AMERICAN UNIVERSITY OF BEIRUT

ON THE ENLARGED KRYLOV SUBSPACE METHOD: SRE-CG2

by

YOUMNA ABBAS ALHAJJ

A thesis
submitted in partial fulfillment of the requirements
for the degree of Master of Science
to the Department of Mathematics
of the Faculty of Arts and Sciences
at the American University of Beirut

Beirut, Lebanon

AMERICAN UNIVERSITY OF BEIRUT

ON THE ENLARGED KRYLOV SUBSPACE METHOD: SRE-CG2

by

YOUMNA ABBAS ALHAJJ

Approved by:

[Signature]

Dr. Sophie Moufawad, Assistant Professor
Department of Mathematics, AUB — Advisor

[Signature]

Dr. Nabil Nassif, Professor
Department of Mathematics, AUB — Co-Advisor

[Signature] Ahmad Sabra

Dr. Ahmad Sabra, Assistant Professor
Department of Mathematics, AUB — Member of Committee

[Signature] Bassam Mourad

Dr. Bassam Mourad, Associate Professor
Department of Mathematics, Lebanese University — Member of Committee

Date of thesis/dissertation defense: September 4,2021

AMERICAN UNIVERSITY OF BEIRUT

THESIS, DISSERTATION, PROJECT RELEASE FORM

Student Name:	Al Hajj	Youmna	Abbas
	Last	First	Middle

☑ Master's Thesis ○ Master's Project ○ Doctoral Dissertation

☐ I authorize the American University of Beirut to: (a) reproduce hard or electronic copies of my thesis, dissertation, or project; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes.

☑ I authorize the American University of Beirut, to: (a) reproduce hard or electronic copies of it; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes after: **One ✓ year from the date of submission of my thesis, dissertation or project.**
Two ___ years from the date of submission of my thesis , dissertation or project.
Three ___ years from the date of submission of my thesis , dissertation or project.

September 11, 2021

Signature Date

This form is signed when submitting the thesis, dissertation, or project to the University Libraries

Acknowledgements

It is exactly 3:09 am in the morning on August,20,2021. The day I knew this space is going to be free of theorems, proofs, corollaries..I couldn't but stay up late and put a few words on something, as my advisor calls it, that will 'remain forever'. I'm not sure how I will feel after the submission and I wonder if the usual cup of coffee that was a must before every edit will miss me.. I worked on the thesis while Lebanon was literally collapsing! I had to leave home and family and move to my grandparents house in the village to 'try to keep my usual life going', I needed internet, electricity..! You see the pages, I FEEL them. I have always believed in the power of hard work but this experience proved me that hard work alone isn't as powerful as I thought. I had the will, the motive ,the love and the passion but was on the verge of quitting. The news weren't helping at all but neither avoiding them did. Moral of the story, that is when I felt my advisor stepped in. Without her actually realizing, with every reply to my email I was recharging..

Allow me now to give my advisor, Assistant Professor Sophie Moufawad, a huge thank you wrapped up with appreciation and gratitude, and another thank you for my co-advisor professor Nabil Nassif who was the reason behind my collaboration with professor Sophie. A big thank you for my committee members: Assistant Professor Ahmad Sabra and professor Bassam Mourad.

I would like to thank my parents and my grandparent who were very welcoming. Special thanks to my boss at work who was very understanding of this phase and to my very sincere work mate who offered an unconditional support! Finally, thanks to my laptop who made it through with me ! Bon Voyage on this thesis that shall forever remain .. !

An Abstract of the Thesis of

Youmna Abbas Alhajj for Master of Science

Major: Mathematics

Title: On The Enlarged Krylov Subspace Method: SRE-CG2

We are interested in solving sparse systems of linear equations $Ax = b$ with Krylov subspace methods. A new approach for reducing communication in Krylov subspace methods was introduced in [6] and [2] that consists of enlarging the Krylov subspace by a maximum of t vectors per iteration, based on a domain decomposition of the graph of A. We think of the solution as belonging to the enlarged Krylov subspace which is a superset of the Krylov subspace. Several enlarged conjugate gradient versions that converge faster than CG in terms of iterations were introduced, such as MSDO-CG [3], SRE-CG2 and SRE-CG [2]. In this thesis we introduce a variant of SRE-CG2: flexible SRE-CG2 and test its effectiveness in terms of iterations and time for convergence as compared to SRE-CG2.

Contents

Acknowledgements v

Abstract vi

1 Introduction 1

2 Krylov Subspace Methods 3

2.1 The Krylov Subspace and its Properties 4

2.2 Projection Methods 5

2.3 Conjugate Gradient Method 6

2.4 Generalized Minimum Residual (GMRES) 12

2.4.1 Arnoldi's Algorithm 13

2.4.2 Minimizing the residual norm 13

2.5 Bi-Conjugate Gradient 16

2.6 Preconditioning 18

2.6.1 Preconditioned Conjugate Gradient 20

3 Enlarged Krylov Subspace Methods 21

3.1 Enlarged Krylov Subspace 23

3.2 Enlarged Krylov Subspace Projection Methods based on CG 24

3.2.1 Multiple Search Direction with Orthogonalization Conjugate Gradient (MSDO-CG) Method 26

3.2.2 SRE-CG (Short Recurrence Enlarged CG) Versions 27

3.2.3 Preconditioning: 29

4 Flexible SRE-CG2 **32**

4.1 SRE-CG2 Variants 32

4.2 Testing 36

5 Conclusion **39**

Bibliography 40

List of Figures

3.1 Partitioning of an 8×8 Sparse symmetric matrix (figure obtained from [10]) and the corresponding projection operator $T(v)$. 21

4.1 Norm of residual vector in SRE-CG2 for partitions: 32, 64 and 128 in Ani3D matrix. 33

List of Tables

4.1 Comparison of the number of iteration k and time needed till convergence in the SRE-CG2 versions for matrices NH2D, Sky2D, Sky3D and Ani3D with number of partitions $t = 32, 64$ and 128. 36

4.2 Comparison of the number of iteration k and time needed till convergence in the original SRE-CG2 and flexible SRE-CG2 version for matrices NH2D, Sky2D, Sky3D and Ani3D with number of partitions $t = 32, 64$ and 128 and three switchTol $10^{-3}, 10^{-5}$ and 10^{-7}. The switch iteration (switch) is reported for flexible SRE-CG2. 37

Chapter 1

Introduction

Solving the system $Ax = b$, for an $n \times n$ sparse matrix A and $n \times 1$ vectors x and b has captured a lot of interest because such a system occurs widely in applied Mathematics either as direct formulations of real world problems or as a part of the numerical analysis of some other problems. Many methods were developed that converge to the solution fast in terms of the number of iterations and runtime. Such methods can be grouped into two categories: Direct methods and Iterative Methods.

Direct methods are methods that solve the linear system $Ax = b$ with a finite number of steps and ends up giving the exact solution to the system. When we speak about direct methods, we are speaking about methods such as LU decomposition which was introduced by Alan Turing in 1948. Another example is Cholesky decomposition which is named after André-Louis Cholesky. For small matrices or matrices with special structures, these methods are used. However, for large sparse matrices, the obtained decomposition matrices (such as the LU decomposition) become dense and require more memory storage. The second category is: Iterative methods. Iterative methods are again methods for solving the linear system $Ax = b$ which are commonly used with sparse large systems, especially those produced by discretizing partial differential equations. These methods use an initial value to generate a sequence of approximate solutions. There is no fixed stopping step, these methods stop after a certain chosen tolerance is reached, and the

exact solution is not always found, an approximate solution is computed instead. These methods require less memory than the direct methods.

In this thesis, we start in chapter 2 by introducing the Krylov subspace methods which are one of the very practical iterative methods used when solving the system $Ax = b$. This method requires only computing the application of the matrix to a vector, i.e, the matrix-vector product.

In chapter 3, we present a new approach of enlarging the Krylov subspace by a maximum of t vectors per iteration as introduced in [6] and [2], then we discuss the enlarged Kyrlov projection CG methods: MSDO-CG [3], SRE-CG2 and its variants [2]. SRE-CG2 with its variants construct and A-orthonormalize the basis vectors of the enlarged Krylov subspace, t vectors at a time. The number of stored and used basis vectors in the A-orthonormalization process yields one of the three versions: SRE-CG2, SRE-CG2(trunc), SRE-CG. SRE-CG2 A-orthonormalizes the t current basis vectors against all previous basis vectors, SRE-CG2 (trunc) A-orthonormalizes basis vectors against the last t trunc number of previous vectors, while in SRE-CG we A-orthonormalize the current basis vectors against the $2t$ previous vectors. SRE-CG2 is the fastest of the three in terms of convergence and sequential runtime but requires more memory storage. For this purpose, i.e the purpose of reducing memory storage and keeping the convergence under control, a flexible SRE-CG2 version is introduced in this thesis.

In chapter 4, after introducing flexible SRE-CG2, we test it for different inputs and compare its behavior to the original versions of SRE-CG2 introduced in [2].

In Chapter 5, we conclude the promising behavior of flexible SRE-CG2 versions in terms of number of iterations and runtime.

Chapter 2

Krylov Subspace Methods

The Krylov Subspace methods are named after the Russian applied mathematician and naval engineer Alexei Krylov. These methods seek a sequence $x_1, x_2, ..x_n$ that minimizes some measure of the error over the corresponding affine space

$$x_0 + \kappa_i(A, r_0),\ i = 1, .., n$$

with $r_0 = b - Ax_0$ and x_0 the initial guess, where at a step i the Krylov subspace is given to be $\kappa_i(A, r_0) = span\{r_0, Ar_0, A^2r_0, .., A^{i-1}r_0\}$ with $r_i = b - Ax_i$ called the $i-th$ residual vector. Any $x \in x_0 + \kappa_i(A, r_0)$ can be expressed as $x = x_0 + \sum_{t=0}^{i-1} A^t r_0$. Once the residual vector is small enough (according to some chosen tolerance), an approximate solution is reached.

