Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación

Cálculo Científico y Tecnológico

Precondicionamiento por aproximaciones a la matriz pseudoinversa para el problema de mínimos cuadrados lineales

Trabajo Especial de Grado
presentado ante la Ilustre
Universidad Central de Venezuela
Por el Bachiller
Oskar Raúl Cahueñas Camargo
para optar al título de
Licenciado en Computación

Tutores
Prof. Marcos Raydán y **Prof. Luis Manuel Hernández**

Caracas, Julio de 2009

ACTA

Quienes suscriben, miembros del jurado designado por el Consejo de Escuela de Computación de la Facultad de Ciencias, para examinar el Trabajo Especial de Grado presentado por el bachiller ***Oskar Raúl Cahueñas Camargo***, portador de la cédula de identidad ***V-12.422.793***, con el título "***Precondicionamiento por aproximaciones a la matriz pseudoinversa para el problema de mínimos cuadrados lineales***", para optar al título de Licenciado en Computación, dejan constancia de lo siguiente:

Leído como fue este trabajo, por cada uno de los miembros del jurado, se fijó el día ***06 de julio de 2009 a las 11:00 am***, para que su autor lo defendiera en forma pública en el ***Salón Leandro Aristiguieta de la Facultad de Ciencias de la Universidad Central de Venezuela***, mediante una presentación oral de su contenido, luego de lo cual respondió a las preguntas formuladas. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobarlo.

En fe de lo cual se levanta la presente Acta, en Caracas el día seis de julio del año dos mil nueve, dejándose también constancia de que actuó como Coordinador del Jurado el profesor Luis Manuel Hernández.

_____

Profa. Joalí Moreno (Jurado Principal)

_____

Prof. Otilio Rojas (Jurado Principal)

_____

Prof. Luis Manuel Hernández (Tutor)

Resumen

Dada una matriz $A \in \mathbb{R}^{m \times n}$ con $m > n$ de rango completo y un vector $b \in \mathbb{R}^n$, se propone utilizar una aproximación a la matriz pseudoinversa de A, $A^\dagger \in \mathbb{R}^{n \times m}$ como precondicionador para el problema de mínimos cuadrados lineales asociado a A y b. Dicha aproximación a $A^\dagger$ se obtendrá a partir del método iterativo de Schulz, un esquema iterativo basado en el método de Newton en espacios de matrices para el cálculo de matrices inversas y pseudoinversas.

El problema de mínimos cuadrados precondicionado se resolverá a través del esquema de aceleración de convergencia Richardson-PR2, un método iterativo-residual obtenido como una generalización del método de Richardson de primer orden para resolver sistemas de ecuaciones lineales. Se estudiarán las propiedades del precondicionador y se realizarán pruebas numéricas con diversas matrices.

Índice general

Índice de Algoritmos

Dedicatoria

A mi madre y mi hermana.

Agradecimientos

A mis tutores, los Profesores Marcos Raydan y Luis Manuel Hernández por su infinita paciencia y apoyo.

La precisión numérica es el alma de toda ciencia.

Sir D' Arcy Wentworth Thompson

En mi teoría del movimiento de los cuerpos celestes, mostré cómo calcular los valores más probables de las incógnitas, dado que las leyes probabilísticas que rigen los errores observacionales son bien conocidas. Sin embargo, en la mayoría de los casos dichas leyes son meras hipótesis. En virtud de ello apliqué a esa teoría a la ley más plausible: que la probabilidad de un error x es proporcional a e^{xxhh}. A partir de esta suposición, nació un método que usé durante algún tiempo, especialmente para cálculos astronómicos. Actualmente es utilizado por muchas personas y es conocido como el método de mínimos cuadrados

Carl Gauss

Introducción

Si una idea no es absurda al principio, entonces no merece la pena

Albert Einstein

Uno de los problemas históricamente más relevantes en las matemáticas aplicadas, es el de la búsqueda de métodos numéricos para el ajuste de datos a alguna función. Este problema es una necesidad fundamental en las ciencias básicas y aplicadas, pues el objetivo último de las disciplinas científicas es la explicación de fenómenos naturales a través de modelos matemáticos que deben obtenerse o ser validados a partir del ajuste de datos provenientes de la observación o experimentación de dichos fenómenos.

En particular, en una gran cantidad de aplicaciones, se busca establecer la existencia de relaciones *lineales* entre varias variables que caracterizan determinado fenómeno. Esto es, supóngase que se tienen variables $x_1, x_2, \dots x_n$ y se desea verificar que alguna variable b_k satisface la relación

$$b_k = \sum_{i=1}^{i=n} c_i x_i$$

para un conjunto de parámetros reales $c_1, c_2, \dots c_n$. La ecuación anterior se conoce como *modelo de ajuste lineal* para n valores.

Si se tienen m conjuntos de datos experimentales $(x_{1k}, x_{2k}, \dots x_{nk}, b_k)$ para $k = 1, \dots, m$ con $m > n$, el problema anterior conlleva a un sistema de ecuaciones lineales *sobredeterminado* de la forma

$$Ax = b$$

con $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$ y $b \in \mathbb{R}^m$. Debido a que este sistema es sobredeterminado, no se garantiza la existencia de una solución en el caso general. No obstante, se desea poder hallar al menos una *aproximación* a la solución de dicho problema que sea la *mejor posible* en algún sentido.

El problema anterior se conoce como *problema de mínimos cuadrados lineales* o simplemente *problema de mínimos cuadrados*.

Históricamente, se tienen referencias de J. Kepler (1571 - 1630) quien usó modelos lineales para ajustar cálculos de órbitas planetarias. R. Cotes (1682 - 1716) y T. Simpson (1710 - 1761) utilizaron ajustes lineales en donde se buscaba minimizar la suma de los cuadrados de los errores de los datos experimentales, aplicados a problemas de astronomía. También S. Laplace (1749-1827) propuso diversas técnicas de ajuste lineal de datos basados en la minimización

del error promedio y del error promedio al cuadrado en aplicaciones de geodesia y mediciones terrestres. No obstante, fue C. Gauss (1777 - 1855) quién postuló el modelo de ajuste por mínimos cuadrados tal como se conoce en la actualidad, y finalmente, A. Legendre (1752-1833) quien acuñó el término *Mínimos cuadrados* al publicar en 1805 el clásico método de las *ecuaciones normales* para la resolución de dicho problema. Para el lector interesado en los aspectos históricos del problema de mínimos cuadrados se sugiere consultar [28] y [38].

Por otro lado, el desarrollo de métodos numéricos para el problema de mínimos cuadrados se inició plenamente después de la segunda mitad del siglo XX. En 1965 G. Golub (1932-2007) publica un artículo donde utiliza la descomposición QR vía transformaciones de Householder a la resolución del problema de mínimos cuadrados [16], el cual es uno de los primeros trabajos de métodos numéricos para mínimos cuadrados. Diversos métodos para la resolución del problema a través de métodos como la eliminación Gaussiana, de la descomposición en valores singulares (SVD) entre otros, fueron propuestos por Golub, Kahan y Wilkinson, entre otros, en las décadas de 1960 y 1970. Así mismo en los últimos años se han hecho grandes progresos en los métodos directos para resolver problemas de mínimos cuadrados, así como métodos iterativos para el caso de matrices *sparse* de grandes dimensiones.

En este trabajo se desarrollará un método numérico *iterativo* para resolver el problema de mínimos cuadrados para matrices densas de grandes dimensiones.

En primer lugar, la matriz A será *precondicionada* usando una aproximación a $A^\dagger$, la matriz pseudoinversa de A. La aproximación a $A^\dagger$ proviene de la iteración de Schulz, que es un caso particular del método de Newton en espacios de matrices, aplicado al cálculo de la inversa de una matriz. Se plantea, por consiguiente, el precondicionamiento de una matriz rectangular, un tópico del cual no se encontró ninguna referencia en la literatura, por lo cual representa el aporte original de este trabajo.

En segundo lugar, el sistema precondicionado, el cual se convierte en un sistema de n ecuaciones con n incógnitas, se resolverá con un método iterativo que proviene de un esquema de aceleración de convergencia del método de Richardson de primer orden desarrollado en [7]. En este esquema, las iteraciones son de la forma

$$\begin{aligned} r^{(0)} &= b - Ax^{(0)} \\ x^{(k+1)} &= x^{(k)} + \lambda_k C_k r^{(k)} \quad k \geq 0 \\ r^{(k+1)} &= r^{(k)} - \lambda_k A C_k r^{(k)} \quad k \geq 0 \end{aligned}$$

Donde en cada iteración, la solución x avanza por una dirección de búsqueda dada por el residual de la iteración anterior premultiplicado por una matriz precondicionadora C_k, la cual puede variar en cada iteración y un tamaño de paso λ_k Además, se definirá una *extensión* de dicho esquema de aceleración de convergencia para iterar sobre la matriz rectangular, usando como familia de matrices precondicionadoras las aproximaciones a $A^\dagger$ generadas por la iteración de Schulz.

Debe observarse que históricamente, los métodos iterativos suelen ser utilizados principalmente en matrices *sparse*, ya que en el caso de matrices densas, estos tienen un costo computacional más elevado respecto a los métodos directos. Sin embargo, en los últimos años, debido a la creciente popularización del procesamiento paralelo, está surgiendo un interés por el uso de métodos iterativos para la resolución de sistemas lineales con matrices densas de grandes dimensiones, pues estos métodos permiten explotar el paralelismo de una manera mucho más eficiente que los métodos directos. Algunas referencias en esta dirección son [10] y [42].

Entre aplicaciones de problemas de mínimos cuadrados que involucran matrices densas se pueden encontrar en áreas como astronomía, geodesia, fotogrametría, estadística entre otras (Ver [11], [21] y [14])

El resto del trabajo se organiza de la siguiente manera. En el capítulo 1 se presentarán los fundamentos teóricos del problema de mínimos cuadrados lineales sobre matrices de rango completo. En el capítulo 2 se dará una breve introducción de los métodos numéricos *clásicos* encontrados en la literatura para la resolución del problema de mínimos cuadrados. En el capítulo 3 se presentará el método de Schulz como un caso especial del método de Newton en varias variables y se probarán algunas propiedades de la iteración de Schulz para el caso de matrices rectangulares, que implican que dichas matrices son una buena elección para precondicionar el problema de mínimos cuadrados. En el capítulo 4 se introducirá el esquema de aceleración de convergencia de Richardson-PR2. En el capítulo 5 se presentarán los algoritmos para la resolución del problema de mínimos cuadrados, obtenidos a partir de la iteración de Richardson-PR2 combinada con el precondicionamiento por aproximaciones a la matriz pseudoinversa. Los dos últimos capítulos estarán dedicados a presentar los resultados de las pruebas numéricas realizadas y las conclusiones del trabajo.

Capítulo 1

El Problema de Mínimos Cuadrados Lineales (PMCL)

La verdad es demasiado complicada como para permitir nada más allá de meras aproximaciones

John von Neumann

Considere el sistema de ecuaciones lineales:

$$Ax = b, \quad A \in \mathbb{R}^{m\times n},\ b \in \mathbb{R}^m\ m > n \tag{1.1}$$

Como este sistema es *sobredeterminado*, esto es, con más ecuaciones que incógnitas, no se puede garantizar la existencia de una solución x para una matriz A y un vector b arbitrarios. Por lo tanto, es deseable hallar un vector x, tal que Ax sea la *mejor* aproximación a b.

