FACULTAD DE MATEMÁTICAS

TRABAJO FIN DE GRADO

Métodos iterativos de resolución de sistemas lineales basados en subespacios de Krylov. Aplicaciones a la resolución numérica de EDP y en matrices test.

Tutor: Samuele Rubino

Marta Guijo Moreno

Grado en Matemáticas

Curso 2022/2023

Resumen

Este trabajo se centra en los diferentes métodos iterativos para resolver sistemas lineales basados en subespacios de Krylov. El objetivo principal es realizar una comparativa entre los distintos métodos a la hora de resolver numéricamente un problema de EDP o problemas test con matrices grandes y dispersas. Para ello, introducimos en el capítulo 1 conceptos ya vistos en Cálculo Numérico II necesarios para poder definir los siguientes métodos. En el capítulo 2 definimos métodos de proyección y en el 3 nos centramos en los métodos iterativos de Krylov. Posteriormente, en el capítulo 4, consideramos un problema de EDP, su discretización e implementación en Matlab para la comparación de dichos métodos. Vamos a ver la diferencia en sistemas precondicionados y no precondicionados en el método GMRES considerando problemas test con matrices grandes y dispersas y por último, analizamos el resultado obtenido.

Abstract

The present work focuses on different iterative methods for solving linear systems based on Krylov subspaces. The main objective is to make a comparison among the different methods when solving a PDE problem numerically or test problems with large and sparse arrays. To do so, we introduce in Chapter 1 concepts already covered in the course Numerical Calculus II (Cálculo Numérico II), which are required to define the following methods. In Chapter 2, the projection methods are defined and in Chapter 3 we focus on Krylov's iterative methods. Subsequently, in Chapter 4, we will consider a PDE problem, its discretization and implementation in MATLAB for the comparison of these methods. We will consider the difference in preconditioned systems and non-preconditioned systems in the GMRES method considering test problems with large and sparse arrays and finally, the result obtained will be analyzed.

Índice general

Introducción III

1. Métodos iterativos básicos 1

1.1. Introducción . . . 1

1.2. Métodos de Jacobi, Gauss-Seidel y relajación . . . 4

1.2.1. Método de Jacobi . . . 5

1.2.2. Método de Gauss-Seidel . . . 6

1.2.3. Método de relajación . . . 7

2. Métodos de proyección para la resolución de sistemas lineales 11

2.1. Definiciones básicas y algoritmos . . . 11

2.1.1. Métodos generales de proyección . . . 12

2.1.2. Representación matricial . . . 12

2.2. Teoría general . . . 14

2.2.1. Dos resultados de optimalidad . . . 14

2.2.2. Interpretación en términos de proyectores . . . 15

2.2.3. Límite de error general . . . 16

2.3. Procesos de proyección unidimensional . . . 18

2.3.1. Descenso más rápido . . . 19

2.3.2. Iteración MR . . . 21

2.3.3. Norma residual del descenso más rápido . . . 22

3. Métodos iterativos de Krylov 23

3.1. Subespacio de Krylov. . . . 23

3.2. El método de Arnoldi 25
3.2.1. El método de Arnoldi para sistemas lineales 25
3.3. El método del residual mínimo generalizado (GMRES) 26
3.3.1. Problemas de implementación práctica 28
3.4. Algoritmo simétrico de Lanczos 31
3.4.1. El algoritmo 31
3.4.2. Relación con los polinomios ortogonales 32
3.5. Algoritmo del gradiente conjugado 33
3.5.1. Derivación y teoría 33

4. Aplicación a la resolución numérica de EDP y en matrices test **37**
4.1. Aplicación a la discretización de EDP lineales elípticas 37
4.1.1. Ejemplo 1 40
4.2. Aplicación en matrices test 42
4.2.1. Precondicionamiento 42
4.2.2. Ejemplo 2 45
4.2.3. Ejemplo 3 45
4.3. Conclusiones 46

Bibliografía **47**

Introducción

En este trabajo nos centramos en los métodos iterativos de resolución de sistemas lineales. Recordamos que los métodos directos de resolución de sistemas lineales (vistos en la asignatura Cálculo Numérico I) proporcionan la solución exacta x de un sistema lineal en un número finito de pasos. A diferencia de los métodos directos, los métodos iterativos conducen a la solución exacta después de un número infinito de pasos. Estos métodos iterativos pueden ser mejores que los métodos directos en algunas situaciones particulares, por ejemplo, para sistemas lineales grandes y, con frecuencia, muy eficientes en el caso de matrices huecas (matrices en las que la mayor parte de sus elementos son cero).

En el capítulo 1 describimos los métodos iterativos de resolución de sistemas lineales vistos en Cálculo Numérico II. Los métodos iterativos son apropiados para sistemas lineales grandes y matrices huecas cuando se discretizan EDP (e.g., ecuación de Poisson, convección-reacción-difusión, etc.) mediante, por ejemplo, el método de diferencias finitas o el método de elementos finitos. Comenzamos el capítulo con conceptos sobre la convergencia de dichos métodos y las estimaciones a priori y a posteriori. Repasamos en particular los métodos de Jacobi, Gauss-Seidel y Relajación.

A lo largo del capítulo 2 definimos métodos de proyección para la resolución de sistemas lineales con el fin de obtener una solución aproximada del problema planteado para un subespacio de $\mathbb{R}^n$. También exponemos dos resultados de optimalidad donde obtendremos la condición de Petrov-Galerkin. A continuación, desarrollamos varios procesos de proyección unidimensional como el algoritmo del descenso más rápido cuyo resultado es una consecuencia de la Desigualdad de Kantorovich; el algoritmo del residuo mínimo; y la norma residual del descenso más rápido.

En el capítulo 3 describimos métodos iterativos de Krylov introduciendo los subespacios de Krylov que denotaremos $\mathcal{K}_m$. A continuación, vemos un método de proyección ortogonal sobre $\mathcal{K}_m$ llamado método de Arnoldi, para matrices no hermitianas. También describimos el método del residual mínimo generalizado (GMRES) en el que nos centramos resolviendo el problema de mínimos cuadrados transformando la matriz de Hessenberg en una de forma triangular superior. A continuación, describimos el algoritmo de Lanczos que se puede ver como una simplificación del método de Arnoldi cuando la matriz es simétrica. Y por último, describimos el algoritmo del gradiente conjugado que es una de las mejores técnicas iterativas conocidas para resolver sistemas lineales definidos positivos, simétricos y dispersos.

Para terminar, en el capítulo 4 vemos la aplicación de dichos métodos en diferentes ejemplos que hemos desarrollado. En el primero observamos la velocidad de convergencia de los diferentes métodos vistos en los capítulos anteriores aplicados a la discretización en diferencias finitas de la ecuación de Poisson en dimensión 2 y 3. En los últimos dos ejemplos, consideramos problemas test con matrices grandes y dispersas. En particular, vemos la diferencia entre usar matrices precondicionadas y no precondicionadas sin reinicialización en el método GMRES. Y por último, consideramos el método GMRES aplicado a una matriz precondicionada para 3 casos distintos de reinicialización.

Capítulo 1

Métodos iterativos básicos

Durante el desarrollo de este trabajo se hará referencia a diferentes métodos básicos. Para su mayor comprensión, y debido a la importancia que poseen, en este capítulo se hace un breve repaso sobre dichos métodos ya vistos en la asignatura Cálculo Numérico II [Apuntes CN2].

1.1 Introducción

A lo largo de este capítulo estudiaremos diferentes métodos para resolver grandes sistemas lineales de ecuaciones

$$Au = b, \tag{1.1}$$

con $A \in \mathcal{L}(\mathbb{K}^n)$ y $b \in \mathbb{K}^n$ (siendo $\mathbb{K}$ un cuerpo de escalares $\mathbb{K} = \mathbb{C}$ o $\mathbb{K} = \mathbb{R}$).

Uno de los objetivos principales cuando se estudian estos métodos es estudiar

- la convergencia del método, es decir, estudiar si se tiene

  $$lim_{k\to\infty} u^k = u,$$

  con u la solución del sistema lineal, y
- analizar la velocidad de convergencia del método, es decir, analizar cómo se reduce la norma del error $\|u_k - u\|$ en función del número de iteraciones k.

Supongamos además que el sistema (1.1) se puede reescribir de manera equivalente como

$$u = Bu + c, \tag{1.2}$$

para cierta matriz $B \in \mathcal{L}(\mathbb{C}^n)$ y cierto vector $c \in \mathbb{K}^n$ tal que $det(I - B) \neq 0$. Obsérvese que el sistema (1.2) tiene una única solución y esta está escrita como ecuación de punto fijo. Eso hace que, para aproximarla, podamos plantear el método de las aproximaciones sucesivas. Este método proporciona el siguiente método iterativo de un paso [1]:

$$\begin{cases} u^0 \in \mathbb{K}^n \ \ arbitrario \\ u^{k+1} = Bu^k + c, \ \ \forall k \geq 0, \end{cases} \tag{1.3}$$

Un resultado de convergencia para el método iterativo general (1.3) sería el siguiente:
Teorema 1.1 (Convergencia). Las siguientes condiciones son equivalentes:

a) El método iterativo (1.3) es convergente.
b) $\rho(B) < 1$, siendo $\rho(B)$ el radio espectral de B.
c) $lim_{k\to\infty} B^k = 0 \quad \text{en} \quad \mathcal{L}(\mathbb{C}^n)$.

Para la demostración de este resultado se puede consultar los apuntes de Cálculo Numérico II.

A continuación, analizamos la velocidad de convergencia de un método iterativo convergente. Se tiene:

Teorema 1.2 (Estimación a priori). Supongamos que el método iterativo (1.3) es globalmente convergente y sea $\|\cdot\|$ una norma vectorial tal que la norma matricial subordinada satisface $\|B\| < 1$. Entonces, se tiene la estimación a priori:

$$\|u^k - u\| \leq \|B\|^k \|u^0 - u\|, \quad \forall k \geq 0.$$

Además, si B es una matriz normal, se satisface

$$\|u^k - u\|_2 \leq \rho(B)^k \|u^0 - u\|_2, \quad \forall k \geq 0.$$

Demostración. La prueba es casi inmediata a la vista de la expresión del error:

$$e^k = B^k e^0, \quad \forall k \geq 0.$$

Sin más que tomar norma se deduce el resultado general.
Si además B es normal, podemos trabajar con la norma vectorial $\|\cdot\|_2$ y con $\|\cdot\|_S$. Efectivamente, como el método (1.3) es convergente, se tiene $\rho(B) < 1$. Así,

$$\|e^k\|_2 = \|B^k e^0\| \leq \|B^k\|_S \|e^0\|_2 = \rho(B^k)\|e^0\|_2 = \rho(B)^k \|e^0\|_2, \quad \forall k \geq 0. \tag{1.4}$$

□

En la desigualdad anterior hemos usado:

- Sea $A \in \mathcal{L}(\mathbb{C}^n)$ una matriz normal, entonces $\|A\|_S = \rho(A)$.

y las siguientes propiedades de los autovalores:

- Sea $A \in \mathcal{L}(\mathbb{C}^n)$ una matriz, entonces

$$det(A) = \prod_{i=1}^{n} \lambda_i(A).$$

  donde λ es un autovalor o valor propio de la matriz A.
- Dada $A \in \mathcal{L}(\mathbb{C}^n)$, se tiene $\lambda_i(A^k)$ para cualquier i: $1 \leq i \leq n$. En particular, $\rho(A^k) = \rho(A)^k$.

- Sea $A \in \mathcal{L}(\mathbb{C}^n)$ una matriz hermítica. Entonces, $sp(A) \subset \mathbb{R}$, es decir, los autovalores de A son reales.

Observación 1.2. De manera general, la velocidad de convergencia del método globalmente convergente (1.3) está asociado al radio espectral de B, $\rho(B)$, en el siguiente sentido:

1. Si B es normal, entonces se tiene (1.4) siendo la desigualdad óptima pues, en primer lugar, existen datos $e^0 \in \mathbb{K}^n$ tales que (1.4) es una igualdad y, en segundo lugar, cualquier norma matricial satisface $\|B\| \geq \rho(B)$.
2. En el caso general, consideramos una norma $\|\cdot\|$ en $\mathbb{C}^n$ y denotamos $\|\cdot\|$ la correspondiente norma matricial subordinada. Por otro lado, dado $\epsilon > 0$, del Teorema de Ciarlet [5], en el cuál $lim_{k\to\infty}\|A^k\|^{\frac{1}{k}} = \rho(A)$, deducimos que existe $k_0 \geq 1$ tal que

$$\|B^k\| \leq (\rho(B) + \epsilon)^k, \quad \forall k \geq k_0.$$

   Por tanto, el método aumenta su velocidad de convergencia cuanto menor sea $\rho(B)$.

Vemos a continuación que podemos obtener también acotaciones del error que son independientes del error inicial, es decir, independientes de la solución exacta. Se tiene:

Teorema 1.3 (Estimación a posteriori). Supongamos que el método iterativo (1.3) es globalmente convergente, y sea $\|\cdot\|$ una norma matricial consistente con la norma vectorial $\|\cdot\|$ tal que $\|B\| < 1$. Entonces, se satisfacen las **estimaciones a posteriori**:

$$\|u^k - u\| \leq \frac{\|B\|}{1-\|B\|}\|u^k - u^{k-1}\| \;\; y \;\; \|u^k - u\| \leq \frac{\|B\|^k}{1-\|B\|}\|u^1 - u^0\|, \;\; \forall k \geq 1.$$

Demostración. Para $m > k$, se tiene

$$\|u^m - u^k\| \leq \|u^m - u^{m-1}\| + ... + \|u^{k+1} - u^k\| \leq (\|B\|m - k + ... + \|B\|)\|u^k - u^{k-1}\|, \tag{1.5}$$

donde se ha usado que $u^{k+1} - u^k = B(u^k - u^{k-1})$ luego $\|u^{k+1} - u^k\| \leq \|B\|\|u^k - u^{k-1}\|$, que $u^{k+2} - u^{k+1} = B^2(u^k - u^{k-1})$, $\|u^{k+2} - uk + 1 \leq \|B\|^2\|u^k - u^{k-1}\|$ y así sucesivamente hasta

$$\|u^m - u^{m-1}\| \leq \|B\|^{m-k}\|u^k - u^{k-1}\|. \tag{1.6}$$

Entonces, aplicando la fórmula de la suma parcial de una serie geométrica a (1.5),

$$\|u^m - u^k\| \leq \frac{\|B\| - \|B\|^{m-k+1}}{1-\|B\|}\|u^k - u^{k-1}\| \leq \frac{\|B\|}{1-\|B\|}\|u^k - u^{k-1}\|, \tag{1.7}$$

y tomando límites con $m \to \infty$ sigue la primera estimación a posteriori. La segunda estimación a posteriori se deduce de (1.6) y (1.7) (para $m = k$ y $k = 1$). □

1.2 Métodos de Jacobi, Gauss-Seidel y relajación

Dado el sistema lineal (1.1) con $A \in \mathcal{L}(\mathbb{K}^n)$ y $b \in \mathbb{K}^n$ tales que $detA \neq 0$. Consideramos en la siguiente sección los métodos iterativos de Jacobi, Gauss-Seidel y relajación. En general, estos métodos consideran un vector inicial arbitrario u^0 y calculan sucesivamente vectores $u^1, u^2, \ldots$ de acuerdo al método aplicado [1].

Para cada método suponemos que los elementos diagonales de A satisfacen

$$a_{ii} \neq 0, \quad \forall i : 1 \leq i \leq n. \tag{1.8}$$

Así, obtenemos estos métodos escribiendo la i-ésima ecuación de (1.1) como:

$$u_i = \frac{1}{a_{ii}}[b_i - \sum_{j=1}^{n} a_{ij}u_j], \quad 1 \leq i \leq n.$$

o como

$$u_i = u_i + \frac{w}{a_{ii}}[b_i - \sum_{j=1}^{n} a_{ij}u_j], \quad 1 \leq i \leq n.$$

para $w \in \mathbb{K}$.
Vemos a continuación que estos tres métodos son casos particulares del algoritmo (1.3) para distintas elecciones de matrices B.