Such methods are considered effective because they rely on matrix-vector multiplication which costs less than matrix- matrix multiplication. At the $i-th$ step $A^i r_0 = A(A^{i-1}r_0)$ obtained by simply multiplying A by the vector $A^{i-1}r_0$ which was previously computed.

We start by introducing the Krylov subspace and its properties in section 2.1. In section 2.2, we discuss projection methods which are based on two conditions: Subspace condition and the Petrov-Galerkin condition. In sections 2.3, 2.4 and 2.5 we present three projection methods: Conjugate gradient, Generalized Minimum Residual and Bi-Conjugate Gradient. In the last section, we explain how the preconditioning of Krylov

projection methods work and give the preconditioned conjugate gradient as an example.

2.1 The Krylov Subspace and its Properties

Definition 2.1. *A Krylov subspace generated by an $n \times n$ matrix A and a vector f in $\mathbb{R}^n$ is the subspace spanned by the vectors of Krylov sequence:*

$$\kappa_m(A, f) = span\{f, Af, A^2 f, ..., A^{m-1} f\}$$

The vectors f and Af are linearly independent if f is non zero and f is not an eigenvector of A.

$\kappa_m(A, f)$ is the subspace of all vectors in $\mathbb{R}^n$ which can be expressed as :

$$x = p(A)f$$

where $p(A)$ is a polynomial of degree not exceeding $m - 1$.

Notice that $\kappa_1(A, f) \subset \kappa_2(A, f) \subset \kappa_3(A, f) \subset ...$

We now present a theorem that helps us see why we can use the Krylov subspace instead of other vector spaces such as $\mathbb{R}^n$ or $\mathbb{C}^n$ to search for the solution of $Ax = b$.

Theorem 2.2. *Let x be the solution to the linear system $Ax = b$ (with $x \in \mathbb{R}^n$ or $\mathbb{C}^n$). If A is non singular (i.e $\det(A) \neq 0$), then $x \in x_0 + \kappa_n$.*

Proof. Using Cayley Hamilton theorem we have that $p(A) = 0$, where $p(\lambda) = \det(A - \lambda I) = \det(A) + \sum_{i=1}^{i=n} c_i \lambda^i$ is the characteristic polynomial of A. Multiplying $p(A)$ by y, with $y \in \mathbb{R}^n$ (or $\mathbb{C}^n$), we get

$$0 = p(A)y = \det(A)y + \sum_{i=1}^{n} c_i A^i y$$

dividing it by $\det(A)$ (since it is non-zero), we get: $y = \sum_{i=1}^{n} d_i A^i y$. Take in particular

$y = x - x_0$. Since $r_0 = b - Ax_0 = A(x - x_0)$ we get:

$$y = A^{-1}r_0 = \sum_{i=0}^{n-1} d_i A^i r_0$$

and so

$$x = x_0 + \sum_{i=0}^{n-1} d_i A^i r_0$$

which means that $x \in x_0 + span\{r_0, Ar_0, A^2r_0, ..., A^{n-1}r_0\}$, i.e $x \in x_0 + \kappa_n(A, r_0)$. □

Lemma 2.3. *There is a positive integer $\nu := \nu(y, A)$ called grade of y with respect to A, such that :*

$$\dim\ \kappa_m(A, y) = \begin{cases} m \text{ if } m \leq \nu \\ \nu \text{ if } m > \nu \end{cases} \tag{2.1}$$

Proof. Refer to [9]. □

Corollary 2.4. *Let $x^\star$ be the solution of $Ax = b$, where A is invertible, and x_0 be any initial approximation of it and $r_0 = b - Ax_0$ be the corresponding residual. Moreover, let $\nu =: \nu(r_0, A)$ then $x^\star \in x_0 + \kappa_\nu(A, r_0)$, i.e ν is the smallest integer satisfying $A^{-1}b \in x_0 + \kappa_\nu(A, r_0)$.*

Proof. Refer to [9] □

The aim of introducing such spaces is solving the Linear system $Ax = b$ for a large sparse matrix A.

2.2 Projection Methods

The Krylov projection methods seek an approximate solution x_m from an affine subspace $x_0 + \kappa_m(A, r_0)$ by imposing the Petrov-Galerkin condition $r_m = b - Ax_m \perp L_m$, where L_m is another subspace of dimension m not necessarily the Krylov subspace and x_0 is the initial guess.

Two conditions govern the Krylov projections methods at each iteration k:

1. Subspace Condition: $x_k \in x_0 + \kappa_k(A, r_0)$

2. Petrov-Galerkin Condition: $r_k = b - Ax_k \perp L_k$

The different versions of Krylov subspace projection methods arise from different choice of L_m.

From now on, we refer to $\kappa_m(A, r_0)$ as κ_m.

Two broad choices for L_m give rise to the best known techniques:

- $L_m = \kappa_m$ full Orthogonal projection method such as Conjugate Gradient specialized for symmetric positive definite matrices.

- $L_m = A\kappa_m$ Oblique projection methods such as GMRES (Generalized Minimum Residual) specialized for non symmetric matrices.

We state the following two key theorems whose proof are provided in [8].

Theorem 2.5. *If $L_m = A\kappa_m$, $\bar{x}$ is the approximate solution provided by the Petrov-Galerkin method if and only if it minimizes the 2-norm of the residual vector $b - Ax$, for all x in the affine subspace $x_0 + \kappa_m$.*

Theorem 2.6. *If A is symmetric positive definite and $L_m = \kappa_m$, $\bar{x}$ is the Petrov-Galerkin approximate solution if and only if it minimizes $(x - \bar{x})^T A(x - \bar{x})$ over $x_0 + \kappa_m$.*

We are now in a position to discuss some of these projection methods for solving the linear system $Ax = b$ using Krylov subspaces.

2.3 Conjugate Gradient Method

This method , introduced by Hestenes and Stiefel in 1952 [4], is a full orthogonal projection method based on taking $L_m = \kappa_m$ and so the Petrov-Galerkin conditions becomes $r_m = b - Ax_m \perp \kappa_m$. It is one of the widely used techniques for solving sparse Symmetric

Positive Definite Linear systems.

We first define

$$f(x) = \frac{1}{2}x^T Ax - x^T b \tag{2.2}$$

Since A is symmetric positive definite

$$\nabla f(x) = Ax - b$$

and the minimum value of $f(x)$ occurs when $Ax = b$.

The CG solves the linear System by using the approximation : $x_{k+1} = x_k + \alpha_{k+1}p_{k+1}$ with some scalar α_{k+1} to be defined. The vector $p_k \in \kappa_k$ is called the search direction will be defined later. The first search direction p_1 is chosen to be r_0.
In the conjugate gradient method, the residuals satisfy

$$r_k^T y = 0, \forall y \in \ \kappa_k(A, r_0)$$

and this is simply due to the Petrov-Galerkin condition.

Proposition 2.7. *$r_i^T r_j = 0$ for $i \neq j$, i.e the residuals are orthogonal.*

Proof. Since $r_k = b - Ax_k$ with $x_k \in \kappa_k$, then $r_k \in \kappa_{k+1}$. And so by using the Petrov-Galerkin condition and using that the sequence of Krylov subspaces is an increasing sequence, the residuals are orthogonal. □

Proposition 2.8. *Assuming the residuals are non-zero, the sequence of residuals $\{r_0, .., r_{k-1}\}$ form an orthogonal basis for $\kappa_k(A, r_0)$.*

Proof. The residuals being orthogonal proves that they are linearly independent. Also, they are in $\kappa_k(A, r_0)$ of dimension k , so they form a basis. □

We now introduce the A-norm.

Definition 2.9. *If A is an $n \times n$ positive definite matrix, x and $y \in \mathbb{R}^n$, then $\langle x, y\rangle_A = x^T Ay$ is an inner product. In case $\langle x, y\rangle_A$=0, x and y are said to be A -conjugate. The corresponding norm $\|x\|_A$=$\sqrt{x^T Ax}$ is called the A-norm or the energy norm.*

Recall $f(x)$=$\frac{1}{2}x^T Ax - x^T b$ which can be written in terms of the A-norm as follows:

$$f(x) = \frac{1}{2}\|x\|_A^2 - \langle b, x\rangle$$

where $\langle b, x\rangle$ is the Euclidean inner product. Thus, if $\bar{x}$ is the approximation for the minimum of $f(x)$, defining the error to be $e = x-\bar{x}$, we can rewrite $f(x) = \frac{1}{2}e^T Ae-\frac{1}{2}\|\bar{x}\|_A^2$. Thus to minimize $f(x)$ we must minimize the A-norm of the error and thus by 2.6, the obtained minimization will be the Petrov-Galerkin approximation.
Before defining the search directions, we give the following proposition.