Una posible escogencia de x, para que Ax sea la *mejor* aproximación a b, consiste en hallar un vector x que minimice la distancia entre los vectores Ax y b. En otras palabras, hallar el vector x que minimice el vector residual $Ax - b$ bajo la norma euclídea. Esto conlleva al siguiente problema de optimización:

$$\min_x \|Ax - b\|_2 \tag{1.2}$$

donde $\|\ \|_2$ es la norma euclídea. Este problema es conocido como el *Problema de Mínimos Cuadrados Lineales (PMCL)* asociado a la matriz A y el vector b. Esta elección de una *mejor* aproximación a una solución de la ecuación (1.1) está motivada por las aplicaciones estadísticas que dieron origen al estudio de los mínimos cuadrados. Para una introducción a los problemas de mínimos cuadrados desde el enfoque de la estadística, ver [40], [21] y [41].

En el resto de este capítulo, se estudiarán los fundamentos teóricos del PMCL y su soluciones. El trabajo se restringirá al PMCL en el caso de matrices $A \in \mathbb{R}^{m\times n}$ de rango completo, esto es, $rango(A) = n$. El lector interesado en métodos numéricos para la resolución del PMCL para matrices con $rango(A) < n$ (conocidas en la literatura como *matrices de rango deficiente*) puede consultar [13] y [5].

1.1. Las ecuaciones normales

A partir de la resolución problema de optimización (1.2) obtiene un sistema de ecuaciones lineales asociado al PMCL, el cual se conoce como *sistema de ecuaciones normales* o simplemente *ecuaciones normales*, tal como se define a continuación.

Definición 1.1.1. *Sea $A \in \mathbb{R}^{m\times n}$ y $b \in \mathbb{R}^m$. El sistema de n ecuaciones y n incógnitas*

$$A^T Ax = A^T b \tag{1.3}$$

Se conoce como sistema de ecuaciones normales *asociado al PMCL.*

Obsérvese que para el caso de una matriz A de rango completo, la matriz $A^T A$ es una matriz *simétrico positivo definida (SPD)*, lo cual implica que todos sus autovalores son positivos y por lo tanto el sistema (1.3) tiene solución única.

La importancia de las ecuaciones normales radica en que la solución de (1.3) es la solución del PMCL asociado a A y b, tal como se establece en el siguiente teorema.

Teorema 1.1.1. *Si $x \in \mathbb{R}^n$ es solución de las ecuaciones normales, para $A \in \mathbb{R}^{m\times n}$ de rango completo y $b \in \mathbb{R}^m$, entonces, x es solución del PMCL asociado a A y b.*

Demostración. Ver [5]. □

Este último teorema, permite obtener una solución cerrada para el PMCL en el caso de que A sea de rango completo: el vector x que satisface la ecuación (1.2) viene dado por

$$x = \left(A^T A\right)^{-1} A^T b \tag{1.4}$$

1.2. La condición de ortogonalidad

A través de las ecuaciones normales, presentadas en la sección anterior, puede deducirse una importante propiedad que caracteriza a las soluciones del PMCL asociado a $A \in \mathbb{R}^{m\times n}$ y $b \in \mathbb{R}^m$. Dicha propiedad se conoce como *la condición de ortogonalidad* de las soluciones del PMCL y se establece en el siguiente teorema.

Teorema 1.2.1. *Sea*

$$\mathcal{S} = \{x \in \mathbb{R}^n : \|Ax - b\|_2 \text{ es mínima}\}$$

El conjunto de soluciones al PMCL asociado a A y b. Entonces, $x \in \mathcal{S}$ si y solo si x satisface la siguiente condición de ortogonalidad*:*

$$A^T(b - Ax) = 0$$

Demostración. ($\Rightarrow$) Suponga que $\hat{x} \in \mathcal{S}$ y que $\hat{x}$ no satisface la condición de ortogonalidad. Esto es, $A^T\hat{r} = z \neq 0$, donde $\hat{r} = b - A\hat{x}$. Considere $x = \hat{x} + \epsilon z$ para algún $\epsilon > 0$. Así

$$\begin{aligned} r &= b - Ax \\ &= b - A\left(\hat{x} + \epsilon z\right) \\ &= (b - A\hat{x}) + \epsilon Az \\ &= \hat{r} - \epsilon Az \end{aligned}$$

y, por otro lado

$$\begin{aligned} \|Ax - b\|_2 &= r^T r \\ &= \hat{r}^T\hat{r} - 2\epsilon z^T z + \epsilon^2 (Az)^T (Az) \end{aligned}$$

Para $\epsilon \ll 1$ el término que tiene ϵ^2 es despreciable y por lo tanto

$$\|Ax - b\|_2 \approx \hat{r}^T\hat{r} - 2\epsilon z^T z < \hat{r}\hat{r}^T = \|A\hat{x} - b\|_2$$

Esto es, $\hat{x}$ no minimiza $\|A\hat{x} - b\|_2$, lo cual es una contradicción. Por lo tanto, queda probado, que $x \in \mathcal{S}$ satisface la condición de ortogonalidad.

($\Leftarrow$) Suponga que $\hat{x}$ satisface la condición de ortogonalidad $A^T\hat{r} = 0$ con $\hat{r} = b - A\hat{x}$. Sea $x \in \mathbb{R}^n$ arbitrario, se tiene:

$$\begin{aligned} r &= b - Ax \\ &= b - Ax + A\hat{x} - A\hat{x} \\ &= (b - A\hat{x}) + A\left(\hat{x} - x\right) \\ &= \hat{r} + Ae \end{aligned}$$

Calculando la norma de r, a partir de la expresión anterior se tiene

$$\begin{aligned} \|b - Ax\|_2 &= r^T r \\ &= \left(\hat{r} + Ae\right)^T \left(\hat{r} + Ae\right) \\ &= \hat{r}^T r + \|Ae\|_2 \end{aligned}$$

de donde se concluye que $\|b - Ax\|_2$ se minimiza cuando $e = 0$, esto es, $\hat{x} - x = 0$. Luego $\hat{x}$ es solución del PMCL. □

Con este resultado, se tiene una importante herramienta teórica para el estudio de los métodos numéricos para resolver el PMCL, pues provee una condición necesaria y suficiente para verificar que un vector $x \in \mathbb{R}^m$ sea solución de la ecuación (1.2).

1.3. Propiedades geométricas de la solución del PMCL

Para ilustrar la interpretación geométrica de la solución al PMCL, considere los subespacios *fundamentales* asociados a la matriz $A \in \mathbb{R}^{m\times n}$:

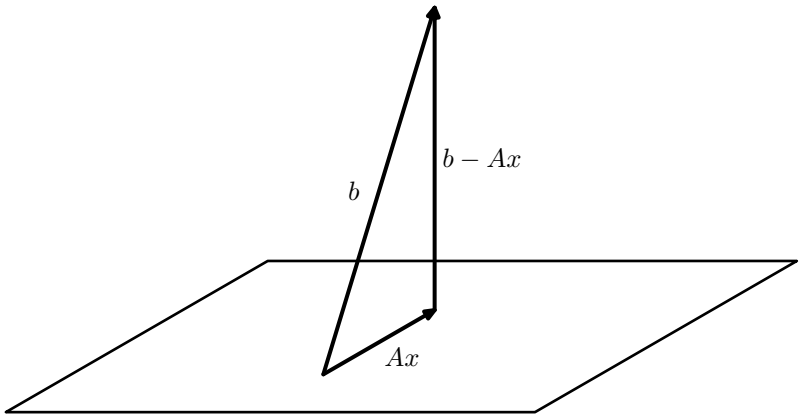

Figura 1.1: Significado geométrico de la solución del PMCL

- $\mathcal{R}(A) = \{z \in \mathcal{R}^n : \text{ para algún } \ x \in \mathbb{R}^m \ Ax = z\}$, el *recorrido* o *espacio columna* de A
- $\mathcal{N}(A) = \{x \in \mathcal{R}^m : Ax = 0\}$, la *nulidad* o *kernel* de A

Obsérvese que si x es una solución del PMCL, entonces la condición de ortogonalidad implica que

$$r = b - Ax \in \mathcal{N}(A^T)$$

Por otro lado, es evidente que

$$Ax \in \mathcal{R}(A)$$

Así, se tiene que x, una solución al PMCL descompone al vector b en dos componentes ortogonales de la forma:

$$b = Ax + r, \quad Ax \in \mathcal{R}(A), \quad r \in \mathcal{N}(A^T) \tag{1.5}$$

En la figura (1.1) se ilustra esta descomposición ortogonal.

Además, a partir de la solución al PMCL de la ecuación (1.4), se obtiene una transformación lineal que *proyecta* cualquier vector de $\mathbb{R}^m$ en $\mathcal{R}(A)$. La definición formal de proyector se presenta a continuación.

Definición 1.3.1. *Sea S un subespacio de $\mathbb{R}^n$, una matriz $P_S \in \mathbb{R}^{n\times n}$ se denomina* proyector ortogonal sobre S *si:*

- *P_S es simétrica.*
- *P_S es idempotente, esto es $P_S^2 = P_S$*
- *Para todo $x \in \mathbb{R}^n$, $P_S x \in S$*

Puede verificarse fácilmente que si P_S es un proyector ortogonal sobre el subespacio S, entonces, $I_m - P_S$ es un proyector ortogonal sobre $S^\perp$, el complemento ortogonal de S.

Con la definición anterior, se tiene que si x es la solución de la ecuación (1.2), entonces Ax es la proyección ortogonal de b sobre $\mathcal{R}(A)$, así, $r = (I_m - P_{\mathcal{R}(A)})$, y usando la ecuación (1.4) se obtiene que

$$P_{\mathcal{R}(A)} = A\left(A^T A\right)^{-1} A^T \tag{1.6}$$

1.4. La matriz pseudoinversa

La matriz pseudoinversa es una *extensión* del concepto de matriz inversa para el caso de matrices rectangulares. Esta matriz es un caso particular de las *Matrices inversas generalizadas*, las cuales se definen a continuación.

Definición 1.4.1. *Sea $A \in \mathbb{R}^{m\times n}$. Una matriz $A^- \in \mathbb{R}^{n\times m}$ es una* inversa generalizada *de A si satisface:*

$$AA^-A = A$$

En esta sección se definirá la matriz pseudoinversa y se presentarán varias de sus propiedades, que serán de utilidad posteriormente cuando se analice el precondicionamiento del PMCL por aproximaciones a la pseudoinversa.

Finalmente, se estudiará la *Descomposición en Valores Singulares* (*SVD*). Esta es una factorización que existe para cualquier matriz $A \in \mathbb{R}^{m\times n}$.

1.4.1. Definición de la pseudoinversa

A continuación se definirá la matriz pseudoinversa de A en función de los proyectores ortogonales asociados a los subespacios $\mathcal{R}(A)$ y $\mathcal{N}(A)$ introducidos en la sección anterior.

Definición 1.4.2. *Sea $A \in \mathbb{R}^{m\times n}$ la* matriz pseudoinversa de A *o* matriz de Moore-Penrose *$A^\dagger \in \mathcal{R}^{n\times m}$ es una matriz que satisface:*

1. *$P \equiv A^\dagger A$ es el proyector ortogonal sobre $\mathcal{N}(A)^\perp$*
2. *$\bar{P} \equiv AA^\dagger$ es el proyector ortogonal sobre $\mathcal{R}(A)^\perp$*

La matriz pseudoinversa satisface una serie de propiedades análogas a las de la matriz inversa, también conocidas como *Condiciones de Penrose*. El siguiente teorema establece dichas propiedades.