Supongamos que podemos escribir $A = M - N$ siendo $M \in \mathcal{L}(\mathbb{K}^n)$ una matriz tal que

$$detM \neq 0, \tag{1.9}$$

matriz "fácil de invertir", en el sentido de que el sistema lineal de matriz M sea fácil de resolver (en la práctica, M va a ser diagonal o triangular). Gracias a la hipótesis (1.9), u es solución del sistema lineal (1.1) si y solo si $Mu = Nu + b$, es decir, si y solo si

$$u = (M^{-1}N)u + M^{-1}b. \tag{1.10}$$

Por tanto, podemos considerar el método iterativo asociado a la ecuación de punto fijo (1.2) con

$$B = M^{-1}N = M^{-1}(M - A) = I - M^{-1}A \quad y \quad c = M^{-1}b.$$

Además, $I - B = M^{-1}A$ y, por tanto, $det(I - B) = detA(detM)^{-1} \neq 0$.

El método iterativo tiene la forma: $\begin{cases} u^0 \in \mathbb{K}^n \quad dado \\ u^{k+1} = (M^{-1}N)u^k + M^{-1}b, \quad \forall k \geq 0. \end{cases}$ que será globalmente convergente si y solo si $\rho(M^{-1}N) < 1$ (Teorema 1.1).

Observación 1.3. Para aplicar el método iterativo, hay que invertir la parte M de la matriz A. Intuitivamente, parece que cuanto más se parezca M a A, mejor será el método iterativo, pero su aplicación será más complicada. En el caso límite, podríamos elegir $M = A$ y $N = 0$. Así, el método es convergente en una única iteración: $u_1 = A^{-1}b = u$.

Para presentar los métodos iterativos de esta sección, haremos la siguiente descomposición de A

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} = D - E - F,$$

siendo

$$D = \begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix}, \quad E = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ -a_{21} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -a_{n1} & -a_{n2} & \cdots & 0 \end{pmatrix}, \quad F = \begin{pmatrix} 0 & -a_{12} & \cdots & -a_{1n} \\ 0 & 0 & \cdots & -a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}.$$

1.2.1 Método de Jacobi

Dada una matriz $A \in \mathbb{K}^{nxn}$ regular y un vector inicial $u^0 \in \mathbb{K}^n$, el método de Jacobi para el sistema $Au = b$ es:

$$u_i^{k+1} = \frac{1}{a_{ii}}(b_i - \sum_{j=1}^{n} a_{ij}u_j^k), \quad 1 \leq i \leq n. \tag{1.11}$$

El método de Jacobi se define invirtiendo la parte diagonal de A, es decir, tomando $M = D$, y entonces $N = E + F$. Gracias a la hipótesis (1.8), se tiene (1.9). El método adopta la forma (1.3) con $B = D^{-1}(E + F)$ y $c = D^{-1}b$ o, equivalentemente,

$$\begin{cases} u^0 \in \mathbb{K}^n \ \ dado, \\ u^{k+1} = D^{-1}(E + F)u^k + D^{-1}b, \quad \forall k \geq 0. \end{cases}$$

Se llamará matriz de Jacobi a

$$J = D^{-1}(E + F) = I - D^{-1}A.$$

Como consecuencia del Teorema (1.1), podemos escribir

Corolario 1.4. Sea $A \in \mathcal{L}(\mathbb{K}^n)$ una matriz satisfaciendo $detA \neq 0$ y (1.8), y sea $b \in \mathbb{K}^n$. Entonces, el método de Jacobi es globalmente convergente si y solo si $\rho(J) < 1$.

Para el cálculo efectivo de la sucesión $\{u^k\}_{k\geq 1} \subset \mathbb{K}^n$ asociada, escribimos el método como $Du^{k+1} = (E + F)u^k + b$ que, matricialmente, nos proporciona

$$\begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} u_1^{k+1} \\ u_2^{k+1} \\ \vdots \\ u_n^{k+1} \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} - \begin{pmatrix} 0 & a_{12} & \cdots & a_{1n} \\ a_{21} & 0 & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & 0 \end{pmatrix} \begin{pmatrix} u_1^k \\ u_2^k \\ \vdots \\ u_n^k \end{pmatrix}.$$

Así pues, para cada i: $1 \leq i \leq n$,

$$\begin{cases} u_i^{k+1} & = \frac{1}{a_{ii}}[b_i - (a_{i1}u_1^k + \ldots + a_{i,i-1}u_{i-1}^k + a_{i,i+1}u_{i+1}^k + \ldots + a_{in}u_n^k)] \\ & = u_i^k + \frac{1}{a_{ii}}[b_i - (Au^k)_i]. \end{cases}$$

Observación 1.4. 1) Los pasos a seguir para el cálculo de u_i^{k+1} son los siguientes. Si en un vector de memoria u tenemos guardado u^k,

1. $r \leftarrow b - Au$ (residuo en la etapa k),
2. $u_i \leftarrow u_i + \frac{r_i}{a_{ii}}$.

2) Para calcular cada componente u_i^{k+1} se utiliza el vector u^k, con lo que se usan 2 vectores de dimensión n en cada iteración, es decir, 2n registros de memoria.

3) Las n componentes del vector x^{k+1} son calculadas de manera similar (así, el método también se conoce como método de iteraciones simultáneas), lo que hace que el método sea fácilmente "paralelizable".

1.2.2 Método de Gauss-Seidel

Para el método de Jacobi (1.11) se consideran los valores $u_1^k, ..., u_{i-1}^k$ para calcular u_i^{k+1}. Sin embargo, podemos considerar intuitivamente que las aproximaciones $u_1^{k+1}, ..., u_{i-1}^{k+1}$ son mejores que $u_1^k, ..., u_{i-1}^k$ para calcular u_i^{k+1}, por lo que se puede definir el siguiente método de iteración de punto fijo en el que se van sustituyendo las antiguas aproximaciones $u_1^k, ..., u_{i-1}^k$ por las nuevas $u_1^{k+1}, ..., u_{i-1}^{k+1}$ conforme se van calculando.

El método de Gauss-Seidel para la solución del sistema lineal de ecuaciones $Au = b$ viene dado por:

$$u_i^{k+1} = \frac{1}{a_i i}(b_i - \sum_{j=1}^{i-1} a_{ij} u_j^{k+1} - \sum_{j=i+1}^{n} a_{ij} u_j^k), \quad 1 \leq i \leq n. \tag{1.12}$$

Con la notación anterior, introducimos el método invirtiendo la parte triangular inferior de A, es decir, tomando $M = D - E$ y $N = F$. Con esta elección y, de nuevo, gracias a la hipótesis (1.8) tenemos $det(D - E) = detD \neq 0$, es decir, se tiene (1.9). El método tiene la forma:

$$\begin{cases} u^0 \in \mathbb{K}^n \ \ dado \\ u^{k+1} = (D - E)^{-1} F u^k + (D - E)^{-1} b, \quad \forall k \geq 0. \end{cases}$$

Se llamará matriz de Gauss-Seidel a la matriz dada por:

$$\mathcal{L}_1 = (D - E)^{-1} F.$$

De nuevo, como consecuencia del Teorema 1.1, podemos escribir
Corolario 1.5. Sea $A \in \mathcal{L}(\mathbb{K}^n)$ una matriz satisfaciendo $detA \neq 0$ y (1.8), y sea $b \in \mathbb{K}^n$. Entonces, el método de Gauss-Seidel es globalmente convergente si y solo si $\rho(\mathcal{L}_1) < 1$.

Obvsérvese que el método puede ser reescrito como

$$(D - E)u^{k+1} = Fu^k + b, \quad \forall k \geq 0.$$

Así, el cálculo efectivo de la sucesión $\{u_k\}_{k\geq 1} \subset \mathbb{K}^n$ asociada al método se lleva a cabo del modo siguiente: $Du^{k+1} = Eu^{k+1} + Fu^k + b$, que, matricialmente, se escribe como:

$$\begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} u_1^{k+1} \\ u_2^{k+1} \\ \vdots \\ u_n^{k+1} \end{pmatrix}$$

$$= \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} - \begin{pmatrix} 0 & 0 & \cdots & 0 \\ a_{21} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & 0 \end{pmatrix} \begin{pmatrix} u_1^{k+1} \\ u_2^{k+1} \\ \vdots \\ u_n^{k+1} \end{pmatrix} - \begin{pmatrix} 0 & a_{12} & \cdots & a_{1n} \\ 0 & 0 & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} \begin{pmatrix} u_1^{k} \\ u_2^{k} \\ \vdots \\ u_n^{k} \end{pmatrix}.$$

Así, la primera componente se calcula como

$$u_1^{k+1} = u_1^k + \frac{1}{a_{11}}[b_1 - (a_{11}u_1^k + a_{12}^k + ... + a_{1n}u_n^k)].$$

Fijemos ahora i: $2 \leq i \leq n$. Calculadas las componentes previas del vector u^{k+1}, es decir, hallados u_j^{j+1} para j<i, determinamos u_i^{k+1} como

$$u_i^{k+1} = \frac{1}{a_{ii}} \left[b_i - \sum_{j<i} a_{ij}u_j^{k+1} - \sum_{j>i} a_{ij}u_j^{k+1} \right] = u_i^k + \frac{1}{a_{ii}} \left[b_i - \sum_{j<i} a_{ij}u_j^{k+1} - \sum_{j\geq i} a_{ij}u_j^{k+1} \right]. \tag{1.13}$$

Observación 1.5. En el método de Gauss-Seidel, el valor u_i^k no interviene en el cálculo de los u_j^{k+1}, para j > i. Por tanto, podemos reemplazar las posiciones de memoria que almacenan los valores u_i^k por los nuevos valores u_i^{k+1}. Eso hace que, para aplicar este método con un programa informático baste utilizar un único vector (n posiciones de memoria) que almacene los distintos vectores u^k que calcula el método.

El cálculo de u_i^{k+1} se hace del modo siguiente: Si en un vector de memoria u tenemos guardado u^k, se hace un ciclo descendente en las componentes, desde $i = 1, ..., n$, calculando

$$u_i \leftarrow u_i + \frac{1}{a_{ii}}(b_i - (Au)_i).$$

En este método, además de necesitar menos memoria (solo el vector u), se "invierte" más parte de la matriz A que en el de Jacobi. Es razonable pensar que será más rápido. Como desventaja respecto al método de Jacobi, ahora el cálculo de las diferentes componentes no se puede hacer "en paralelo".

1.2.3 Método de relajación

A continuación, introducimos el método de relajación para la solución de $Au = b$ de la forma:

$$u_i^{k+1} = u_i^k + \frac{w}{a_i i}(b_i - \sum_{j=1}^{n} a_{ij}u_j^{k+1}), \quad 1 \leq i \leq n. \tag{1.14}$$

Con la notación anterior consideramos la siguiente descomposición de D:

$$D = \frac{1}{w}D + (1 - \frac{1}{w})D, \quad con \quad w \in \mathbb{R} \setminus \{0\}.$$

De esta forma, podemos escribir A como:

$$A = \frac{1}{w}D - E - \Big(\frac{1-w}{w}\Big)D - F$$

y se pueden tomar

$$M = \frac{1}{w}D - E, \quad N = \Big(\frac{1-w}{w}\Big)D + F,$$

de modo que se ha pasado parte de la diagonal D de A a la matriz N.

La matriz M es invertible gracias a la hipótesis (1.8). Por tanto, el sistema lineal (1.1) es equivalente a (1.10) y podemos plantear el correspondiente método iterativo. Este tiene la forma:

$$\begin{cases} u^0 \in \mathbb{K}^n \quad dado, \\ u^{k+1} = \Big(\frac{1}{w}D - E\Big)^{-1}\Big[\Big(\frac{1-w}{w}D + F\Big)u^k + b\Big], \quad \forall k \geq 0 \end{cases}$$

Se llamará matriz de relajación a la matriz dada por

$$\mathcal{L}_w = \Big(\frac{1}{w}D - E\Big)^{-1}\Big(\frac{1-w}{w}D + F\Big) = (D - wE)^{-1}[(1-w)D + wF].$$

Con esta notación y, como consecuencia del Teorema 1.1, se tiene:

Corolario 1.6. Sean $A \in \mathcal{L}(\mathbb{K}^n)$, una matriz satisfaciendo $detA \neq 0$ y (1.8), $b \in \mathbb{K}^n$ y $w \in \mathbb{R} \setminus \{0\}$. Entonces, el método de relajación es globalmente convergente si y solo si $\rho(\mathcal{L}_w) < 1$.

El cálculo efectivo que se lleva a cabo es el siguiente:

$$\Big(\frac{1}{w}D - E\Big)u^{k+1} = \Big(\frac{1-w}{w}D + F\Big)u^k + b \iff (D - wE)u^{k+1} = ((1-w)D + wF)u^k + wb.$$

De manera equivalente

$$Du^{k+1} = Du^k + w[Eu^{k+1} - (D - F)u^k + b].$$

En notación vectorial, la igualdad anterior proporciona:

$$\begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix}\begin{pmatrix} u_1^{k+1} \\ u_2^{k+1} \\ \vdots \\ u_n^{k+1} \end{pmatrix} = \begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix}\begin{pmatrix} u_1^{k} \\ u_2^{k} \\ \vdots \\ u_n^{k} \end{pmatrix}$$

$$+w\left[\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} - \begin{pmatrix} 0 & 0 & \cdots & 0 \\ a_{21} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & 0 \end{pmatrix}\begin{pmatrix} u_1^{k+1} \\ u_2^{k+1} \\ \vdots \\ u_n^{k+1} \end{pmatrix} - \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix}\begin{pmatrix} u_1^{k} \\ u_2^{k} \\ \vdots \\ u_n^{k} \end{pmatrix}\right].$$

Así pues, la primera componente viene dada por:

$$u_1^{k+1} = u_1^k + \frac{w}{a_{11}}[b_1 - (a_{11}u_1^k + ... + a_{1n}u_n^k)].$$

y, una vez hallados u_j^{k+1} para j < i, se determina

$$u_i^{k+1} = u_i^k + \frac{w}{a_{ii}}[b_i - (a_{i1}u_1^{k+1} + ... + a_{i,i-1}u_{i-1}^{k+1} + a_{ii}u_i^k + a_{i,i+1}u_{i+1}^k + ... + a_{in}u_n^k)].$$

Observación 1.6. 1) El método de relajación para w = 1 coincide con el de Gauss-Seidel. De ahí la notación usada para la matriz de Gauss-Seidel. El cálculo efectivo de una etapa es similar a Gauss-Seidel, para i = 1,...,n, se calcula

$$u_i \leftarrow u_i + \frac{w}{a_{ii}}(b_i - (Au)_i).$$

2) Aunque en principio el parámetro w podría ser un número real no nulo, para que el método converja, es necesario que $w \in (0, 2)$. El método se llamará de sobrerrelajación si $w \in (1, 2)$ y de subrrelajación si $w \in (0.1)$.

3) Si $\rho(\mathcal{L}_w)$ es una función continua de w, entonces, el estudio del método consiste en

a) Determinar un intervalo $I \subset \mathbb{R} \setminus \{0\}$, tal que se tenga $\rho(\mathcal{L}_w) < 1$.
b) Determinar $w_0 \in I$ tal que

$$\rho(\mathcal{L}_{w_0}) \approx inf_{w \in I}\rho(\mathcal{L}_w).$$

4) Para ciertos valores del parámetro de relajación se obtiene una convergencia más rápida que para $w = 1$ y por tanto un tiempo de cálculo menor que para el método de Gauss-Seidel. El número de operaciones es similar en ambos métodos. No obstante, hay que tener en cuenta el tiempo utilizado en la estimación preliminar del parámetro w_0 para comparar la eficacia de ambos métodos.

Capítulo 2

Métodos de proyección para la resolución de sistemas lineales

La mayoría de los métodos iterativos existentes para resolver grandes sistemas de ecuaciones lineales utilizan un proceso de proyección. Este proceso representa una forma canónica de extraer una aproximación de la solución de un sistema lineal a partir de un subespacio. Este capítulo describe estas técnicas en un marco muy general. El caso unidimensional está cubierto en detalle al final del capítulo, que proporciona una vista previa de los métodos de proyección más complejos que veremos en el capítulo 3.

2.1 Definiciones básicas y algoritmos

Comenzamos con algunas definiciones y algoritmos necesarios para describir los métodos de proyección.

Consideramos el sistema lineal

$$Ax = b, \tag{2.1}$$

donde A es una matriz real nxn. La idea de los métodos de proyección es extraer una solución aproximada del problema anterior para un subespacio de $\mathbb{R}^n$. Si $\mathcal{K}$ es este subespacio de candidatos aproximados, también llamado el subespacio buscado, y si m es su dimensión, entonces, las restricciones de m deben imponerse para poder extraer dicha aproximación. Un camino típico para describir estas restricciones es imponer a m (independiente) condiciones ortogonales. Especialmente, el vector residual $b - Ax$ es restringido para ser ortogonal a m vectores linealmente independientes. Esto define otro subespacio $\mathcal{L}$ de dimensión m, que llamaremos el subespacio de restricciones.