Proposition 2.10. *The directions $\{p_1, .., ..., p_k\}$ are A-orthogonal basis of the Krylov subspace κ_k as long as the residuals are non zero.*

Proof. $r_{k+1} = b - Ax_{k+1} = b - A(x_k + \alpha_{k+1}p_{k+1}) = r_k - \alpha_{k+1}Ap_{k+1}$. Using Petrov-Galerkin condition, we see that Ap_{k+1} is orthogonal to the Krylov subspace $\kappa_k(A, r_0)$, but $p_j \in \kappa_j(A, r_0)$ so $(Ap_{k+1})^T p_j = 0$ when $j \leq k$ and by symmetry of A, we get $p_k^T Ap_{j+1} = 0$ k $\leq$ j, which means that the direction vectors are A-orthogonal implying that they are linearly independent. □

Note: The current residual is orthogonal to the previous search directions in addition to the previous residual. Remember that $p_k \in \kappa_k$ and thus using the Petrov-Galerkin condition, as $r_k \perp \kappa_k$, then $p_k^T r_k = 0$, which is a strength point for CG as it's guaranteed always to produce a new, linearly independent search direction unless the residual is zero, in which case the problem is already solved.

We choose α_k as follow:

$$
\begin{aligned}
& p_k^T r_k = 0 \\
& p_k^T (b - Ax_k) = 0 \\
& p_k^T (b - A(x_{k-1} + \alpha_k p_k)) = 0 \\
& p_k^T r_{k-1} - \alpha_k p_k^T A p_k = 0 \\
& \therefore \alpha_k = \frac{p_k^T r_{k-1}}{p_k^T A p_k}
\end{aligned}
$$

This expression for α_k will be modified later on after giving the approximation of the search directions.

Lemma 2.11. *The step size α_k given above gives the exact minimum along the direction p_k.*

Proof. We want to minimize $g(\alpha_i)$=$f(x_{i-1} + \alpha_i p_i)$

$$
\begin{aligned}
g(\alpha_i) &= \frac{1}{2}(x_{i-1} + \alpha_i p_i)^T A(x_{i-1} + \alpha_i p_i) - (x_{i-1} + \alpha_i p_i)^T b \\
&= \frac{1}{2} x_{i-1}^T A x_{i-1} + \frac{\alpha_i^2}{2} p_i^T A p_i + \alpha_i x_{i-1} A p_i - x_{i-1}^T b - \alpha_i p_i^T b
\end{aligned}
$$

and

$$
\begin{aligned}
g'(\alpha_i) &= \alpha_i p_i^T A p_i + x_{i-1}^T A p_i - p_i^T b \\
&= \alpha_i p_i^T A p_i + p_i^T A x_{i-1} - p_i^T b \\
&= \alpha_i p_i^T A p_i + p_i^T (A x_{i-1} - b) \\
&= \alpha_i p_i^T A p_i - p_i^T r_{i-1}
\end{aligned}
$$

Thus the minimum occurs when

$$\alpha_i = \frac{p_i^T r_{i-1}}{p_i^T A p_i}$$

□

p_i is defined iteratively according to the following iterative relation:

$$p_{k+1} = r_k + \beta_{k+1} p_k \tag{2.3}$$

with $p_1 = r_0$. We can see that from 2.3, we can simplify the expression of α_{k+1}.

Multiply 2.3 by r_k^T we get, $r_k^T p_{k+1} = \|r_k\|_2^2 + \beta_{k+1} r_k^T p_k$. This now yields

$$r_k^T p_{k+1} = \|r_k\|_2^2 \tag{2.4}$$

Substituting for α_{k+1}, we get

$$\alpha_{k+1} = \frac{\|r_k\|_2^2}{p_{k+1}^T A p_{k+1}}$$

Now referring to 2.3 and by using the A-orthogonality property of the search directions, we have:

$$\begin{aligned}
p_{k+1}^T A p_k &= r_k^T A p_k + \beta_{k+1} p_k^T A p_k \\
0 &= r_k^T A p_k + \beta_{k+1} p_k^T A p_k \text{ (by } A\text{-orthogonality of the search directions)} \\
\beta_{k+1} &= -\frac{r_k^T A p_k}{p_k^T A p_k}
\end{aligned}$$

Now let's try to explore how we can simplify $r_k^T A p_k$ and thus simplify β_{k+1}. By multiplying the expression of r_k by r_{k+1}^T:

$$\begin{aligned}
r_k^T r_{k+1} &= r_k^T r_k - \alpha_k r_k^T A p_k \\
\alpha_k r_k^T A p_k &= \|r_k\|_2^2 - r_k^T r_{k+1} \\
r_k^T A p_k &= \frac{\|r_k\|_2^2}{\alpha_k}
\end{aligned}$$

Now, $\beta_{k+1} = -\frac{\frac{\|r_k\|_2^2}{\alpha_k}}{p_k^T A p_k}$. However, from the equation of α_k, $p_k^T A p_k = \frac{\|r_{k-1}\|_2^2}{\alpha_k}$, finally $\beta_{k+1} = \frac{\|r_k\|_2^2}{\|r_{k-1}\|_2^2}$. This will be discussed in Algorithm 1.

Convergence: Let x_m be the approximate solution obtained at the m-th step of the conjugate Gradient Algorithm, and $\bar{x}$ the exact solution and define

$$\eta = \frac{\lambda_{min}}{\lambda_{max} - \lambda_{min}}$$

where λ_{min} and λ_{max} are the minimum and maximum eigenvalues of A respectively with $\lambda_{min} \neq \lambda_{max}$. Then

$$\|\bar{x} - x_m\|_A \leq \frac{\|\bar{x} - x_0\|_A}{C_m(1 + 2\eta)}$$

where C_m is the Chebyshev polynomial of degree m of first kind.

Let k be the spectral condition number of matrix A, i.e $k = \frac{\lambda_{max}}{\lambda_{min}}$ then:

$$\|\bar{x} - x_m\|_A \leq 2 \left(\frac{\sqrt{k} - 1}{\sqrt{k} + 1} \right)^m \|\bar{x} - x_0\|_A$$

Hence, the convergence is a function of the spectral condition number, refer to [9].

The CG fails if A is not symmetric positive definite, thus a method that works for any matrix A shall be introduced.

After discussing an orthogonal projection method(GC) where $L_m = \kappa_m$, we are in a position to introduce an oblique projection method where $L_m = A\kappa_m$ i.e $(b - Ax_m \perp A\kappa_m)$ known as GMRES that works for any sparse matrix A not necessarily symmetric.

Algorithm 1 CG Algorithm

Input: A: an $n \times n$ matrix ; b: $n \times 1$ right hand side vector; x_0 initial guess; tol the given tolerance.

Output: x_k approximate solution of the system $Ax = b$.

$r_0 = b - Ax_0, \rho_0 = \|r_0\|_2$ and $k = 1$;
while $\sqrt{\rho_{k-1}} > \epsilon\|b\|_2$ and $k < k_{max}$ **do**
    if $k == 1$ **then**
        $p_1 = r_0$;
    else
        $\beta_k = \dfrac{\rho_{k-1}}{\rho_{k-2}}$ and $p_k = r_{k-1} + \beta_k p_{k-1}$;
        $w_k = Ap_k$;
        $\alpha_k = \dfrac{\rho_{k-1}}{p_k^t w}$;
        $x_k = x_{k-1} + \alpha_k p_k$;
        $r_k = r_{k-1} - \alpha_k w_k$
        $\rho_k = \|r_k\|_2^2$;
    end if
end while

2.4 Generalized Minimum Residual (GMRES)

Introduced by Yousef Saad and Martin H. Schultz ,1986 [9], this method is a projection method based on taking $L_m = A\kappa_m$ in which κ_m is the m-th Krylov subspace with $v_1 = \dfrac{r_0}{\|r_0\|_2}$, where v_1 is the first basis vector constructed using Arnoldi's Algorithm as will be discussed later. Such a technique minimizes residual norm $\|b - Ax\|_2$ over all vectors in $x_0 + \kappa_m$.The implementation of the algorithm is based on an approach which finds a basis of κ_m. This method works for any non-singular matrix.

Any vector x in $x_0 + \kappa_m$ can be written as $x = x_0 + V_m y$, where y is an m-vector and V_m is the $n \times n$ matrix containing the orthonormal column basis vectors of the Krylov subspace κ_m which are computed using Arnoldi's Algorithm.

2.4.1 Arnoldi's Algorithm

One algorithm for building an orthonormal basis for the Krylov subspace is Arnoldi's method. At each step, the algorithm multiplies Arnoldi vector v_j by A and then orthonormalizes the resulting vector w_j against all previous v_j's by a standard Gram-Schmidt procedure.

This method yields to a Hessenberg matrix $\bar{H}_m$ with elements $h_{ij} = \langle w_j, v_i \rangle$.

Denote by V_m the $n \times m$ matrix with column vectors $v_1, v_2, ..v_m$,by $\bar{H}_m$ the $(m+1) \times m$ Hessenberg matrix whose non-zero entries h_{ij} are defined by Arnoldi-modified gram-Schmidt Algorithm and H_m the matrix obtained from the Hessenberg matrix by deleting its last row. Then the following relations hold: please refer to [8].

$$AV_m = V_m H_m + w_m e_m^T$$

$$V_m^T A V_m = H_m$$

2.4.2 Minimizing the residual norm

Define

$$J(y) = \|b - Ax\|_2 = \|b - A(x_0 + V_m y)\|_2$$

Now,

$$b - Ax = b - A(x_0 + V_m y) = r_0 - AV_m y$$

From Arnoldi's Algorithm we have:

$$v_1 = \frac{r_0}{\|r_0\|}, \|r_0\| = \beta$$

In addition

$$AV_m = V_m H_m + w_m e_m^T = V_{m+1} \bar{H}_m$$

Thus

$$b - Ax = r_0 - AV_m y = \beta v_1 - V_{m+1}\bar{H}_m y = (\beta e_1 - \bar{H}_m y)V_{m+1}$$

Then

$$J(y) = \|(\beta e_1 - \bar{H}_m y)V_{m+1}\|_2$$

However, since the column vectors of V_{m+1} are orthonormal then $\|V_{m+1}\|_2 = 1$, we get

$$J(y) = \|\beta e_1 - \bar{H}_m y\|$$

The GMRES approximation is the unique vector of $x_0 + \kappa_m$ which minimizes $J(y)$ i.e

$$x_m = x_0 + V_m y_m$$

where

$$y_m = \min_y \|\beta e_1 - \bar{H}_m y\|_2$$

The minimizer is inexpensive to compute as it requires the solution of $(m+1) \times m$ least square problems where m is typically small compared to the actual matrix size n.
In order to solve the least squares problem

$$y_m = \min_y \|\beta e_1 - \bar{H}_m y\|_2$$

it may be convenient to transform the Hessenberg matrix into upper triangular form using given rotations, refer to [9].
Considering this, we get

$$\|\beta e_1 - \bar{H}_m y\|_2 = |\gamma_{m+1}|^2 + \|g_m - R_m y\|_2^2$$

where R_m is an $n \times n$ upper triangular matrix obtained by transforming the Hessenberg matrix into upper triangular form by using plane rotations, g_m and $m \times 1$ vectors, $|\gamma_{m+1}|$ is the residual norm, refer to [8].