Teorema 1.4.1. *Sea $A \in \mathbb{R}^{m\times n}$. La pseudoinversa $A^\dagger$ satisface las siguientes propiedades (Condiciones de Penrose):*

1. *$A^\dagger A$ es simétrica.*
2. *$AA^\dagger$ es simétrica.*
3. *$AA^\dagger A = A$.*
4. *$A^\dagger AA^\dagger = A$.*

Demostración. Ver [39] □

Existen otras propiedades relevantes de la pseudoinversa, que pueden deducirse de las condiciones dadas en el teorema (1.4.1) y que también mantienen una analogía con las propiedades de la inversa para el caso de matrices cuadradas.

Teorema 1.4.2. *Sea $A \in \mathbb{R}^{m\times n}$. La pseudoinversa $A^\dagger$ satisface las siguientes propiedades:*

1. $\left(A^\dagger\right)^\dagger = A$.
2. $\left(A^\dagger\right)^T = \left(A^T\right)^\dagger$.
3. $(\alpha A)^\dagger = \alpha^{-1}A^\dagger$ *Para todo* $\alpha \in \mathbb{R}$ *tal que* $\alpha \neq 0$
4. $\left(A^T A\right)^\dagger = A^\dagger \left(A^\dagger\right)^T$.
5. *Si* $U \in \mathbb{R}^{n\times m}$ *y* $V \in \mathbb{R}^{m\times n}$ *son matrices unitarias, esto es* $U^{-1} = U^T$ *y* $V^{-1} = V^T$, $\left(UAV^T\right)^\dagger = VA^\dagger U^T$.
6. *Si* $AA^T = A^T A$ *entonces* $A^\dagger A = AA^\dagger$ *y* $(A^n)^\dagger = \left(A^\dagger\right)^n$ *para* $n \in \mathbb{Z}$.
7. $rango\,(A) = rango\left(A^T\right) = rango\left(A^\dagger\right) = rango\left(A^\dagger A\right) = tr\left(A^\dagger A\right)$.

Demostración. Ver [5] □

1.4.2. La Descomposición en Valores Singulares

La *Descomposición en Valores Singulares (SVD)* de una matriz $A \in \mathbb{R}^{m\times n}$ es una herramienta de gran importancia teórica para el estudio del PMCL. La SVD provee una extensión al concepto de *Diagonalización* de matrices cuadradas, esto es, una transformación ortogonal de A a una matriz diagonal de $n \times n$ que preserva la norma $\| \ \|_2$. El siguiente resultado establece la existencia de la *SVD* para cualquier matriz rectangular.

Teorema 1.4.3. *Sea $A \in \mathbb{R}^{m\times n}$ con $rango\,(A) = r \leq n$, entonces existen matrices unitarias $U \in \mathbb{R}^{m\times m}$ y $V \in \mathbb{R}^{n\times n}$ y una matriz $\Sigma \in \mathbb{R}^{m\times n}$ tales que:*

$$A = U\Sigma V^H, \quad \Sigma = \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix}, \quad \Sigma_1 = \begin{pmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ & & \ddots & \\ 0 & 0 & \dots & \sigma_r \end{pmatrix} \in \mathbb{R}^{r\times r}$$

Los elementos no nulos en la diagonal principal del bloque Σ_1 satisfacen $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n > 0$ y se denominan valores singulares *de A.*

Demostración. Ver [15]. □

Obsérvese que si $U = (u_1 \cdots u_r)$ y $V = (v_1, \cdots v_r)$ donde u_i y v_i son las columnas de U y V respectivamente, SVD de A se puede reescribir de la forma:

$$A = U\Sigma H = \sum_{i=1}^{r} \sigma_i u_i v_i^H \tag{1.7}$$

Así, se tiene que la SVD de A induce a una descomposición de A como la suma de $r = rango(A)$ matrices de rango 1 (estas son las matrices $\sigma_i u_i v_i$)

La SVD de A, es de interés, ya que permite definir la solución al PMCL en función de las matrices U, V y Σ mencionadas en el teorema (1.4.3)

Teorema 1.4.4. *Considere el problema general de mínimos cuadrados lineales:*

$$\min_{x\in\mathcal{S}} \|x\|_2, \quad \mathcal{S} = \{x \in \mathbb{R}^n : \|Ax - b\|_2 \text{ es mínima}\}$$

donde $A \in \mathbb{C}^{m\times n}$ y $rango(A) = r \leq \min(m, n)$. Este problema siempre tiene una solución única x, la cual se puede escribir en términos de las matrices resultantes de la SVD de A como:

$$x = V \begin{pmatrix} \Sigma_r^{-1} & 0 \\ 0 & 0 \end{pmatrix} U^H b$$

Donde U^H denota la matriz traspuesta conjugada *de U.*

Demostración. En vista de que V y U son matrices unitarias, se tiene que $VV^H = I_n$. Así

$$b - Ax = b - AVV^H x$$

Por otro lado

$$\begin{aligned} \|b - Ax\|_2 &= \|U^H (b - Ax)\|_2 \\ &= \|U^H \left(b - AVV^H\right)\|_2 \end{aligned}$$

Ahora considere:

$$z = V^H x = \begin{pmatrix} z_1 \\ z_2 \end{pmatrix}, \quad c = UHb = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} \left[U^H b\right]_r \\ \left[U^H b\right]_{m-r} \end{pmatrix}$$

Donde $z_1, c_1 \in \mathbb{C}^r$. Con lo anterior, podemos reescribir la norma del residual $b - Ax$ como:

$$\begin{aligned} \|b - Ax\|_2 &= \|U^H \left(b - AVV^H x\right)\|_2 \\ &= \|U^H b - \left(U^H AV\right) V^H x\|_2 \\ &= \left\| \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} - \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \right\|_2 \\ &= \left\| \begin{pmatrix} c_1 - \Sigma_1 z_1 \\ c_2 \end{pmatrix} \right\|_2 \end{aligned}$$

Puede observarse que para c_1 y c_2 fijos, el residual se minimiza cuando $c_1 - \Sigma_1 z_1 = 0$, esto es, $z_1 = \Sigma_1^{-1} c_1$, mientras que z_2 puede ser cualquier valor. En particular, si se escoge $z_2 = 0$, también se minimiza $\|z\|_2$, lo cual implica que se minimiza $\|x\|_2$, siendo el valor de x que

minimiza la norma igual a:

$$\begin{aligned} x = Vz &= V \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \\ &= V \begin{pmatrix} \Sigma_1^{-1} c_1 \\ 0 \end{pmatrix} \\ &= V \begin{pmatrix} \Sigma_1^{-1} \left[U^H b\right]_r + 0 \left[U^H b\right]_{m-r} \\ 0 \left[U^H b\right]_r + 0 \left[U^H b\right]_{m-r} \end{pmatrix} \\ &= V \begin{pmatrix} \Sigma_1^{-1} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \left[U^H b\right]_r \\ \left[U^H b\right]_{m-r} \end{pmatrix} \\ &= V \begin{pmatrix} \Sigma_1^{-1} & 0 \\ 0 & 0 \end{pmatrix} U^H b \end{aligned}$$

□

La construcción de la solución del problema de mínimos cuadrados generalizados en la demostración del teorema (1.4.4), define la matriz

$$C = V \begin{pmatrix} \Sigma_1^{-1} & 0 \\ 0 & 0 \end{pmatrix} U^H \tag{1.8}$$

La cual resulta ser $A^\dagger$, tal como se deduce del siguiente resultado:

Teorema 1.4.5. *La matriz*

$$C = V \begin{pmatrix} \Sigma_1^{-1} & 0 \\ 0 & 0 \end{pmatrix} U^H$$

Satisface las condiciones de Penrose establecidas en el teorema (1.4.1) y por consiguiente, $C = A^\dagger$.

Demostración. Ver [5] □

Con este último resultado, y a partir del teorema (1.4.4) se deriva que la solución del PMCL se puede expresar, en términos de $A^\dagger$ como:

$$x = A^\dagger b \tag{1.9}$$

Para el caso de A de rango completo, usando la ecuación (1.4) se obtiene una expresión para $A^\dagger$.

$$A^\dagger = \left(A^T A\right)^{-1} A^T \tag{1.10}$$

Obsérvese que la cálculo de la SVD provee desde el punto de vista numérico de un método directo para la resolución del PMCL. Sin embargo,el cálculo de la SVD tiene un altísimo costo computacional, lo cual hace que el enfoque de la SVD sea poco atractivo para la resolución del PMCL en el caso general. No obstante, para el caso de matrices densas de rango deficiente, la SVD es la primera elección para la resolución del PMCL. Ver [15] para un estudio de los métodos clásicos para el cálculo de la SVD y sus aplicaciones al PMCL en matrices de rango deficiente.

Cabe destacar, además, que la SVD por sí sola, es un tópico de gran interés en el álgebra lineal numérica debido a la cantidad de aplicaciones existentes en áreas como el tratamiento digital de imágenes, filtros digitales, reconocimiento de patrones y series temporales, entre otras. Para un estudio más detallado de la SVD y referencias a sus distintas aplicaciones ver [24] y [1]. Para una introducción al estudio de los orígenes y desarrollo histórico de la SVD ver [36].

Para finalizar este capítulo, se demostrará una de las propiedades más importantes de la matriz pseudoinversa: $AA^\dagger \approx I_n$ en el sentido de que la distancia entre $AA^\dagger$ e I_n es la mínima posible respecto a AX con $X \in \mathbb{R}^{n\times m}$.

Teorema 1.4.6. *$X = A^\dagger$ minimiza la norma $\|AX - I_m\|_F$, donde $\| \ \|_F$ es la norma de Frobenious.*

Demostración. Sea X la matriz que minimiza la norma. Observe que:

$$\|AX - I\|_F = \sum_{i=1}^{m} \|Ax_i - e_i\|_2^2$$

Donde x_i es la i-ésima columna de X y e_i es el i-ésimo vector de la base canónica de $\mathbb{R}^m$. Para todo i en $1 \dots n$, el término $\|Ax_i - e_i\|_2$ se minimiza cuando x_i es solución al PMCL $Ax = e_i$. Por la ecuación (1.10), la solución del i-ésimo PMCL es $x_i = A^\dagger e_i$, es decir, x_i es la i-ésima columna de $A^\dagger$. Luego, $X = A^\dagger$. □

Capítulo 2

Métodos Numéricos Clásicos para el PMCL

Donde quiera que haya un número está la belleza

Proclo

En este capítulo se dará una breve introducción a los métodos numéricos más conocidos para la resolución del PMCL que se encuentranen la literatura. Los métodos numéricos para el PMCL, se dividen en dos grandes familias: *métodos directos*, los cuales se aplican principalmente a matrices densas y los *métodos iterativos* usados principalmente en matrices *sparse*. También, se revisarán algunas ideas del precondicionamiento de las ecuaciones normales aplicado al PMCL.