Hay dos clases de métodos de proyección: ortogonal y oblicua. En la proyección ortogonal, el subespacio $\mathcal{L}$ es el mismo que $\mathcal{K}$. En el método de proyección oblicua, $\mathcal{L}$ es diferente de $\mathcal{K}$ y puede no estar totalmente relacionado con esto. Esta distinción es importante y da lugar a diferentes tipos de algoritmos [7].

2.1.1 Métodos generales de proyección

Sea A una matriz real nxn y $\mathcal{K}$ y $\mathcal{L}$ dos subespacios m-dimensional de $\mathbb{R}^n$. Una técnica de proyección sobre el subespacio $\mathcal{K}$ y ortogonal a $\mathcal{L}$ es el proceso que busca una solución aproximada $\tilde{x}$ para (2.1) imponiendo las condiciones:

$$Hallar\ \tilde{x} \in \mathcal{K}\ tal\ que\ b - A\tilde{x} \perp \mathcal{L}. \tag{2.2}$$

Si deseamos aprovechar el conocimiento de una suposición inicial x_0 de la solución, luego la aproximación debe buscarse en el espacio afín $x_0 + \mathcal{K}$ en lugar del espacio vectorial homogéneo $\mathcal{K}$. Esto requiere una leve modificación en la formulación anterior,

$$Hallar\ \tilde{x} \in x_0 + \mathcal{K}\ tal\ que\ b - A\tilde{x} \perp \mathcal{L}. \tag{2.3}$$

Notar que, si escribimos $\tilde{x}$ de la forma $\tilde{x} = x_0 + \delta$ y el vector residual inicial r_0 es definido como

$$r_0 = b - Ax_0, \tag{2.4}$$

luego la ecuación anterior se convierte en $b - A(x_0 + \delta) \perp \mathcal{L}$ ó $r_0 - A\delta \perp \mathcal{L}$.
En otras palabras, la solución aproximada puede ser definida como

$$\tilde{x} = x_0 + \delta, \quad \delta \in \mathcal{K} \tag{2.5}$$

$$(r_0 - A\delta, w) = 0, \quad \forall w \in \mathcal{L} \tag{2.6}$$

Esto es un paso básico de proyección, en la forma más general. La mayoría de las técnicas estándar utilizan una sucesión de tales proyecciones. Normalmente, un nuevo paso de proyección usa un nuevo par de subespacios $\mathcal{K}$ y $\mathcal{L}$ y una conjetura inicial X_0 igual a la aproximación más reciente obtenida para los pasos de proyección previos. Los métodos de proyección forman una estructura unificadora para muchos de los métodos conocidos en el cálculo científico. De hecho, practicamente todas las técnicas iterativas básicas vistas en el capítulo 1 pueden ser consideradas técnicas de proyección. Cuando una aproximación es definida a través de m grados de libertad (subespacio $\mathcal{K}$) y m restricciones ($\mathcal{L}$) resulta un proceso de proyección.

Los métodos de proyección ortogonales corresponden al caso particular en el que los dos subespacios $\mathcal{K}$ y $\mathcal{L}$ son idénticos. La diferencia es particularmente importante en el caso Hermitiano ya que se garantiza que el problema de proyección será Hermitiano en esta situación. Además, muchos resultados teóricos son ciertos para el caso ortogonal. Cuando $\mathcal{L} = \mathcal{K}$, las condiciones de Petrov-Galerkin, que definiremos más adelante (Proposición 2.3), son llamadas a menudo las condiciones de Galerkin.

2.1.2 Representación matricial

Sea $V = [v_1, ..., v_m]$, una matriz nxm cuyos vectores columna forman una base de $\mathcal{K}$, y sea $W = [w_1, ..., w_m]$, una matriz nxm cuyos vectores columna forman una base de $\mathcal{L}$. Si la solución aproximada es escrita como

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

luego la condición de ortogonalidad conduce inmediatamente al siguiente sistema de ecuaciones para el vector y:

$$W^T AV y = W^T r_0.$$

Si se hace la suposición de que la matriz *mxm*, $W^T AV$ es no singular, la expresión para la solución aproximada resulta ser:

$$\tilde{x} = x_0 + V(W^T AV)^{-1} W^T r_0 \tag{2.7}$$

En algunos algoritmos, la matriz $W^T AV$ no es necesaria calcularla ya que la obtenemos como subproducto del algoritmo. Un pseudo-código de la técnica de proyección es representado por el siguiente algoritmo.

ALGORITMO 2.1. Pseudo-código de la técnica de proyección.

1. Seleccionamos un par de subespacios $\mathcal{K}$ y $\mathcal{L}$.
2. Elegimos las bases $V = [v_1, ..., v_m]$ y $W = [w_1, ..., w_m]$ para $\mathcal{K}$ y $\mathcal{L}$
3. $r := b - Ax$
4. $y := (W^T AV)^{-1} W^T r$
5. $x := x + Vy$

La solución aproximada está definida solo cuando la matriz $W^T AV$ es no singular, una propiedad que no se garantiza que sea verdadera incluso cuando A es no singular.

Ejemplo 2.1. Consideramos la matriz

$$A = \begin{pmatrix} O & I \\ I & I \end{pmatrix}$$

donde I es la matriz identidad *mxm* y O es la matriz cero *mxm*, y sea $V = W = [e_1, e_2, ..., e_m]$ la base canónica. Aunque A es no singular, la matriz $W^T AV$ es precisamente el bloque O en la esquina superior izquierda de A y es entonces singular.

Puede ser fácilmente verificado que $W^T AV$ es no singular si ningún vector del subespacio $A\mathcal{K}$ es ortogonal al subespacio $\mathcal{L}$. Hay dos casos particulares importantes donde la no singularidad de $W^T AV$ está garantizada. Estos son discutidos en la siguiente proposición.

Proposición 2.1. Sean A, $\mathcal{L}$ y $\mathcal{K}$ satisfaciendo cualquiera de las dos condiciones siguientes:

(i) A es definida positiva y $\mathcal{L} = \mathcal{K}$
(ii) A es no singular y $\mathcal{L} = A\mathcal{K}$.

Luego la matriz $B = W^T AV$ es no singular para cualquier base V y W de $\mathcal{K}$ y $\mathcal{L}$, respectivamente.

Demostración. Consideramos el primer caso (i). Sea V alguna base de $\mathcal{K}$ y sea W alguna base de $\mathcal{L}$. De hecho, ya que $\mathcal{L}$ y $\mathcal{K}$ son iguales, W puede ser siempre expresada como $W = VG$, donde G es una matriz no singular *mxm*. Luego

$$B = W^T AV = G^T V^T AV.$$

Como A es definida positiva, también lo es $V^T AV$, que nos demuestra que B es no singular. Consideramos ahora el caso (ii). Sea V alguna base de $\mathcal{K}$ y sea W alguna base de $\mathcal{L}$. Ya que $\mathcal{L} = A\mathcal{K}$, W puede ser expresado en este caso como $W = AVG$, donde G es una matriz mxm no singular. Luego

$$B = W^T AV = G^T (AV)^T AV \tag{2.8}$$

Como A es no singular, la matriz nxm, AV es de rango completo y, como consecuencia, $(AV)^T AV$ es no singular. Esto, junto con (2.8), demuestra que B es no singular. □

Ahora consideramos el caso particular donde A es simétrica (real) y se utiliza una técnica de proyección ortogonal. En esta situación, las mismas bases pueden ser usadas para $\mathcal{L}$ y $\mathcal{K}$, que son subespacios idénticos, y la matriz de proyección, que es $B = V^T AV$, es simétrica. Además, si la matriz A es simétrica definida positiva (SPD), entonces también lo es B.

2.2 Teoría general

En esta sección daremos algunos resultados generales de teoría sin ser específicos sobre los subespacios $\mathcal{L}$ y $\mathcal{K}$ que se utilizan. El objetivo es aprender sobre la clase de la aproximación obtenida de un proceso general de proyección. Dos herramientas principales se utilizan para ello. La primera sirve para usar de manera óptima las propiedades de los métodos de proyección. La segunda consiste en interpretar el problema de proyección con la ayuda de los operadores de proyección en un intento de extraer límites residuales [9].

2.2.1 Dos resultados de optimalidad

En esta sección definimos dos resultados importantes de optimalidad considerarando el caso cuando A es (SPD).

Proposición 2.2. Suponiendo que A es SPD y $\mathcal{L} = \mathcal{K}$, entonces un vector $\tilde{x}$ es el resultado de un método de proyección (ortogonal) sobre $\mathcal{K}$ con el vector inicial x_0 si se minimiza la norma de A del error sobre $x_0 + \mathcal{K}$, es decir, si

$$E(\tilde{x}) = min_{x \in x_0 + \mathcal{K}} E(x),$$

donde

$$E(x) = (A(x_* - x), x_* - x)^{1/2}.$$

siendo x_* la solución exacta.

Demostración. Para que $\tilde{x}$ sea el mínimo de $E(x)$, es necesario y suficiente que $x_* - \tilde{x}$ sea A ortogonal para todo el subespacio $\mathcal{K}$. Esto produce

$$(A(x_* - \tilde{x}), v) = 0 \quad \forall v \in \mathcal{K}$$

o, equivalentemente,

$$(b - A\tilde{x}, v) = 0 \qquad \forall v \in \mathcal{K}$$

que es la condición de Galerkin definiendo un proceso de proyección ortogonal para la aproximación $\tilde{x}$. □

Ahora tomamos el caso donde $\mathcal{L}$ es definido por $\mathcal{L} = A\mathcal{K}$.

Proposición 2.3. Sea A una matriz cuadrada arbitraria y suponiendo que $\mathcal{L} = A\mathcal{K}$. Entonces, un vector $\tilde{x}$ es el resultado de un método de proyección (oblicuo) sobre $\mathcal{K}$ ortogonalmente a $\mathcal{L}$ con el vector inicial x_0 si se minimiza la norma-2 del vector residual $b - Ax$ sobre $x \in x_0 + \mathcal{K}$, es decir, si

$$R(\tilde{x}) = min_{x \in x_0 + \mathcal{K}} R(x),$$

donde $R(x) = \|b - Ax\|_2$.

Demostración. Para que $\tilde{x}$ sea el mínimo de $R(x)$, es necesario y suficiente que $b - A\tilde{x}$ sea ortogonal a todos los vectores de la forma $v = Ay$, donde y pertenece a $\mathcal{K}$, es decir,

$$(b - A\tilde{x}, v) = 0 \qquad \forall v \in A\mathcal{K},$$

que es precisamente la condición de Petrov-Galerkin que define la solución aproximada $\tilde{x}$. □

2.2.2 Interpretación en términos de proyectores

Volvemos ahora a los dos casos particulares importantes señalados en la sección anterior, es decir, los casos $\mathcal{L} = \mathcal{K}$ y $\mathcal{L} = A\mathcal{K}$. En estos casos, el resultado del proceso de proyección puede ser interpretado fácilmente en cuanto a las acciones de las proyecciones ortogonales sobre el residual inicial o error inicial. Consideremos primero el segundo caso, ya que es más simple. Sea r_o el residuo inicial $r_0 = b - Ax_0$ y $\tilde{r} = b - A\tilde{x}$ el residuo obtenido después del proceso de proyección con $\mathcal{L} = A\mathcal{K}$. Entonces

$$\tilde{r} = b - A(x_0 + \delta) = r_0 - A\delta. \tag{2.9}$$

Además, δ es obtenido imponiendo la condición de que $r_0 - A\delta$ sea ortogonal a $A\mathcal{K}$. Por lo tanto, el vector $A\delta$ *es la proyección ortogonal* del vector r_0 sobre el subespacio $A\mathcal{K}$.

Proposición 2.4. Sea $\tilde{x}$ la solución aproximada obtenida de un proceso de proyección sobre $\mathcal{K}$ ortogonal a $\mathcal{L} = A\mathcal{K}$ y sea $\tilde{r} = b - A\tilde{x}$ el residuo asociado. Entonces

$$\tilde{r} = (I - \mathbf{P})r_0, \tag{2.10}$$

donde $\mathbf{P}$ denota la proyección ortogonal sobre el subespacio $A\mathcal{K}$.

Un resultado de la proposición 2.4 es que la norma-2 del vector residual obtenido después de un paso de proyección no excederá la norma-2 inicial del residuo, es decir,

$$\|\tilde{r}\| \leq \|r_0\|_2.$$

Esta clase de métodos puede denominarse *métodos de proyección del residuo.*

Ahora consideramos el caso donde $\mathcal{L} = \mathcal{K}$ y A es SPD. Sea $d_0 = x_* - x_0$ el error inicial, donde x_* denota la solución exacta del sistema, y, análogamente, sea $\tilde{d} = x_* - \tilde{x}$, donde $\tilde{x} = x_0 + \delta$ es la solución aproximada obtenida de un paso de proyección. Luego (2.9) nos da la relación

$$A\tilde{d} = \tilde{r} = A(d_0 - \delta),$$

donde δ es ahora obtenida al restringir el vector residual $r_0 - A\delta$ para que sea ortogonal a $\mathcal{K}$:

$$(r_0 - A\delta, w) = 0 \qquad \forall w \in \mathcal{K}.$$

La condición anterior es equivalente a

$$(A(d_0 - \delta), w) = 0 \qquad \forall w \in \mathcal{K}.$$

Como A es SPD, define un producto interno que es usualmente denotado por $(\cdot, \cdot)_A$, y la condición anterior se convierte en

$$(d_0 - \delta, w)_A = 0 \qquad \forall w \in \mathcal{K}.$$

La condición anterior es ahora más fácil de interpretar: *El vector δ es la proyección ortogonal A del error inicial d_0 sobre el subespacio $\mathcal{K}$.*

Proposición 2.5. Sea $\tilde{x}$ la solución aproximada obtenida de un proceso de proyección ortogonal sobre $\mathcal{K}$ y sea $\tilde{d} = x_* - \tilde{x}$ el vector error asociado. Entonces

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

donde P_A denota la proyección sobre el subespacio $\mathcal{K}$, que es ortogonal con respecto al producto interno A.

Un resultado de la proposición es que la norma A del vector error obtenida después de un paso de la proyección no excede la norma A inicial del error, es decir,

$$\|\tilde{d}\|_A \leq \|d_0\|_A,$$

que es lo esperado ya que se sabe que la norma A del error es minimizada en $x_0 + \mathcal{K}$. Esta clase de métodos pueden denominarse *métodos de proyección de errores.*

2.2.3 Límite de error general

Si ningún vector del subespacio $\mathcal{K}$ se acerca a la solución exacta x, entonces es imposible encontrar una buena aproximación $\tilde{x}$ a x de $\mathcal{K}$. Por lo tanto, la aproximación obtenida por algún proceso de proyección basada en $\mathcal{K}$ será pobre. Por otro lado, si hay algún vector en $\mathcal{K}$ que es una distancia pequeña ϵ lejos de x, entonces la pregunta es la siguiente: ¿Cómo de buena puede ser la solución aproximada? El objetivo de esta sección es intentar contestar esta pregunta.

Sea $\mathcal{P}_\mathcal{K}$ la proyección ortogonal sobre el subespacio $\mathcal{K}$ y sea $\mathcal{Q}_\mathcal{K}^\mathcal{L}$ la proyección (oblicua) sobre $\mathcal{K}$ y ortogonal a $\mathcal{L}$. Estas proyecciones están definidas como

$$\mathcal{P}_\mathcal{K} x \in \mathcal{K}, \;\; x - \mathcal{P}_\mathcal{K} x \perp \mathcal{K},$$

$$\mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} x \in \mathcal{K},\ x - \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} \perp \mathcal{L}.$$

El símbolo A_m es usado a continuación para denotar el operador

$$A_m = \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A \mathcal{P}_{\mathcal{K}},$$

y se asume, sin pérdida de generalidad, que $x_0 = 0$. Entonces el problema aproximado definido en (2.5)-(2.6) puede ser reformulado como: Hallar $\tilde{x} \in \mathcal{K}$ tal que

$$\mathcal{Q}_{\mathcal{K}}^{\mathcal{L}}(b - A\tilde{x}) = 0,$$

o equivalentemente,

$$A_m \tilde{x} = \mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} b,\ \tilde{x} \in K.$$

Por lo tanto, un sistema lineal n-dimensional es aproximado por uno m-dimensional.
La siguiente proposición expone que ocurre en el caso particular cuando el subespacio $\mathcal{K}$ es invariante bajo A.

