

# Iterative Methods

Jayadev Naram

October 26, 2019

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Iterative Methods . . . . .	2
1.2	Polynomial Methods . . . . .	2
<b>2</b>	<b>General Projection Methods</b>	<b>3</b>
<b>3</b>	<b>One-Dimensional Projection Methods</b>	<b>6</b>
3.1	Steepest Descent . . . . .	6
<b>4</b>	<b>Krylov Subspace Methods</b>	<b>8</b>
<b>5</b>	<b>Arnoldi's Method for Linear Systems (FOM)</b>	<b>9</b>
5.1	Variation 1: Restarted FOM . . . . .	10
5.2	Variation 1: IOM and DIOM . . . . .	10
<b>6</b>	<b>Symmetric Lanczos Algorithm</b>	<b>13</b>
<b>7</b>	<b>Conjugate Gradient</b>	<b>13</b>
<b>8</b>	<b>Convergence Analysis</b>	<b>16</b>
8.1	Real Chebyshev Polynomials . . . . .	16
8.2	Convergence of CG Algorithm . . . . .	17
<b>9</b>	<b>Deflated-CG Algorithm</b>	<b>18</b>
9.1	Deflated Lanczos algorithm . . . . .	18
9.2	Deflated-CG algorithm . . . . .	19

# 1 Introduction

We want to solve  $Ax = b$ , where  $A \in \mathbb{C}^{n \times n}$  is a large non-singular matrix. To solve such a large system using Gaussian elimination is very costly and hence we turn to iterative methods for approximate solutions.

## 1.1 Iterative Methods

An iterative method is defined by an initial guess  $x_0$  to the exact solution  $\tilde{x} = A^{-1}b$ , followed by a sequence of further approximations  $\{x_i\}_{i>0}$ . The error is given by  $e_i = \tilde{x} - x_i$  and residual by  $r_i = b - Ax_i = Ae_i$ . For a given choice of  $x_0$ , an iterative method converges if  $\|\tilde{x} - x_m\| \rightarrow 0$  as  $m \rightarrow \infty$ . An iterative method experiences finite termination if for some  $d$  we have  $x_m = \tilde{x} \forall m \geq d$ .

## 1.2 Polynomial Methods

A polynomial method is an iterative method satisfying,  $e_m = P_m(A)e_0$ , for some polynomial  $P_m(z) = 1 - zQ_{m-1}(z)$  of degree no greater than  $m$  satisfy  $P_m(0) = 1$ .

## 2 General Projection Methods

Let  $A \in \mathbb{R}^{n \times n}$  and  $\mathcal{K}$  and  $\mathcal{L}$  be two  $m$ -dimensional subspaces of  $\mathbb{R}^n$ . A projection technique onto the subspace  $\mathcal{K}$  and orthogonal to  $\mathcal{L}$  with an initial guess  $x_0$  is a process which finds an approximate solution  $\tilde{x}$  by imposing the conditions that  $\tilde{x}$  belong to  $x_0 + \mathcal{K}$  and that the new residual vector be orthogonal to  $\mathcal{L}$ , i.e,

$$\text{find } \tilde{x} \in x_0 + \mathcal{K}, \text{ such that } b - A\tilde{x} \perp \mathcal{L}.$$

$$\tilde{x} = x_0 + \delta, \delta \in \mathcal{K}$$

$$(r_0 - A\delta, w) = 0, \forall w \in \mathcal{L}, \text{ where } r_0 = b - Ax_0.$$

Let  $V = [v_1, \dots, v_m]_{n \times m}$  and  $W = [w_1, \dots, w_m]_{n \times m}$  whose column-vectors form a basis of  $\mathcal{K}$  and  $\mathcal{L}$ , respectively. Then approximate solution can be written as:

$$\tilde{x} = x_0 + Vy,$$

where  $y$  can found from the orthogonality constraint:

$$W^T AVy = W^T r_0.$$

If  $W^T AV$  is non-singular, then  $\tilde{x} = x_0 + V(W^T AV)^{-1}W^T r_0$ .

---

### Algorithm 1 Prototype Projection Method

---

- 1: **repeat**
  - 2:   Select a pair of subspaces  $\mathcal{K}$  and  $\mathcal{L}$
  - 3:   Choose basis  $V=[v_1, \dots, v_m]$ ,  $W=[w_1, \dots, w_m]$  for  $\mathcal{K}$  and  $\mathcal{L}$
  - 4:    $r \leftarrow b - Ax$
  - 5:    $y \leftarrow (W^T AV)^{-1}W^T r$
  - 6:    $x \leftarrow x + Vy$
  - 7: **until** Convergence
- 

Non-singularity of  $A$  is not sufficient condition for non-singularity of  $W^T AV$ .

**Proposition 1.** *Let  $A$ ,  $\mathcal{L}$  and  $\mathcal{K}$  satisfy either one of the two following conditions:*

- i.  $A$  is SPD and  $\mathcal{L} = \mathcal{K}$ , or
- ii.  $A$  is non-singular and  $\mathcal{L} = A\mathcal{K}$ .

*Then  $B = W^T AV$  is non-singular for any bases  $V$  and  $W$  of  $\mathcal{K}$  and  $\mathcal{L}$ .*

*Proof.* Consider case(i). Since  $\mathcal{L} = \mathcal{K}$ , then  $W = VG$ , where  $G$  is a non-singular  $m \times m$  matrix. Then  $B = W^T AV = G^T V^T AV$ . Since  $A$  is SPD, so is  $V^T AV$  and since  $G$  is non-singular,  $B$  is non-singular.

Now, consider case(ii). Since  $\mathcal{L} = A\mathcal{K}$ , then  $W = AVG$ , where  $G$  is a non-singular  $m \times m$  matrix. Then  $B = W^T AV = G^T (AV)^T AV$ . Since  $A$  is non-singular, then  $(AV)_{n \times m}$  full rank matrix and so is  $(AV)^T AV$  and therefore,  $B$  is non-singular.  $\square$

**Theorem 1.** Assume that  $A$  is SPD and  $\mathcal{L} = \mathcal{K}$ . Then a vector  $\tilde{x}$  is the result of an (orthogonal) projection method onto  $\mathcal{K}$  with the starting vector  $x_0$  iff it minimizes the  $A$ -norm of the error over  $x_0 + \mathcal{K}$ , i.e., iff

$$\tilde{x} = \arg \min_{x \in x_0 + \mathcal{K}} \|x_* - x\|_A = \arg \min_{x \in x_0 + \mathcal{K}} (A(x_* - x), x_* - x)^{\frac{1}{2}}$$

*Proof.* First we prove that if  $\tilde{x}$  minimizes  $A$ -norm of the error, then it is the result of orthogonal projection method with  $x_0$  onto  $\mathcal{K}$ . Assume columns of  $V$  to be basis vectors of  $\mathcal{K}$ , then the objective function can be written as:

$$\begin{aligned} E(x) &= (A(x_* - x), x_* - x)^{\frac{1}{2}}, \quad (x \in x_0 + \mathcal{K}) \\ \implies E(y) &= (A(x_* - x_0 - Vy), x_* - x_0 - Vy)^{\frac{1}{2}}, \quad (y \in \mathbb{R}^m) \\ \implies E^2(y) &= (A(x_* - x_0 - Vy), x_* - x_0 - Vy), \\ &= (x_* - x_0 - Vy)^T A(x_* - x_0 - Vy), \\ &= c + 2y^T V^T (Ax_0 - Ax_*) + y^T V^T A V y, \\ &= c - 2y^T V^T (b - Ax_0) + y^T V^T A V y = f(y), \\ \frac{\partial f(y)}{\partial y} = 0 &\implies V^T (b - A(x_0 + Vy)) = 0 \\ &\implies V^T (b - A\tilde{x}) = 0 \\ &\implies b - A\tilde{x} \perp \mathcal{K}. \end{aligned}$$

Therefore the residue of vector which minimizes  $A$ -norm of error over  $x_0 + \mathcal{K}$  is orthogonal to  $\mathcal{K}$ , therefore it is the result of orthogonal projection method onto  $\mathcal{K}$  starting with  $x_0$ . Now we prove the converse, i.e., the result of orthogonal projection method onto  $\mathcal{K}$  starting with  $x_0$  minimizes  $A$ -norm of error over  $x_0 + \mathcal{K}$ . We know  $V^T (b - A\tilde{x}) = 0$ , i.e.,  $(x_* - \tilde{x}, v)_A = 0 \forall v \in \mathcal{K}$ .

$$\begin{aligned} \implies \|x_* - x\|_A &= \|x_* - \tilde{x} + \tilde{x} - x\|_A, \quad (\tilde{x}, x \in x_0 + \mathcal{K}) \\ &= \|x_* - \tilde{x}\|_A + \|\tilde{x} - x\|_A, \quad (\text{since } x_* - \tilde{x} \text{ is } A\text{-orthogonal to } \mathcal{K}) \\ \implies \|x_* - \tilde{x}\|_A &\leq \|x_* - x\|_A, \quad \forall x \in x_0 + \mathcal{K}. \end{aligned}$$

Therefore  $\tilde{x}$  minimizes the  $A$ -norm of the error.  $\square$

**Corollary 1.1.** Let  $A$  be an arbitrary square matrix and assume that  $\mathcal{L} = AK$ . Then a vector  $\tilde{x}$  is the result of an (oblique) projection method onto  $\mathcal{K}$  orthogonally to  $\mathcal{L}$  with the starting vector  $x_0$  iff it minimizes the 2-norm of the residual vector  $b - Ax$  over  $x \in x_0 + \mathcal{K}$ , i.e., iff

$$\tilde{x} = \arg \min_{x \in x_0 + \mathcal{K}} \|b - Ax\|_2$$

**Proposition 2.** Let  $\tilde{x}$  be the approximate solution obtained from a projection process onto  $\mathcal{K}$  orthogonally to  $\mathcal{L} = AK$ , and let  $\tilde{r} = b - A\tilde{x}$ . Then,

$$\tilde{r} = (I - P)r_0,$$

where  $P$  denotes the orthogonal projector onto  $\mathcal{K}$ .