El principal compendio de métodos numéricos para el PMCL es [5]; los métodos directos clásicos como la factorización QR para matrices rectagulares son también cubiertos por [15]. La teoría general para los métodos iterativos estacionarios así como del método del Gradiente Conjugado puede consultarse en [33]. Esta obra igualmente aborda los métodos iterativos para la resolución del sistema (1.3) en particular. Un artículo introductorio al tema del precondicionamiento de matrices es [3]; para el caso particular del precondicionamiento de las ecuaciones normales, puede consultarse [34] y [5]. Un bosquejo histórico del desarrollo de los métodos iterativos para la resolución de sistemas lineales, desde un punto de vista computacional, se encuentra en [35].

Previo a la presentación de los métodos para el PMCL, se introducirá un concepto de gran importancia para el estudio de los métodos numéricos: el *condicionamiento numérico*.

El *condicionamiento numérico* de un problema, se refiere a una medida de la *sensibilidad* que puede tener la solución de dicho problema a pequeñas perturbaciones en los datos de entrada. Para el caso de los sistemas de ecuaciones lineales, sea $M \in \mathbb{R}^{n\times n}$ y $b \in \mathbb{R}^n$ y supóngase que se desea resolver el sistema

$$Mx = b \tag{2.1}$$

si los valores numéricos de M y b provienen, por ejemplo, de algún experimento, estos serán susceptibles de tener alguna perturbación debido al redondeo u otros errores experimentales como instrumentos de medición mal calibrados, una lectura incorrecta por parte del experimentador, o la propagación de errores en mediciones indirectas, entre otros. Por lo tanto, en

la práctica, en lugar de resolver el sistema (2.1) se tendrá que resolver el siguiente sistema asociado

$$(M + \Delta M)x = b + \Delta b \tag{2.2}$$

donde ΔM y Δb son una matriz y un vector que contienen los errores experimentales o perturbaciones asociados a M y b respectivamente. Entonces, se dice que el sistema (2.1) está *bien condicionado* si el error relativo de la solución del sistema perturbado (2.2) respecto a la solución del sistema original es pequeño. En caso contrario, se dice que el problema está *mal condicionado*.

Una medida para el condicionamiento de una matriz en el caso general, es el *número de condición*, el cual se define a continuación

Definición 2.0.3. *Sea $A \in \mathbb{R}^{n\times n}$, no sigular, y sea $\| \ \|$ una norma matricial inducida. El* número de condición de *A, el cual se denota como $\kappa(A)$ se define como*

$$\kappa(A) \equiv \|A\|\|A^{-1}\|$$

Para el caso de una matriz rectangular $A \in \mathbb{R}^{m\times n}$ el número de condición se define como

$$\kappa(A) \equiv \|A\|\|A^{\dagger}\|$$

Obsérvese que para toda matriz A, se cumple que $\kappa(A) \geq 1$.

El número de condición da una medida del condicionamiento de un sistema de ecuaciones lineales que involucre a la matriz A. Si $\kappa(A) \approx 1$ (esto es, un número de condición *pequeño*) entonces la matriz está bien condicionada, mientras que un número de condición que sea mucho mayor a 1 (esto es, un número de condición *grande*) indica que la matriz está mal condicionada, y por lo tanto el resultado de un método numérico que involucre a dicha matriz podría ser poco confiable.

Además, debe destacarse que en el caso del PMCL asociado a una matriz A y un vector b, el condicionamiento del problema no depende sólo de A, sino que depende además del valor de b, por lo cual se puede obtener una *extensión* del número de condición para el PMCL (Ver [5]) tal como se define a continuación

Definición 2.0.4. *Sea $A \in \mathbb{R}^{m\times n}$ y $b \in \mathbb{R}^n$. Si x_{LS} es una solución al PMCL asociado a A y b, entonces el número de condición del PMCL, usando la norma euclídea, viene dado por*

$$\kappa_{LS}(A, b) \equiv \kappa(A)\left(1 + \kappa(A)\frac{\|r\|_2}{\|A\|_2\|x_{LS}\|_2}\right)$$

donde $r = b - Ax_{LS}$.

2.1. Métodos directos para el PMCL

2.1.1. Factorización de Cholesky

El método directo más simple para la resolución de las ecuaciones normales se basa en la factorización de Cholesky, la cual existe para cualquier matriz SPD, tal como se establece en el siguiente teorema.

Teorema 2.1.1. *Sea $C \in \mathbb{R}^{n\times n}$ una matriz SPD, entonces existe una única matriz $R \in \mathbb{R}^{n\times n}$ triangular superior tal que:*

$$C = R^T R$$

La matriz R se denomina Factor de Cholesky *asociado a C.*

Demostración. Ver [22] □

Para el caso de una matriz C almacenada por filas, la factorización de Cholesky se puede calcular con el algoritmo (2.1).

Algoritmo 2.1 Factorización de Cholesky de la matriz A

- **Algoritmo**
  FactChol
- **Descripción**
  Calcula la factorización de Cholesky de una matriz A SPD.
- **Entradas**
  - A Matriz de n filas por n columnas SPD.
- **Salidas**
  - R Matriz triangular de n filas por n columnas tal que $A = R^T R$.

1: **función FactChol**(A)
2: $R \leftarrow 0$
3: **para** $j \leftarrow 1 \dots n$ **hacer**
4: **para** $i \leftarrow 1 \dots j-1$ **hacer**
5: $r_{ij} \leftarrow c_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj}$
6: $r_{ij} \leftarrow r_{ij}/r_{ii}$
7: **fin para**
8: $r_{jj} \leftarrow \left(c_{jj} - \sum_{k=1}^{j-1} r_{kj}^2\right)^{1/2}$
9: **fin para**
10: $\leftarrow R$
11: **fin función**

Finalmente, conociendo la factorización de Cholesky de A, el algoritmo (2.2) define un método para la resolución del PMCL.

2.1.2. Descomposición QR

Uno de los métodos directos más utilizados para la resolución del PMCL en sistemas sobredeterminados, es el de la descomposición ortogonal de la matriz A. Esta estrategia es factible ya que, de que si $Q \in \mathbb{R}^{m\times m}$ es una matriz ortogonal, esta preserva la norma euclídea y por consiguiente, el problema de optimización

$$\min_x \left\| Q^T (Ax - b) \right\|_2 \tag{2.3}$$

Algoritmo 2.2 Solución del PMCL vía factorización de Cholesky

- **Algoritmo**
  CholPMCL
- **Descripción**
  Resuelve el PMCL asociado a A y b usando la factorización de Cholesky.
- **Entradas**
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - b Vector columna de n filas.
- **Salidas**
  - x_{LS} Solución al PMCL asociado a A y b.

1: **función CholPMCL**(A, b)
2: $R \leftarrow$ **FactChol**(C)
3: $d \leftarrow A^T b$
4: Resolver el sistema $Ry = d$
5: Resolver el sistema $R^T x_{LS} = y$
6: $\leftarrow x_{LS}$
7: **fin función**

es equivalente al planteado en la ecuación (1.2). El objetivo es, entonces, hallar alguna matriz $Q \in \mathbb{R}^{m\times m}$ tal que la ecuación (2.3) sea más fácil de resolver que la ecuación (1.2).

En general, para cualquier matriz rectangular, existe una descomposición ortogonal conocida como *descomposición QR* cuya existencia se garantiza por el siguiente teorema

Teorema 2.1.2. *Sea $A \in \mathbb{R}^{m\times n}$ con $m \geq n$, entonces existe una matriz ortogonal $Q \in \mathbb{R}^{m\times m}$ y una matriz triangular $R \in \mathbb{R}^{n\times n}$ tales que*

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$$

esta factorización se conoce como descomposición QR *de A y la matriz R se conoce como* factor R *asociado a A.*

Demostración. Ver [5] □

En los siguientes apartados se estudiarán algunos de los principales métodos numéricos para la descomposición QR de A y su respectiva aplicación al PMCL

2.1.3. Descomposición QR vía transformaciones ortogonales

Una estrategia para hallar la descomposición QR de A, consiste en hallar matrices $P_1, P_2, \ldots, P_n$ de dimensión $m \times m$, tales que $Q = P_n P_{n-1} \ldots P_1$, donde P_i es un proyector ortogonal sobre algún subespacio de $\mathcal{R}(A)$, tal como se describe a continuación.

Para comenzar, suponga que A se puede descomponer a partir de la siguiente relación de recurrencia

$$\begin{aligned} A^{(1)} &= A \\ A^{(k+1)} &= P_k A^{(k-1)}, \quad k = 1 \dots n \end{aligned} \tag{2.4}$$

donde P_k, $k = 1, 2, \dots$ es una matriz ortogonal que debe construirse de tal forma que $A^{(k+1)}$ tenga una estructura triangular superior sobre las primeras k columnas, es decir, $A^{(k+1)}$ debe tener la estructura

$$A^{(k+1)} = P_k P_{k-1} \dots P_1 A = \begin{pmatrix} R_{11} & R_{12} \\ 0 & \widetilde{A}^{(k+1)} \end{pmatrix} \tag{2.5}$$

donde $R_{11} \in \mathbb{R}^{k \times k}$ es una matriz triangular superior y la matriz $\widetilde{A}^{(k+1)} \in \mathbb{R}^{(m-k)\times(n-k-1)}$ es ortogonal. Si la estructura de $\widetilde{A}^{(k)}$ por columnas es: $\widetilde{A}^{(k)} = (\widetilde{a}_k^{(k)}, \widetilde{a}_{k+1}^{(k)}, \dots, \widetilde{a}_n^{(k)})$ entonces, la matriz P_k debe tener la siguiente forma

$$P_k = \begin{pmatrix} I_{k-1} & 0 \\ 0 & \widetilde{P}_k \end{pmatrix} \tag{2.6}$$

donde $\widetilde{P}_k$ es una matriz ortogonal de tamaño $(m-k+1) \times (m-k+1)$ tal que

$$P_k \widetilde{a}_k^{(k)} = \sigma_k e_1, \quad \sigma_k = r_{kk} = \|\widetilde{a}_k^k\|_2^2, \quad e_1 = (1, 0, \dots 0)^T$$

Obsérvese que en el k-ésimo paso de la recurrencia de la ecuación (2.5) la transformación ortogonal sólo afecta a la submatriz $\widetilde{A}^{(k+1)}$ de A y (R_{11}, R_{12}) son las primeras $k-1$ filas de R. Después de n pasos se obtiene

$$A^{(n+1)} = P_n P_{n-1} \dots P_1 A = \begin{pmatrix} R \\ 0 \end{pmatrix} \tag{2.7}$$

de donde se deduce que

$$Q = (P_n P_{n-1} \dots P_1)^{-1} = (P_n P_{n-1} \dots P_1)^T = P_1^T \dots P_n^T \tag{2.8}$$

Una vez calculada la descomposición QR a través de transformaciones ortogonales, el PMCL se resuelve a través del *método de Golub* [16] cuya implementación (restringida al caso de matrices de rango completo) se presenta en el algoritmo (2.3).

Para efectuar la descomposición QR de A falta conocer algíun método que permita calcular la colección de matrices $\widetilde{P}_k$. Los métodos numéricos más populares para la obteción de matrices de transformación ortogonal son los siguientes:

- Transformaciones ortogonales de Householder.
- Matrices de rotación de Givens

El lector interesado en los diversos algoritmos existentes para la obtención de la descomposición ortogonal de una matriz rectangular a través de los métodos anteriores puede consultar [15] y [5].