We see that the minimum will be obtained when the second term of the above expression is zero i.e

$$\|g_m - R_m y\|_2^2 = 0$$

Therefore, the least square problem is converted into $m \times m$ matrix solver with R_m upper triangular which can be easily solved by backward substitution.

The GMRES process must be stopped once the residual norm $|\gamma_{m+1}|$ is small enough as will be in algorithm 2.

Convergence: For an $n \times n$ matrix, GMRES converges by at most n-steps.
Assume that A is a diagonalizable matrix and let $A = XDX^{-1}$,where D is the diagonal eigenvalue matrix , D=diag$\{\lambda_1, ..., \lambda_n\}$. Define

$$\epsilon^{(m)} = \min_{p \in P_m, p(0)=1} \max_{i=1,2,..n} |p(\lambda_i)|$$

where P_m is the space of polynomials of degree at most m. Thus, the residual norm achieved by the $m - th$ step of GMRES satisfies the inequality:

$$\|r_m\|^2 \leq \kappa_2(X)\epsilon^{(m)}\|r_0\|_2$$

Where $\kappa_2(X) = \|X\|_2\|X\|_2^{-1}$. Thus, the method converges for any guess value x_0. The convergence rate depends on the condition number of the eigenvector matrix.

Algorithm 2 GMRES

Input: A: an $n \times n$ matrix ; b: $n \times 1$ right hand side vector; x_0 initial guess; tol the given tolerance.

Output: x_k approximate solution of the system $Ax = b$.

1: $r_0 = b - Ax_0, \beta := \|r_0\|_2$,and $v_1 := \frac{r_0}{\beta}$;
2: **for** $j = 1, 2, ..., m$ **do**
3: $w_j = Av_j$;
4: **for** $i = 1, ..., j$ **do**
5: $h_{i,j} = (w_j, v_i)$;
6: $w_j = w_j - h_{i,j}v_i$;
7: **end for**
8: $h_{j+1,j} = \|w_j\|_2$;
9: **if** $h_{j+1,j} \leq$ tol **then**
10: $m = j$;
11: break (go to line 16)
12: **else**
13: $v_{j+1} = \frac{w_j}{h_{j+1,j}}$;
14: **end if**
15: **end for**
16: Define the $(m+1) \times m$ matrix $\bar{H}_m = \{h_{i,j}\}_{1 \leq i \leq m+1, 1 \leq j \leq m}$ and $W_m = w_{j\,1 \leq j \leq m}$;
17: Compute y_m the minimizer $\|\beta e_1 - \bar{H}_m y\|_2$ and $x_m = x_0 + W_m y_m$;

2.5 Bi-Conjugate Gradient

The Bi-Conjugate Gradient method (BiCG) is a Krylov projection process onto

$$\kappa_m = span\{v_1, Av_1, ..., A^{m-1}v_1\}$$

orthogonality to

$$L_m = span\{w_1, A^T w_1, ..., (A^T)^{m-1} w_1\}$$

As in CG, it is defined by the subspace condition:

$$x_{k+1} = x_k + \alpha_{k+1} p_{k+1}, \alpha_{k+1} \in \mathbb{R}, p_{k+1} \in \kappa_{k+1}(A, r_0)$$

as well as by the orthogonality property :

$$r_k^T v = 0, \forall v \in \kappa_k(A^T, \tilde{r}_0) \tag{2.5}$$

where $\tilde{r}_0$ is a given vector.

We define a sequence of shadow residuals by :

$\tilde{r}_j = P_j(A)\tilde{r}_0$, where $P_j(A)$ is a polynomial of A of degree j.

The subspace condition includes a shadow condition:

$$\tilde{x}_{k+1} = \tilde{x}_k + \alpha_{k+1} \tilde{p}_{k+1}$$

The bi-orthogonality property of the residuals include equation 2.5 along with:

$$\tilde{r_k}^T v = 0 \, \forall v \in \kappa_k(A, r_0)$$

The search directions satisfy the bi-conjugacy condition:

$$(\tilde{p}_k)^T A p_i = 0, \text{ for } \ i \neq k$$

In practice, $\tilde{r_0} = r_0$ and $p_1 = r_0 = \tilde{p_1}$.

The search directions and the shadow search directions are given by the following approximations:

- $p_{k+1} = r_k + \beta_{k+1} p_k$
- $\tilde{p}_{k+1} = \tilde{r}_k + \beta_{k+1} \tilde{p}_k$

Proposition 2.12. *If* $\tilde{r_j}^T r_j \neq 0$ *for* $0 \leq j \leq k-1$ *then the sequence of residuals obtained*

from Bi-CG: $\{r_0, .., r_{k-1}\}$ *forms an orthogonal basis for* $\kappa_k(A, r_0)$. *Also, the sequence of the shadow residuals* $\{\tilde{r}_0, ...\tilde{r}_{k-1}\}$ *form an orthogonal basis for* $\kappa_k(A^T, \tilde{r}_0)$.

Proposition 2.13. *If* $p_j^T A p_j \neq 0 \ \forall \ \ 0 \leq j \leq k-1$, *then the sequence of directions* $\{p_1..., p_k\}$ *forms an orthogonal basis to the Krylov subspace* $\kappa_k(A, r_0)$ *also the sequence of shadow directions* $\{\tilde{p_1}, .., \tilde{p_k}\}$ *forms an orthogonal basis to* $\kappa_k(A^T, \tilde{r_0})$.

Proof. Similar to that of conjugate gradient. □

Algorithm 3 Bi-CG

Input: A: an $n \times n$ matrix ; b: $n \times 1$ right hand side vector; x_0 initial guess; tol the given tolerance.

Output: x_k approximate solution of the system $Ax = b$.

1: $r_0 = b - Ax_0$;

2: Choose $\tilde{r}_0$ so that $\langle r_0, \tilde{r}_0 \rangle \neq 0$;

3: set $p_1 := r_0$ and $\tilde{p}_1 = \tilde{r}_0$

4: **for** $i = 1, ...$till convergence **do**

5: $\alpha_i = \dfrac{\langle r_{i-1}, \tilde{r}_{i-1} \rangle}{\langle Ap_i, \tilde{p}_i \rangle}$;

6: $x_i = x_{i-1} + \alpha_i p_i$;

7: $r_i = r_{i-1} - \alpha_i A p_i \ \tilde{r}_i = \tilde{r}_{i-1} - \alpha_i A^T \tilde{p}_i$;

8: $\beta_i = \dfrac{\langle r_i, \tilde{r}_i \rangle}{\langle r_{i-1}, \tilde{r_{i-1}} \rangle}$;

9: $p_{i+1} = r_i + \beta_i p_i$;

10: $\tilde{p}_{i+1} = \tilde{r}_i + \beta_i \tilde{p}_i$;

11: **end for**

2.6 Preconditioning

Krylov subspace methods are rarely used without preconditioning. In conjunction with iterative methods, preconditioning is often the vital component in enabling the solution of such systems when the dimension is large. The term 'preconditioning' was introduced by Alan Turing and it corresponds to the reduction of the condition number k to guarantee faster convergence. It is a mean of transforming the original linear system into one

which has the same solution, but which is likely to be easier to solve with an iterative solver (such as Krylov subspace methods).

Without the use of preconditioning, the Krylov solver may not converge, or may take many iterations to return an acceptably accurate solution.

Starting with the equation $Ax = b$, we want the preconditioning matrix M to be non-singular to resemble A in some sense. We are going to solve the system for $M^{-1}A$, so if M resembles A, then $M^{-1}A$ will start resembling the identity which is a 'nice' matrix. Once the preconditioning matrix M is chosen, there are three ways in which we precondition:

1. applying the preconditioning matrix to the left:

$$M^{-1}Ax = M^{-1}b$$

2. applying the preconditioning matrix to the right:

$$AM^{-1}u = b \quad x = M^{-1}u$$

3. Mixed preconditioning, Left and Right:

$$M = M_L M_R$$

$$M_L^{-1}AM_R^{-1}u = M_L^{-1}b, \text{where} \;\; x = M_R^{-1}u$$

The latter is used when we want to preserve the symmetry of the matrix in the linear system because it is not always the case that the product of symmetric matrices is symmetric. Namely if M^{-1} and A are symmetric, the product $M^{-1}A$ may not be symmetric. There are a lot of preconditioning techniques. We mention some of these techniques briefly and for more information please see [1]

- Jacobi, SOR, and SSOR Preconditioners: Where A is split into $A{=}M-N$, and M is the preconditioner.
- Block Preconditioners: Where A is written as $A = L + D + U$ where D is a diagonal matrix, L block strictly lower triangular matrix and U block strictly upper triangular matrix, and the preconditioner is the product of the three matrices.