*Proof.* Let  $r_0 = b - Ax_0$ , then

$$\begin{aligned}\tilde{r} &= b - A\tilde{x} \\ &= b - A(x_0 + \delta), \quad (\delta \in \mathcal{K}) \\ &= r_0 - A\delta.\end{aligned}$$

By orthogonality condition we have  $\tilde{r} \perp A\mathcal{K}$ , i.e,  $A\delta$  is the projection of  $r_0$  onto  $A\mathcal{K}$ . Therefore, if  $P$  is the orthogonal projector onto  $A\mathcal{K}$ , then

$$Pr_0 = A\delta \implies \tilde{r} = (I - P)r_0$$

It follows from the above that  $\|\tilde{r}\|_2 \leq \|r_0\|_2$ . Therefore, this class of methods can be termed as **Residual Projection Methods**.  $\square$

**Proposition 3.** Let  $\tilde{x}$  be the approximate solution obtained from an orthogonal projection process onto  $\mathcal{K}$ , and let  $\tilde{d} = x_* - \tilde{x}$ . Then,

$$\tilde{d} = (I - P_A)d_0,$$

where  $P_A$  denotes the projector onto  $\mathcal{K}$ , which is orthogonal with respect to  $A$ -inner product.

*Proof.* Let  $d_0 = x_* - x_0$  be the initial error, and let  $\tilde{d} = x_* - \tilde{x}$ , where  $\tilde{x} = x_0 + \delta$  is the approximate solution resulting from the projection step. We know that residual of the approximate solution is orthogonal to  $\mathcal{K}$ , i.e,  $\tilde{r} = A\tilde{d} = A(d_0 - \delta)$ ,  $\tilde{r} \perp \mathcal{K}$ .

$$\begin{aligned}\implies (A(d_0 - \delta), w) &= 0 \quad \forall w \in \mathcal{K} \\ \implies (d_0 - \delta, w)_A &= 0 \quad \forall w \in \mathcal{K}\end{aligned}$$

Therefore, if  $P_A$  is the projector onto  $A\mathcal{K}$ , which is orthogonal with respect to  $A$ -inner product, then  $\delta$  is the  $A$ -orthogonal projection of  $d_0$ , i.e,

$$P_A d_0 = \delta \implies \tilde{d} = (I - P_A)d_0.$$

It follows from the above that  $\|\tilde{d}\|_A \leq \|d_0\|_A$ . Therefore, this class of methods can be termed as **Error Projection Methods**.  $\square$

Define  $\mathcal{P}_{\mathcal{K}}$  to be the orthogonal projector onto  $\mathcal{K}$  and let  $\mathcal{Q}_{\mathcal{K}}^{\mathcal{L}}$  be the (oblique) projector onto  $\mathcal{K}$  and orthogonally to  $\mathcal{L}$ . Then

$$\begin{aligned}\mathcal{P}_{\mathcal{K}}x &\in \mathcal{K} \text{ and } x - \mathcal{P}_{\mathcal{K}}x \perp \mathcal{K}, \\ \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}}x &\in \mathcal{K} \text{ and } x - \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}}x \perp \mathcal{L}\end{aligned}$$

**Theorem 2.** Assume that  $\mathcal{K}$  is invariant under  $A$  and the initial residue, i.e,  $r_0 = b - Ax_0$  belongs to  $\mathcal{K}$ . Then the approximate solution obtained from any (oblique or orthogonal) projection method onto  $\mathcal{K}$  is exact.

*Proof.* An approximate solution  $\tilde{x}$  is defined by

$$\mathcal{Q}_{\mathcal{K}}^{\mathcal{L}}(b - A\tilde{x}) = 0, \text{ where } \tilde{x} = x_0 + \delta, \delta \in \mathcal{K}.$$

$$\implies \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}}(b - Ax_0 - A\delta) = 0$$

$$\implies \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}}r_0 = \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}}A\delta$$

But  $\mathcal{K}$  is invariant under  $A$ , then  $A\delta \in \mathcal{K}$ .

$$\implies r_0 = A\delta, \text{ (since } r_0 \in \mathcal{K} \text{ and } \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}}A\delta = A\delta)$$

$$\implies A\tilde{x} = b$$

□

**Theorem 3 (General Error Bound).** *Let  $\gamma = \|\mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A(I - \mathcal{P}_{\mathcal{K}})\|_2$  and assume that  $b$  is a member of  $\mathcal{K}$  and  $x_0 = 0$ . Then the exact solution  $x_*$  of the problem is such that*

$$\|b - \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A \mathcal{P}_{\mathcal{K}} x_*\|_2 \leq \gamma \|(I - \mathcal{P}_{\mathcal{K}})x_*\|_2.$$

*Proof.* Since  $b \in \mathcal{K}$ ,

$$\begin{aligned} b - \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A \mathcal{P}_{\mathcal{K}} x_* &= \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} b - \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A \mathcal{P}_{\mathcal{K}} x_* \\ &= \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} (b - A \mathcal{P}_{\mathcal{K}} x_*) \\ &= \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A (I - \mathcal{P}_{\mathcal{K}}) x_* \\ &= \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A (I - \mathcal{P}_{\mathcal{K}}) (I - \mathcal{P}_{\mathcal{K}}) x_* \\ \implies \|b - \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A \mathcal{P}_{\mathcal{K}} x_*\|_2 &= \|\mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A (I - \mathcal{P}_{\mathcal{K}}) (I - \mathcal{P}_{\mathcal{K}}) x_*\|_2 \\ &\leq \|\mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A (I - \mathcal{P}_{\mathcal{K}})\|_2 \|(I - \mathcal{P}_{\mathcal{K}})x_*\|_2 \\ \implies \|b - \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A \mathcal{P}_{\mathcal{K}} x_*\|_2 &\leq \gamma \|(I - \mathcal{P}_{\mathcal{K}})x_*\|_2 \end{aligned}$$

□

### 3 One-Dimensional Projection Methods

One-dimensional projection processes are defined when  $\mathcal{K} = \text{span}\{v\}$  and  $\mathcal{L} = \text{span}\{w\}$ . In this case, the new approximation takes the form  $x \leftarrow x + \alpha v$ , where the orthogonality condition  $r - A\delta \perp w$  yields,

$$\alpha = \frac{(r, w)}{(Av, w)}, \text{ where } r = b - Ax_0.$$

#### 3.1 Steepest Descent

The steepest descent algorithm is defined when  $A$  is SPD and  $v = w = r$ .