Algoritmo 2.3 Método de Golub para la solución del PMCL vía descomposición QR

- **Algoritmo**
  GolubPMCL
- **Descripción**
  Resuelve el PMCL asociado a A y b usando la factorización QR.
- **Entradas**
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - b Vector columna de n filas.
- **Salidas**
  - x_{LS} Solución al PMCL asociado a A y b.

1: **función GolubPMCL**(A, b)
2: $[Q, R] \leftarrow$ **FactQR**(A)
3: $d \leftarrow Q^T b$
4: $c \leftarrow d(1:n)$ ▷ $n < m$
5: Resolver el sistema $Rx_{LS} = c$
6: $\leftarrow x_{LS}$
7: **fin función**

2.1.4. Descomposición QR vía ortogonalización

Un *proceso de ortogonalización* es un problema clásico del álgebra lineal, el cual tiene como objetivo, dado un conjunto de vectores $V = \{v_1, v_2, \dots v_r\}$ en $\mathbb{R}^n$, con $r \leq n$, hallar un nuevo conjunto $U = \{u_1, u_2, \dots u_r\}$ tal que $span(V) = span(U)$ y los vectores de U son ortogonales entre sí. Si además, los vectores resultantes de U son unitarios, el proceso se denomina *proceso de ortonormalización* El algoritmo clásico para la ortonormalización de un conjunto de vectores es el *método de Gram Schmidt* (Ver [18]).

Dada una matriz $A \in \mathbb{R}^{m \times n}$ con $m \geq n$, se puede utilizar un proceso de ortonormalización sobre las columnas de A para obtener su descomposición QR, de la siguiente manera: si $A = (a_1, a_2, \dots, a_n)$, donde a_i es la i-ésima columna de A, entonces

$$A = (q_1, q_2, \dots q_n)R \tag{2.9}$$

donde $Q = (q_1, q_2, \dots q_n)$ se obtiene a partir de la ortonormalización de las columnas de A. Evidentemente Q será una matriz ortogonal.

En esta sección se presentará el *método de Gram Schmidt modificado* para obtener la descomposición QR de A a través de a ortogonalización de sus columnas.

La estrategia del método de Gram Schmidt modificado, consiste en construir una sucesión de matrices $A = A^{(1)}, A^{(2)}, \dots A^{(n)} = Q$ donde $A^{(k)}$ tiene la forma

$$A^{(k)} = \left(q_1, q_2, \dots q_{k-1}, a_k^{(k)}, \dots, a_n^{(k)}\right)$$

Así, las k primeras columnas de $A^{(k)}$ correponden a los k primeros pasos del proceso de

ortonormalización de las columnas de A. En el k-ésimo paso, en primer lugar se obtiene el vector q_k normalizando el vector $a_k^{(k)}$, esto es

$$\widetilde{q}_k = a_k^{(k)}, \quad r_{kk} = \|\widetilde{q}_k\|_2, \quad q_k = \widetilde{q}_k / r_{kk} \tag{2.10}$$

y luego se ortogonalizan $a_{k+1}^{(k)}, \ldots, a_n^{(k)}$ respecto a q_k como sigue

$$a_j^{(k+1)} = a_j^{(k)} - r_{kj} q_k, \quad r_{kj} = q_k^T a_j^{(k)}, \quad j = k+1, \ldots n \tag{2.11}$$

Existen varias implementaciones algoritmicas del método de Gram Schmidt modificado para la descomposición QR de una matriz rectangular. Los detalles de estos algoritmos pueden consultarse en [5] y [22].

2.2. Métodos iterativos para el PMCL

En esta sección se considerarán varios métodos iterativos clásicos, con algunas modificaciones para la solución de la ecuación (1.3). El objetivo de los métodos iterativos es generar una sucesión de soluciones aproximadas $\left\{x^{(0)}, x^{(1)}, \cdots\right\}$ donde $x^{(0)}$ es una aproximación inicial dada a la solución o *iterado inicial* y $x^{(k)}$ para $k \geq 1$ puede calcularse a través de una relación de recurrencia bien definida, en función de $x^{(0)}, x^{(1)}, \cdots, x^{(k-1)}$. Otra característica deseable es que el cálculo de $x^{(n)}$ sea de bajo costo computacional.

Una condición necesaria para que un método iterativo pueda dar la solución de un sistema de ecuaciones lineales, es que el método sea *convergente*. La definición formal de convergencia es la siguiente

Definición 2.2.1. *Un método iterativo, con un iterado inicial $x^{(0)}$ y una relación de recurrencia bien definida para el cálculo de $x^{(k)}$ para $k \geq 1$ es* convergente *si la sucesión $\left\{x^{(k)}\right\}_{k=0}^{\infty}$ es convergente para cualquier iterado inicial $x^{(0)}$.*

Una de las propiedades de los métodos iterativos, es que el proceso de generación de los $x^{(k)}$ puede detenerse en cualquier momento, así se tiene la ventaja respecto a los métodos directos, en el hecho de que el método iterativo tenga una menor complejidad en tiempo en el caso de que se deseen soluciones de poca precisión numérica, lo cual implica un número reducido de iteraciones, o bien, para el caso de matrices de dimensión *grande* el número de iteraciones para observar convergencia puede ser mucho menor que el tamaño de la matriz.

Para el caso de los métodos iterativos aplicados a las ecuaciones normales, es deseable que estos eviten la formación explícita de la matriz $A^T A$, y para ello se suele trabajar con las ecuaciones normales en su *forma factorizada* que es

$$A^T (Ax - b) \tag{2.12}$$

Trabajar con A^T y A de manera separada, tiene varias ventajas: En primer lugar, la matriz $A^T A$ suele tener un número de condición mucho mayor a A ($\kappa\left(A^T A\right) = \kappa(A)^2$ en el caso de que se tome el número de condición usando la norma euclídea), por lo cual una pequeña perturbación en $A^T A$ podría producir un error considerable en la resolución del problema. En segundo lugar, para el caso de matrices *sparse* se evita el relleno que aparece al formar la matriz $A^T A$.

2.2.1. Métodos Estacionarios

Los métodos iterativos más sencillos para la resolución de sistemas de ecuaciones lineales son los métodos estacionarios. Un método estacionario para la resolución de (1.3) se construye definiendo un par de matrices M y N asociadas a A^TA y la iteración:

$$Mx^{(k+1)} = Nx^{(k)} + b,\ k = 0, 1, \ldots \tag{2.13}$$

Las matrices M y N de la ecuación (2.13) son tales que $A^TA = M - N$ y además se impone la condición de que M sea no singular. Este esquema de definición de M y N propio de los métodos estacionarios también es conocido como *particionamiento* o *splitting regular* de las ecuaciones normales.

Para el análisis de la convergencia de los métodos estacionarios, se define:

$$G = M^{-1}N = I - M^{-1}A^TA \tag{2.14}$$

$$c = M^{-1}b \tag{2.15}$$

Así, (2.13) puede reescribirse como:

$$x^{(k+1)} = Gx^{(k)} + c,\ k = 0, 1, \ldots \tag{2.16}$$

La matriz G se conoce como *matriz de iteración* asociada al método estacionario. Usando dicha matriz de iteración, se puede establecer una condición necesaria y suficiente para la convergencia de un método estacionario, la cual se presenta en el siguiente teorema

Teorema 2.2.1. *El método estacionario de la ecuación (2.16) es convergente si y sólo si* $\rho(G) < 1$.

Demostración. Ver [22] □

A continuación se presentarán algunos de los métodos estacionarios que se consiguen con mayor frecuencia en la literatura, cada uno de estos métodos se define a partir de un *splitting* distinto para la matriz A^TA.

El método de Richardson de primer orden

El método de Richardson de primer orden se define a partir del siguiente *splitting*

$$\begin{aligned} A^TA &= \frac{1}{\alpha}I_n - \left(\frac{1}{\alpha}I_n - A^TA\right) \\ M &= \frac{1}{\alpha}I_n \\ N &= \frac{1}{\alpha}I_n - A^TA \end{aligned} \tag{2.17}$$

Donde $\alpha > 0$ es un parámetro fijo. Para este método se tiene que $G = I - \alpha A^TA$ y $c = \alpha A^Tb$, así, luego de una manipulación algebraica de la ecuación (2.16) se obtiene la siguiente iteración:

$$x^{(k+1)} = x^{(k)} + \alpha A^T \left(b - Ax^{(k)}\right) \tag{2.18}$$

En donde no hace falta realizar el producto $A^T A$ de manera explícita, el algoritmo (2.4) sumariza el método.

Algoritmo 2.4 Método de Richardson de primer orden para las ecuaciones normales

- **Algoritmo**
  RICHARDSON1ORDPMCL
- **Descripción**
  Resuelve el PMCL asociado a A y b usando la iteración de Richardsond de primer orden para las ecuaciones normales.
- **Entradas**
  - $x^{(0)}$ Vector columna de n filas, iterado inicial.
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - b Vector columna de n filas.
  - α Real positivo que proviene del *splitting* de A.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - x Solución al PMCL asociado a A y b.

1: **función RICHARDSON1ORDPMCL**$(x^{(0)}, A, b, \alpha, N_{max}, \epsilon)$
2: **para** $i \leftarrow 1, \ldots, N_{max}$ **hacer**
3: $x^{(i)} \leftarrow x^{(i-1)} + \alpha A^T \left(b - Ax^{(i-1)}\right)$
4: $r^{(i)} \leftarrow b - Ax^{(i)}$
5: **si** $\|r^{(i)}\|_2 < \epsilon$ **entonces**
6: $\leftarrow x^{(i)}$
7: **fin si**
8: **fin para**
9: **ERROR** (Máximo de iteraciones alcanzado)
10: **fin función**

Para garantizar que el método de Richardson de primer orden converge a la solución del PMCL, esto es, converge a $x = A^\dagger b$ se establece la siguiente condición

Teorema 2.2.2. *El método de Richardson de primer orden converge a la solución del PMCL si* $x^{(0)} \in \mathcal{R}\left(A^T\right)$ *y* $0 < \alpha < 2/\sigma_1^2$ *donde* σ_1 *es el máximo de los valores singulares de* A.

Demostración. Ver [22] □

El método de Jacobi

El método de Jacobi se define a partir del siguiente *splitting*

$$\begin{aligned} A^TA &= L_A + D_A + L_A^T \\ M &= D_A \\ N &= -\left(L_A + L_A^T\right) \end{aligned} \tag{2.19}$$

Donde D_A es la matriz con la diagonal principal de A^TA, L_A es la matriz que contiene la porción triangular inferior de A^TA y L_A^T es la matriz que contiene la porción triangular superior de A^TA.