Proposición 2.6. Suponiendo que $\mathcal{K}$ es invariante bajo A, $x_0 = 0$, y b pertenece a $\mathcal{K}$. Entonces la solución aproximada obtenida de algún método de proyección sobre $\mathcal{K}$ (oblicuo u ortogonal) es exacta.

Demostración. Una solución aproximada $\tilde{x}$ es definida por

$$\mathcal{Q}_{\mathcal{K}}^{\mathcal{L}}(b - A\tilde{x}) = 0,$$

donde $\tilde{x}$ es un vector distinto de cero en $\mathcal{K}$. El lado derecho b está en $\mathcal{K}$, así que tomamos $\mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} b = b$. Análogamente, $\tilde{x}$ pertenece a $\mathcal{K}$, que es invariante bajo A y, por lo tanto, $\mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A\tilde{x} = A\tilde{x}$. Entonces la ecuación anterior se convierte en

$$b - A\tilde{x} = 0,$$

demostrando que $\tilde{x}$ es la solución exacta. □

El resultado puede ser extendido al caso donde $x_0 \neq 0$. La suposición requerida en este caso es que el residuo inicial $r_0 = b - Ax_0$ pertenece al subespacio invariante $\mathcal{K}$.

Una cantidad importante para las propiedades de convergencia de los métodos de proyección es la distancia $\|(I - \mathcal{P}_{\mathcal{K}})x_*\|_2$ de la solución exacta x_* del subespacio $\mathcal{K}$. Esta cantidad juega un papel clave en el análisis de los métodos de proyección. Señalar que la solución x_* no puede estar bien aproximada en $\mathcal{K}$ si $\|(I - \mathcal{P}_{\mathcal{K}})x_*\|_2$ no es pequeña porque

$$\|\tilde{x} - x_*\|_2 \geq \|(I - \mathcal{P}_{\mathcal{K}})x_*\|_2.$$

La cantidad fundamental $\|(I - \mathcal{P}_{\mathcal{K}})x_*\|_2 / \|x_*\|_2$ es el seno del ángulo agudo entre la solución x_* y el subespacio $\mathcal{K}$. El siguiente teorema establece un límite superior para la norma residual de la solución exacta con respecto al operador aproximado A_m.

Teorema 2.7. Sea $\gamma = \|\mathcal{Q}_{\mathcal{K}}^{\mathcal{L}} A(I - \mathcal{P}_{\mathcal{K}})\|_2$ y suponiendo que b es un miembro de $\mathcal{K}$ y $x_0 = 0$. Entonces la solución exacta x_* del problema original es tal que

$$\|b - A_m x_*\|_2 \leq \gamma \|(I - \mathcal{P}_{\mathcal{K}})x_*\|_2 \tag{2.11}$$

Demostración. Como $b \in \mathcal{K}$, entonces

$$\begin{aligned} b - A_m x_* &= Q_{\mathcal{K}}^{\mathcal{L}}(b - A\mathcal{P}_{\mathcal{K}} x_*) \\ &= Q_{\mathcal{K}}^{\mathcal{L}}(Ax_* - A\mathcal{P}_{\mathcal{K}} x_*) \\ &= Q_{\mathcal{K}}^{\mathcal{L}} A(x_* - \mathcal{P}_{\mathcal{K}} x_*) \\ &= Q_{\mathcal{K}}^{\mathcal{L}} A(I - \mathcal{P}_{\mathcal{K}}) x_*. \end{aligned}$$

Observando que $I - \mathcal{P}_{\mathcal{K}}$ es un proyector, se deduce que

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

lo que completa la prueba. □

Observación: Es útil considerar una interpretación matricial de los teoremas. Consideramos solo el caso particular de los métodos de proyección ortogonal ($\mathcal{L} = \mathcal{K}$). Suponemos que V es unitario, es decir, que la base $\{v_1, ..., v_m\}$ es ortonormal, y que $W = V$. Observar que $b = VV^T b$. La ecuación (2.11) puede ser representada en la base V como

$$\|b - V(V^T AV)V^T x_*\|_2 \leq \gamma \|(I - \mathcal{P}_{\mathcal{K}}) x_*\|_2.$$

Sin embargo,

$$\begin{aligned} \|b - V(V^T AV)V^T x_*\|_2 &= \|V(V^T b - (V^T AV))V^T x_*\|_2 \\ &= \|V^T b - (V^T AV)V^T x_*\|_2. \end{aligned}$$

Así, la proyección de la solución exacta tiene una norma residual con respecto a la matriz $B = V^T AV$, que es del orden de $\|(I - \mathcal{P}_{\mathcal{K}}) x_*\|_2$.

2.3 Procesos de proyección unidimensional

Esta sección expone los ejemplos proporcionados por los procesos de proyección unidimensional. En lo que sigue, el vector r denota el vector residual $r = b - Ax$ para la aproximación actual x. Para evitar subíndices, se utiliza la notación de flecha para denotar actualizaciones de vectores. Así, "$x \leftarrow$x+$\alpha\ r$" significa "calcular $x + \alpha\ r$ y sobreescribir el resultado en la x actual".

Los procesos de proyección unidimensional son definidos cuando

$$\mathcal{K} = span\{v\} \qquad y \qquad \mathcal{L} = span\{w\},$$

donde v y w son dos vectores. En este caso, la nueva aproximación coge la forma x $\leftarrow$ x+αv y la condición de Petrov-Galerkin $r - A\delta \perp w$ produce

$$\alpha = \frac{(r, w)}{(Av, w)}. \tag{2.12}$$

Las siguientes son tres opciones conocidas de proyección unidimensional a considerar.

2.3.1 Descenso más rápido

El algoritmo del descenso más rápido es definido para el caso donde la matriz A es SPD. Consiste en tomar en cada paso $v = r$ y $w = r$. Esto produce el siguiente procedimiento iterativo:

$$r \leftarrow b - Ax$$

$$\alpha \leftarrow (r, r)/(Ar, r)$$

$$x \leftarrow x + \alpha r.$$

Sin embargo, el procedimiento anterior requiere dos productos de matriz por vector, que pueden ser reducido a uno solo reorganizando ligeramente el cálculo. La variación consiste en calcular r diferentemente, como se muestra a continuación.

ALGORITMO 2.2. Algoritmo del descenso más rápido.

1. Calcular $r = b - Ax$ y $p = Ar$
2. $\alpha \leftarrow (r, r)/(p, r)$
3. $x \leftarrow x + \alpha r$
4. $r \leftarrow r - \alpha p$
5. Calcular p:=Ar

Cada paso de la iteración anterior minimiza

$$f(x) = \|x - x_*\|_A^2 = (A(x - x_*), (x - x_*)),$$

sobre todos los vectores de la forma $x+\alpha d$, donde d es el negativo de la dirección del gradiente $-\nabla f$. El negativo de la dirección del gradiente es localmente la dirección que produce la mayor velocidad de disminución para f. A continuación, probamos que la convergencia está garantizada cuando A es SPD. El resultado es una consecuencia del lema siguiente conocido como la desigualdad de Kantorovich [10].

Lema 2.8. (Desigualdad de Kantorovich). Sea B alguna matriz real SPD y λ_{max}, λ_{min} su mayor y menor autovalor, respectivamente. Entonces

$$\frac{(Bx, x)(B^{-1}x, x)}{(x, x)^2} \leq \frac{(\lambda_{max} + \lambda_{min})^2}{4\lambda_{max}\lambda_{min}} \quad \forall x \neq 0. \tag{2.13}$$

Demostración. Demostrar (2.13) es quivalente a demostrar que el resultado es verdadero para cualquier vector unitario x. Como B es simétrica, esto es unitariamente similar a la matriz diagonal, $B = Q^T DQ$, y

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

Estableciendo $y = Qx = (y_1, ..., y_n)^T$ y $\beta_i = y_i^2$, indicar que

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

es una combinación convexa de los autovalores λ_i, $i = 1, .., n$. Se cumple la siguiente relación:

$$(Bx, x)(B^{-1}x, x) = \lambda\psi(y), \quad con \;\; \psi(y) = (D^{-1}y, y) = \sum_{i=1}^{n} \beta_i \frac{1}{\lambda_i}.$$

Señalar que la función $f(\lambda) = 1/\lambda$ es convexa, $\psi(y)$ está delimitada por arriba por la curva lineal que une los puntos $(\lambda_1, 1/\lambda_1)$ y $(\lambda_n, 1/\lambda_n)$; es decir,

$$\psi(y) \leq \frac{1}{\lambda_1} + \frac{1}{\lambda_n} - \frac{\lambda}{\lambda_1\lambda_n}.$$

Por lo tanto,

$$(Bx, x)(B^{-1}x, x) = \lambda\psi(y) \leq \lambda(\frac{1}{\lambda_1} + \frac{1}{\lambda_n} - \frac{\lambda}{\lambda_1\lambda_n}).$$

El máximo del lado derecho es alcanzado por $\lambda = \frac{1}{2}(\lambda_1 + \lambda_n)$, produciendo

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

que da el resultado deseado. □

Este lema ayuda a establecer el resultado siguiente con respecto a la velocidad de convergencia del método.

Teorema 2.9. Sea A una matriz SPD. Entonces las normas A de los vectores de error $d_k = x_* - x_k$ generados por el algoritmo 2.2 satisfacen la relación

$$\|d_{k+1}\|_A \leq \frac{\lambda_{max} - \lambda_{min}}{\lambda_{max} + \lambda_{min}} \|d_k\|_A, \tag{2.14}$$

y el algoritmo 2.2 (descenso más rápido) converge para alguna estimación inicial x_0.

Demostración. Empezamos por expandir el cuadrado de la norma A de $d_{k+1} = d_k - \alpha_k r_k$ como

$$\|d_{k+1}\|_A^2 = (d_{k+1}, d_k - \alpha_k r_k)_A = (d_{k+1}, d_k)_A - \alpha_k (d_{k+1}, r_k)_A = (d_{k+1}, r_k).$$

La última igualdad es debido a la ortogonalización entre r_k y r_{k+1}. Así,

$$\begin{aligned} \|d_{k+1}\|_A^2 &= (d_k - \alpha r_k, r_k) \\ &= (A^{-1}r_k, r_k) - \alpha_k (r_k, r_k) \\ &= \|d_k\|_A^2 \left(1 - \frac{(r_k, r_k)}{(r_k, Ar_k)} \times \frac{(r_k, r_k)}{(r_k, A^{-1}r_k} \right). \end{aligned}$$

El resultado se obtiene aplicando la desigualdad de Kantorovich (2.13). □

2.3.2 Iteración MR

Ahora asumimos que A es no necesariamente simétrica pero solo definida positiva, es decir, su parte simétrica $A + A^T$ es SPD. Tomando en cada paso $v = r$ y $w = Ar$ obtenemos el siguiente procedimiento:

$$r \leftarrow b - Ax,$$

$$\alpha \leftarrow (Ar, r)/(Ar, Ar),$$

$$x \leftarrow x + \alpha r.$$

Este procedimiento puede ser ligeramente reordenado de nuevo para reducir el número de productos vector por matriz requeridos solo uno por paso, como se hizo en el algoritmo del descenso más rápido. Este resultado es el algoritmo siguiente.

ALGORITMO 2.3. Iteración del residuo mínimo.

1. Calcular $r = b - Ax$ y $p = Ar$
2. $\alpha \leftarrow (p, r)/(p, p)$
3. $x \leftarrow x + \alpha r$
4. $r \leftarrow r - \alpha p$
5. Calcular p:=Ar

Aquí, cada paso minimiza $f(x) = \|b - Ax\|_2^2$ en la dirección r. La iteración converge bajo la condición de que A es definida positiva, como se indica en el siguiente teorema.

Teorema 2.10. Sea A una matriz real, definida positiva y sea

$$\mu = \lambda_{min}(A + A^T)/2, \qquad \sigma = \|A\|_2.$$

Entonces los vectores residuales generados por el algoritmo 2.3 (residuo mínimo) satisfacen la relación

$$\|r_{k+1}\|_2 \leq \left(1 - \frac{\mu^2}{\sigma^2}\right)\|r_k\|_2, \tag{2.15}$$

y el algoritmo 2.3 converge para cualquier estimación inicial x_0.

Demostración. Procedemos de manera similar al método del descenso más rápido, empezando con la relación

$$\|r_{k+1}\|2_2 = (r_k - \alpha_k A r_k, r_k - \alpha_k A r_k) \tag{2.16}$$

$$= (r_k - \alpha_k A r_k, r_k) - \alpha_k (r_k - \alpha_k A r_k, A r_k). \tag{2.17}$$

Por construcción, el nuevo vector residual $r_k - \alpha_k A r_k$ debe ser ortogonal a la dirección buscada Ar_k y, como consecuencia, el segundo término del lado derecho de la ecuación anterior desaparece y obtenemos

$$\|r_{k+1}\|_2^2 = (r_k - \alpha_k A r_k, r_k)$$

$$= (r_k, r_k) - \alpha_k (Ar_k, r_k)$$

$$= \|r_k\|_2^2 \left(1 - \frac{(Ar_k, r_k)}{(r_k, r_k)} \frac{(Ar_k, r_k)}{(Ar_k, Ar_k)} \right) \tag{2.18}$$

$$= \|r_k\|_2^2 \left(1 - \frac{(Ar_k, r_k)^2}{(r_k, r_k)^2} \frac{\|r_k\|_2^2}{\|Ar_k\|_2^2} \right).$$

Aplicando la desigualdad de Kantorovich (2.13) se puede afirmar que

$$\frac{(Ax, x)}{(x, x)} \geq \mu > 0 \tag{2.19}$$

donde $\mu = \lambda_{min}(A+A^T)/2$. El resultado deseado se obtiene usando la inecuación $\|Ar_k\|_2 \leq \|A\|_2 \|r_k\|_2$. □

2.3.3 Norma residual del descenso más rápido

En el algoritmo de la norma residual del descenso más rápido, la suposición de que A es definida positiva se relaja. De hecho, el único requisito es que A sea una matriz no singular. En cada paso, el algoritmo usa $v = A^T r$ y $w = Av$, dando la siguiente secuencia de operaciones:

$$r \leftarrow b - Ax, v = A^T r$$

$$\alpha \leftarrow \|v\|_2^2 / \|Av\|_2^2 \tag{2.20}$$

$$x \leftarrow x + \alpha v$$

Sin embargo, un algoritmo basado en la secuencia anterior requeriría tres productos vector por matriz, que es tres veces más que los otros algoritmos vistos en esta sección. El número de operaciones de vector por matriz puede reducirse de manera diferente. Una manera está representada por esta variante.

ALGORITMO 2.4. Norma residual del descenso más rápido.

1. Calcular $r = b - Ax$
2. $v := A^T r$
3. Calcular Av y $\alpha := \|v\|_2^2 / \|Av\|_2^2$
4. $x := x + \alpha v$
5. $r := r - \alpha Av$

En esta variante, cada paso minimiza $f(x) = \|b - Ax\|_2^2$ en la dirección $-\nabla f$. Como resultado, esto es equivalente al algoritmo del descenso más rápido de la sección 2.3.1 aplicado a las ecuaciones normales $A^T Ax = A^T b$. Como $A^T A$ es definida positiva cuando A es no singular, entonces, acorde con el Teorema 2.9, el método convergerá siempre que A sea no singular.

Capítulo 3

Métodos iterativos de Krylov

En este capítulo desarrollamos métodos que son considerados actualmente como los métodos iterativos más importantes disponibles para resolver grandes sistemas lineales. Estos métodos se basan en procesos de proyección, ortogonales u oblicuos, sobre subespacios de Krylov, que son subespacios con vectores de la forma $p(A)v$, donde p es un polinomio. Este capítulo describe métodos relacionados con la ortogonalización de Arnoldi y métodos basados en la biortogonalización de Lanczos.

3.1 Subespacio de Krylov.

Recordemos del capítulo anterior que un método de proyección general para resolver el sistema lineal

$$Ax = b, \tag{3.1}$$

extrae una solución aproximada x_m de un subespacio afín $x_0 + \mathcal{K}_m$ de dimensión m al imponer la condición de Petrov-Galerkin

$$b - Ax_m \perp \mathcal{L}_m,$$

donde $\mathcal{L}_m$ es otro subespacio de dimensión m. Aquí, x_0 representa una estimación inicial arbitraria de la solución.