**Lemma 4 (Kantorovich inequality).** *Let  $B$  be any real SPD matrix and  $\lambda_1, \lambda_n$  its largest and smallest eigenvalues. Then,*

$$\frac{(Bx, x)(B^{-1}x, x)}{(x, x)} \leq \frac{(\lambda_1 + \lambda_n)^2}{4\lambda_1\lambda_n}, \forall x \neq 0$$

*Proof.* It is equivalent to prove the statement for any unit vector  $x$ . Since  $B$  is SPD, it can be diagonalized by similarity transformation with an orthogonal matrix  $Q$ ,  $B = Q^T D Q$ .

$$(Bx, x)(B^{-1}x, x) = (Q^T D Q x, x)(Q^T D^{-1} Q x, x) = (D Q x, Q x)(D^{-1} Q x, Q x).$$

Define  $y = Qx = (y_1, y_2, \dots, y_n)^T$ , and  $\beta_i = y_i^2$ . Then,

$$\lambda \equiv (Dy, y) = \sum_{i=1}^n \beta_i \lambda_i, \quad \sum_{i=1}^n \beta_i = 1$$

$$\psi(y) = (D^{-1}y, y) = \sum_{i=1}^n \beta_i \frac{1}{\lambda_i}.$$

Note that  $\lambda$  is a convex combinations of eigenvalues of  $B$ . Then,

$$(Bx, x)(B^{-1}x, x) = \lambda\psi(y).$$

Noting that  $f(\lambda) = 1/\lambda$  is a convex function for  $x \in \mathbb{R}_{++}$ ,  $\psi(y)$  contains all the convex combinations of  $1/\lambda_i$ s which is bounded above by line passing through  $(\lambda_1, 1/\lambda_1)$  and  $(\lambda_n, 1/\lambda_n)$ , i.e.,

$$\begin{aligned} \psi(y) &\leq \frac{1}{\lambda_1} + \frac{1}{\lambda_n} - \frac{\lambda}{\lambda_1\lambda_n}. \\ \implies (Bx, x)(B^{-1}x, x) &= \lambda\psi(y) \leq \lambda \left( \frac{1}{\lambda_1} + \frac{1}{\lambda_n} - \frac{\lambda}{\lambda_1\lambda_n} \right). \end{aligned}$$

The right-hand side is maximum when  $\lambda = \frac{\lambda_1 + \lambda_n}{2}$  yielding,

$$(Bx, x)(B^{-1}x, x) \leq \frac{(\lambda_1 + \lambda_n)^2}{4\lambda_1\lambda_n}$$

□

---

**Algorithm 2** Steepest Descent Algorithm

---

- 1: Compute  $r = b - Ax$  and  $p = Ar$
  - 2: **repeat**
  - 3:    $\alpha \leftarrow (r, r)/(p, r)$
  - 4:    $x \leftarrow x + \alpha r$
  - 5:    $r \leftarrow r - \alpha p$
  - 6:   Compute  $p = Ar$
  - 7: **until** Convergence
- 

**Theorem 5.** Let  $A$  be a SPD. Then,  $A$ -norms of the error vectors  $d_k = x_* - x_k$  generated by the above algorithm satisfy the following relation:

$$\|d_{k+1}\|_A \leq \left( \frac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n} \right) \|d_k\|_A,$$

and the algorithm converges for any initial guess  $x_0$ .

*Proof.* We know that  $d_{k+1} = x_* - x_{k+1}$ , but  $x_{k+1} = x_k + \alpha_k r_k$ .

$$\implies d_{k+1} = x_* - (x_k + \alpha_k r_k) = d_k - \alpha_k r_k.$$

Now consider,

$$\begin{aligned} \|d_{k+1}\|_A^2 &= (d_{k+1}, d_{k+1})_A = (d_k - \alpha_k r_k, d_k - \alpha_k r_k)_A \\ &= (d_k, d_k)_A - 2\alpha_k (d_k, r_k)_A + \alpha_k^2 (r_k, r_k)_A \\ (d_{k+1}, \alpha_k r_k)_A &= (\alpha_k d_{k+1}, r_k)_A = (\alpha_k (d_k - \alpha_k r_k), r_k)_A \\ &= (\alpha_k d_k, r_k)_A - \alpha_k^2 (r_k, r_k)_A, \text{ where } \alpha_k = \frac{(r_k, r_k)}{(Ar_k, r_k)}, \\ &= (\alpha_k d_k, r_k)_A - \alpha_k^2 (r_k, r_k)_A \\ &= (\alpha_k d_k, r_k)_A - \alpha_k^2 (r_k, r_k)_A = 0 = (r_{k+1}, r_k)_A. \\ \implies (d_{k+1}, \alpha_k r_k)_A &= 0. \end{aligned}$$

$$\begin{aligned}
\implies \|d_{k+1}\|_A^2 &= (d_{k+1}, d_k)_A \\
&= (d_{k+1}, Ad_k) \quad (\text{since } A \text{ is SPD}), \\
&= (d_k - \alpha_k r_k, r_k) \\
&= (A^{-1}r_k, r_k) - \alpha_k (r_k, r_k) \\
\text{But, } \|d_k\|_A^2 &= (Ad_k, d_k) = (r_k, d_k) = (A^{-1}r_k, r_k), \\
\implies \|d_{k+1}\|_A^2 &= (A^{-1}r_k, r_k) \left(1 - \frac{(r_k, r_k)^2}{(Ar_k, r_k)(A^{-1}r_k, r_k)}\right),
\end{aligned}$$

From Kantorovich inequality,

$$\begin{aligned}
&\leq \|d_k\|_A^2 \left(1 - \frac{4\lambda_1\lambda_n}{(\lambda_1 + \lambda_n)^2}\right), \\
\implies \|d_{k+1}\|_A &\leq \left(\frac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n}\right) \|d_k\|_A.
\end{aligned}$$

□

## 4 Krylov Subspace Methods

We define Krylov Subspace to be

$$\mathcal{K}_m(A, v) = \text{span}\{v, Av, A^2v, \dots, A^{m-1}v\}.$$

Then,  $x = p(A)v$ ,  $\forall x \in \mathcal{K}_m$ , where  $\deg(p) < m$ .

**Definition 1** (Minimal Polynomial of a vector). *Monic polynomial of least degree such that  $p(A)v = 0$  is called minimal polynomial of  $v$  and degree of such polynomial is called  $\text{grade}(\mu)$ .*

**Theorem 6.** *Let  $\mu$  be the grade of  $v$ . Then  $\mathcal{K}_\mu$  is invariant under  $A$  and  $\mathcal{K}_\mu = \mathcal{K}_m \forall m \geq \mu$ .*

*Proof.* Since, grade of  $v$  is  $\mu$  there exists a polynomial  $p$  of degree  $\mu$ , such that  $p(A)v = 0$ , where  $p(A) = p_0I + p_1A + \dots + p_{\mu-1}A^{\mu-1} + A^\mu$ .

$$\implies A^\mu v = -(p_0I + p_1A + \dots + p_{\mu-1}A^{\mu-1})v \quad (1)$$

But,  $\forall x \in \mathcal{K}_\mu$ ,  $x = q(A)v$ ,  $\deg(q) < \mu$ , i.e.,

$$x = q_0v + q_1Av + \dots + q_{\mu-1}A^{\mu-1}v, \quad \forall x \in \mathcal{K}_\mu,$$

$$\implies Ax = q_0Av + q_1A^2v + \dots + q_{\mu-1}A^\mu v,$$

Case 1:  $q_{\mu-1} = 0$ , then  $Ax \in \mathcal{K}_\mu$ .

Case 2:  $q_{\mu-1} \neq 0$ , then replace  $A^\mu v$  by (1),  $Ax \in \mathcal{K}_\mu$ .

Therefore,  $\mathcal{K}_\mu$  is invariant under  $A$ . Similarly it can be seen that  $\mathcal{K}_\mu = \mathcal{K}_m \forall m \geq \mu$ . □

**Corollary 6.1.**  $\dim(\mathcal{K}_m) = \min\{m, \text{grade}(v)\}$ .



## 5 Arnoldi's Method for Linear Systems (FOM)

Arnoldi's procedure is an algorithm for building an orthogonal basis of the Krylov subspace  $\mathcal{K}_m$ .

---

**Algorithm 3** Arnoldi-Modified Gram-Schmidt

---

```

1: Choose a vector  $v_1$  of norm 1
2: for  $j = 1, 2, \dots, m$  do
3:   Compute  $w_j = Av_j$ 
4:   for  $i = 1, 2, \dots, j$  do
5:      $h_{ij} = (w_j, v_i)$ 
6:      $w_j = w_j - h_{ij}v_i$ 
7:   EndDo
8:    $h_{j+1,j} = \|w_j\|_2$ 
9:   If  $h_{j+1,j} = 0$  Stop; found an invariant subspace  $[v_1, \dots, v_j]$ 
10:   $v_{j+1} = w_j/h_{j+1,j}$ 
11: EndDo

```

---

**Proposition 4.** Denote by  $V_m = [v_1, v_2, \dots, v_m]_{n \times m}$  and  $\bar{H}_m$ , the  $(m+1) \times m$  Hessenberg matrix whose non-zero entries  $h_{ij}$  are defined by the above algorithm and by  $H_m$  the matrix obtained from  $\bar{H}_m$  by removing the last row. Then,

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

$$V_m^T AV_m = H_m.$$

*Proof.* From lines 6,8 we have,  $w_j = Av_j - h_{ij}v_i$  and  $w_j = v_{j+1}h_{j+1,j}$ .

$$\implies Av_j = \sum_{i=1}^{j+1} h_{ij}v_i \implies AV_m = V_m H_m + w_m e_m^T = V_{m+1} \bar{H}_m.$$

Since  $V_m^T$  is orthogonal, we get  $V_m^T AV_m = H_m$ .