In this thesis, we discuss the preconditioned CG , for more details refer to [8].

2.6.1 Preconditioned Conjugate Gradient

Starting with a preconditioner matrix M that is also assumed to be symmetric positive definite as A, we compute its Cholesky factorization: $M = L_M L_M^T$ and this yields the equation:

$$L_M^{-1} A L_M^{-T} u = L_M^{-1} b, x = L_M^{-T} u \tag{2.6}$$

and we solve the system for the new matrix $B = L_M^{-1} A L_M^{-T}$ where the matrix obtained is now symmetric positive definite with a better condition number. After finding u, we get x using backward substitution.

Another, yet identical approach has been discussed where the usual Euclidean inner product in the Conjugate Gradient algorithm is replaced by the M-inner product (refer to [8]).

Chapter 3

Enlarged Krylov Subspace Methods

As discussed in the previous chapter, these methods seek a solution for $Ax = b$ in the so-called 'Krylov Subspace', where with every iteration, the dimension of the Krylov subspace increases, and the solution will eventually be found in 'The Largest Krylov Subspace', i.e κ_ν, however it is approximated from a smaller subspace.

A new approach for reducing communication in Krylov subspace methods was introduced in [2]. It consists of enlarging the Krylov subspace by t vectors per iteration. First, the domain of the $n \times n$ matrix A is partitioned into t subdomains according to a graph partitioning method. Assuming the index domain δ=$\{1, 2, .., n\}$ is divided into t subdomains, then $\delta = \bigcup_{i=1}^{t} \delta_i$.

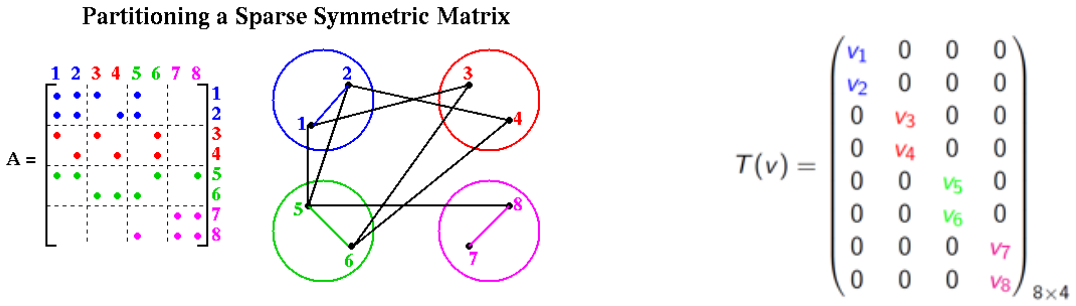

Figure 3.1: Partitioning of an 8×8 Sparse symmetric matrix (figure obtained from [10]) and the corresponding projection operator $T(v)$.

Then the residual vector is split into t vectors, i.e a vector per domain. Defining $T_i(x)$ to be the operator that projects the $n \times 1$ vector x into the i-th subdomain δ_i, so T_i replaces all vector elements not in the i-th subdomain by zero. We also define $T(x)$ to be the operator that takes an $n \times 1$ vector x and gives as an output $(T_1(x), .., T_t(x))$. At each iteration k, the residual vector is multiplied by A. Thus, at a step k the new vectors produced are $\{A^{k-1}T_1(r_0), ..., A^{k-1}T_t(r_0)\}$. These vectors make up the new basis vectors of the enlarged Krylov subspace $\varkappa_{t,k}$, where t corresponds to the number of partitions of the matrix A and k the iteration. Such vectors along with the previous vectors span the enlarged Krylov subspace as will be defined in this chapter.

The obtained enlarged Krylov subspace $\varkappa_{t,k}(A, r_0)$ is a superset of the Krylov subspace $\kappa_k(A, r_0)$, thus it makes sense to search for the solution in the enlarged Krylov subspace. The enlarged Krylov projection methods lead to faster convergence in terms of iterations , [2]. In section 3.1, we present two enlarged Krylov subspace versions of the classical CG (discussed in section 2.1) that solve one system with one residual vector per iteration but with multiple basis vectors, namely t vectors per iteration, or multiple search directions.

The versions presented are:

- MSDO-CG which is nothing but MSD-CG that was introduced by Gu et al. [3] but with A-orthornormalized search directions to ensure that it is an enlarged Krylov projection CG method.
- the SRE-CG methods with its variants.

We shed light on the SRE-CG methods introduced in [2], that are a class of enlarged Krylov projection CG methods that solves the system $Ax = b$ by the following approximation: $x_k = x_{k-1} + Q_k\alpha_k$ where Q_k is the $n \times (tk)$ matrix containing the basis vectors of the enlarged Krylov subspace $\varkappa_{t,k}$, α_k step size vector of length tk and $x_k \in x_0 + \varkappa_{t,k}(A, r_0)$. Three versions of this class of methods will be introduced that differ in the number of the basis vectors that are stored and used per iteration for A-orthonormalization.

We start by defining The Krylov subspace and its properties.

3.1 Enlarged Krylov Subspace

Definition 3.1. *Let*

$$\varkappa_{t,k}(A, r_0) = span\{T(r_0), AT(r_0), A^2T(r_0), .., A^{k-1}T(r_0)\}$$
$$= span\{T_1(r_0), .., T_t(r_0), AT_1(r_0), .., AT_t(r_0), ..., A^{k-1}T_1(r_0), ..., A^{k-1}T_t(r_0)\}$$

be an enlarged Krylov subspace of dimension $k \leq z \leq tk$ generated by the matrix A and the vector r_0, and associated to a given partition defined by δ_i for $i = 1, 2, ..t$.

Some of the important properties will be mentioned without the proof, however a reference for them is [6].

Theorem 3.2. *The Krylov subspace $\kappa_k A, r_0) \subset \varkappa_{t,k}(A, r_0)$.*

Proof. For the proof we define $R_0 = [T_1(r_0)T_2(r_0)..,T_t(r_0)]$, the block containing the t vectors obtained from $T(r_0)$. Clearly, $r_0 = R_0 * \mathbb{1}_t$, with the $t \times 1$ vector of ones. Now,

letting $y \in \kappa_t$, $y = \sum_{j=0}^{k-1} a_j A^j r_0 = \sum_{j=0}^{k-1} a_j A^j R_0 * \mathbb{1}_t = \sum_{j=0}^{k-1} \sum_{i=1}^{t} a_j A^j T_i(r_0) \in \varkappa_{t,k}(A, r_0)$. □

Krylov subspace methods seek a solution $x_k \in x_0 + \kappa_k$

A corollary for 3.2 is that we can view the solution as belonging to the enlarged Krylov subspaces, $x_k \in x_0 + \varkappa_{t,k}(A, r_0)$.

The goal is to search for the solution in the enlarged Krylov subspace and approximate it in less iterations. This is justified by the following theorems.

Theorem 3.3. *There exists a smallest integer k_{max} such that $\varkappa_{t,k_{max}}(A, r_0) = \varkappa_{t,k_{max+q}}(A, r_0), \forall q > 0$.*

Remark 3.4. $\varkappa_{t,k+1}(A, r_0) = \varkappa_{t,k}(A, r_0) + span\{A^kT_1(r_0), ..., A^kT_t(r_0)\}$.

The sum in the remark is not direct because it doesn't always happen that the intersection is empty. In fact, we give the following theorem:

Theorem 3.5. *If* $A^k T_\nu(r_0) \in \varkappa_{t,k}(A, r_0), \quad \forall \quad 1 \leq \nu \leq t$, *then* $A^{k+q} T_i(r_0) \in \varkappa_{t,k}(A, r_0)$ *for all* $1 \leq i \leq t$ *and for all* $q > 0$.

The proof of the theorem is done using induction.

The following theorem gives the relationship between the grade of $\kappa_k(A, r_0)$ and $\varkappa_{t,k}(A, r_0)$.

Theorem 3.6. *Let* p_{max}, k_{max} *be such that* $\kappa_{p_{max}}(A, r_0) = \kappa_{p_{max}+q}(A, r_0)$ *and* $\varkappa_{t,k_{max}}(A, r_0) = \varkappa_{t,k_{max}+q}(A, r_0)$ *for* $q > 0$, *then* $k_{max} \leq p_{max}$.

The above theorem justifies our preference of having the solution belong to the enlarged Krylov subspace, as we reach it faster because it has a grade less than that of the Krylov subspace which is explained more in the following theorem.

Theorem 3.7. *The solution* $\bar{x}$ *of* $Ax = b$ *belongs to* $\varkappa_{t,k_{max}}(A, r_0)$ *where* $\varkappa_{t,k_{max}}(A, r_0) = \varkappa_{t,k_{max}+q}(A, r_0)$ *for* $q > 0$

Proof. The solution $\bar{x}$ belongs to $x_0 + \kappa_{p_{max}}(A, r_0) \subset x_0 + \varkappa_{t,p_{max}}(A, r_0)$ with $\kappa_{p_{max}}(A, r_0) = \kappa_{p_{max}+q}(A, r_0) \quad \forall \quad q > 0$ and $p_{max} \geq k_{max}$ by 3.6. Suppose $\bar{x} \in \varkappa_{t,p_{max}}(A, r_0)$ and $\bar{x} \notin$

$\varkappa_{t,k_{max}}(A, r_0)$, then $\varkappa_{t,k_{max}}(A, r_0) \neq \varkappa_{t,p_{max}}(A, r_0)$. However, by definition of k_{max}, $\varkappa_{t,k_{max}}(A, r_0) = \varkappa_{t,k_{max}+q}(A, r_0)$ for $q > 0$, and since $p_{max} \geq k_{max}$, then $\varkappa_{t,k_{max}}(A, r_0) = \varkappa_{t,p_{max}}(A, r_0)$. Contradiction. □

3.2 Enlarged Krylov Subspace Projection Methods based on CG

Very similar to Krylov subspace projection methods, this class of methods is based on two conditions: the subspace condition and the Petrov-Galerkin.