En esta sección consideramos métodos de proyección sobre subespacios de Krylov [10], es decir, subespacios de la forma

$$\mathcal{K}_m = span\{v, Av, A^2v, ..., A^{m-1}v\}, \tag{3.2}$$

que denotaremos por $\mathcal{K}_m$. La dimensión del subespacio de aproximaciones aumenta en cada paso del proceso de aproximaciones. Algunas propiedades del subespacio de Krylov se pueden establecer. Una primera propiedad es que $\mathcal{K}_m$ es el subespacio de todos los vectores en $\mathbb{R}^n$ que pueden escribirse como $x = p(A)v$, donde p es un polinomio de grado no superior a m-1. Recordar que el polinomio mínimo de un vector v es el polinomio mónico distinto de cero p de menor grado tal que $p(A)v = 0$.

Hemos mencionado que la dimensión de $\mathcal{K}_m$ es creciente. En efecto, la siguiente proposición determina la dimensión de $\mathcal{K}_m$ en general.

Proposición 3.1. El subespacio de Krylov $\mathcal{K}_m$ es de dimensión m si y sólo si el grado μ de v con respecto a A no es menor que m, es decir,

$$dim(\mathcal{K}_m) = m \Leftrightarrow grado(v) \geq m. \tag{3.3}$$

Por lo tanto,

$$dim(\mathcal{K}_m) = min\{m, grado(v)\}. \tag{3.4}$$

Demostración. Los vectores $v, Av, ..., A^{m-1}v$ forman una base de $\mathcal{K}_m$ si y sólo si para cualquier conjunto de m escalares α_i, i = 0,...,m-1, donde al menos un α_i es distinto de cero, la combinación lineal $\sum_{i=0}^{m-1} \alpha_i A^i v$ es distinta de cero. Esto es equivalente a la condición de que el único polinomio de grado no superior a m-1 para el cual $p(A)v = 0$ es el polinomio cero. □

Dado un cierto subespacio X, recordemos que $A_{|X}$ denota la restricción de A a X. Si Q es un proyector para X, la sección del operador A en X es el operador para X sobre sí mismo definido por $QA_{|X}$. La siguiente proposición caracteriza el producto de polinomios de A con v términos de la sección de A en $\mathcal{K}_m$.

Proposición 3.2. Sea Q_m algún proyector sobre $\mathcal{K}_m$ y sea A_m la sección de A a K_m; que es, $A_m = Q_m A_{|\mathcal{K}_m}$. Luego, para algún polinomio q de grado no superior a m-1,

$$q(A)v = q(A_m)v,$$

y, para algún polinomio de grado no superior a m,

$$Q_m q(A)v = q(A_m)v.$$

Demostración. Primero probamos que $q(A)v = q(A_m)v$ para algún polinomio q de grado no superior a m-1. Esto es suficiente para probar la propiedad para los polinomios mónicos $q_i(t) \equiv t^i$, i = 0,...,m-1. La demostración es por inducción. La propiedad es cierta para el polinomio $q_0(t) \equiv 1$. Asumiendo que esto es cierto para $q_i(t) \equiv t^i$:

$$q_i(A)v = q_i(A_m)v.$$

Multiplicando la ecuación anterior por A en ambos lados tenemos

$$q_{i+1}(A)v = Aq_i(A_m)v.$$

Si $i + 1 \leq m - 1$ el vector en el lado izquierdo pertenece a $\mathcal{K}_m$ y, por lo tanto, si la ecuación anterior es multiplicada en ambos lados por Q_m, luego

$$q_{i+1}(A)v = Q_m A q_i(A_m)v.$$

Mirando el lado derecho observamos que $q_i(A_m)v$ pertenece a $\mathcal{K}_m$. Por tanto,

$$q_{i+1}(A)v = Q_m A_{|\mathcal{K}_m} q_i(A_m)v = q_{i+1}(A_m)v,$$

que prueba que la propiedad es cierta para i+1, siempre que $i + 1 \leq m - 1$. Para el caso, $i + 1 = m$, solo queda demostrar que $Q_m q_m(A)v = q_m(A_m)v$, que se sigue de $q_{m-1}(A)v = q_{m-1}(A_m)v$ simplemente multiplicando ambos lados por $Q_m A$. □

3.2 El método de Arnoldi

El método de Arnoldi es un método de proyección ortogonal sobre $\mathcal{K}_m$ para matrices no hermitianas generales. El procedimiento de Arnoldi es un algoritmo para construir una base ortogonal del subespacio de Krylov $\mathcal{K}_m$. En la aritmética exacta, una variante del algoritmo es la siguiente.

ALGORITMO 3.1. Arnoldi

1. Elegir un vector v_1 tal que $\|v_1\|_2 = 1$
2. Para $j = 1, 2, ..., m$
3. Calcular $h_{ij} = (Av_j, v_i)$ para i = 1,2,...,j
4. Calcular $w_j := Av_j - \sum_{i=1}^{j} h_{ij} v_i$
5. $h_{j+1,j} = \|w_j\|_2$
6. Si $h_{j+1,j} = 0$ entonces parar
7. $v_{j+1} = w_j / h_{j+1,j}$

En cada paso, el algoritmo multiplica el vector de Arnoldi previo v_j por A y luego ortonormaliza el vector resultante w_j contra todos los anteriores v_i por un procedimiento Gram-Schmidt estándar. Esto parará si el vector w_j calculado en la linea 4 es nulo. Ahora demostraremos algunas propiedades simples del algoritmo.

Proposición 3.3. Asumiendo que el algoritmo 3.1 no se detiene antes del m-ésimo paso. Entonces, los vectores $v_1, ..., v_m$ forman una base ortonormal del subespacio de Krylov

$$\mathcal{K}_m = span\{v_1, Av_1, ..., A^{m-1}v_1\}.$$

Demostración. Los vectores v_j, j = 1,2,...,m son ortonormales por construcción. $\mathcal{K}_m$ se obtiene del hecho de que cada vector v_j es de la forma $q_{j-1}(A)v_1$, donde q_{j-1} es un polinomio de grado j-1. Esto puede ser demostrado por inducción en j como sigue. El resultado es claramente cierto para j = 1, ya que $v_1 = q_0(A)v_1$, con $q_0(t) \equiv 1$. Asumir que el resultado es cierto para todos los enteros no superiores a j y considerar v_{j+1}. Tenemos

$$h_{j+1,j}v_{j+1} = Av_j - \sum_{i=1}^{j} h_{ij}v_i = Aq_{j-1}(A)v_1 - \sum_{i=1}^{j} h_{ij}q_{i-1}(A)v_1, \tag{3.5}$$

que demuestra que v_{j+1} puede ser expresada como $q_j(A)v_1$, donde q_j es de grado j, y se concluye la prueba. □

3.2.1 El método de Arnoldi para sistemas lineales

Dada una suposición inicial x_0 al sistema lineal original $Ax = b$, consideramos ahora un método de proyección ortogonal, como se define en el capítulo anterior, que toma $\mathcal{L} = \mathcal{K} = \mathcal{K}_m(A, r_0)$, con

$$\mathcal{K}_m(A, r_0) = span\{r_0, Ar_0, A^2r_0, ..., A^{m-1}r_0\}, \tag{3.6}$$

en el cuál, $r_0 = b - Ax_0$. Este método busca una solución aproximada x_m del subespacio afín $x_0 + \mathcal{K}_m$ de dimensión m imponiendo la condición de Galerkin

$$b - Ax_m \perp \mathcal{K}_m. \tag{3.7}$$

Si $v_1 = r_0/\|r_0\|_2$ en el método de Arnoldi y si $\beta = \|r_0\|_2$, entonces

$$V_m^T A V_m = H_m,$$

y

$$V_m^T r_0 = V_m^T(\beta v_1) = \beta e_1.$$

Como consecuencia, la solución aproximada usando los subespacios anteriores m-dimensionales es dada por

$$x_m = x_0 + V_m y_m, \tag{3.8}$$
$$y_m = Hm^{-1}(\beta e_1). \tag{3.9}$$

Un método basado en lo anterior y llamado método de ortogonalización completa es descrito a continuación. El método de Gram-Schmidt es usado en el procedimiento de Arnoldi.

ALGORITMO 3.2. Ortogonalización completa

1. Calcular $r_0 = b - Ax_0$, $\beta := \|r_0\|_2$ y $v_1 := r_0/\beta$
2. Definir la matriz m x m $H_m = \{h_{ij}\}_{i,j=1,...,m}$; Tomar $H_m = 0$
3. Para $j = 1, 2, ..., m$
4. Calcular $w_j := Av_j$
5. Para $i = 1, ..., j$
6. $h_{ij} = (w_j, v_i)$
7. $w_j := w_j - h_{ij}v_i$
8. Calcular $h_{j+1,j} = \|w_j\|_2$. Si $h_{j+1,j} = 0$, tomar $m := j$ e ir a 12
9. Calcular $v_{j+1} = w_j/h_{j+1,j}$
10. Calcular $y_m = H_m^{-1}(\beta e_1)$ y $x_m = x_0 + V_m y_m$.

El algoritmo anterior depende de un parámetro m que es la dimensión del subespacio de Krylov.

3.3 El método del residual mínimo generalizado (GMRES)

El método del residual mínimo generalizado es un método de proyección basado en tomar $\mathcal{K} = \mathcal{K}_m$ y $\mathcal{L} = A\mathcal{K}_m$, donde $\mathcal{K}_m$ es el m-ésimo subespacio de Krylov, con $v_1 = r_0/\|r_0\|_2$. Como hemos visto en el capítulo anterior, tal técnica minimiza la norma residual sobre todos los vectores en $x_0 + \mathcal{K}_m$.

La implementación de un algoritmo basado en esta propuesta es similar al algoritmo de ortogonalización completa.

Hay dos maneras de describir el algoritmo. La primera usa la propiedad de optimización y la relación $AV_m = V_{m+1}\bar{H}_m$. Cualquier vector x en $x_0 + \mathcal{K}_m$ puede ser escrito como

$$x = x_0 + V_m y, \tag{3.10}$$

donde y es un m-vector. Definiendo

$$J(y) = \|b - Ax\|_2 = \|b - A(x_0 + V_m y)\|_2, \tag{3.11}$$

obtenemos

$$\begin{aligned} b - Ax &= b - A(x_0 + V_m y) \\ &= r_0 - AV_m y \\ &= \beta v_1 - V_{m+1}\bar{H}_m y \\ &= V_{m+1}(\beta e_1 - \bar{H}_m y). \end{aligned}$$

Como los vectores columna de V_{m+1} son ortonormales, entonces

$$J(y) \equiv \|b - A(x_0 + V_m y)\|_2 = \|\beta e_1 - \bar{H}_m\|_2. \tag{3.12}$$

La aproximación de GMRES es el único vector de $x_0 + \mathcal{K}_m$ que minimiza (3.11). Por (3.10) y (3.12), esta aproximación puede ser obtenida de forma muy simple como $x_m = x_0 + V_m y_m$, donde y_m minimiza la función $J(y) = \|\beta e_1 - \bar{H}_m y\|_2$, es decir,

$$x_m = x_0 + V_m y_m, \quad donde \tag{3.13}$$

$$y_m = argmin_y \|\beta e_1 - \bar{H}_m y\|_2 \tag{3.14}$$

El minimizador y_m es fácil de calcular ya que requiere la solución de un problema de mínimos cuadrados (m+1) x m, donde m es típicamente pequeño. Esto nos devuelve el siguiente algoritmo:

ALGORITMO 3.3: GMRES

1. Calcular $r_0 = b - Ax_0$, $\beta := \|r_0\|_2$, y $v_1 := r_0/\beta$
2. Para j = 1,2,...,m
3. Calcular $w_j := Av_j$
4. Para i = 1,...,j
5. $h_{ij} := (w_j, v_i)$
6. $w_j := w_j - h_{ij}v_i$
7. $h_{j+1,j} = \|w_j\|_2$. Si $h_{j+1,j} = 0$, tomar $m := j$ e ir a 11
8. $v_{j+1} = w_j/h_{j+1,j}$
9. Definir la matriz (m+1) x m de Hessengerg $\bar{H}_m = \{h_{ij}\}_{1 \le i \le m+1, 1 \le j \le m}$
10. Calcular y_m, el minimizador de $\|\beta e_1 - \bar{H}_m y\|_2$, y $x_m = x_0 + V_m y_m$

3.3.1 Problemas de implementación práctica

Una clara dificultad de usar el algoritmo 3.3 es que no proporciona una solución aproximada x_m explícita en cada paso. Como resultado, no es fácil determinar cuando parar. Un remedio es calcular la solución aproximada x_m en intervalos regulares y comprobar la convergencia mediante una prueba sobre el residuo, por ejemplo. Sin embargo, hay una solución más elegante, que está relacionada con la forma en la que el problema de mínimos cuadrados (3.14) está resuelto.

Una técnica común para resolver el problema de mínimos cuadrados (3.14) es tránsformar la matriz de Hessenberg en una de forma triangular superior usando notaciones planas. Definimos las matrices de notación dadas,

$$\Omega_i = \begin{pmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & & & \\ & & & c_i & s_i & & \\ & & & -s_i & c_i & & \\ & & & & & 1 & \\ & & & & & & \ddots \\ & & & & & & 1 \end{pmatrix} \tag{3.15}$$

donde $c_i^2 + s_i^2 = 1$. Si los m pasos de la iteración GMRES son realizados, entonces estas matrices tienen dimensión $(m+1)x(m+1)$.

Se multiplica la matriz de Hessenberg $\bar{H}_m$ y el correspondiente lado derecho $\bar{g}_0 \equiv \beta e_1$ por una secuencia de tales matrices desde la izquierda. Los coeficientes s_i, c_i son seleccionadas para eliminar h_{i+1} en cada iteración. Por lo tanto, si $m = 5$, tendriamos

$$\bar{H}_5 = \begin{pmatrix} h_{11} & h_{12} & h_{13} & h_{14} & h_{15} \\ h_{21} & h_{22} & h_{23} & h_{24} & h_{25} \\ & h_{32} & h_{33} & h_{34} & h_{35} \\ & & h_{43} & h_{44} & h_{45} \\ & & & h_{54} & h_{55} \\ & & & & h_{65} \end{pmatrix}, \qquad \bar{g}_0 = \begin{pmatrix} \beta \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Luego, se multiplica $\bar{H}_5$ por

$$\Omega_1 = \begin{pmatrix} c_1 & s_1 & & & \\ -s_1 & c_1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 1 \end{pmatrix}$$

con

$$s_1 = \frac{h_{21}}{\sqrt{h_{11}^2 + h_{21}^2}}, \qquad c_1 = \frac{h_{11}}{\sqrt{h_{11}^2 + h_{21}^2}},$$

para obtener la matriz y el lado derecho

$$\bar{H}_5^{(1)} = \begin{pmatrix} h_{11}^{(1)} & h_{12}^{(1)} & h_{13}^{(1)} & h_{14}^{(1)} & h_{15}^{(1)} \\ & h_{22}^{(1)} & h_{23}^{(1)} & h_{24}^{(1)} & h_{25}^{(1)} \\ & h_{32} & h_{33} & h_{34} & h_{35} \\ & & h_{43} & h_{44} & h_{45} \\ & & & h_{54} & h_{55} \\ & & & & h_{65} \end{pmatrix}, \qquad \bar{g}_1 = \begin{pmatrix} c_1\beta \\ -s_1\beta \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{3.16}$$

Ahora podemos multiplicar la matriz anterior y el lado derecho de nuevo por una matriz de notación Ω_2 para eliminar h_{32}. Esto se consigue tomando

$$s_2 = \frac{h_{32}}{\sqrt{(h_{22}^{(1)})^2 + h_{32}^2}}, \qquad c_2 = \frac{h_{22}^{(1)}}{\sqrt{(h_{22}^{(1)})^2 + h_{32}^2}}.$$

Este proceso de eliminación continúa hasta que la m-ésima notación es aplicada, que transforma el problema en uno que involucra la matriz y el lado derecho

$$\bar{H}_5^{(5)} = \begin{pmatrix} h_{11}^{(5)} & h_{12}^{(5)} & h_{13}^{(5)} & h_{14}^{(5)} & h_{15}^{(5)} \\ & h_{22}^{(5)} & h_{23}^{(5)} & h_{24}^{(5)} & h_{25}^{(5)} \\ & & h_{33}^{(5)} & h_{34}^{(5)} & h_{35}^{(5)} \\ & & & h_{44}^{(5)} & h_{45}^{(5)} \\ & & & & h_{55}^{(5)} \\ & & & & 0 \end{pmatrix}, \qquad \bar{g}_5 = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \vdots \\ \gamma_6 \end{pmatrix}. \tag{3.17}$$

Generalmente, los escalares c_i y s_i de la i-ésima notación Ω_i son definidas como

$$s_i = \frac{h_{i+1,i}}{\sqrt{(h_{ii}^{(i-1)})^2 + h_{i+1,i}^2}}, \qquad c_i = \frac{h_{ii}^{(i-1)}}{\sqrt{(h_{ii}^{(i-1)})^2 + h_{i+1,i}^2}}. \tag{3.18}$$

Definir Q_m como el producto de matrices Ω_i:

$$Q_m = \Omega_m \Omega_{m-1} ... \Omega_1, \tag{3.19}$$

y tomar

$$\bar{R}_m = \bar{H}_m^{(m)} = Q_m \bar{H}_m, \tag{3.20}$$

$$\bar{g}_m = Q_m(\beta e_1) = (\gamma_1, ..., \gamma_{m+1})^T. \tag{3.21}$$

Como Q_m es unitario

$$min\|\beta e_1 - \bar{H}_m y\|_2 = min\|\bar{g}_m - \bar{R}_m y\|_2.$$

La solución del problema de mínimos cuadrados anterior es obtenida simplemente resolviendo el sistema triangular resultante de la eliminación de la última fila de la matriz $\bar{R}_m$ y del lado derecho $\bar{g}_m$ en (3.17).