□

Given an initial guess  $x_0$  to the original linear system  $Ax = b$ , we now consider an orthogonal projection method which takes  $\mathcal{L} = \mathcal{K} = \mathcal{K}_m(A, r_0)$ , with

$$\mathcal{K}_m(A, r_0) = \text{span}\{r_0, Ar_0, A^2 r_0, \dots, A^{m-1} r_0\},$$

in which  $r_0 = b - Ax_0$ . This method seeks an approximate solution  $x_m$  from the affine subspace  $x_0 + \mathcal{K}_m$  of dimension  $m$  by imposing the following orthogonality constraint:

$$b - Ax_m \perp \mathcal{K}_m.$$

If  $v_1 = r_0/\|r_0\|_2$  in Arnoldi's method, and we set  $\beta = \|r_0\|_2$ , then

$$V_m^T AV_m = H_m, \quad V_m^T r_0 = V_m^T (\beta v_1) = \beta e_1.$$

As a result, the approximate solution using the above  $m$ -dimensional subspaces is given by:

$$x_m = x_0 + V_m y_m,$$

where  $y_m$  can be found by imposing orthogonality constraint that

$$V_m^T (b - Ax_m) = 0 \implies y_m = H_m^{-1}(\beta e_1).$$

---

**Algorithm 4** Full Orthogonalization Method (FOM)

---

```
1: Compute  $r_0 = b - Ax_0$ ,  $\beta = \|r_0\|_2$ , and  $v_1 = r_0/\beta$ 
2: Define the  $m \times m$  matrix  $H_m = \{h_{ij}\}_{i,j=1,2,\dots,m}$ ; Set  $H_m = 0$ 
3: for  $j = 1, 2, \dots, m$  do
4:   Compute  $w_j = Av_j$ 
5:   for  $i = 1, 2, \dots, j$  do
6:      $h_{ij} = (w_j, v_i)$ 
7:      $w_j = w_j - h_{ij}v_i$ 
8:   EndDo
9:    $h_{j+1,j} = \|w_j\|_2$ . If  $h_{j+1,j} = 0$  Stop
10:   $v_{j+1} = w_j/h_{j+1,j}$ 
11: EndDo
12: Compute  $y_m = H_m^{-1}\beta e_1$  and  $x_m = x_0 + V_my_m$ 
```

---

**Proposition 5.** *The residual vector of the approximate solution  $x_m$  computed by the FOM Algorithm is such that*

$$b - Ax_m = -h_{m+1,m}e_m^T y_m v_{m+1}$$

and, therefore,

$$\|b - Ax_m\|_2 = h_{m+1,m}|e_m^T y_m|.$$

*Proof.*

$$\begin{aligned} b - Ax_m &= b - Ax_0 - AV_my_m \\ &= r_0 - (V_m H_m + w_m e_m^T) y_m \\ &= r_0 - V_m H_m (H_m^{-1} \beta e_1) - w_m e_m^T y_m \\ &= r_0 - V_m V_m^T r_0 - h_{m+1,m} e_m^T y_m v_{m+1} = -h_{m+1,m} e_m^T y_m v_{m+1}. \end{aligned}$$

□

## 5.1 Variation 1: Restarted FOM

---

**Algorithm 5** Restarted FOM (FOM(m))

---

```
1: Compute  $r_0 = b - Ax_0$ ,  $\beta = \|r_0\|_2$ , and  $v_1 = r_0/\beta$ 
2: Generate  $V_m$  and  $H_m$  using Arnoldi algorithm starting with  $v_1$ .
3: Compute  $y_m = H_m^{-1}\beta e_1$  and  $x_m = x_0 + V_my_m$ . If satisfied then Stop.
4: Set  $x_0 = x_m$  and go to 1.
```

---

## 5.2 Variation 1: IOM and DIOM

A formula can be developed whereby the current approximate solution  $x_m$  can be computed from the previous approximation  $x_{m-1}$  and a small number vectors are updated at each step. This progressive formulation of the solution leads to an algorithm termed as Direct IOM (DIOM).

---

**Algorithm 6** Incomplete Orthogonalization Method (IOM)

---

```

1: Compute  $r_0 = b - Ax_0$ ,  $\beta = \|r_0\|_2$ , and  $v_1 = r_0/\beta$ 
2: Define the  $m \times m$  matrix  $H_m = \{h_{ij}\}_{i,j=1,2,\dots,m}$ ; Set  $H_m = 0$ 
3: for  $j = 1, 2, \dots, m$  do
4:   Compute  $w_j = Av_j$ 
5:   for  $i = \max\{1, j - (k - 1)\}, 2, \dots, j$  do
6:      $h_{ij} = (w_j, v_i)$ 
7:      $w_j = w_j - h_{ij}v_i$ 
8:   EndDo
9:    $h_{j+1,j} = \|w_j\|_2$ . If  $h_{j+1,j} = 0$  Stop
10:   $v_{j+1} = w_j/h_{j+1,j}$ 
11: EndDo
12: Compute  $y_m = H_m^{-1}\beta e_1$  and  $x_m = x_0 + V_my_m$ 

```

---

The Hessenberg matrix obtained from IOM has a band structure with band-width  $k + 1$ , i.e.,

$$\begin{aligned}
H_m &= \begin{pmatrix} h_{11} & h_{12} & h_{13} & & \\ h_{21} & h_{22} & h_{23} & h_{24} & \\ & h_{32} & h_{33} & h_{34} & h_{35} \\ & & h_{43} & h_{44} & h_{45} \\ & & & h_{54} & h_{55} \end{pmatrix} = L_m U_m \\
&= \begin{pmatrix} 1 & & & & \\ l_{21} & 1 & & & \\ & l_{32} & 1 & & \\ & & l_{43} & 1 & \\ & & & l_{54} & 1 \end{pmatrix} \times \begin{pmatrix} u_{11} & u_{12} & u_{13} & & \\ & u_{22} & u_{23} & u_{24} & \\ & & u_{33} & u_{34} & u_{35} \\ & & & u_{44} & u_{45} \\ & & & & u_{55} \end{pmatrix}
\end{aligned}$$

The approximate solution then is given by

$$x_m = x_0 + V_m U_m^{-1} L_m^{-1} (\beta e_1).$$

Define  $P_m \equiv V_m U_m^{-1}$  and  $z_m = L_m^{-1}(\beta e_1)$ , we have  $x_m = x_0 + P_m z_m$ . Because of the structure of  $U_m$ ,  $P_m$  can be updated easily. Indeed, equating the last columns of the matrix relation  $P_m U_m = V_m$  yields,

$$\sum_{i=m-k+1}^m u_{im} p_i = v_m \implies p_m = \frac{1}{u_{mm}} \left( v_m - \sum_{i=m-k+1}^{m-1} u_{im} p_i \right).$$

Therefore,  $p_m$  can be computed using previous  $p_i$ 's and  $v_m$ . In addition, due to the structure of  $L_m$ , we have compute  $z_m$  by,

$$z_m = \begin{bmatrix} z_{m-1} \\ \zeta_m \end{bmatrix}, \text{ where } \zeta_m = -l_{m,m-1} \zeta_{m-1}.$$

Now, the approximate solution is,

$$x_m = x_0 + \begin{bmatrix} P_{m-1} & p_m \end{bmatrix} \begin{bmatrix} z_{m-1} \\ \zeta_m \end{bmatrix} = x_0 + P_{m-1} z_{m-1} + p_m \zeta_m.$$

Noting that  $x_{m-1} = P_{m-1}z_{m-1}$ ,  $x_m$  can be updated as follows:

$$x_m = x_{m-1} + \zeta_m p_m.$$

This gives the following algorithm, called **Incomplete Orthogonalization Method(DIOM)**.

---

**Algorithm 7** Direct Incomplete Orthogonalization Method (DIOM)

---

- 1: Choose  $x_0$  and compute  $r_0 = b - Ax_0$ ,  $\beta = \|r_0\|_2$ , and  $v_1 = r_0/\beta$
  - 2: **for**  $m = 1, 2, \dots$ , until convergence **do**
  - 3:   Compute  $w_m = Av_m$
  - 4:   **for**  $i = \max\{1, m - k + 1\}, 2, \dots, m$  **do**
  - 5:      $h_{im} = (w_m, v_i)$
  - 6:      $w_m = w_m - h_{im}v_i$
  - 7:    $h_{m+1,m} = \|w_m\|_2$ . If  $h_{m+1,m} = 0$  Stop
  - 8:    $v_{m+1} = w_m/h_{m+1,m}$
  - 9:   Update the LU factorization of  $H_m$ , i.e, obtain the last column
  - 10:    $U_m$  using the previous  $k$  pivots. If  $u_{mm} = 0$  Stop.
  - 11:    $\zeta_m = \beta$  if  $m = 1$  else  $-l_{m,m-1}\zeta_{m-1}$
  - 12:    $p_m = u_{mm}^{-1} \left( v_m - \sum_{i=m-k+1}^{m-1} u_{im}p_i \right)$  (for  $i \leq 0$  set  $u_{im}p_i \equiv 0$ )
  - 13:    $x_m = x_{m-1} + \zeta_m p_m$
  - 14: EndDo
- 

**Remark.** Observe that  $V_m^T A V_m = H_m$  is still valid because the orthogonality properties were not used to derive this relation. As a consequence the following result is also valid,

$$\begin{aligned} b - Ax_m &= -h_{m+1,m} e_m^T y_m v_{m+1} \\ \implies \|b - Ax_m\|_2 &= h_{m+1,m} |e_m^T y_m| \\ \text{But, } y_m = H_m^{-1}(\beta e_1) &= U_m^{-1} z_m \implies e_m^T y_m = \zeta_m / u_{mm} \\ \implies \|b - Ax_m\|_2 &= h_{m+1,m} \left| \frac{\zeta_m}{u_{mm}} \right| \end{aligned}$$

Since the residual vectors is a scalar multiple of  $v_{m+1}$  and since the  $v_i$ 's are no longer orthogonal, IOM and DIOM are not orthogonal projection techniques. They can however be viewed as oblique projection techniques onto  $\mathcal{K}_m$  orthogonally to an artificially constructed subspace.