- Subspace condition:

$$x_k \in x_0 + \varkappa_{t,k}(A, r_0)$$

- Petrov Galerkin Condition:

$$r_k \perp \mathscr{K}_{t,k}(A, r_0)$$

The enlarged Krylov subspace CG methods minimize the 'famous' function $f(x)$ defined in section 2.3, over $x_0 + \mathscr{K}_{t,k}(A, r_0)$.

We now present theorems concerning this topic.

Theorem 3.8. *If* $r_k \perp \mathscr{K}_{t,k}(A, r_0)$ *then* $f(x_k) = min\{f(x), x \in x_0 + \mathscr{K}_{t,k}(A, r_0)\}$

Proof. refer to [6] □

Another theorem of great use:

Theorem 3.9. x_k *minimizes* f *over* $x_0 + \mathscr{K}_{t,k}$ *if and only if it minimizes* $\|x^\star - x\|_A$ *over* $x_0 + \mathscr{K}_{t,k}$ *where* x^* *is the exact solution for* $Ax = b$.

Proof. Define $\phi(x) = \|x^\star - x\|_A = (x^\star - x)^T A(x^\star - x) = (x^\star)^T Ax\star - 2(x^\star)^T Ax + x^T Ax = b^T x^\star - 2b^T x + x^T Ax = b^T x^\star + 2f(x)$ □

The minimum is achieved when $\phi'(x) = 0$, i.e when $f'(x) = 0$.

Convergence:

We know that the classical CG converges in l iterations where $l \leq n$ with $A \in \mathbb{R}^{n \times n}$.

Using the above theorems we can prove that the enlarged CG converges in less than or equal iterations as compared to the classical CG.

$$\begin{aligned}
&\|\bar{e}_k\|_A \\
&= \|x^\star - x_k\|_A \\
&= \min\{\|x^\star - x\|_A, x \in x_0 + \mathscr{K}_{t,k}\} \\
&\leq \min\{\|x^\star - x\|_A, x \in x_0 + \kappa_k\} (\kappa_k \subset \mathscr{K}_{t,k}) \\
&= \|e_k\|_A
\end{aligned}$$

3.2.1 Multiple Search Direction with Orthogonalization Conjugate Gradient (MSDO-CG) Method

I want to start introducing this method by stating the following fact: if there is only one subdomain, then MSDO-CG reduces to CG.

The MSD-CG method introduced by Gu et al. [3] has the same structure as the classical CG method where first t new search directions are defined on the t subdomains, then the t step lengths are obtained by solving a $t \times t$ system, and finally the solution and the residual are updated. However it is not a projection method because the search directions aren't necessarily A-orthonormal, thus MSDO-CG was introduced to solve this matter.

The way MSD-CG works is that at each iteration k, we choose a search direction p_i^k associated with the i-th subdomain. We call $p_k = \sum_{i=1}^{t} p_i^k$ the global search direction. Now, p_i^k can be viewed as projecting the global search direction to the i-th subdomain.
Defining N_i to be the number of elements in the i-th subdomain, then p_i^k, being an $n \times 1$ vector, can at most have N_i non zero elements.

Define P_k the matrix with the search directions at the k-th iterations as its column vectors, i.e $P_k = [p_1^k, ..., p_t^k]$. We set $p_i^1(\delta_i) = r_0(\delta_i)$ for $i = 1, 2, .., t$ and zero otherwise. The updated search directions are $p_i^k = T_i(r_{k-1}) + \beta_i^k p_i^{k-1}$ and we have

$$x_k = x_{k-1} + P_k \alpha_k$$

and

$$r_k = r_{k-1} - AP_k \alpha_k$$

with $\alpha_k = (P_k^T A P_k)^{-1} P_k^T r_{k-1}$ and $\beta_k = [\beta_1^k, .., \beta_t^k] = (P_{k-1}^t A P_{k-1})^{-1} P_{k-1}^T A r_{k-1}$ where α_k is chosen so that it minimizes $f(x_{k-1} + P_k \alpha_k)$ and β so that the global search direction p_k is A-orthogonal to the previous domain search direction p_i^{k-1} i.e $p_k^T A p_i^{k-1} = 0, \ \forall i = 1, 2, .., t$.

However the P_k's aren't A-orthogonal implying that this is not a projection method satisfying the famous Petrov-Galerkin condition $r_k \perp \varkappa_{t,k}(A, r_0)$.

The way the A-orthogonalization of MSDO-CG search directions works is that at each iteration k the block vector P_k is A-orthonormalized against all previous block vectors $P_i, i = 1, .., k-1$ by using an A-orthonormalization process such as the classical Gram-Schmidt or the modified Gram-Schmidt. Then the column vectors are A-orthonormalized against each other using many methods such as CGS, A-CholQR (refer to [2]).
We are now interested in presenting an enlarged CG projection method that seeks a basis' construction for $\varkappa_{t,k}$

3.2.2 SRE-CG (Short Recurrence Enlarged CG) Versions

For this class of methods, we present three versions that differ in the way the basis is constructed. However, the general derivations are the same.
The update x_k is as follows:

$$x_k = x_{k-1} + Q_k \alpha_k$$

where Q_k is the $n \times tk$ matrix containing the basis vectors of $\varkappa_{t,k}$, α_k is the step size vector of length tk. Then $r_k = r_{k-1} - AQ_k\alpha_k$.
As previously, α_k is found by setting the derivative of the function g defined in the proof of 2.11 to zero so that $\alpha_k = (Q_k^t A Q_k)^{-1}(Q_k^t r_{k-1})$ minimizes the function $f(x)$. Thus the orthogonality condition $r_k \perp \varkappa_{t,k}$ is satisfied and the residuals make up an orthogonal set.
This method can be thought of as undergoing two phases: building the candidate A-orthonormal basis and updating the solution.
We start with the basis vectors of $\varkappa_{t,k}(A, r_0) = span\{T(r_0), AT(r_0), ..., A^{k-1}T(r_0)\}$, we A-orthonormalize these basis vectors so we get that α_k defined above reduces to only the term multiplied by r_{k-1}.

In other words, $Q_k^T A Q_k = I$, implying that $\alpha_k = Q_k^T r_{k-1}$ and by the orthogonality condition, as the residuals are orthogonal to the enlarged Krylov subspace we get that $Q_{k-1}^T r_{k-1} = 0$.

Now splitting Q_k into the previous matrix of basis vectors built along with the t new basis vector at the k-th step we get:

$$Q_k = [Q_{k-1}, W_k]$$

where W_k is the set of t newly computed vectors. Then α_k becomes:

$$\alpha_k = Q_k^T r_{k-1} = [Q_{k-1}, W_k] r_{k-1} = [0_{t(k-1)}, W_k^T r_{k-1}] = [0_{t(k-1)}, \tilde{\alpha}_k]$$

Thus, $x_k = x_{k-1} + W_k \tilde{\alpha}_k$. Similarly, $r_k = r_{k-1} - A W_k \tilde{\alpha}_k$.

Let's now delve deeper into how these basis vectors are A-orthonormalized against each other. At each step we compute $W_k = AW_{k-1}$, set $Q_{k-1} = [W_1 W_2 .. W_{k-1}]$, A-orthonormalize W_k against Q_{k-1} as follows:

$$W_k = AW_{k-1} - Q_{k-1} Q_{k-1}^t A(AW_{k-1}) \tag{3.1}$$

$$= AW_{k-1} - \sum_{i=1}^{k-1} W_i W_i^t A(AW_{k-1}) \tag{3.2}$$

$$= AW_{k-1} - W_{k-1} W_{k-1}^t A(AW_{k-1}) - W_{k-2} W_{k-2}^t A(AW_{k-2}) \tag{3.3}$$

given the fact that $(AW_i)^T A w_{k-1} = 0, \forall i < k-2$. Notice that this requires storing the last $3t$ computed vectors, x_{k-1} and r_{k-1}.

This gives rise to the first version: SRE-CG which will be briefly discussed next.

As shown in Algorithm 4 [4], we set the first basis matrix W_1=$T(r_0)$. We then A-orthonormalize its column vectors (using A-CholQR or Pre-CholQR for example).

Then we starting updating: $W_k = AW_{k-1}$. Here we have to A-othoronormalize W_k

against W_{k-1} and W_{k-2} using equation 3.3. Right after that, we A-orthonormalize the vectors of W_k against each other (using classical Gram-Schmidt for example).
We update x_k and r_k and repeat until convergence.
The second version SRE-CG2 is very similar to the previous one except here we are A-orthonormalizing W_k against all previous basis vectors Q_{k-1}, as shown in Algorithm 5 [5]. Notice that to do so we need to store the tk basis vectors, x_{k-1} and r_{k-1} to define x_k and r_k.

Now, in case there is not enough memory to store all the tk basis vectors, we use a truncated version where we choose the last trunc$\times$t number of basis vectors and A-orthonormalize against it.