Proposición 3.4. Sea $m \leq n$ y Ω_i, $i = 1, ..., m$, las matrices de notación usadas para transformar $\bar{H}_m$ en una de forma triangular superior. Denotar por $\bar{R}_m$, $\bar{g}_m = (\gamma_1, ..., \gamma_{m+1})^T$ la matriz resultante y el lado derecho, definidos por (3.20) y (3.21), y por R_m, g_m la matriz triangular superior mxm y el vector m-dimensional obtenido de $\bar{R}_m$, $\bar{g}_m$ por eliminar su última fila y componente, respectivamente. Entonces tenemos las siguientes propiedades:

1. El rango de AV_m es igual al rango de R_m. En particular, si $r_{mm} = 0$, entonces A debe ser singular.
2. El vector y_m que minimiza $\|\beta e_1 - \bar{H}_m y\|_2$ es dado por

$$y_m = R_m^{-1} g_m.$$

3. El vector residual en el paso m satisface

$$b - Ax_m = V_{m+1}(\beta e_1 - \bar{H}_m y_m) = V_{m+1} Q_m^T (\gamma_{m+1} e_{m+1}), \tag{3.22}$$

y como resultado,

$$\|b - Ax_m\|_2 = |\gamma_{m+1}|. \tag{3.23}$$

Hasta ahora hemos descrito un proceso para calcular la solución de mínimos cuadrados y_m (3.14). Esto junto con las notaciones del plano sirve para resolver el sistema lineal (3.9) del método de ortogonalización completa.

Es posible implementar el proceso anterior de una manera progresiva, es decir, en cada paso del algoritmo GMRES. Este enfoque permitirá obtener la norma residual en cada paso, sin operaciones adicionales de aritmética. Para ilustrar esto, asumir que la primera m rotación ya ha sido aplicada, es decir, comenzar en (3.17). Ahora, la norma residual es válida para x_5 y el criterio de parada puede ser aplicado. Un paso más en el algoritmo de Arnoldi debe ser ejecutado para obtener Av_6 y la sexta columna de $\bar{H}_6$. Esta columna se adjunta a $\bar{R}_5$, que ha sido aumentada por una fila de ceros para que coincida la dimensión. A continuación, hemos obtenido la siguiente matriz y lado derecho

$$\bar{H}_6^{(5)} = \begin{pmatrix} h_{11} & h_{12} & h_{13} & h_{14} & h_{15} & h_{16} \\ & h_{22} & h_{23} & h_{24} & h_{25} & h_{26} \\ & & h_{33} & h_{34} & h_{35} & h_{36} \\ & & & h_{44} & h_{45} & h_{46} \\ & & & & h_{55} & h_{56} \\ & & & & 0 & h_{66} \\ & & & & 0 & h_{76} \end{pmatrix}, \qquad \bar{g}_6^{(5)} = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \vdots \\ \gamma_6 \\ 0 \end{pmatrix}. \tag{3.24}$$

El algoritmo ahora continúa de la misma forma que antes. Necesitamos multiplicar la matriz por una matriz de notación Ω_6 (ahora de dimensión 7 x 7), con

$$s_6 = \frac{h_{76}}{\sqrt{(h_{66})^2 + h_{76}^2}}, \qquad c_6 = \frac{h_{66}^{(5)}}{\sqrt{(h_{66}^{(5)})^2 + h_{76}^2}}, \tag{3.25}$$

para obtener la matriz y lado derecho

$$\bar{R}_6 = \begin{pmatrix} r_{11} & r_{12} & r_{13} & r_{14} & r_{15} & r_{16} \\ & r_{22} & r_{23} & r_{24} & r_{25} & r_{26} \\ & & r_{33} & r_{34} & r_{35} & r_{36} \\ & & & r_{44} & r_{45} & r_{46} \\ & & & & r_{55} & r_{56} \\ & & & & & r_{66} \\ & & & & & 0 \end{pmatrix}, \qquad \bar{g}_6 = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \vdots \\ c_6\gamma_6 \\ -s_6\gamma_6 \end{pmatrix}. \tag{3.26}$$

Si la norma residual dada por $|\gamma_{m+1}|$ es suficientemente pequeña, el proceso debe ser parado. Las últimas filas de $\bar{R}_m$ y $\bar{g}_m$ son eliminadas y el sistema triangular superior resultante es resuelto para obtener y_m. Entonces, la solución aproximada $x_m = x_0 + V_m y_m$ es calculada.

Anotar para (3.26) que la siguiente relación conveniente para γ_{j+1} sea:

$$\gamma_{j+1} = -s_j \gamma_j. \tag{3.27}$$

En particular, si $s_j = 0$, entonces la norma residual debe ser igual a cero, que significa que la solución es exacta en el paso j.

3.4 Algoritmo simétrico de Lanczos

Este algoritmo se puede ver como una simplificación del método de Arnoldi para el caso particular cuando la matriz es simétrica. Cuando A es simétrica, entonces, la matriz de Hessenberg H_m se convierte en una tridiagonal simétrica. Esto lleva a tres términos recurrentes en el proceso de Arnoldi y recurrencia a corto plazo para soluciones algorítmicas como la ortogonalización completa y GMRES.

3.4.1 El algoritmo

Para introducir el algoritmo de Lanczos comenzamos haciendo la observación escrita en el siguiente teorema.

Teorema 3.5. Asumimos que el método de Arnoldi es aplicado a una matriz real simétrica A. Entonces, los coeficientes h_{ij} generados por el algoritmo son tales que

$$h_{ij} = 0 \quad para \quad 1 \leq i < j - 1 \tag{3.28}$$

$$h_{j,j+1} = h_{j+1,j}, j = 1, ..., m. \tag{3.29}$$

En otras palabras, la matriz H_m obtenida del proceso de Arnoldi es tridiagonal y simétrica.

Demostración. La prueba es una consecuencia inmediata de que $H_m = V_m^T A V_m$ sea una matriz simétrica que es también una matriz de Hessenberg por construcción. Por lo tanto, debe ser una matriz tridiagonal simétrica. □

La notación estándar usada para describir el algoritmo de Lanczos es obtenida imponiendo

$$\alpha_j \equiv h_{ij}, \quad \beta_j \equiv h_{j-1,j}$$

y, si T_m denota la matriz H_m resultante, esta es de la forma

$$T_m = \begin{pmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & & \cdots & & \\ & & \beta_{m-1} & \alpha_{m-1} & \beta_m \\ & & & \beta_m & \alpha_m \end{pmatrix}. \tag{3.30}$$

ALGORITMO 3.4: Algoritmo de Lanczos

1. Elegir un vector inicial v_1 de norma 2 unitario. Imponer $\beta_1 \equiv 0$, $v_0 \equiv 0$.
2. Para $j = 1, 2, ..., m$.
3. $w_j := Av_j - \beta_j v_{j-1}$.
4. $\alpha_j := (w_j, v_j)$.
5. $w_j := w_j - \alpha_j v_j$.
6. $\beta_{j+1} := \|w_j\|_2$. Si $\beta_{j+1} = 0$ entonces parar.
7. $v_{j+1} := w_j / \beta_{j+1}$.

El algoritmo anterior garantiza, al menos en la aritmética exacta, que los vectores v_i, $i = 1, 2, ...$, son ortogonales. En realidad, la ortogonalidad exacta de estos vectores se observa solo en el comienzo del proceso. En algún momento, los vecteres v_i empiezan perdiendo su ortogonalización global rápidamente. Para ello, se debe encontrar y eliminar los valores propios "con errores" [6].

La mayor diferencia práctica con el método de Arnoldi es que la matriz H_m es tridiagonal y, más importante, que sólo tres vectores deben ser salvados, a no ser que alguna forma de reortogonalización sea empleada.

3.4.2 Relación con los polinomios ortogonales

En la aritmética exacta, el núcleo del Algoritmo 3.4 es una relación de la forma

$$\beta_{j+1} v_{j+1} = Av_j - \alpha_j v_j - \beta_j v_{j-1}.$$

Esta relación de recurrencia de tres términos es una reminiscencia de la relación de recurrencia de tres términos estándar de los polinomios ortogonales. Para empezar, recordar que, si el grado de v_1 no es menor que m, entonces el subespacio $\mathcal{K}_m$ es de dimensión m y consiste en todos los vectores de la forma $q(A)v_1$, donde q es un polinomio de grado (q) no superior a m-1. En este caso, hay incluso un isomorfismo entre $\mathcal{K}_m$ y $\mathbb{P}_{m-1}$, el espacio de polinomios de grado no superior a m-1, que es definido como

$$q \in \mathbb{P}_{m-1} \rightarrow x = q(A)v_1 \in \mathcal{K}_m.$$

Por otra parte, podemos considerar que el subespacio $\mathbb{P}_{m-1}$ es proporcionado con el producto interno

$$\langle p, q \rangle_{v_1} = (p(A)v_1, q(A)v_1). \tag{3.31}$$

Esto es de hecho una forma bilineal no degenerada bajo la hipótesis de que m no es superior a μ, el grado de v_i. Ahora observamos que los vectores v_i son de la forma

$$v_i = q_{i-1}(A)v_1,$$

y la ortogonalidad de estos se traduce dentro de la ortogonalidad de los polinomios con respecto al producto interno.

Se sabe que los polinomios ortogonales reales satisfacen una recurrencia de tres términos. La relación de recurrencia entre los polinomios característicos de matrices tridiagonales también demuestran que la recurrencia de Lanczos calcula la secuencia de vectores $p_{T_m} v_1$, donde p_{T_m} es el polinomio característico de T_m.

3.5 Algoritmo del gradiente conjugado

El algoritmo del gradiente conjugado (CG) es una de las mejores técnicas iterativas conocidas para resolver sistemas lineales definidos positivos, simétricos y dispersos. Este método es una realización de una técnica de proyección ortogonal sobre el subespacio de Krylov $\mathcal{K}_m(r_0, A)$, donde r_0 es el residuo inicial. Por lo tanto, es matemáticamente equivalente al método de ortogonalización completa. Sin embargo, como A es simétrica, algunas simplificaciones resultantes de la recurrencia de tres términos de Lanczos dará lugar a algoritmos más elegantes [8].

3.5.1 Derivación y teoría

Primero derivamos el análogo del método de ortogonalización completa o método de Arnoldi, para el caso cuando A es simétrica. Dada una estimación inicial x_0 del sistema lineal $Ax = b$ y los vectores de Lanczos v_i, i = 1,...,m, junto con la matriz tridiagonal T_m, la solución aproximada obtenida para un método de proyección ortogonal sobre $\mathcal{K}_m$ es dada por

$$x_m = x_0 + V_m y_m, \quad y_m = T_m^{-1}(\beta e_1) \tag{3.32}$$

ALGORITMO 3.5: Método de Lanczos para sistemas lineales

1. Calcular $r_0 = b - Ax_0$, $\beta := \|r_0\|_2$, y $v_1 := r_0/\beta$.
2. Para $j = 1, 2, ..., m$.
3. $w_j = Av_j - \beta_j v_{j-1}$. (Si j = 1 fijar $\beta_1 v_0 \equiv 0$)
4. $\alpha_j = (w_j, v_j)$
5. $w_j := w_j - \alpha_j v_j$
6. $\beta_{j+1} = \|w_j\|_2$. Si $\beta_{j+1} = 0$ fijar $m := j$ e ir al paso 8.
7. $v_{j+1} = w_j/\beta_{j+1}$.
8. Fijar $T_m = tridiag(\beta_i, \alpha_i, \beta_{i+1})$ y $v_m = [v_1, ..., v_m]$.
9. Calcular $y_m = Tm^{-1}(\beta e_1)$ y $x_m = x_0 + v_m y_m$.

Algunos de los resultados obtenidos del método de Arnoldi para sistemas lineales son todavia válidos. Por ejemplo, el vector residual de la solución aproximada x_m es tal que

$$b - Ax_m = -\beta_{m+1} e_m^T y_m V_{m+1}. \tag{3.33}$$

El algoritmo del gradiente conjugado se puede derivar del algoritmo de Lanczos. Seguiremos los mismos pasos que el método de ortogonalización incompleta directa.

Primero se escribe la factorización LU de T_m como $T_m = L_m U_m$. La matriz L_m es unitaria bidiagonal inferior y U_m es bidiagonal superior. Por lo tanto, la factorización de T_m es de la forma

$$T_m = \begin{pmatrix} 1 & & & & \\ \lambda_2 & 1 & & & \\ & \lambda_3 & 1 & & \\ & & \lambda_4 & 1 & \\ & & & \lambda_5 & 1 \end{pmatrix} x \begin{pmatrix} \eta_1 & \beta_2 & & & \\ & \eta_2 & \beta_3 & & \\ & & \eta_3 & \beta_4 & \\ & & & \eta_4 & \beta_5 \\ & & & & \eta_5 \end{pmatrix}.$$

La solución aproximada está entonces dada por

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

Tomando

$$P_m \equiv V_m U_m^{-1},$$

y

$$z_m = L_m^{-1} \beta e_1,$$

entonces

$$x_m = x_0 + P_m z_m.$$

La última columna de P_m puede ser calculada de los vectores p_i anteriores y v_m por la simple actualización

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

Anotar que β_m es un escalar calculado del algoritmo de Lanczos, mientras que η_m resulta del m-ésimo paso de la eliminación Gausiana sobre la matriz tridiagonal, es decir,

$$\lambda_m = \frac{\beta_m}{\eta_{m-1}}, \tag{3.34}$$

$$\eta_m = \alpha_m - \lambda_m \beta_m. \tag{3.35}$$

Además, obtenemos que

$$z_m = \begin{pmatrix} z_{m-1} \\ \zeta_m \end{pmatrix},$$

de donde $\zeta_m = -\lambda_m \zeta_{m-1}$. Como resultado, x_m puede ser actualizado en cada paso como

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

donde p_m es definida anteriormente.

Esto nos da el siguiente algoritmo, que llamamos la versión directa del algoritmo de Lanczos para sistemas lineales (D-Lanczos).

ALGORITMO 3.6: D-Lanczos

1. Calcular $r_0 = b - Ax_0$, $\zeta_1 := \beta := \|r_0\|_2$, y $v_1 := r_0/\beta$.
2. Imponer $\lambda_1 = \beta_1 = 0$, $p_0 = 0$.
3. Para m = 1,2,...,hasta convergencia.
4. Calcular $w := Av_m - \beta_m v_{m-1}$ y $\alpha_m = (w, v_m)$.
5. Si m > 1 entonce calcular $\lambda_m = \frac{\beta_m}{\eta_{m-1}}$ y $\zeta_m = -\lambda_m \zeta_{m-1}$.
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. Si x_m ha convergido entonces parar.
10. $w := w - \alpha_m v_m$
11. $\beta_{m-1} = \|w\|_2$, $v_{m+1} = w/\beta_{m+1}$

Este algoritmo calcula la solución del sistema tridiagonal $T_m y_m = \beta e_1$ progresivamente usando la eliminación Gaussiana sin pivotar.