Para este método se tiene que $G = I_n - D_A^{-1}A^TA$ y $c = D_A^{-1}A^Tb$. Reescribiendo la ecuación (2.16) para evitar el producto explícito de A^TA, la iteración de Jacobi es la siguiente:

$$x^{(k+1)} = x^{(k)} + D_A^{-1}A^T\left(b - Ax^{(k)}\right) \tag{2.20}$$

Finalmente el método se presenta en el algoritmo (2.5)

El método de Gauss-Seidel

El método de Gauss-Seidel se define a partir del siguiente *splitting*

$$A^TA = L_A + D_A + L_A^T \tag{2.21}$$
$$M = L_A + D_A \tag{2.22}$$
$$N = -L_A \tag{2.23}$$

A partir de las definiciones anteriores, se tiene que $G = -(L_A + D_A)^{-1}L_A^T$ y $c = (L_A + D_A)^{-1}b$, y así la iteración de Gauss-Seidel queda como:

$$x^{(k+1)} = x^{(k)} + D_A^{-1}\left(A^Tb - L_Ax^{(k+1)} + \left(D_A + L_A^T\right)x^{(k)}\right) \tag{2.24}$$

Obsérvese que en este caso, la iteración para $x^{(k+1)}$ incluye el propio término en la relación de recurrencia, por lo cual no es posible manipular algebraicamente la expresión (2.24) para evitar la formación explícita de A^TA. No obstante, se presentará una alternativa para la construcción de la iteración de Gauss-Seidel sin necesidad de realizar produtos matriciales, conocida como *método de reducción de residual* ya que genera una sucesión de aproximaciones $x^{(j)}$ tales que $\|r^{(j+1)}\|_2^2 \leq \|r^{(j)}\|_2^2$ donde $r^{(j)} = b - Ax^{(j)}$.

Sea $\{p_j\}$, $j = 1, 2, \cdots$ una sucesión de vectores no nulos tales que $p_j \notin \mathcal{N}(A)$, para $j = 1, 2, \cdots$ y considérese una sucesión de aproximaciones de la forma

$$x^{(k+1)} = x^{(j)} + \alpha_j p_j,\ \alpha_j = \frac{p_j^TA^T\left(b - Ax^{(j)}\right)}{\|Ap_j\|_2^2} \tag{2.25}$$

Dada una aproximación inicial $x^{(0)}$, puede verificarse que (Ver [5])

$$\|r^{(j+1)}\|_2^2 = |\alpha_j|^2\|Ap_j\|_2^2 \leq \|r^{(j)}\|_2^2 \tag{2.26}$$

En particular, si $A \in \mathbb{R}^{m\times n}$ es de rango completo, el método de Gauss-Seidel para las ecuaciones normales puede derivarse de la ecuación (2.26) escogiendo la sucesión $p_1, p_2, \ldots$

Algoritmo 2.5 Método de Jacobi para las ecuaciones normales

- **Algoritmo**
  JacobiPMCL
- **Descripción**
  Resuelve el PMCL asociado a A y b usando la iteración de Jacobi para las ecuaciones normales.
- **Entradas**
  - $x^{(0)}$ Vector columna de n filas, iterado inicial.
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - b Vector columna de n filas.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - x Solución al PMCL asociado a A y b.

función **JacobiPMCL**$(x^{(0)}, A, b, N_{max}, \epsilon)$
  $r^{(0)} \leftarrow b - Ax^{(0)}$
  para $i \leftarrow 1, \ldots, N_{max}$ **hacer**
    $x^{(i)} = x^{(i-1)} + D_A^{-1} A^T r^{(i-1)}$
    $r^{(i)} \leftarrow b - Ax^{(i)}$
    si $\|r^{(i)}\|_2 < \epsilon$ **entonces**
      $\leftarrow x^{(i)}$
    fin si
  fin para
  Error (Máximo de iteraciones alcanzado)
fin función

Algoritmo 2.6 Método de Gauss-Seidel para las ecuaciones normales

- **Algoritmo**
  GaussSeidelPMCL
- **Descripción**
  Resuelve el PMCL asociado a A y b usando la iteración de Gauss-Seidel para las ecuaciones normales.
- **Entradas**
  - $x^{(0)}$ Vector columna de n filas, iterado inicial.
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - b Vector columna de n filas.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - x Solución al PMCL asociado a A y b.

función GaussSeidelPMCL$(x^{(0)}, A, b, N_{max}, \epsilon)$
  $r^{(0)} = b - Ax^{(0)}$
  para $i \leftarrow 1, \ldots, N_{max}$ **hacer**
    $z^{(1)} \leftarrow x^{(i-1)}$ ▷ Iterado inicial para el barrido de n subiteraciones
    $\widetilde{r}^{(1)} \leftarrow r^{(0)}$
    para $j \leftarrow 1, 2, \cdots, n$ **hacer**
      $a \leftarrow Ae_j$ ▷ j-ésima columna de A
      $d \leftarrow \|a\|_2^2$
      $\delta_j \leftarrow \left(a^T \widetilde{r}^{(j)}\right)/d$
      $z^{(j+1)} \leftarrow z^{(j)} + \delta_j e_j$
      $\widetilde{r}^{(j+1)} \leftarrow \widetilde{r}^{(j)} - \delta_j a$
    fin para
    $x^{(i)} \leftarrow z^{(n+1)}$
    $r^{(i)} \leftarrow \widetilde{r}^{(n+1)}$
    si $\|r^{(i)}\|_2 < \epsilon$ **entonces**
      $\leftarrow x^{(i)}$
    fin si
  fin para
  Error (Máximo de iteraciones alcanzado)
fin función

como $e_1, e_2, \ldots$, donde e_i es el i-ésimo vector de la base canónica en orden cíclico. Con esta elección, se tiene que $Ap_j = Ae_j = a_j$ donde a_j es la j-ésima columna de A.

A partir de la elección anterior, una iteración de Gauss-Seidel se realiza a través de un *barrido* de n sub-iteraciones. Los pasos para obtener la aproximación $x^{(k)}$ son los siguientes:

- Sea $z^{(1)} = x^{(k)}$, y $\widetilde{r}^{(1)} = b - Ax^{(k)}$
- Calcular las $n-1$ subiteraciones restantes $z^{(2)}, \cdots, z^{(n+1)}$ usando la recurrencia:

$$\begin{aligned} \delta_j &= \frac{a_j^T \widetilde{r}^{(j)}}{\|a_j\|_2^2} \\ z^{(j+1)} &= z^{(j)} + \delta_j e_j \\ \widetilde{r}^{(j+1)} &= \widetilde{r}^{(j)} - \delta_j a_j \end{aligned}$$

- Sea $x^{(k+1)} = z^{(n+1)}$

Finalmente, se presenta el método de Gauss-Seidel en el algoritmo (2.6).

El Método de Sobrerrelajación sucesiva (SOR)

El método SOR es una mejora del método de Gauss-Seidel, la cual consiste en introducir un parámetro $\omega \in \mathbb{R}$ a las sub-iteraciones de un barrido en la iteración de Gauss-Seidel con la finalidad de acelerar la convergencia. Dicho factor se conoce como *factor de relajación*. Con la introducción del factor de relajación, las recurrencias (2.27) se convierten en:

$$\delta_j = \omega \frac{a_j^T \widetilde{r}^{(j)}}{\|a_j\|_2^2} \tag{2.27}$$

$$z^{(j+1)} = z^{(j)} + \delta_j e_j \tag{2.28}$$

$$\widetilde{r}^{(j+1)} = \widetilde{r}^{(j)} - \delta_j a_j \tag{2.29}$$

La principal desventaja que presenta este método es que para el caso general no es posible determinar el factor de relajación óptimo para garantizar la máxima aceleración de la convergencia, no obstante, en vista de que la matriz A^TA es SPD, es condición necesaria que $0 < \omega < 2$. El método sor se presenta en el algoritmo (2.7).

Para cierta clase de matrices puede hallarse un factor de relajación óptimo ω_{opt} que garantiza la máxima velocidad de convergencia del de convergencia del método SOR, estas son las *matrices de orden consistente* cuya definición es la siguiente

Definición 2.2.2. *Sea una matriz de la forma $M = A^TA$ donde $A \in \mathbb{R}^{m\times n}$ y considérese una descomposición $M = D_A(I_m - L - U)$ donde:*

- *D_A es una matriz no singular.*
- *L es una matriz estrictamente triangular inferior.*
- *U es una matriz estrictamente triangular superior.*

Algoritmo 2.7 Método SOR para las ecuaciones normales

- **Algoritmo**
  SOR_PMCL
- **Descripción**
  Resuelve el PMCL asociado a A y b usando la iteración SOR para las ecuaciones normales.
- **Entradas**
  - $x^{(0)}$ Vector columna de n filas, iterado inicial.
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - b Vector columna de n filas.
  - ω Real positivo, factor de relajación de SOR.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - x Solución al PMCL asociado a A y b.

función SOR_PMCL$(x^{(0)}, A, b, \omega, N_{max}, \epsilon)$
  $r^{(0)} = b - Ax^{(0)}$
  para $i \leftarrow 1, \ldots, N_{max}$ **hacer**
    $z^{(1)} \leftarrow x^{(i-1)}$ ▷ Iterado inicial para el barrido de n subiteraciones
    $\widetilde{r}^{(1)} \leftarrow r^{(0)}$
    para $j \leftarrow 1, 2, \cdots, n$ **hacer**
      $a \leftarrow Ae_j$ ▷ j-ésima columna de A
      $d \leftarrow \|a\|_2^2$
      $\delta_j \leftarrow \omega((a^T\widetilde{r}^{(j)})/d)$
      $z^{(j+1)} \leftarrow z^{(j)} + \delta_j e_j$
      $\widetilde{r}^{(j+1)} \leftarrow \widetilde{r}^{(j)} - \delta_j a$
    fin para
    $x^{(i)} \leftarrow z^{(n+1)}$
    $r^{(i)} \leftarrow \widetilde{r}^{(j+1)}$
    si $\|r^{(i)}\|_2 < \epsilon$ **entonces**
      $\leftarrow x^{(i)}$
    fin si
  fin para
  Error (Máximo de iteraciones alcanzado)
fin función

Se dice que M es una matriz de orden consistente *si los autovalores del funcional*

$$J(\alpha) = \alpha L + \alpha^{-1} U, \quad \alpha \in \mathbb{R}, \alpha \neq 0$$

Teorema 2.2.3. *Sea $A \in \mathbb{R}^{m \times n}$, si AA^T una matriz de orden consistente con $AA^T = D_A(I_m - L - U)$ tal como en la definicion (2.2.2), entonces si $B = L + U$ tiene todos sus autovalores reales y $\rho(B) < 1$, entonces el factor de relajación óptimo para el método SOR aplicado a la resolución de las ecuaciones normales es*

$$\omega_{opt} = \frac{2}{1 + \sqrt{1 - \rho(B)^2}}$$

Demostración. Ver [44] □

2.2.2. El método del Gradiente Conjugado para las Ecuaciones Normales

El método del *Gradiente Conjugado (GC)* fue propuesto originalmente en [20] como un método directo para la resolución de sistemas de ecuaciones lineales de la forma $Cx = d$, $C \in \mathbb{R}^{n \times n}$ y C es SPD. No obstante, GC adquirió relevancia al ser reformulado como un método iterativo, en este ámbito el método es muy eficiente sobre todo para el caso de matrices *sparse*.

Para el caso de matrices SPD, el GC proviene a partir de un problema de optimización de funcionales cuadráticas. En efecto para $M \in \mathbb{R}^{n \times n}$, una matriz SPD, el problema de minimizar el funcional cuadrático

$$q(x) = \frac{1}{2} x^T M x - x^T b + c \tag{2.30}$$

Es equivalente a resolver el sistema $Mx = b$.