**Proposition 6.** *IOM and DIOM are mathematically equivalent to projection process onto  $\mathcal{K}_m$  and orthogonally to*

$$\mathcal{L}_m = \text{span}\{z_1, z_2, \dots, z_m\},$$

$$\text{where } z_i = v_i - (v_i, v_{m+1})v_{m+1}, \quad i = 1, 2, \dots, m.$$

*Proof.* From the construction of  $\mathcal{L}_m$ ,  $v_{m+1}$  is orthogonal to  $\mathcal{L}_m$  and we know the final residue  $r_m$  is a scalar multiple of  $v_{m+1}$ , hence the approximate solution  $x_m \in \mathcal{K}_m$  and residue vector  $r_m \perp \mathcal{L}_m$ .  $\square$

## 6 Symmetric Lanczos Algorithm

The symmetric lanczos algorithm can be viewed as a simplification of Arnoldi's method for the particular case of symmetric matrix. When  $A$  is symmetric, then the Hessenberg matrix  $H_m$  will become symmetric tridiagonal. The standard notation used to describe the Lanczos algorithm is obtained by setting

$$\alpha_j = h_{jj}, \beta_j = h_{j-1,j},$$

and if  $T_m$  denotes the resulting  $H_m$  matrix, it is of the form,

$$T_m = \begin{pmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \cdot & \cdot & \cdot & \\ & & \beta_{m-1} & \alpha_{m-1} & \beta_m \\ & & & \beta_m & \alpha_m \end{pmatrix}.$$

This leads to the following form of Modified Gram-Schmidt variant of Arnoldi's method:

---

### Algorithm 8 Lanczos Method for Linear Systems

---

- 1: Compute  $r_0 = b - Ax_0$ ,  $\beta = \|r_0\|_2$ , and  $v_1 = r_0/\beta$
  - 2: Set  $\beta_1 = 0$  and  $v_0 = 0$
  - 3: **for**  $j = 1, 2, \dots, m$  **do** ▷ Orthogonalization Procedure
  - 4:    $w_j = Av_j - \beta_j v_{j-1}$
  - 5:    $\alpha_j = (w_j, v_j)$
  - 6:    $w_j = w_j - \alpha_j v_j$
  - 7:    $\beta_{j+1} = \|w_j\|_2$ . If  $\beta_{j+1} = 0$  then Stop
  - 8:    $v_{j+1} = w_j/\beta_{j+1}$
  - 9: **EndDo**
  - 10: Set  $T_m = \text{tridiag}(\beta_i, \alpha_i, \beta_{i+1})$ , and  $V_m = [v_1, \dots, v_m]$ .
  - 11: Compute  $y_m = H_m^{-1} \beta e_1$  and  $x_m = x_0 + V_m y_m$
- 

## 7 Conjugate Gradient

The conjugate gradient algorithm can be derived from the Lanczos algorithm in the same way DIOM was derived from IOM. Infact, the conjugate gradient algorithm can be viewed as a variation of DIOM for the case when  $A$  is symmetric.

First write the LU factorization of  $T_m$  as  $T_m = L_m U_m$ . The matrix  $L_m$  is unit lower bidiagonal and  $U_m$  is unit upper bidiagonal matrix. Thus the factorization of  $T_m$  is of the form

$$T_m = \begin{pmatrix} 1 & & & & \\ \lambda_2 & 1 & & & \\ & \cdot & \cdot & \cdot & \\ & & \lambda_{m-1} & 1 & \\ & & & \lambda_m & 1 \end{pmatrix} \times \begin{pmatrix} \eta_1 & \beta_2 & & & \\ \eta_2 & \beta_3 & & & \\ & \cdot & \cdot & \cdot & \\ & & \eta_{m-1} & \beta_m & \\ & & & \eta_m & \end{pmatrix}.$$

The approximate solution is then given by,

$$x_m = x_0 + V_m U_m^{-1} L_m^{-1} (\beta e_1) = x_0 + P_m z_m.$$

As for DIOM,  $p_m$ , the last column of  $P_m$ , can be computed from the previous  $p_i$ 's and  $v_m$  by the simple update

$$p_m = \eta_m^{-1} [v_m - \beta_m p_{m-1}].$$

Note that  $\beta_m$  is a scalar computed from the Lanczos algorithm, while  $\eta_m$  results from the  $m$ -th Gaussian elimination step on the tridiagonal matrix, i.e,  $\lambda_m = \frac{\beta_m}{\eta_{m-1}}$ ,  $\eta_m = \alpha_m - \lambda_m \beta_m$ . In addition, following again what has been shown for DIOM,  $z_m = \begin{bmatrix} z_{m-1} \\ \zeta_m \end{bmatrix}$ , where  $\zeta_m = -\lambda_m \zeta_{m-1}$ . As a result,  $x_m$  can be updated at each step as follows:

$$x_m = x_{m-1} + \zeta_m p_m.$$

This gives the following algorithm, which we call as direct version of Lanczos algorithm for linear systems.

---

**Algorithm 9** D-Lanczos

---

- 1: Choose  $x_0$  and compute  $r_0 = b - Ax_0$ ,  $\zeta_1 = \beta = \|r_0\|_2$ , and  $v_1 = r_0/\beta$
  - 2:  $\lambda_1 = \beta_1 = 0$ ,  $p_0 = 0$
  - 3: **for**  $m = 1, 2, \dots$ , until convergence **do**
  - 4:   Compute  $w_m = Av_m - \beta_m v_{m-1}$  and  $\alpha_m = (w, v_m)$
  - 5:   If  $m > 1$  then compute  $\lambda_m = \frac{\beta_m}{\eta_{m-1}}$  and  $\zeta_m = -\lambda_m \zeta_{m-1}$
  - 6:    $\eta_m = \alpha_m - \lambda_m \beta_m$
  - 7:    $p_m = \eta_m^{-1} [v_m - \beta_m p_{m-1}]$
  - 8:    $x_m = x_{m-1} + \zeta_m p_m$
  - 9:   If  $x_m$  has converged then Stop
  - 10:    $w = w - \alpha_m v_m$
  - 11:    $\beta_{m+1} = \|w\|_2$ ,  $v_{m+1} = w/\beta_{m+1}$
  - 12: **EndDo**
- 

Observe that the residual vector for this algorithm is in the direction of  $v_{m+1}$ . Therefore, the residual vectors are orthogonal to each other as in FOM. Likewise, the vectors  $p_i$  are A-orthogonal or conjugate to each other.

**Proposition 7.** Let  $r_m = b - Ax_m$ , and  $p_m, m = 0, 1, \dots$ , be the residual vectors and auxiliary vectors produced by D-Lanczos algorithm. Then,

- i) Each residual vector  $r_m$  is such that  $r_m = \sigma_m v_{m+1}$ , where  $\sigma_m$  is a certain scalar. As a result, the residual vectors are orthogonal to each other.
- ii) The auxiliary vectors  $p_i$  form an A-conjugate set, i.e,

$$(Ap_i, p_j) = 0, \text{ for } i \neq j.$$

*Proof.* The first part is immediate consequence of the following relation:

$$r_m = b - Ax_m = -h_{m+1,m} e_m^T y_m v_{m+1}$$

Now we prove the second part. Consider the matrix  $P_m^T A P_m$ ,

$$P_m^T A P_m = U_m^{-T} V_m^T A V_m U_m^{-1} = U_m^{-T} T_m U_m^{-1} = U_m^{-T} L_m$$

Now we observe  $U_m^{-T} L_m$  is a lower triangular matrix which is also symmetric since it is equal to the symmetric matrix  $P_m^T A P_m$ . Therefore it must be diagonal.  $\square$

A consequence of the above proposition is that a version of the algorithm can be derived by imposing the orthogonality and conjugacy conditions. This gives the Conjugate Gradient algorithm which we now derive.