Proposición 3.5. Sea $r_m = b - Ax_m$, m = 0,1,..., el vector residual producido por los algoritmos Lanczos y D-Lanczos y p_m, m = 0,1,..., el vector auxiliar producido por el algoritmo 3.6. Entonces

1. Cada vector residual r_m es tal que $r_m = \sigma_m v_{m+1}$, donde σ_m es un cierto escalar. Como resultado, los vectores residuales son ortogonales el uno al otro.
2. Los vectores auxiliares p_m forman un conjunto A conjugado, es decir, $(Ap_i, p_j) = 0$ para $i \neq j$.

Demostración. La primera parte de la proposición es una consecuencia inmediata de la relación (3.33). Para la segunda parte, debe ser probado que $P_m^T A P_m$ es una matriz diagonal, donde $P_m = V_m U_m^{-1}$. Esto se sigue de

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

Ahora observamos que $U_m^{-T} L_m$ es una matriz triangular inferior que es también simétrica, ya que es igual a la matriz simétrica $P_m^T A P_m$. Por lo tanto, debe ser una matriz diagonal. □

Una consecuencia de la proposición anterior es que una versión del algoritmo puede ser derivada imponiendo las condiciones de ortogonalidad y conjugación. Esto nos da el algoritmo del gradiente conjugado, el que ahora derivamos. El vector x_{j+1} puede ser expresado como

$$x_{j+1} = x_j + \alpha_j p_j. \tag{3.36}$$

Ahora los vectores residuales deben satisfacer la recurrencia

$$r_{j+1} = r_j - \alpha_j A p_j. \tag{3.37}$$

Si los vectores r_j son ortogonales, entonces es necesario que $(r_j - \alpha_j A p_j, r_j) = 0$ y, como resultado,

$$\alpha_j = \frac{(r_j, r_j)}{(Ap_j, r_j)}. \tag{3.38}$$

También, se sabe que la siguiente dirección buscada p_{j+1} es una combinación lineal de r_{j+1} y p_j y, después de reescalar los p-vectores adecuadamente, se sigue que

$$p_{j+1} = r_{j+1} + \beta_j p_j. \tag{3.39}$$

Por lo tanto, una primera consecuencia de la relación anterior es que

$$(Ap_j, r_j) = (Ap_j, p_j - \beta_{j-1} p_{j-1}) = (Ap_j, p_j),$$

ya que Ap_j es ortogonal a p_{j-1}. Entonces (3.38) se convierte en $\alpha_j = (r_j, r_j)/(Ap_j, p_j)$. Además, escribiendo que p_{j+1} definida por (3.39) es ortogonal a Ap_j nos devuelve

$$\beta_j = -\frac{(r_{j+1}, Ap_j)}{(p_j, Ap_j)}.$$

Anotar que, para (3.37),

$$Ap_j = -\frac{1}{\alpha_j}(r_{j+1} - r_j), \tag{3.40}$$

y, por lo tanto,

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

Poniendo estar relaciones juntas obtenemos el siguiente algoritmo.

ALGORITMO 3.7: Gradiente conjugado

1. Calcular $r_0 = b - Ax_0$, $p_0 := r_0$
2. Para j = 0,1,...,hasta convergencia.
3. $\alpha_j := (r_j, r_j)/(Ap_j, p_j)$
4. $x_{j+1} := x_j + \alpha_j p_j$
5. $r_{j+1} := r_j - \alpha_j Ap_j$
6. $\beta_j := (r_{j+1}, r_{j+1})/(r_j, r_j)$
7. $p_{j+1} := r_{j+1} + \beta_j p_j$

Es importante anotar que los escalares α_j, β_j en este algoritmo son diferentes de aquellos del algoritmo de Lanczos. Los vectores p_j son múltiplos de los p_j del Algoritmo 3.6. En términos de almacenamiento, además de la matriz A, cuatro vectores (x, p, Ap, y r) deben ser salvados en el algoritmo 3.7, frente a cinco vectores (v_m, v_{m-1}, w, p, y x) en el algoritmo 3.6.

Capítulo 4

Aplicación a la resolución numérica de EDP y en matrices test

En este capítulo vamos a estudiar el comportamiento de los métodos numéricos vistos en los capítulos anteriores cuando los aplicamos a EDPs lineales elípticas (e.g., ecuación de Poisson) y en matrices test [4].

4.1 Aplicación a la discretización de EDP lineales elípticas

La EDP es de la forma

$$-d\sum_{i=1}^{n}\frac{\partial^2 u}{\partial^2 x_i}+a\sum_{i=1}^{n}\frac{\partial u}{\partial x_i}+r\cdot u=f,\quad en\quad \Omega=(0,1)^n, \tag{4.1}$$

con condiciones de frontera en $\Gamma = \partial\Omega$ de tipo Dirichlet, donde a,d y r son constantes, $u = u(x_1, ..., x_n)$, $f = f(x_1, ..., x_n)$ y además, $\frac{\partial u}{\partial x_i} = u_{x_i}$, $\frac{\partial^2 u}{\partial^2 x_i} = u_{x_i x_i}$, $1 \leq i \leq n$.

Vamos a repasar primero el caso unidimensional que nos lleva a una EDO para la cual ya hemos estudiado la discretización en diferencias finitas para problemas de contorno lineales. Consideramos el siguiente problema de contorno [Apuntes CN2]:

$$\begin{cases} u'' = a(x)u' + r(x)u + f(x), \quad x \in [0,1] \\ u(0) = \alpha, u(1) = \beta \end{cases} \tag{4.2}$$

Suponemos que $a, r, f \in C[0,1]$ y $r < 0$ en $[0,1]$. Sea $N \in \mathbb{N}$ dado, dividimos el intervalo $[0,1]$ en $N+1$ subintervalos de longitud $h = \frac{1}{N+1}$, con nodos $0 = x_0 < x_1 < ... < x_N < x_{N+1} = 1$, siendo $x_j = jh, j = 0, ..., N+1$. Se trata de calcular $u_j \approx u(x_j)$ para $j = 1, ..., N$.

Para ello, se aproximan derivadas por cocientes centrados de segundo orden en una malla de amplitud $h > 0$.

$$u'(x_j) = \frac{u(x_{j+1}) - u(x_{j-1})}{2h} - \frac{1}{6}u'''(\zeta_1) \cdot h^2, \;\; \zeta_1 \in (x_{j-1}, x_{j+1}), \tag{4.3}$$

$$u''(x_j) = \frac{u(x_{j+1}) - 2u(x_j) + u(x_{j-1})}{h^2} - \frac{1}{12}u^{(4)}(\zeta_2) \cdot h^2, \;\; \zeta_2 \in (x_{j-1}, x_{j+1}), \tag{4.4}$$

Usando (4.3) y (4.4) llegamos a:

$$\begin{cases} u_0 = \alpha, u_{N+1} = \beta \\ \frac{u_{j-1}-2u_j+u_{j+1}}{h^2} = a(x_j)\frac{u_{j+1}-u_{j-1}}{2h} + r(x_j)u_j + f(x_j), \quad j = 1, ..., N. \end{cases} \tag{4.5}$$

En el siguientes ejemplo consideramos la ecuación de Poisson para n = 2,3. Para ello vamos a tomar $d = 1$, $a = r = 0$, con lo que obtenemos,

$$-\sum_{i=1}^{n} u_{x_i x_i} = f(x). \tag{4.6}$$

$\boxed{n = 2}$

Tenemos el problema

$$\begin{cases} -u_{x_1 x_1} - u_{x_2 x_2} = f, & (x_1, x_2) \in \Omega = (0,1)^2 \\ u = g, & (x_1, x_2) \in \Gamma = \partial\Omega \end{cases} \tag{4.7}$$

y si consideramos la malla $\Omega_h^{(2)} = \{(ih, jh)\}_{i,j=0}^{N+1}$, con $h = \frac{1}{N+1}$, y diferencias centrales de orden dos:

$$u_{x_1 x_1}(x_1, x_2) \simeq \frac{u(x_1 + h, x_2) - 2u(x_1, x_2) + u(x_1 - h, x_2)}{h^2},$$
$$u_{x_2 x_2}(x_1, x_2) \simeq \frac{u(x_1, x_2 + h) - 2u(x_1, x_2) + u(x_1, x_2 - h)}{h^2},$$

se obtiene el esquema en diferencias finitas:

$$\frac{(-U_{i+1,j} + 2U_{i,j} - U_{i-1,j})}{h^2} + \frac{(-U_{i,j+1} + 2U_{i,j} - U_{i,j-1})}{h^2} = f_{i,j} \;\; 1 \le i, j \le N, \tag{4.8}$$

que corresponde a un sistema lineal de dimensión N^2

$$A^{(2)} \cdot U = b, \tag{4.9}$$

donde $b \in \mathbb{R}^{N^2}$ contiene la discretización del dato $f(x_1, x_2)$ y las condiciones de frontera $g(x_1, x_2)$, $U = (U_{11}, ..., U_{N1}, U_{12}, ..., U_{N2}, ..., U_{1N}, ..., U_{NN})^T \in \mathbb{R}^{N^2}$, $U_{ij} \simeq U(ih, jh)$ y $A^{(2)}$ es la matriz de dimensión $N^2 x N^2$,

$$
\mathrm{A}^{(2)} = \left[\begin{array}{cccc|cccc|cccc|cccc}
4 & -1 & & & -1 & & & & & & & & & & & \\
-1 & \ddots & \ddots & & & \ddots & & & & & & & & & & \\
 & \ddots & \ddots & -1 & & & \ddots & & & & & & & & & \\
 & & -1 & 4 & & & & -1 & & & & & & & & \\
\hline
-1 & & & & 4 & -1 & & & \ddots & & & & & & & \\
 & \ddots & & & -1 & \ddots & \ddots & & & \ddots & & & & & & \\
 & & \ddots & & & \ddots & \ddots & -1 & & & \ddots & & & & & \\
 & & & -1 & & & -1 & 4 & & & & \ddots & & & & \\
\hline
 & & & & \ddots & & & & \ddots & & & & -1 & & & \\
 & & & & & \ddots & & & & \ddots & & & & \ddots & & \\
 & & & & & & \ddots & & & & \ddots & & & & \ddots & \\
 & & & & & & & \ddots & & & & \ddots & & & & -1 \\
\hline
 & & & & & & & & -1 & & & & 4 & -1 & & \\
 & & & & & & & & & \ddots & & & -1 & \ddots & \ddots & \\
 & & & & & & & & & & \ddots & & & \ddots & \ddots & -1 \\
 & & & & & & & & & & & -1 & & & -1 & 4
\end{array}\right] \tag{4.10}
$$

$\boxed{n = 3}$

Tenemos el problema

$$
\begin{cases} -u_{x_1x_1} - u_{x_2x_2} - u_{x_3x_3} = f, & (x_1, x_2, x_3) \in \Omega = (0,1)^3, \\ u = g, & (x_1, x_2, x_3) \in \Gamma = \partial\Omega, \end{cases} \tag{4.11}
$$

y consideramos la malla $\Omega_h^{(3)} = \{(ih, jh, kh)\}_{i,j,k=1}^{N+1}$, con $h = \frac{1}{N+1}$, se obtiene el esquema en diferencias finitas:

$$(-U_{i+1,j,k}+2U_{i,j,k}-U_{i-1,j,k})+(-U_{i,j+1,k}+2U_{i,j,k}-U_{i,j-1,k})+(-U_{i,j,k+1}+2U_{i,j,k}-U_{i,j,k-1}) = f_{i,j,k}, \quad 1 \leq i, j, k \leq N, \tag{4.12}$$

que corresponde a un sistema lineal de dimensión N^3

$$A^{(3)} \cdot U = b, \tag{4.13}$$

donde $b \in \mathbb{R}^{N^3}$ contiene la discretización del dato $f(x_1, x_2, x_3)$ y las condiciones de frontera $g(x_1, x_2, x_3)$, $U = (U_{i.j,k})_{i,j,k=1}^{N} \in \mathbb{R}^{N^3}$, $U_{ij} \simeq U(ih, jh, kh)$ y $A^{(3)}$ es la matriz de dimensión $N^3 x N^3$,

$$A^{(3)} = \left[\begin{array}{c|c|c|c|c|c} A^{(2)} & -I & 0 & \cdots & 0 & 0 \\ \hline -I & A^{(2)} & -I & \cdots & 0 & 0 \\ \hline & \ddots & \ddots & \ddots & & \\ \hline & & \ddots & \ddots & \ddots & \\ \hline & & & \ddots & \ddots & -I \\ \hline 0 & 0 & & & -I & A^{(2)} \end{array}\right] \tag{4.14}$$

A continuación, describimos un ejemplo en dimensiones $n = 2$ y $n = 3$ donde obtenemos que la solución de (4.6) es un polinomio de grado menor o igual que dos. El vector U se usará para medir el error de las iteraciones provistas por los métodos vistos en los capítulos anteriores. La función f de la ecuación de Poisson la obtenemos a partir de la solución exacta que tomemos y el vector b se obtiene discretizando dicha f junto con las condiciones de frontera.

4.1.1 Ejemplo 1

En este ejemplo consideramos la ecuación de Poisson con condiciones de frontera de tipo Dirichlet homogéneas en $[0, 1]^n$, n = 2,3, con solución exacta $u = u(x_1, ..., x_n) = 4^n \cdot \prod_{i=1}^{n} x_i(1 - x_i)$.
Para las dimensiones n = 2 y n = 3, la discretización de (4.2) mediante diferencias centrales de orden 2 produce los sistemas lineales (4.9) y (4.13) $A^{(n)} \cdot U = b$, de dimensión N^n, con $A^{(n)}$ simétrica y definida positiva. Tomamos en cada caso N = 100.

•Para n = 2, la solución exacta es $u(x_1, x_2) = 4^2 \cdot x_1(1 - x_1) \cdot x_2(1 - x_2)$, de donde obtenemos $f(x_1, x_2) = 32 \cdot [x_1(1-x_1)+x_2(1-x_2)]$. Y obtenemos la matriz b discretizando la f. A continuación, mostramos el comportamiento de la norma del error en relación al número de iteraciones. Hemos comparado los métodos de GMRES (algoritmo 3.3), del Gradiente conjugado (algoritmo 3.7) y del Residual mínimo (algoritmo 2.3).

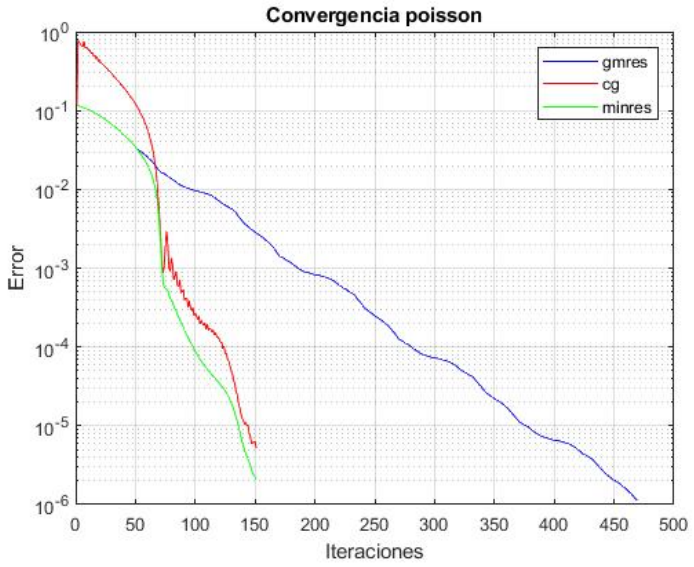

Figura 4.1: Velocidad de convergencia en los métodos GMRES, GC y MINRES aplicados a la ecuación de Poisson 2D con solución exacta $u(x_1, x_2) = 4^2 \cdot x_1(1 - x_1) \cdot x_2(1 - x_2)$.

Observamos en la Figura 4.3 que la convergencia de GC y MINRES acaban muy cerca en apenas 150 iteraciones, mientras que GMRES necesita más iteraciones para converger.

•Para n = 3, la solución exacta es $u(x_1, x_2, x_3) = 4^3 \cdot x_1(1 - x_1) \cdot x_2(1 - x_2) \cdot x_3(1 - x_3)$, de donde obtenemos la $f(x_1, x_2, x_3) = 192 \cdot [x_1(1 - x_1) + x_2(1 - x_2) + x_3(1 - x_3)]$. Y obtenemos la matriz b discretizando el dato f. A continuación, mostramos el comportamiento del error para los métodos GC y GMRES, y sus variantes con reinicialización GMRES(m) tras m iteraciones, con m = 10,50,100.