El objetivo del método es generar un conjunto de direcciones de búsqueda que cumplen una propiedad conocida como *M-Conjugancia* respecto a una matriz M, la cual se define a continuación

Definición 2.2.3. *Sea $\mathcal{U} = \{u_1, u_2, \dots u_n\}$ un conjunto de vectores en $\mathbb{R}^n$. Dada una matriz $M \in \mathbb{R}^{n \times n}$ SPD, se dice que $\mathcal{U}$ posee la propiedad de* M-conjugancia *si para todo $1 \leq i, j \leq n$ se cumple*

$$u_j^T M u_i = 0 \text{ si } i \neq j$$

Así, dado un conjunto $\mathcal{U}$ de vectores que poseen la propiedad de *M-Conjugancia*, existe un método iterativo tal que, usando los vectores de dicho conjunto como direcciones de búsqueda, obtiene la solución al sistema de ecuaciones $Mx = b$ en un número finito de iteraciones, tal como se establece en el siguiente resultado

Teorema 2.2.4. *Sea $M \in \mathbb{R}^{n \times n}$ SPD, $b \in \mathbb{R}^n$ y $\mathcal{U} = \{u_1, u_2, \dots u_n\}$ un conjunto de vectores no nulos con la propiedad de* M-Conjugancia*, entonces, para todo iterado inicial $x_0 \in \mathbb{R}^n$ la sucesión*

$$x_k = x_{k-1} + \frac{r_{k-1}^T u_k}{u_k^T M u_k} u_k, \quad (r_k = b - M x_k)$$

satisface $Mx_n = b$

Demostración. Ver [27] □

La conexión entre el método iterativo definido por el teorema anterior y la optimización de funcionales cuadráticas está en el hecho de que el factor $(r_{k-1}^T u_k / u_k^T M u_k) u_k$ es el argumento que minimiza a $q(x + \alpha u)$ donde α es un parámetro real, x y u son valores fijos y q es el funcional definido en la ecuación (2.30) (Ver [27])

En el algoritmo (2.8) presenta el método GC en su forma genérica para una matriz M SPD. Finalmente el algoritmo (2.9) presenta el método GC aplicado a las ecuaciones normales para una matriz $A \in \mathbb{R}^{m \times n}$, en donde no es necesario formar explícitamente la matriz $A^T A$.

Algoritmo 2.8 Gradiente Conjugado Genérico para una matriz SPD

- **Algoritmo**
  GC_Generico
- **Descripción**
  Resuelve el sistema $Mx = b$ usando GC.
- **Entradas**
  - $x^{(0)}$ Vector columna de n filas, iterado inicial.
  - M Matriz SPD de n filas por n columnas.
  - b Vector columna de n filas. iteración.
- **Salidas**
  - x solución del sistema de ecuaciones.

1: **función GC_Generico**$(x^{(0)}, M, b)$
2: $k \leftarrow 0$
3: $r^{(0)} \leftarrow b - Mx^{(0)}$
4: **mientras** $\|r_k\|_2 \neq 0$ **hacer**
5: $k \leftarrow k + 1$
6: **si** $k = 1$ **entonces**
7: $u_1 \leftarrow r^{(0)}$
8: **si no**
9: $\beta_k \leftarrow r^{(k-1)T} r^{(k-1)} / r^{(k-2)T} r^{(k-2)}$
10: $u_k \leftarrow r^{(k-1)} - \beta_k u_{k-1}$
11: **fin si**
12: $\alpha_k \leftarrow r^{(k-1)T} r^{(k-1)} / (u_k^T M^T u_k)$
13: $x^{(k)} \leftarrow x^{(k-1)} + \alpha_k u_k$
14: $r^{(k)} \leftarrow r^{(k-1)} - \alpha_k A u_k$
15: **fin mientras**
16: $\leftarrow x^{(k)}$
17: **fin función**

Algoritmo 2.9 GC aplicado a las ecuaciones normales

- **Algoritmo**
  GC_PMCL
- **Descripción**
  Resuelve el PMCL asociado a A y b usando GC para las ecuaciones normales.
- **Entradas**
  - $x^{(0)}$ Vector columna de n filas, iterado inicial.
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - b Vector columna de n filas.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - x Solución al PMCL asociado a A y b.

1: **función GC_PMCL**$(x^{(0)}, A, b, N_{max}, \epsilon)$
2: $p^{(0)} \leftarrow A^T(b - Ax^{(0)})$
3: $s^{(0)} \leftarrow p^{(0)}$
4: $\gamma^{(0)} \leftarrow \|s^{(0)}\|_2^2$
5: **para** $k = 0, 1, \ldots N_{max}$ **hacer**
6: $q^{(k)} \leftarrow Ap^{(k)}$
7: $\alpha_k \leftarrow \gamma^{(k)}/\|q^{(k)}\|_2^2$
8: $x^{(k+1)} \leftarrow x^{(k)} + \alpha_k p^{(k)}$
9: $s^{(k+1)} \leftarrow s^{(k)} - \alpha_k(A^T q^{(k)})$
10: $\gamma^{(k+1)} \leftarrow \|s^{(k+1)}\|_2^2$
11: **si** $\gamma^{(k+1)} < \epsilon$ **entonces**
12: $\leftarrow x^{(k+1)}$
13: **fin si**
14: $\beta_k \leftarrow \gamma^{(k+1)}/\gamma^{(k)}$
15: $p^{(k+1)} \leftarrow s^{(k+1)} + \beta_k p^{(k)}$
16: **fin para**
17: **Error** (Máximo de iteraciones alcanzado)
18: **fin función**

2.3. Precondicionamiento para las ecuaciones normales

En muchos casos, la convergencia de los métodos iterativos puede resultar lenta, principalmente en los casos de matrices mal condicionadas. Es por ello, que en los últimos años, se ha puesto mayor énfasis en *precondicionamiento* de la matriz de entrada más que en los propios métodos iterativos. Dada $A \in \mathbb{R}^{n\times n}$ y $b \in \mathbb{R}^n$, el *precondicionamiento* es una técnica que tiene como objetivo transformar el sistema de ecuaciones original $Ax = b$ en otro sistema equivalente $\widetilde{A}x = \widetilde{b}$ tal que $\widetilde{A}$ sea una matriz mejor condicionada, y por consiguiente un método iterativo aplicado a este último sistema tenga una mayor velocidad de convergencia que para el primer sistema.

Existen dos maneras fundamentales para precondicionar el sistema $Ax = b$: la primera es premultiplicar el sistema por una matriz $M \in \mathbb{R}^{n\times n}$ no singular, así se obtiene el sistema equivalente

$$MAx = Mb \tag{2.31}$$

Esta forma de precondicionamiento se denomina *precondicionamiento por la izquierda*. La otra forma de precondicionamiento, el *precondicionamiento por la derecha* proviene de postmultiplicar A por M^{-1}, obteniendo el problema equivalente:

$$AM^{-1}y = b, \quad Mx = y \tag{2.32}$$

En ambos casos la matriz M se denomina *precondicionador* del sistema de ecuaciones.

Para el caso particular de las ecuaciones normales, en la literatura se encuentran técnicas de precondicionamiento por la derecha. En este caso, sea $M \in \mathbb{R}^{n\times n}$ no singular, entonces el problema de optimización de la ecuación (1.2) es equivalente a

$$\min_y \|(AS^{-1})y - b\|_2, \quad Sx = y \tag{2.33}$$

Donde se elige S tal que la matriz AS^{-1} tenga un espectro más favorable que la matriz A. En general, es deseable que la matriz S tenga las siguientes características:

- AS^{-1} debe tener un número de condición más pequeño que A y debe tener pocos valores singulares distintos.
- En el caso de matrices *sparse*, S debe tener el mismo *patrón de sparsidad* de A, esto es, las posiciones de los elementos no nulos de S debería ser igual a las de A.
- Debe ser de bajo costo computacional resolver sistemas de ecuaciones que involucren a S y a S^T.

Una vez hallado un precondicionador S, se procede a resolver el sistema de ecuaciones normales precondicionadas

$$(AS^{-1})^T(AS^{-1})x = (AS^{-1})^T b \tag{2.34}$$

2.3.1. Gradiente Conjugado Precondicionado para las ecuaciones normales

Dado un precondicionador $S \in \mathbb{R}^{n\times n}$, puede reformularse el algoritmo (2.8) para resolver el sistema de ecuaciones (2.34).

El GC precondicionado se presenta en el algoritmo (2.10)

2.3.2. Algunos precondicionadores conocidos

Uno de los problemas actuales en el cálculo numérico es conseguir buenos precondicionadores. Lamentablemente no existe una teoría general para hallar precondicionadores que sean aplicables para cualquier sistema de ecuaciones, sino que lo usual es hallar precondicionadores aplicables a algún problema o familia de problemas particulares, basándose en información adicional del contexto de dichos problemas.

No obstante, existen unos pocos precondicionadores que pueden aplicarse a una amplia gama de sistemas de ecuaciones. En este trabajo se presentarán los precondicionadores más populares para el PMCL.

Precondicionador basado en la Factorización de Cholesky incompleta Sea $S = R$, donde R es el factor de Cholesky asociado a AA^T, puede verificarse que (Ver [5]) $\kappa(AS^{-1}) = 1$, lo cual implica que, en teoría, el sistema de ecuaciones normales precondicionado con S necesitaría de sólo una iteración para converger a la solución, para cualquier método iterativo.

Este hecho teórico hace intuir que una aproximación al factor de Cholesky de AA^T puede ser un buen precondicionador para el PMCL. Una forma de aproximación al factor de Cholesky, es la que se conoce como *factorización de Cholesky incompleta*, la cual consiste en una matriz $\widetilde{R} \equiv R$ tal que

$$C = A^T A = R^T R - E$$

Donde E es una *matriz de error*. Se desea que $\|E\|$ sea pequeña y $\widetilde{R}$ sea *sparse* con un patrón de *sparsidad* similar a A. Para calcular dicha $\widetilde{R}$ puede usarse el mismo algoritmo (2.1) de la factorización de Cholesky, pero calculando sólo aquellos elementos $\widetilde{r}_{ij}$ tales que $A_{ij} \neq 0$.

El algoritmo (2.11) presenta la factorización de Cholesky incompleta. Para el algoritmo $C = AA^T$ y se define el patrón de *sparsidad* de A como el conjunto $P = \{(i,j), A_{ij} \neq 0\}$ y es una de las entradas del algoritmo. Obsérvese que no hace falta calcular de manera explícita la matriz AA^T, sino que sólo hace falta acceder a las filas i y j cada vez que se desee calcular c_{ij}.

Precondicionador basado en la factorización LU Una opción para el precondicionamiento del PMCL en sistemas sobredeterminados, es usar la factorización de un subbloque cuadrado de A. Para una matriz de rango completo $A \in \mathbb{R}^{m\times n}$, si luego de permutaciones de filas sobre A se puede llegar a

$$A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix}, \quad A_1 \in \mathbb{R}^{n\times n} \tag{2.35}$$

Tal que A_1 sea no singular, entonces, A_1^{-1} se puede usar como precondicionador derecho, obteniendo el siguiente problema de optimización, equivalente al PMCL.