**Remark.** *In order to conform with the standard notation used in the literature to describe the algorithm, the indexing of the  $p$  vectors now begins at zero instead of one as was done so far.*

The vector  $x_{j+1}, r_{j+1}, p_{j+1}$  can be expressed as follows:

$$x_{j+1} = x_j + \alpha_j p_j, \quad r_{j+1} = r_j - \alpha_j A p_j, \quad p_{j+1} = r_{j+1} + \beta_j p_j.$$

It can be seen that  $(A p_j, r_j) = (A p_j, p_j + \beta_j p_{j-1}) = (A p_j, p_j)$ . Now imposing constraints,

1. Orthogonality Conditions on  $r_i$ 's gives  $(r_j - \alpha_j A p_j, r_j) = 0$ , as a result,

$$\alpha_j = \frac{(r_j, r_j)}{(A p_j, r_j)} = \frac{(r_j, r_j)}{(A p_j, p_j)}$$

2. Conjugacy Conditions on  $p_i$ 's gives  $(p_{j+1}, A p_j) = 0$ . But,

$$(p_{j+1}, A p_j) = (r_{j+1} + \beta_j p_j, A p_j) = -\frac{1}{\alpha_j} (r_{j+1} + \beta_j p_j, r_{j+1} - r_j)$$

$$(p_{j+1}, A p_j) = 0 \implies \beta_j = \frac{(r_{j+1}, r_{j+1})}{(r_j, r_j)}$$

It is important to note that the scalar  $\alpha_j, \beta_j$  in this algorithm and D-Lanczos are different.

Putting these relation together gives the following algorithm.

---

**Algorithm 10** Conjugate Gradient

---

- 1: Choose  $x_0$  and compute  $r_0 = b - A x_0$ ,  $p_0 = r_0$ .
  - 2: **for**  $j = 0, 1, 2, \dots$ , until convergence **do**
  - 3:    $\alpha_j = (r_j, r_j) / (A p_j, p_j)$
  - 4:    $x_{j+1} = x_j + \alpha_j p_j$
  - 5:    $r_{j+1} = r_j - \alpha_j A p_j$
  - 6:    $\beta_j = (r_{j+1}, r_{j+1}) / (r_j, r_j)$
  - 7:    $p_{j+1} = r_{j+1} + \beta_j p_j$
  - 8: **EndDo**
-

## 8 Convergence Analysis

Here we show the convergence of CG Algorithm using Chebyshev polynomials.

### 8.1 Real Chebyshev Polynomials

**Definition 2.** The Chebyshev polynomial of the first kind of degree  $k$  is defined by:

$$C_k(t) = \begin{cases} \cos[k \cos^{-1}(t)], & \text{for } |t| \leq 1 \\ \cosh[k \cosh^{-1}(t)], & \text{otherwise.} \end{cases}$$

It can be seen that  $C_0(t) = 1$ ,  $C_1(t) = t$ , can be easily extended by the following trigonometric relation

$$\cos[(k+1)\theta] + \cos[(k-1)\theta] = 2\cos\theta \cos k\theta.$$

This also shows the important three-term recurrence relation

$$C_{k+1}(t) = 2tC_k(t) - C_{k-1}(t).$$

**Proposition 8.** If  $|t| > 1$ , then  $C_k(t) = \frac{1}{2} \left[ \left( t + \sqrt{t^2 - 1} \right)^k + \left( t - \sqrt{t^2 - 1} \right)^{-k} \right]$

*Proof.* We know,  $C_k(t) = \cosh[k \cosh^{-1}(t)]$ ,  $|t| > 1$ ,  $\cosh \theta = \frac{1}{2}(e^\theta + e^{-\theta})$ . Then,  $C_k(t) = \frac{1}{2} \left[ (e^\theta)^k + (e^\theta)^{-k} \right]$ , where  $\theta = \cosh^{-1}(t)$ , i.e,  $t = \cosh \theta$

$$\begin{aligned} \implies t &= \frac{1}{2}(e^\theta + e^{-\theta}) \implies e^{2\theta} - 2te^\theta + 1 = 0 \\ \implies e^\theta &= t \pm \sqrt{t^2 - 1} \end{aligned}$$

Notice that  $t + \sqrt{t^2 - 1} = \frac{1}{t - \sqrt{t^2 - 1}}$ . Then, for  $|t| > 1$ ,

$$C_k(t) = \frac{1}{2} \left[ \left( t + \sqrt{t^2 - 1} \right)^k + \left( t - \sqrt{t^2 - 1} \right)^{-k} \right] \gtrsim \frac{1}{2} \left( t + \sqrt{t^2 - 1} \right)^k.$$

□

In what follows we denote by  $\mathbb{P}_k$  as the set of all polynomials of degree  $k$ .

**Remark.** The Chebyshev polynomials are polynomials with the largest possible leading coefficient whose absolute value on the interval  $[-1, 1]$  is bounded by 1.

**Theorem 7.** Let  $[\alpha, \beta]$  be a non-empty interval in  $\mathbb{R}$  and let  $\gamma$  be any real scalar outside  $[\alpha, \beta]$ . Then,

$$\hat{C}_k(t) = \frac{C_k \left( 1 + 2 \frac{t-\beta}{\beta-\alpha} \right)}{C_k \left( 1 + 2 \frac{\gamma-\beta}{\beta-\alpha} \right)} = \arg \min_{p \in \mathbb{P}_k, p(\gamma)=1} \max_{t \in [\alpha, \beta]} |p(t)|.$$

**Remark.** If  $\gamma \leq \alpha$ , the absolute values are needed in the denominators and exchanging the roles of  $\alpha$  and  $\beta$ , i.e,

$$\hat{C}_k(t) = \frac{C_k \left( 1 + 2 \frac{\alpha-t}{\beta-\alpha} \right)}{C_k \left( 1 + 2 \frac{\alpha-\gamma}{\beta-\alpha} \right)}.$$



We can further state following which is the result of the above remark and theorem.

**Corollary 7.1.**  $\arg \min_{p \in \mathbb{P}_k, p(\gamma)=1} \max_{t \in [\alpha, \beta]} |p(t)| = \frac{1}{|C_k(1 + 2\frac{\gamma-\beta}{\beta-\alpha})|}$

## 8.2 Convergence of CG Algorithm

**Lemma 8.** Let  $x_m$  be the approximate solution obtained from the  $m$ -th step of the CG algorithm, and let  $d_m = x_* - x_m$  where  $x_*$  is the exact solution. Then,  $x_m = x_0 + q_m(A)r_0$ , where  $q_m$  is a polynomial of degree  $m-1$  such that

$$\|(I - Aq_m(A))d_0\|_A = \min_{q \in \mathbb{P}_{m-1}} \|(I - Aq(A))d_0\|_A.$$

*Proof.* Consider the following objective function in the polynomial  $q$

$$\|(I - Aq(A))d_0\|_A = \|d_0 - Aq(A)d_0\|_A = \|x_* - (x_0 + Aq(A)r_0)\|_A = \|x_* - x\|_A.$$

And also observe that  $\forall x \in x_0 + \mathcal{K}_m(A, r_0)$ ,  $x = x_0 + q(A)r_0$ ,  $q \in \mathbb{P}_{m-1}$ . Therefore minimizing  $x$  over  $x_0 + \mathcal{K}_m(A, r_0)$  and  $q$  over  $\mathbb{P}_{m-1}$  are equivalent. We know that  $x_m = \arg \min_{x \in x_0 + \mathcal{K}} \|x_* - x\|_A$ . Hence,  $q_m = \arg \min_{q \in \mathbb{P}_{m-1}} \|(I - Aq(A))d_0\|_A$ .  $\square$

**Theorem 9.** Let  $\eta = \frac{\lambda_n}{\lambda_1 - \lambda_n}$  and  $\kappa = \frac{\lambda_1}{\lambda_n}$ , then

$$\|x_* - x_m\|_A \leq \frac{\|x_* - x_0\|_A}{C_m(1 + 2\eta)} \leq 2 \left[ \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right]^m \|x_* - x_0\|_A.$$

*Proof.* From the previous lemma, it is known that  $\|x_* - x_m\|_A$  is the minimum of the error over polynomials  $r(t) = 1 - tq(t)$  which take the value one at 0, i.e,

$$\|x_* - x_m\|_A = \min_{r \in \mathbb{P}_m, r(0)=1} \|r(A)d_0\|_A.$$

Let  $\lambda_i, i = 1, 2, \dots, n$  are the eigenvalues of  $A$ , and  $\xi_i, i = 1, 2, \dots, n$  are the components of the initial error  $d_0$  in the eigenbasis. Let eigen decomposition of  $A = U\Lambda U^T$ , where  $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ , then  $d_0 = U\xi$ ,  $\xi = [\xi_1 \ \xi_2 \ \dots \ \xi_n]^T$  and  $r(A) = Ur(\Lambda)U^T$ . Then,

$$\begin{aligned} \|r(A)d_0\|_A^2 &= (Ar(A)d_0, r(A)d_0) = d_0^T r(A)A r(A)d_0 \\ &= \xi^T U^T U r(\Lambda) U^T U \Lambda U^T U r(\Lambda) U^T U \xi \\ &= \xi^T \Lambda r(\Lambda)^2 \xi = \sum_{i=1}^n \lambda_i r(\lambda_i)^2 \xi_i^2 \leq \max_i (r(\lambda_i)^2) \|d_0\|_A^2 \\ \implies \|r(A)d_0\|_A^2 &\leq \max_{\lambda \in [\lambda_n, \lambda_1]} (r(\lambda)^2) \|d_0\|_A^2. \end{aligned}$$