Convergence: As shown in [2] all the SRE-CG versions converge faster than CG in number of iterations. Very similar results were obtained for MSDO-CG except for Poisson 2D with two partitions.

3.2.3 Preconditioning:

The way the enlarged Krylov conjugate gradient methods are preconditioned is either by using left, right or mixed preconditioning discussed in section 2.6. However, because the matrix A in conjugate gradient methods is spd (symmetric positive definite), the preconditioning matrix M should be spd as well.
It is not easy to find a matrix M such that $M^{-1}A$ or AM^{-1} is spd. Assuming we have the Cholesky factorization of $M = L_M L_M^T$, the matrix $L_M^{-1} A L_M^{-T}$ is spd for an spd matrix A.
So now substituting A by $\hat{A} = L_M^{-1} A L_M^{-T}$, the enlarged Krylov subspace corresponding to the system $\hat{A}y = L_M^{-1} b$, with $y = L^T x$, becomes:

$$\varkappa_{t,k}(\hat{A}, r_0) \tag{3.4}$$

$$= span\{T(r_0), \hat{A}T(r_0), ..., (\hat{A}^{k-1}T(r_0)\} \tag{3.5}$$

with $r_0 = L_M^{-1}(b - AL_M^{-T}y_0)$, b the $n \times 1$ vector, $y_0 = L^T x_0$, x_0 the initial guess.

Thus, the subspace condition for the preconditioned enlarged CG methods becomes: $y_k \in y_0 + \mathscr{K}_{t,k}(\hat{A}, r_0)$, and the Petrov-Galerkin condition becomes: $r_k \perp \mathscr{K}_{t,k}(\hat{A}, r_0)$ and so everything follows for the new system $\hat{A}y = L_M^{-1}b$.

In case of preconditioned SRE-CG2 and SRE-CG, the A-orthonormalization of basis vectors is replaced by $\hat{A}$ orthonormalization as discussed in [6].

Another version that avoids the use of $\hat{A}$- orthonormalization is presented in [7].

Algorithm 4 SRE-CG

Input: A: an $n \times n$ SPD matrix defined ; b: $n \times 1$ right hand side vector; T: partitioning operator defined above; x_0 initial guess; ϵ stopping tolerance; k_{max} the maximum number of iterations allowed.

Output: x_k approximate solution of the system $Ax = b$.

$r_0 = b - Ax_0, \rho_0 = \|r_0\|_2^2, k = 1$

while $\sqrt{\rho_{k-1}} > \epsilon\|b\|_2$ and $k < k_{max}$ **do**

  if k==1 **then**

    Let $W_1 = T(r_0)$, and A-orthonormalize its vectors

  else

    Let $W_k = AW_{k-1}$;

    A-orthonormalize the vectors of W_k against vectors of W_{k-1} and W_{k-2} if $k > 2$

    A-orthonormalize the vectors of W_k

  end if

  $\tilde{\alpha}_k = W_k^t r_{k-1}$;

  $x_k = x_{k-1} + W_k\tilde{\alpha}_k$

  $r_k = r_{k-1} - AW_k\tilde{\alpha}_k$

  $\rho_k = \|r_k\|_2^2$

  $k = k + 1$;

end while

Algorithm 5 SRE-CG2

Input: A: an $n \times n$ SPD matrix defined ; b: $n \times 1$right hand side vector; T: partitioning operator defined above; x_0 initial guess; ϵ stopping tolerance; k_{max} the maximum number of iterations allowed

Output: x_k approximate solution of the system $Ax = b$

$r_0 = b - Ax_0, \rho_0 = \|r_0\|_2^2, k = 1$

while $\sqrt{\rho_{k-1}} > \epsilon\|b\|_2$ and $k < k_{max}$ **do**

  if k==1 **then**

    Let $W_1 = T(r_0)$, and A-orthonormalize its vectors, and let $Q = W_1$

  else

    Let $W_k = AW_{k-1}$;

    A-orthonormalize the vectors of W_k against Q

    A-orthonormalize the vectors of W_k and let $Q = [Q, W_k]$

  end if

  $\tilde{\alpha}_k = W_k^t r_{k-1}$;

  $x_k = x_{k-1} + W_k\tilde{\alpha}_k$

  $r_k = r_{k-1} - AW_k\tilde{\alpha}_k$

  $\rho_k = \|r_k\|_2^2$

  $k = k + 1$;

end while

Chapter 4

Flexible SRE-CG2

In this chapter, we will introduce a flexible version of SRE-CG2 and test it.

4.1 SRE-CG2 Variants

As previously mentioned in chapter 3, SRE-CG2 methods solve that system $Ax = b$ by approximating its solution x_k at the k-th step by the following iterate:

$$x_k = x_{k-1} + Q_k \alpha_k$$

At each iteration, x_k is such that it minimizes the function $f(x)$ introduced in 2.2, over $x_0 + \varkappa_{t,k}(A, r_0)$.

The way the three methods differ is the number of previous basis vectors to which the current t vectors are A-orthonormalized against.

The first is SRE-CG2 which is based on A-orthonormalizing the t basis vectors at the k-th iteration against all the previous vectors and so this requires storing tk basis vectors.

Second is SRE-CG2(trunc): the t basis vectors are A-orthonormalized against the basis vectors computed in the last 'trunc' iterations and thus storing at most (trunc+1)$\times t$ basis vectors.

Third is SRE-CG that A-orthonormalizes the t basis vectors at the current iteration against the previous 2t vectors,which is basically nothing but SRE-CG2(2), i.e trunc=2, meaning against the vectors obtained just before two steps from our current step,thus needing to store 3t basis vectors.
However, the problem with the last two versions is that finding the solution requires more iterations as compared to SRE-CG2.

Thus SRE-CG2 is very costly memory-wise however faster than SRE-CG and SRE-CG2(trunc) as will be shown in table 4.1. The goal of this thesis is to look into other variants of SRE-CG2 that would reduce memory requirements without affecting the number of iterations to convergence and if possible reduce runtime.

The idea of these variants is based on the observation that the norm of the residual stagnates as shown in figure 4.1 for partitions $t = 32, 64$ and 128. And so one can reduce the number basis of vectors produced into half. Thus, instead of producing ' number of partitions' basis vectors per iteration, after some tolerance is reached, we compute half this number. We chose to switch when the relative residual norm $\frac{|\|r_{k+1}\|_2 - \|r_k\|_2|}{\|r_0\|_2}$ becomes smaller than one of the following switch tolerances: $10^{-3}, 10^{-5}$ and 10^{-7}. This modification is detailed in Algorithm 6.

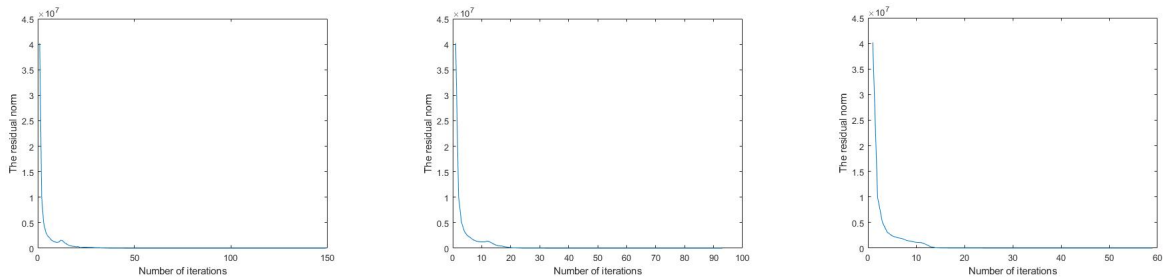

Figure 4.1: Norm of residual vector in SRE-CG2 for partitions: 32, 64 and 128 in Ani3D matrix.

Here are some notations that will be used in the algorithm of flexible SRE-CG2.

T^t is the operator that projects the vector over t subdomains δ_i where $\delta = \cup_{i=1}^{t} \delta_i$, whereas $T^{t/2}$ is the operator that projects the vector over the $t/2$ subdomains $\tilde{\delta}_i$ where $\delta = \cup_{i=1}^{t/2} \tilde{\delta}_i$, and $\tilde{\delta}_i = \delta_{2i-1} \cup \delta_{2i}$ for $i = 1, 2, ..., t/2$.

$$T^t(v) = \begin{pmatrix} * & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ * & 0 & 0 & 0 \\ 0 & * & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ 0 & * & 0 & 0 \\ \vdots & 0 & * & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & * & 0 \\ 0 & 0 & 0 & * \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & * \end{pmatrix}_{n \times t} \qquad T^{\frac{t}{2}}(v) = \begin{pmatrix} * & 0 \\ \vdots & \vdots \\ * & 0 \\ * & 0 \\ \vdots & \vdots \\ * & 0 \\ 0 & * \\ \vdots & \vdots \\ 0 & * \\ 0 & * \\ \vdots & \vdots \\ 0 & * \end{pmatrix}_{n \times \frac{t}{2}}$$

The following modification in the algorithm yields a reduction in the number of partitions and thus the vectors produced and stored per iteration after the switch. Thus, if the switch is occuring at the i-th iteration, flexible SRE-CG2 seeks the approximate solution at the k-th iteration, $k > i$ in

$$x_0 + \varkappa_{t,i}(A, r_0) + \varkappa_{\frac{t}{2},k}(A, r_i)$$

where: $\varkappa_{\frac{t}{2},k}(A, r_i) = span\{T^{\frac{t}{2}}(r_i), .., A^{k-1}T^{\frac{t}{2}}(r_i)\}$, $\varkappa_{t,i}(A, r_0) = \{T^t(r_0), .., T^t(r_0)A^{i-1}\}$

Algorithm 6 flexible SRE-CG2

Input: A: an $n \times n$ SPD matrix defined ; b: $n \times 1$ right hand side vector; $x_0$initial guess;ϵ stopping tolerance; k_{max} the maximum number of iterations allowed, switchTol.