Observamos nuevamente que el decaimiento del error con las iteraciones es similar para GC y GMRES con m = 50,100. Mientras que para GMRES con m = 10 la convergencia necesita el máximo de iteraciones.

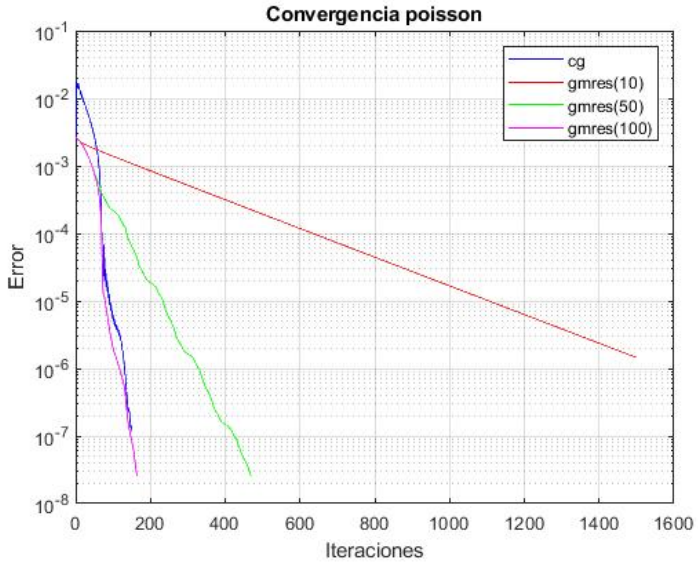

Figura 4.2: Error frente al número de iteraciones de los métodos GC y GMRES(m), m = 10,50,100, aplicados a la ecuación de Poisson 3D.

4.2 Aplicación en matrices test

En esta sección vamos a centrarnos en el método GMRES precondicionado y vamos a observar la convergencia frente al número de iteraciones en dos ejemplos, uno con reinicialización y otro sin reinicialización usando problemas test con matrices grandes y dispersas [4].

4.2.1 Precondicionamiento

El precondicionamiento se basa en intentar encontrar una matriz M tal que $M^{-1}A$ tenga mejores propiedades espectrales de forma que se tenga convergencia rápida cuando se aplica al sistema modificado $M^{-1}A = M^{-1}b$. Para el caso $M = A$, todos los métodos alcanzarían la solución exacta en un único paso. Por tanto, buscamos una aproximación M de A de manera que el correspondiente método de Krylov aplicado a $M^{-1}A = M^{-1}b$ solo necesite una pocas iteraciones para alcanzar una buena aproximación de la solución del sistema. La matriz M en este contexto recibe el nombre de precondicionador para la matriz A.

El mismo precondicionador puede ser implementado de diferentes formas. Esto no cambia los autovalores de la matriz precondicionada pero sí los autovectores, lo cual puede repercutir en el comportamiento de la convergencia. Existen tres implementaciones diferentes:

1. Precondicionamiento por la izquierda.
   Aplicamos el método iterativo a
   $$M^{-1}A = M^{-1}b.$$
   Notemos que la simetría de A y M no implica la simetría de $M^{-1}A$. Sin embargo, si M es simétrica definida positiva entonces $[x, y] = (x, My)$ define un producto interno adecuado.

Es fácil verificar que $M^{-1}A$ es simétrica respecto a este nuevo producto interno, y por tanto, podemos usar métodos como MINRES y GC.
Si usamos un método que minimice la norma del residuo (como GMRES o MINRES) debemos tener en cuenta que estamos minimizando el residuo precondicionado $M^{-1}(b - Ax_k)$, que podría ser bastante diferente del residuo $b - Ax_k$.

2. Precondicionamiento por la derecha.
Aplicamos el método iterativo a

$$AM^{-1}y = b \quad con \quad x = M^{-1}y.$$

Esta forma tampoco deja un producto simétrico aunque A y M sean simétricas. Este precondicionamiento tiene la ventaja de que sólo afecta al operador y no al lado derecho del sistema.

3. Precondicionamiento por ambos lados.
Para un precondicionador M con $M = M_1M_2$, el método iterativo puede ser aplicado a

$$M_1^{-1}AM_2^{-1}z = M_1^{-1}b \quad con \quad x = M_2^{-1}x.$$

Esta forma podría ser útil para aquellos precondicionadores que vienen en forma factorizada o para obtener un operador simétrico cuando M no pueda ser usada para la definición de producto interno (es decir, M no es simétrica definida positiva).

Observación 4.1. La elección del precondicionador M puede influir bastante en la eficiencia del correspondiente método precondicionado y, por lo general, M ha de elegirse adaptado al sistema lineal que se desea resolver. No obstante, en este trabajo consideraremos un tipo de precondicionador general basado en las denominadas factorizaciones matriciales incompletas. Consideramos el sistema lineal disperso $Ax = b$, con $A \in \mathbb{R}^{NxN}$ una matriz nxn. Con frecuencia este es resuelto utilizando eliminación Gaussiana que es equivalente a factorizar la matriz A como $A = LU$. Con lo cuál, elegimos $M = L \cdot U$, donde L y U forman una descomposición LU de la matriz A, con L triangular inferior y U triangular superior. Sin embargo, tal descomposición puede ser muy costosa si la dimensión de la matriz A es suficientemente grande. Una primera estrategia para relajar la exigencia computacional de la descomposición LU consiste en fijar un subconjunto de índices

$$S \subseteq \{1, .., N\}x\{1, .., N\},$$

y seguir los pasos usuales de los algoritmos para computar una descomposición LU pero sólo en aquellos índices $(i, j) \in S$. En particular, una usual opción a considerar es tomar $S = \{(i, j) \in \{1, .., N\}^2 | a_{i,j} \neq 0\}$, esto es, realizar los pasos de la descomposición LU pero respetando las componentes nulas de la matriz A original. Esta elección da lugar a la factorización incompleta LU con "nivel 0 de relleno" denotado como $ILU(0)$. Observar que si A es una matriz sparse, es decir, con una elevada proporción de elementos nulos, el cálculo de factorizaciones LU hace que los factores L y U tengan un mayor número de componentes no nulas, lo cual conlleva un mayor gasto computacional en la resolución de sistemas $L \cdot U \cdot y = C$ por sustitución hacia atrás y hacia adelante. Por ello, la elección de un precondicionador $M = L \cdot U$ a través de una factorización incompleta suele ser ventajosa si la dimensión N de A es elevada, a pesar de que $A - LU \neq 0$.
Consideramos en particular el siguiente pseudocódigo para la factorización $ILU(0)$ [2].

ALGORITMO 4.1: ILU(0) (por el método de Doolittle, esto es, L con unos en la diagonal).

1. Para $r = 1 : N - 1$
2. Para $i = r + 1 : N$
3. Si $(i, r) \in S$, entonces $a_{i,r} = a_{i,r} / a_{r,r}$.
4. Para $j = r + 1 : N$
5. Si $(i, j) \in S$ y $(r, j) \in S$, entonces

$$a_{i,j} = a_{i,j} - \frac{a_{i,r} \cdot a_{r,j}}{a_{r,r}}. \tag{4.15}$$

Observación 4.2: (Factorizaciones incompletas modificadas).
En la factorización $ILU(0)$, si $(i, j) \notin S$, entonces el valor $a_{i,r} \cdot a_{r,j}/a_{r,r}$ no se sustrae de $a_{i,j}$, como correspondería a una factorización LU completa.
Una opción alternativa que suele ser ventajosa en la práctica consiste en no despreciar completamente el valor $a_{i,r} \cdot a_{r,j}/a_{r,r}$ cuando $(i, j) \notin S$ y sustraerlo de la componente diagonal de la fila correspondiente, eso es, $a_{i,i}$. Esto da lugar a la descomposición imcompleta modificada LU, $MILU(0)$.

ALGORITMO 4.2: MILU(0)

1. Si $(i, j) \in S$, entonces
2. Si $(i, j) \in S$, entonces $a_{i,j} = a_{i,j} - \frac{a_{i,r} \cdot a_{r,j}}{a_{r,r}}$; en caso contrario,

$$a_{i,i} = a_{i,i} - \frac{a_{i,r} \cdot a_{r,j}}{a_{r,r}} \tag{4.16}$$

A continuación, nos centraremos en el método GMRES precondicionado. Para ello, primero desarrollamos su algoritmo:

ALGORITMO 4.1: MÉTODO GMRES PRECONDICIONADO (PGMRES):
Consideramos el sistema lineal $Ax = b$ donde $A \in \mathbb{R}^{NxN}$ es una matriz regular, $b \in \mathbb{R}^N$ un vector no nulo y M una matriz regular. Dado $x_0 \in \mathbb{R}^N$ y $r_0 = b - Ax_0$,

1. Definimos $\tilde{r}_0 := M^{-1} r_0$, $v_1 = \frac{\tilde{r}_0}{\|\tilde{r}_0\|_2}$ y $c = \|\tilde{r}_0\|_2 e_1 \in \mathbb{R}^N$.
2. Sean $\tilde{v}_n = M^{-1} A v_n$ y $h_{in} := \langle \tilde{v}_n, v_i \rangle_2$ para $1 \leq i \leq n$. Calcular $\hat{v}_{n+1} = \tilde{v}_n - \sum_{i=1}^{n} h_{in} v_i$ y $h_{n+1,n} = \|\hat{v}_{n+1}\|_2$.
3. Continuar con los pasos 7-10 del Algoritmo del Método GMRES.

Para los siguientes ejemplos vamos a tomar una matriz dispersa no simétrica real de dimensión 479 por 479 predefinida en MATLAB [4]. Esta matriz puede surgir de la discretización de ciertas EDP (e.g., ecuación de convección-difusión-reacción).

4.2.2 Ejemplo 2

Vamos a observar el efecto de usar una matriz precondicionada y no precondicionada sin reinicialización en el método GMRES para resolver un sistema lineal. Para el caso precondicionado, como A es no simétrica, usamos el método de precondicionamiento ILU descrito anteriormente para generar la matriz precondicionada $M = LU$. A continuación, resolvemos el sistema precondicionado $M^{-1}Ax = M^{-1}b$ donde L y U son argumentos de entrada de GMRES.

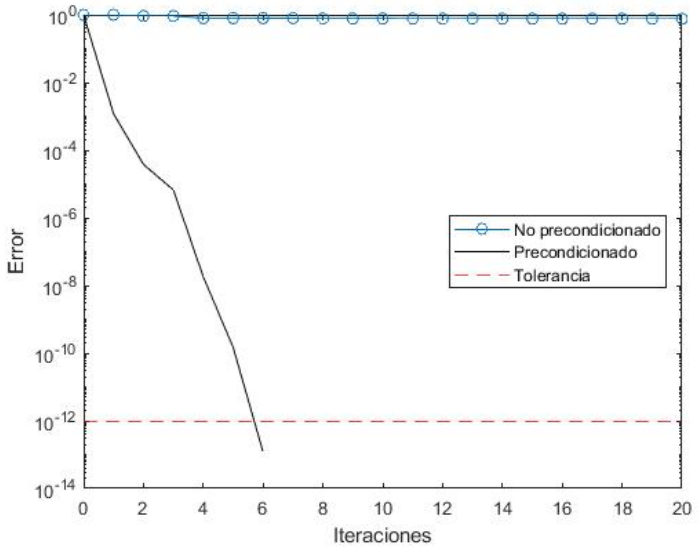

Figura 4.3: Error frente al número de iteraciones del método GMRES sin reinicialización para los casos precondicionado y no precondicionado.

Para el método no precondicionado se ha realizado el número máximo de iteraciones sin conseguirse la convergencia. Mientras que el método precondicionado ha convergido con la tolerancia deseada en un número menor de iteraciones que las máximas establecidas. El uso del método de precondicionamiento ILU produce un error menor que la tolerancia descrita en la sexta iteración.

4.2.3 Ejemplo 3

Veamos ahora el efecto de usar una matriz precondicionada con distintas reinicialización en el método GMRES(m), m = 3,4,5. La siguiente gráfica muestra el historial de convergencia de cada método GMRES reiniciado.

Observamos que gmres(3) converge en la iteración 5 por encima de la tolerancia, por otro lado, para el caso gmres(4) y gmres(5), la convergencia se aproxima a la iteración 3. Podemos observar que a mayor reinicialización converge más rápido en menos iteraciones.

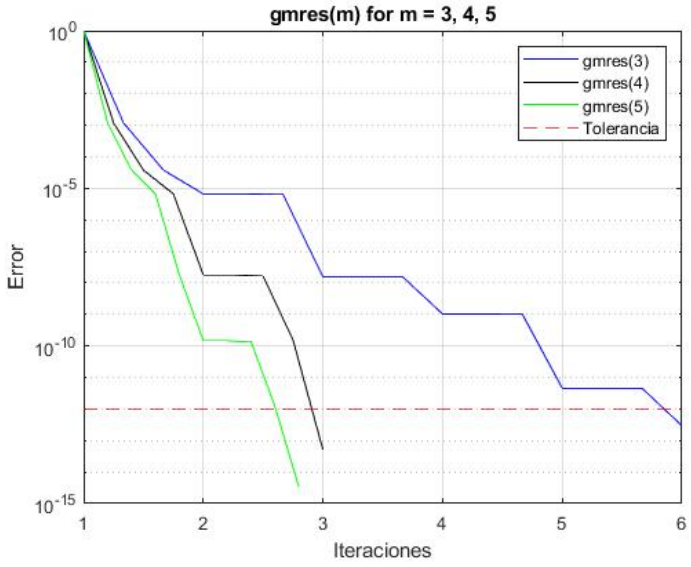

Figura 4.4: Error frente al número de iteraciones del método GMRES con reinicialización para m = 3,4 y 5.

4.3 Conclusiones

El comportamiento de los métodos de proyección presentados en este trabajo queda recogido en los ejemplos pertenecientes al último capítulo. Se intuye que el método del Gradiente Conjugado y del Residual mínimo resultan más eficientes que el método GMRES para una ecuación de Poisson con condiciones de frontera tipo Dirichlet homogéneas, para una matriz simétrica definida positiva, tomando n = 2. Para el caso en el que n = 3, tomando la misma ecuación y condiciones que para n = 2 y con matriz simétrica, concluimos que el método GMRES con reinicialización mayor nos lleva a una convergencia muy parecida a la obtenida por el método del gradiente conjugado.

A continuación, para el caso de una matriz test no simétrica que puede surgir, por ejemplo, de la discretización de una EDP de la forma convección-difusión-reacción, estudiamos el error frente al número de iteraciones para el método GMRES precondicionado. En el ejemplo 2, en el cuál vemos la diferencia entre dicho método precondicionado y no precondicionado sin reinicialización, obtenemos que el método no precondicionado no converge mientras que el precondicionado si lo hace y además en pocas iteraciones. Para finalizar, en el ejemplo 3, donde vemos el comportamiento del método gmres precondicionado con reinicialización, obtenemos que mientras mayor reinicialización, menos iteraciones son necesarias para que converga dicho método.

Por lo tanto, el objetivo de nuestro trabajo queda expuesto en estas gráficas como resultado del comportamiento de los diferentes métodos de proyección basados en subespacios de Krylov cuando se tratan de distintos sistemas lineales mostrando las condiciones más favorables a cada uno.

Bibliografía

[1] Aubanell, A., Benseny, A. Delshams, A., *Útiles básicos de Cálculo Numérico.* Labor (1993).

[2] Axelsson, 0. *Iterative solution methods. Cambridge University Press* (1996).

[3] Ayres, F. *Ecuaciones diferenciales.* Mc Graw-Hill (1991).

[4] Barrett, R., Berry, M., Chan, T., Demmen, J., Donat, J., Dongarra, J., Eijkhout,V., Pozo, R., Romine, C., Van der Vorst, H. *Templates for the solution of linear systems: building blocks for iterative systems.* SIAM (1994)

[5] Ciarlet, P.G., *Introducción à l'analyse numérique matricielle et à l'optimization.* Masson (1990).

[6] Cullum, J., Willoughby, R. *Lanczos Algoritms for Large Symmetric Eigenvalue Computations.* Birkhäuser (1984).

[7] Hackbusch, W. *Iterative Solution of Large Sparse Systems of Equations.* Springer (2016).

[8] Meurant, G., Duintjer Tebbens, J. *Krylov methods for nonsymmetric linear systems: from theory to computations.* Springer (2020).

[9] Quarteroni, A., Sacco, R., Saleri, F. *Numerical Mathematics.* Springer (2007).

[10] Saad, Y. *Iterative methods for sparse linear systems.* SIAM (2003).