Therefore,

$$\begin{aligned} \|x_* - x_m\|_A &\leq \min_{r \in \mathbb{P}_m, r(0)=1} \max_{\lambda \in [\lambda_n, \lambda_1]} |r(\lambda)| \|d_0\|_A \\ &= \frac{\|x_* - x_0\|_A}{C_m(1 + 2\eta)}. \end{aligned}$$

We know,  $C_m(t) \geq \frac{1}{2} \left( t + \sqrt{t^2 - 1} \right)^m$ , then

$$C_m(1 + 2\eta) \geq \frac{1}{2} \left( 1 + 2\eta + 2\sqrt{\eta(\eta - 1)} \right)^m.$$

Now notice that

$$1 + 2\eta + 2\sqrt{\eta(\eta - 1)} = (\sqrt{\eta} + \sqrt{\eta + 1})^2 = \frac{(\sqrt{\lambda_n} + \sqrt{\lambda_1})^2}{\lambda_1 - \lambda_n} = \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}.$$

$$\text{Therefore, } \|x_* - x_m\|_A \leq 2 \left[ \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right]^m \|x_* - x_0\|_A.$$

□

## 9 Deflated-CG Algorithm

### 9.1 Deflated Lanczos algorithm

$A \in \mathbb{R}^{n \times n}$  is SPD and define  $W = [w_1 \ w_2 \ \cdots \ w_k]$ ,  $w_i \in \mathbb{R}^n$  are linearly independent vectors. Let  $v_1 \in \mathbb{R}^n$  be a unit vector such that  $W^T v_1 = 0$ .

**Proposition 9.** *Let  $W$  and  $A$  be defined as above, then  $W^T A W$  is SPD.*

*Proof.* Let  $x \in \mathbb{R}^k$ . We know that the columns of  $W$  are linearly independent, hence  $Wx = 0 \Leftrightarrow x = 0$ . Let  $y = Wx$ , then  $x^T W^T A W x = y^T A y \geq 0$  (since  $A$  is SPD). □

We want to generate a sequence  $\{v_j\}_{j=1,2,\dots}$  of vectors such that

$$v_{j+1} \perp \text{span}\{W, v_1, v_2, \dots, v_j\}, \quad \|v_{j+1}\|_2 = 1.$$

To obtain such a sequence, we apply Lanczos algorithm to the following auxiliary matrix:

$$B = A - A W (W^T A W)^{-1} W^T A,$$

with the given initial vector  $v_1$ . Then we get a sequence  $\{v_j\}_{j=1,2,\dots}$  which satisfies

$$B V_j = V_j T_j + \sigma_{j+1} v_{j+1} e_j^T, \quad \text{where } T_j = \begin{pmatrix} \rho_1 & \sigma_2 & & & \\ \sigma_2 & \rho_2 & \sigma_3 & & \\ & \cdot & \cdot & \cdot & \\ & & \sigma_{j-1} & \rho_{j-1} & \sigma_j \\ & & & \sigma_j & \rho_j \end{pmatrix}$$

It is guaranteed by the Lanczos algorithm that the vectors  $v_j$  are orthogonal to each other. By comparing the last column on the both sides of the above equation, we get

$$\sigma_{j+1} v_{j+1} = B v_j - \sigma_j v_{j-1} - \rho_j v_j.$$

It can also be seen that  $W^T B = 0$ .

**Proposition 10.** *Let  $W$  and  $v_{j+1}$  be defined as above. Then  $W^T v_{j+1} = 0 \ \forall \ j = 1, 2, \dots$ .*

*Proof.* We prove by induction on  $j$ . It is clear from the definition that  $W^T v_1 = 0$ . Now assume that the proposition is true till  $j - 1$ , i.e,  $W^T v_{k+1} = 0 \forall k = 1, 2, \dots, j$ . Now we prove for  $k = j + 1$ .

We know,  $\sigma_{j+1} v_{j+1} = Bv_j - \sigma_j v_{j-1} - \rho_j v_j$ , then

$$\implies \sigma_{j+1} W^T v_{j+1} = W^T Bv_j - \sigma_j W^T v_{j-1} - \rho_j W^T v_j = 0.$$

□

Therefore, we arrive at the following algorithm:

---

**Algorithm 11** Deflated Lanczos Algorithm

---

- 1: Choose  $k$  linearly independent vectors  $w_1, w_2, \dots, w_k$  and
  - 2:     define  $W = [w_1 \ w_2 \ \dots \ w_k]$
  - 3: Choose an initial vector  $v_1$  such that  $W^T v_1 = 0$  and  $\|v_1\|_2 = 1$ .
  - 4: **for**  $j = 1, 2, \dots, m$  **do**
  - 5:     Solve  $W^T A W \hat{v}_j = W^T A v_j$  for  $\hat{v}_j$
  - 6:      $z_j = A v_j - A W \hat{v}_j$
  - 7:      $\rho_j = v_j^T z_j$
  - 8:      $v_{j+1} = z_j - \sigma_j v_{j-1} - \rho_j v_j$
  - 9:      $\sigma_{j+1} = \|v_{j+1}\|_2$ ; If  $\sigma_{j+1} = 0$  Exit.
  - 10:     $v_{j+1} = v_{j+1} / \sigma_{j+1}$
  - 11: **EndDo**
- 

## 9.2 Deflated-CG algorithm

We want to solve  $Ax = b$ ,  $A$  is SPD. Based on deflated Lanczos algorithm we wish to derive a projection method similar to CG-algorithm.

Assume  $x_0$  is the initial guess and initial residue  $r_0 = b - Ax_0$  such that  $r_0 \perp W$ . Now set  $v_1 = r_0 / \|r_0\|_2$  and define  $\mathcal{K}_{k,j}(A, W, r_0) \equiv \text{span}\{W, V_j\}$ . At  $j$ -th step of the projection, we seek for the approximate solution  $x_j \in x_0 + \mathcal{K}_{k,j}(A, W, r_0)$  and  $r_j = b - Ax_j \perp \mathcal{K}_{k,j}(A, W, r_0)$ .

**Lemma 10.** *If  $x_j$  and  $r_j$  satisfy the above conditions, then  $r_j = c_j v_{j+1}$ , for some scalar  $c_j$ . Thus  $\mathcal{K}_{k,j}(A, W, r_0) = \text{span}\{W, r_0, r_1, \dots, r_{j-1}\}$  and the residuals  $r_i$ 's are orthogonal to each other.*

*Proof.* Let  $x_j = x_0 + W \hat{\xi}_j + V_j \hat{\eta}_j$ , for some  $\hat{\xi}_j, \hat{\eta}_j$ . And we know  $B = A - AW(W^T AW)^{-1} W^T A$  and  $BV_j = V_j T_j + \sigma_{j+1} v_{j+1} e_j^T$ , then we get

$$AV_j = AW \Delta_j + V_j T_j + \sigma_{j+1} v_{j+1} e_j^T, \quad \Delta_j = (W^T AW)^{-1} W^T A.$$

Hence,  $r_j = r_0 - AW \hat{\xi}_j - AV_j \hat{\eta}_j = r_0 - AW \hat{\xi}_j - (AW \Delta_j + V_j T_j + \sigma_{j+1} v_{j+1} e_j^T) \hat{\eta}_j$ .

$$W^T r_j = W^T r_0 - (W^T AW)(\hat{\xi}_j - \Delta_j \hat{\eta}_j) - W^T (V_j T_j + \sigma_{j+1} v_{j+1} e_j^T) \hat{\eta}_j$$

$$\implies 0 = -(W^T AW) \hat{\xi}_j - (W^T AW) \Delta_j \hat{\eta}_j$$

$$\implies (W^T AW) \hat{\xi}_j + (W^T AW) \Delta_j \hat{\eta}_j = 0 \implies \hat{\xi}_j = -\Delta_j \hat{\eta}_j$$

Therefore, we have

$$x_j = x_0 - W \Delta_j \hat{\eta}_j + V_j \hat{\eta}_j$$

and

$$r_j = r_0 - (V_j T_j + \sigma_{j+1} v_{j+1} e_j^T) \hat{\eta}_j.$$

We know,  $V_j^T r_j = 0 \implies 0 = V_j^T r_0 - T_j \hat{\eta}_j \implies \hat{\eta}_j = \|r_0\|_2 T_j^{-1} e_1$ . Now substituting this equation in the above equation, we get

$$\begin{aligned} r_j &= r_0 - \|r_0\|_2 V_j T_j (T_j^{-1} e_1) - \|r_0\|_2 \sigma_{j+1} v_{j+1} e_j^T (T_j^{-1} e_1) \\ &= (-\|r_0\|_2 \sigma_{j+1} e_j^T T_j^{-1} e_1) v_{j+1} = c_j v_{j+1} \end{aligned}$$

□

Let  $T_j = L_j D_j L_j^T$  and define  $P_j \equiv [p_0 \ p_1 \ \cdots \ p_{j-1}] = (-W \Delta_j + V_j) L_j^{-T} \Lambda_j$ , where  $\Lambda_j = \text{diag}\{c_0, c_1, \dots, c_{j-1}\}$ .