Output: x_k approximate solution of the system $Ax = b$.

counter=0; tol1=1;

$r_0 = b - Ax_0, \rho_0 = \|r_0\|_2^2, k = 1;$

while $\sqrt{\rho_{k-1}} > \epsilon\|b\|_2$ and $k < k_{max}$ **do**

  if k==1 **then**

    Let $W_1 = T^t(r_0)$, and A-orthonormalize its vectors, and let $Q = W_1$

  else

    if (tol1< switchTol) and (counter==0) **then**

      $W_k = T^{t/2}(r_{k-1})$;

      counter=counter+1;

    else

      Let $W_k = AW_{k-1}$;

    end if

    A-orthonormalize the vectors of W_k against Q;

    A-orthonormalize the vectors of W_k and let $Q = [Q, W_k]$;

  end if

  $\tilde{\alpha}_k = W_k^t r_{k-1}$; $x_k = x_{k-1} + W_k\tilde{\alpha}_k$;

  $r_k = r_{k-1} - AW_k\tilde{\alpha}_k$; $\rho_k = \|r_k\|_2^2$;

  $k = k + 1$;

  tol1=$\dfrac{\rho_{k+1} - \rho_k}{\rho_0}$;

end while

4.2 Testing

To further investigate the consequences, we have tested this modification on large sparse matrices that were partitioned according to k-way partitioning [5], where the testing was performed sequentially and not in parallel. The matrices referred to as NH2D, Sky2D, Sky3D, and Ani3D, arise from boundary value problems of the convection diffusion equations (for a detailed description refer to [2]). The Algorithm was implemented and tested in MATLAB R2020b on a PC with the following specifications: The Operating system used is Windows 10 Pro,Version 20H2, Installed RAM is 8.00 GB and processor Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz 2.00 GHz.

We test the methods SRE-CG2, SRE-CG2(20), SRE-CG and flexible SRE-CG2 where the matrices were partitioned into t= 32, 64 and 128 partitions and stopping tolerance 10^{-8}. The initial guess x_0 is chosen to be 0 and b is chosen A times a random vector.

		CG	SRE-CG		SRE-CG2(20)		SRE-CG2	
	t	k	k	time	k	time	k	time
NH 2D	32		82	3.5337	82	10.7903	82	11.0413
	64	259	60	5.969	60	18.2912	60	16.4959
	128		43	9.219	43	31.3037	43	28.4432
Sky 2D	32		993	42.737	860	142.992	126	23.0239
	64	5934	497	48.03	386	145.2916	75	26.5711
	128		235	54.284	142	131.3585	47	36.3737
Sky 3D	32		282	9.7987	222	26.7138	69	6.4683
	64	903	155	11.412	102	26.8459	42	6.9513
	128		80	13.679	41	23.6833	28	11.5984
Ani 3D	32		1620	59.284	1226	155.0953	148	24.9361
	64	4183	927	68.685	485	161.0091	92	30.1556
	128		380	67.016	163	156.4313	58	43.7699

Table 4.1: Comparison of the number of iteration k and time needed till convergence in the SRE-CG2 versions for matrices NH2D, Sky2D, Sky3D and Ani3D with number of partitions $t = 32, 64$ and 128.

It is clear from table 4.1 that SRE-CG2 is the most effective between SRE-CG and SRE-CG2(20) as it required less iterations and less time to converge.

				flexible SRE-CG2								
		SRE-CG2		switchTol=10^{-3}			switchTol=10^{-5}			switchTol=10^{-7}		
	t	k	time	k	switch	time	k	switch	time	k	switch	time
NH 2D	32	82	11.0413	109	14	8.3972	85	61	9.9475	82	75	9.9395
	64	60	16.4959	79	13	13.207	61	47	14.1285	60	57	15.6853
	128	43	28.4432	52	15	19.9319	43	35	24.1147	43	41	27.2315
Sky 2D	32	126	23.0239	207	14	24.1259	191	32	28.9739	126	116	23.6397
	64	75	26.5711	116	12	24.7772	91	41	23.1654	75	68	24.7205
	128	47	36.3737	67	9	26.6768	53	29	28.8202	47	44	37.201
Sky 3D	32	69	6.4683	108	13	6.5514	73	51	5.9714	69	66	6.337
	64	42	6.9513	60	12	6.2409	43	33	6.4983	42	41	7.2134
	128	28	11.5984	36	9	8.2446	29	21	9.0356	28	27	10.4188
Ani3D	32	148	24.9361	231	24	27.6881	207	49	25.4582	151	126	25.4697
	64	92	30.1556	138	11	23.8468	123	28	24.9233	94	78	27.9326
	128	58	43.7699	79	15	29.7068	66	32	33.4797	60	50	40.4119

Table 4.2: Comparison of the number of iteration k and time needed till convergence in the original SRE-CG2 and flexible SRE-CG2 version for matrices NH2D, Sky2D, Sky3D and Ani3D with number of partitions $t = 32, 64$ and 128 and three switchTol $10^{-3}, 10^{-5}$ and 10^{-7}. The switch iteration (switch) is reported for flexible SRE-CG2.

As for table 4.2, it is first important to note that the switch iteration corresponds to the iteration when the relative residual becomes less than the switchTol.

We can see that when switching early on for switchTol=10^{-3} the number of iterations increased for the matrix NH2D compared to SRE-CG2, flexible SRE-CG2 (switchTol=10^{-5}) and SRE-CG2 (switchTol=10^{-7}). However, for the matrices Sky2D,Sky3D and Ani3D it increased compared to the original SRE-CG2, but was still less than SRE-CG and SRE-CG2 (20). As for the convergence time, in all matrices and all partitions, except Ani3D (32), the runtime of flexible SRE-CG2 (switchTol=10^{-3}) was less than that of SRE-CG2.

When switching for swichTol=10^{-5}, for all matrices and all number of partitions the

number of iterations compared to original SRE-CG2 increased a bit but was still less than that of SRE-CG and SRE-CG(20). The time for convergence was less than the time of SRE-CG2 except for Sky2D(32) and Ani3D(32).

As for SwitchTol=10^{-7}, the number of iterations were exactly the same for NH2D,Sky2D and SKY3D as SRE-CG2, this is because we were so close to the stopping tolerance 10^{-8} and so the needed enlarged Krylov subspace was built already. In ANI3D the number of iterations increased compared to the original SRE-CG2 but was still small compared to both SRE-CG and SRE-CG(20). As for the time it took, it was less than the time taken by SRE-CG and SRE-CG(20) and very close to the time taken by SRE-CG2.

The increase in the number of iterations in flexible SRE-CG2 for the three switchTol's was expected as the dimension of the enlarged Krylov subspace is smaller compared to the original Krylov subspace, this is due to building half the number of basis vectors after the switch. Thus finding the solution will require more iterations.The decrease in time isn't a surprise either and is due to the reduction of the number of the basis vectors built.

For all tested matrices and all partitions, flexible SRE-CG2 with switchTol=10^{-7} required less iterations compared to switchTol=10^{-3} and switchTol=10^{-5} because it is the last to switch and so 'enough' basis vectors are built. However, it took more time as more basis vectors were computed.
Based on the results presented in table 4.2, the best switchTol would be 10^{-5}. Although it needed more iterations as compared to flexible SRE-CG2 with switchTol=10^{-7}, but taking into consideration the three variables: iterations till convergence, runtime and memory storage, it is better than switchTol=10^{-3} and 10^{-7}.

Chapter 5

Conclusion

In this thesis, a flexible version of SRE-CG2 method was introduced. This version has shown its effectiveness in both number of iterations and time compared to SRE-CG and SRE-CG(20). As compared to SRE-CG2 , the number of iterations in flexible SRE-CG2 was still acceptable as it didn't increase much. Moreover, the time was reduced in most cases which is promising.

Future work would be to test the same strategy on MSDO-CG.

Bibliography

[1] M. Benzi, (2002). Preconditioning Techniques for Large Linear Systems: A Survey. *Journal of Computational Physics, 182, 418477.*

[2] L. Grigori, S. Moufawad, and F. Nataf. Enlarged Krylov Subspace Conjugate Gradient Methods for Reducing Communication. *SIAM Journal on Matrix Analysis and Applications (SIMAX)*, 37(2):744-773, 2016

[3] T. Gu, X. Liu, Z. Mo, and X. Chi. Multiple search direction conjugate gradient method I: methods and their propositions. *International Journal of Computer Mathematics*, 81(9):1133-1143, 2004.

[4] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49:409–436, 1952.

[5] G. Karypis and V. Kumar, Metis 4.0: Unstructured graph partitioning and sparse matrix ordering system, Technical Report, Department of Computer Science, University of Minnesota, 1998.

[6] S. Moufawad. *Enlarged Krylov Subspace Methods and Preconditioners for Avoiding Communication*. PhD thesis,Université Pierre et Marie Curie-Paris VI, 2014.

[7] S. Moufawad. S-step Enlarged Krylov Subspace Conjugate Gradient Methods, *SIAM Journal on Scientific Computing*, 42(1), A187-A219, January 2020

[8] Y. Saad, (2003). *Iterative Methods for Sparse Linear Systems*. 2nd edition. Philadelphia, PA: SIAM.

[9] Y. Saad and M. H. Schultz. Gmres: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, July 1986.

[10] https://people.eecs.berkeley.edu/ demmel/cs267/lecture18/lecture18.html