**Proposition 11.** *The approximate solution  $x_j$ , the residual  $r_j$  and the descent direction  $p_j$  satisfy the following recurrence relations:*

$$x_j = x_{j-1} + \alpha_{j-1} p_{j-1}, \quad r_j = r_{j-1} - \alpha_{j-1} A p_{j-1}, \quad p_j = r_j + \beta_{j-1} p_{j-1} - W \hat{\mu}_j,$$

for some  $\alpha_{j-1}, \beta_{j-1}, \hat{\mu}_j$ . Thus  $\mathcal{K}_{k,j}(A, W, r_0) = \text{span}\{W, p_0, p_1, \dots, p_{j-1}\}$ .

*Proof.* Let  $\hat{\zeta}_j = \|r_0\|_2 (L_j D_j \Lambda_j)^{-1} e_1$ . We know,

$$\begin{aligned} x_j &= x_0 + \|r_0\|_2 (-W \Delta_j + V_j) T_j^{-1} e_1 \\ &= x_0 + [(-W \Delta_j + V_j) L_j^{-T} \Lambda_j] [\|r_0\|_2 \Lambda_j^{-1} L_j^T (L_j D_j L_j^T)^{-1} e_1] \\ &= x_0 + P_j \zeta_j \end{aligned}$$

Now notice that  $L_j D_j \Lambda_j$  is lower bidiagonal matrix, then we have,

$$\zeta_j = \begin{bmatrix} \zeta_{j-1} \\ \alpha_{j-1} \end{bmatrix}, \quad \text{for some scalar } \alpha_{j-1}.$$

Hence,

$$x_j = x_{j-1} + \alpha_{j-1} p_{j-1} \quad \text{and} \quad r_j = r_{j-1} - \alpha_{j-1} A p_{j-1}.$$

We know that  $P_j = (-W \Delta_j + V_j) L_j^{-T} \Lambda_j$ , then

$$P_j \Lambda_j^{-1} L_j^T \Lambda_j = (-W \Delta_j \Lambda_j + V_j \Lambda_j).$$

Now notice that  $\Lambda_j^{-1} L_j^T \Lambda_j$  is a unit upper bidiagonal matrix. Comparing the last columns of the both sides of this equation

$$p_{j-1} - \beta_{j-2} p_{j-2} = -W \hat{\mu}_{j-1} + c_{j-1} v_j,$$

where  $\hat{\mu}_{j-1} = c_{j-1} \hat{v}_j$  and  $\beta_{j-2} = -c_{j-1} u_{j-1,j} / c_{j-2}$ . □

**Proposition 12.** *The vectors  $p_j$  are  $A$ -orthogonal to each other, i.e.,  $P_j^T A P_j$  is diagonal. In addition, they are also  $A$ -orthogonal to all  $w_i$ 's, i.e.,  $W^T A P_j = 0$ .*

*Proof.*

$$\begin{aligned} P_j^T A P_j &= P_j^T (-A W \Delta_j + A V_j) L_j^{-T} \Lambda_j \\ &= P_j^T (V_j T_j + \sigma_{j+1} v_{j+1} e_j^T) L_j^{-T} \Lambda_j \\ &= \Lambda_j L_j^{-1} (-\Delta_j^T W^T + V_j^T) (V_j T_j + \sigma_{j+1} v_{j+1} e_j^T) L_j^{-T} \Lambda_j \\ &= \Lambda_j L_j^{-1} T_j L_j^{-T} \Lambda_j = \Lambda_j^2 D_j, \quad \text{which is diagonal.} \end{aligned}$$

Consider,

$$W^T AP_j = W^T (V_j T_j + \sigma_{j+1} v_{j+1} e_j^T) L_j^{-T} \Lambda_j = 0.$$

□

**Proposition 13.** *The coefficients in Delfated-CG satisfy the relations*

$$\alpha_j = \frac{(r_j, r_j)}{(Ap_j, p_j)}, \quad \hat{\mu}_j = (W^T AW)^{-1} Ar_j, \quad \beta_j = \frac{(r_{j+1}, r_{j+1})}{(r_j, r_j)}.$$

*Proof.* From the orthogonality constraints on residuals,  $r_{j-1}^T r_j = 0$ . Then,

$$\implies r_{j-1}^T (r_{j-1} - \alpha_{j-1} Ap_{j-1}) = 0$$

$$\implies \alpha_{j-1} = \frac{(r_{j-1}, r_{j-1})}{(r_{j-1}, Ap_{j-1})}$$

But,  $(p_{j-2}, Ap_{j-1}) = 0$  and  $(W\hat{\mu}_{j-1}, Ap_{j-1}) = 0$

$$\implies \alpha_{j-1} = \frac{(r_{j-1}, r_{j-1})}{(\beta_{j-2} p_{j-2} + r_{j-1} - W\hat{\mu}_{j-1}, Ap_{j-1})}$$

$$\implies \alpha_{j-1} = \frac{(r_{j-1}, r_{j-1})}{(p_{j-1}, Ap_{j-1})}$$

We know,

$$p_j = r_j + \beta_{j-1} p_{j-1} - W\hat{\mu}_j$$

$$\implies W^T Ap_j = W^T Ar_j + \beta_{j-1} W^T Ap_{j-1} - W^T AW\hat{\mu}_j$$

$$\implies \hat{\mu}_j = (W^T AW)^{-1} Ar_j$$

Similarly,  $p_{j-1}^T Ap_j = p_{j-1}^T Ar_j + \beta_{j-1} p_{j-1}^T Ap_{j-1} - p_{j-1}^T AW\hat{\mu}_j$

$$\implies \beta_{j-1} = -\frac{(Ap_{j-1}, r_j)}{(p_{j-1}, Ap_{j-1})} = \frac{1}{\alpha_{j-1}} \frac{(r_j - r_{j-1}, r_j)}{(p_{j-1}, Ap_{j-1})}$$

$$\implies \beta_{j-1} = \frac{(r_{j+1}, r_{j+1})}{(r_j, r_j)}$$

□

**Remark.** *To guarantee that initial guess  $x_0$  satisfies  $W^T r_0 = 0$ , we can choose  $x_0$  in the form*

$$x_0 = x_{-1} + W(W^T AW)^{-1} W^T r_{-1},$$

where  $x_{-1}$  is arbitrary and  $r_{-1} = b - Ax_{-1}$ .

Then,

$$W^T r_0 = W^T r_{-1} - (W^T AW)(W^T AW)^{-1} W^T r_{-1} = 0.$$

Putting these relation together gives the following algorithm.

---

**Algorithm 12** Deflated-CG

---

```
1: Choose k linearly independent vectors  $w_1, w_2, \dots, w_k$  and
2:   define  $W = [w_1 \ w_2 \ \dots \ w_k]$ 
3: Choose an initial guess  $x_0$  such that  $W^T r_0 = 0$ , where  $r_0 = b - Ax_0$ 
4: Solve  $W^T A W \hat{\mu}_0 = W^T A r_0$  for  $\hat{\mu}_0$  and set  $p_0 = r_0 - W \hat{\mu}_0$ 
5: for  $j = 1, 2, \dots, m$ , do
6:    $\alpha_{j-1} = (r_{j-1}, r_{j-1}) / (A p_{j-1}, p_{j-1})$ 
7:    $x_j = x_{j-1} + \alpha_{j-1} p_{j-1}$ 
8:    $r_j = r_{j-1} - \alpha_{j-1} A p_{j-1}$ 
9:    $\beta_{j-1} = (r_j, r_j) / (r_{j-1}, r_{j-1})$ 
10:  Solve  $W^T A W \hat{\mu}_j = W^T A r_j$  for  $\hat{\mu}_j$ 
11:   $p_j = r_j + \beta_{j-1} p_{j-1} - W \hat{\mu}_j$ 
12: EndDo
```

---

## References

- [1] Y. Saad, Iterative Methods for Sparse Linear Systems, PWS publishing company, Boston, MA, 1996.
- [2] Y. Saad, M. Yeung, J. Erhel, F. Guyomarc'H, A deflated version of the conjugate gradient algorithm, SIAM J. Sci. Comput., 21 (2000), pp. 1909–1926.
- [3] W. D. Joubert and T. A. Manteuffel, Iterative Methods for Nonsymmetric Linear Systems, Academic Press, New York, 1990, pp. 149–171