Algoritmo 2.10 GC Precondicionado

- **Algoritmo**
  GCPrecondicionadoPMCL
- **Descripción**
  Resuelve el PMCL asociado a A y b usando GC precondicionado para las ecuaciones normales.
- **Entradas**
  - $x^{(0)}$ Vector columna de n filas, iterado inicial.
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - b Vector columna de n filas.
  - S Matriz precondicionadora de n filas por n columnas.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - x Solución al PMCL asociado a A y b.

función **GCPrecondicionadoPMCL**$(x^{(0)}, A, b, S, N_{max}, \epsilon)$
  $r^{(0)} \leftarrow (b - Ax^{(0)})$
  Resolver el sistema $S^T y = A^T r^{(0)}$
  $p^{(0)} \leftarrow y$
  $s^{(0)} \leftarrow p^{(0)}$
  $\gamma^{(0)} \leftarrow \|s^{(0)}\|_2^2$
  para $k \leftarrow 0, 1, \ldots N_{max}$ **hacer**
    Resolver el sistema $Sw = p^{(k)}$
    $t^{(k)} \leftarrow w$
    $q^{(k)} \leftarrow At^{(k)}$
    $\alpha_k \leftarrow \gamma^{(k)} / \|q^{(k)}\|_2^2$
    $x^{(k+1)} \leftarrow x^{(k)} + \alpha_k t^{(k)}$
    $r^{(k+1)} \leftarrow r^{(k)} + \alpha_k q^{(k)}$
    $u \leftarrow A^{(t)} r^{(k+1)}$
    Resolver el sistema $S^T z = u$
    $s^{(k+1)} \leftarrow z$
    $\gamma^{(k+1)} \leftarrow \|s^{(k+1)}\|_2^2$
    si $\gamma^{(k+1)} < \epsilon$ **entonces**
      $\leftarrow x^{(k+1)}$
    fin si
    $\beta_k \leftarrow \gamma^{(k+1)} / \gamma^{(k)}$
    $p^{(k+1)} \leftarrow s^{(k+1)} + \beta_k p^{(k)}$
  fin para
  Error (Máximo de iteraciones alcanzado)
fin función

Algoritmo 2.11 Factorización de Cholesky incompleta

- **Algoritmo**
  CholeskyIncompleto
- **Descripción**
  Calcula la factorización de Cholesky incompleta de una matriz C SPD con un patrón de *sparsidad* P.
- **Entradas**
  - C Matriz *sparse* n filas por n columnas SPD.
  - P Patrón de *sparsidad* para la factorización incompleta.
- **Salidas**
  - R Aproximación al factor de Cholesky de C con patrón de *sparsidad* P.

1: **función** **CholeskyIncompleto**(C, P)
2: **para** $i \leftarrow 1 \dots n$ **hacer**
3: $r_{ii} \leftarrow \left(c_{ii} - \sum_{k=1}^{i-1} r_{ki}^2\right)^{1/2}$
4: **para** $j \leftarrow 1 \dots n$ **hacer**
5: **si** $(i, j) \in P$ **entonces**
6: $r_{ij} \leftarrow c_{ij} - \sum_{k=1}^{i-1} r_{ki} r_{kj}$
7: $r_{ij} \leftarrow r_{ij}/rii$
8: **fin si**
9: **fin para**
10: **fin para**
11: $\leftarrow C$
12: **fin función**

$$\min_y \|AA_1^{-1}y - b\|_2, \quad y = A_1 x \tag{2.36}$$

En donde el PMCL asociado a AA_1^{-1} y b está mejor condicionado que el problema original y se puede resolver con cualquiera de los métodos iterativos cubiertos en las secciones anteriores.

Una vez resuelto el PMCL precondicionado, la solución al PMCL original es $x = A_1^{-1}y$. En el caso general para evitar el cálculo explícito de la inversa, se prefiere trabajar con la factorización LU de A_1.

Precondicionador para GC basado en la factorización LU Un esquema de precondicionamiento basado en la factorización LU combinado con GC, y que puede usarse tanto para matrices *sparse*, como para matrices densas, se presenta en [5]. Si la ecuación (2.35) se sustituye en el problema de optimización (1.2), se obtiene el problema de optimización equivalente

$$\min_x \left\| \begin{pmatrix} I_n \\ C \end{pmatrix} y - \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \right\|_2, \quad b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \tag{2.37}$$

donde

$$b_1 \in \mathbb{R}^n, \quad b_2 \in \mathbb{R}^{m-n}, \quad C = A_2 A_1^{-1}$$

En este caso, con $r = b - Ax$, las ecuaciones normales pueden ser reescritas como

$$\begin{pmatrix} I_n & 0 & I_n \\ 0 & I_{m-n} & 0 \\ I_n & C^T & 0 \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ y \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ 0 \end{pmatrix}, \quad r = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix} \tag{2.38}$$

donde

$$r_1 \in \mathbb{R}^n, \quad r_2 \in \mathbb{R}^{m-n}$$

Haciendo $r_2 = 0$ se obtiene el siguiente sistema SPD

$$\left(I_n + C^T C\right) r_1 = C^T \left(b_2 - C b_1\right) \tag{2.39}$$

Con la solución del sistema de ecuaciones (2.39), se obtiene, finalmente, la solución al PMCL resolviendo el sistema

$$A_1 x = b_1 - r_1 \tag{2.40}$$

El algoritmo (2.12) muestra cómo se adapta el GC al precondicionamiento LU explicado anteriormente. Sin embargo, se requiere del cálculo explícito de la matriz $C = A_2 A_1^{-1}$. Una mejora de dicho algoritmo para evitar el cálculo de C, se logra observando que, para cada iteración

$$s = r + C^T r = r + (L^T)^{-1}((U^T)^{-1}(A_2^T r))$$

y

$$q = Cp = A_2(U^{-1}(L^{-1}p))$$

Por consiguiente, sólo basta conocer la factorización LU de A_1, y en cada paso del algoritmo (2.12) en donde esté involucrada la matriz C, será preciso resolver varios sistemas de ecuaciones

lineales donde existan productos matriz-vector de las formas $L^{-1}v$, $U^{-1}v$, $(L^T)^{-1}v$ y $(U^T)^{-1}v$. Estos sistemas de ecuaciones son de bajo costo computacional, debido a que las matrices U y L son triangulares y sparse.

Algoritmo 2.12 GC Precondicionado con Factorización LU

- **Algoritmo**
  GC_LU_PMCL
- **Descripción**
  Resuelve el PMCL asociado a A y b usando la GC precondicionado por factorización LU incompleta de un subbloque cuadrado de A.
- **Entradas**
  - $x^{(0)}$ Vector columna de n filas, iterado inicial.
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - C Matriz precondicionadora de la forma $C = A_2 A_1^{-1}$ donde A_1 y A_2 son subbloques de A luego de permutaciones de filas.
  - b Vector columna de n filas.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - x Solución al PMCL asociado a A y b.

función GC_LU_PMCL$(x^{(0)}, A, C, b, N_{max}, \epsilon)$
  $r^{(0)} \leftarrow b - Ax^{(0)}$
  $\rho_1^{(0)} \leftarrow r^{(0)}(1:n)$
  $\rho_2^{(0)} \leftarrow r^{(0)}((n+1):m)$
  $p^{(0)} \leftarrow \rho_1^{(0)} + C^T \rho_2^{(0)}$
  $s^{(0)} \leftarrow p^{(0)}$
  $\gamma^{(0)} \leftarrow \|s^{(0)}\|_2^2$
  para $k = 0, 1, \ldots N_{max}$ **hacer**
    $q^{(k)} \leftarrow Cp^{(k)}$
    $\alpha_k \leftarrow \gamma^{(k)} / \|q^{(k)}\|_2^2$
    $\rho_1^{(k+1)} \leftarrow \rho_1^{(k)} - \alpha_k p^{(k)}$
    $\rho_2^{(k+1)} \leftarrow \rho_2^{(k)} - \alpha_k q^{(k)}$
    $s^{(k+1)} \leftarrow \rho_1^{(k+1)} + C^T \rho_2^{(k+1)}$
    $\gamma^{(k+1)} \leftarrow \|s^{(k+1)}\|_2^2$
    si $\gamma_{k+1} < \epsilon$ **entonces**
      Resolver el sistema $A_1 x = b_1 - \rho_1$
      $\leftarrow x$
    fin si
    $\beta_k \leftarrow \gamma^{(k+1)} / \gamma^{(k)}$
    $p^{(k+1)} \leftarrow s^{(k+1)} + \beta_k p^{(k)}$
  fin para
  Error (Máximo de iteraciones alcanzado)
fin función

Capítulo 3

El método de Schulz

El centro no es un punto
Si lo fuera, resultaría fácil acertarlo
Roberto Juarroz - Segunda poesía vertical

Como se vio en el capítulo 1, $x = A^{\dagger}b$, es la solución al PMCL para el caso de una matriz A de rango completo. Esto permite pensar que una matriz $B \approx A^{\dagger}$ podría ser un buen precondicionador para la ecuación (1). Por esto, es de interés hallar algún método iterativo que permita obtener una aproximación a $A^{\dagger}$ con la precisión que se desee, y además, sería deseable que dicho método posea una alta velocidad de convergencia.

Un método que puede utilizarse para calcular una aproximación a la matriz pseudoinversa de A es el *Método iterativo de Schulz* [37], un método que se deriva a partir de la *iteración de Newton* en espacios de matrices. Originalmente, este método fue ideado para el cálculo de la inversa de una matriz cuadrada; no obstante, en este trabajo se reutilizará para calcular una aproximación a la pseudoinversa de una matriz rectangular de rango completo.

El resto del capítulo se organiza así: en la primera sección se dará una breve introducción al método de Newton en espacios de matrices. Posteriormente se presentará el método iterativo de Schultz y se estudiarán sus propiedades en cuanto a la convergencia y otros aspectos relevantes. Finalmente, el capítulo concluirá con el estudio de algunas propiedades interesantes de la iteración de Schultz para el caso de matrices rectangulares, las cuales dan una pista teórica de que la aproximación de Schultz es un buen precondicionador.

3.1. El método de Newton

Uno de los problemas clásicos del cálculo numérico es hallar el cero de una función; esto es, dada $F : \mathbb{R}^n \to \mathbb{R}^n$ hallar ξ tal que $F(\xi) = 0$. Para el caso general de funciones no lineales diferenciables, el *Método iterativo de Newton* representa una de las alternativas más porderosas para el cálculo de los ceros de una función no lineal.

Sea $F : \mathbb{R}^n \to \mathbb{R}^n$, si para $h \in \mathbb{R}^n$ F es diferenciable en algún entorno de h, entonces, el teorema de Taylor [26], implica que para toda matriz X en un subconjunto convexo de dicho entorno de h se tiene la siguiente aproximación a $F(X)$

$$F(X) \approx F(h) + (X - h)D_F(h) \tag{3.1}$$

donde $D_F(h)$ es la derivada de F en h. En particular, si ξ es un cero de F, para alguna matriz X_0 en un entorno de ξ, usando la ecuación (3.1) se tiene

$$0 = F(\xi) \approx F(X_0) + (\xi - X_0)D_F(X_0) \tag{3.2}$$

Con la información anterior, puede deducirse la iteración de Newton en espacios de matrices a través de un modelo general para el estudio de los métodos iterativos para funciones no lineales: la *iteración funcional de punto fijo.*

En el esquema de *iteración funcional de punto fijo*, un método iterativo para la función $F : \mathbb{R}^n \to \mathbb{R}^n$ se define a partir de una función Φ asociada a F de la siguiente manera

$$x_{k+1} = \Phi(x_k) \tag{3.3}$$

esto es, que cualquier método iterativo para F puede redefinirse como una iteración que converja al *punto fijo* de una función Φ. Para un estudio general de las propiedades y condiciones de convergencia de los métodos de punto fijo, ver [22] y [39].

Así, a partir de la ecuación (3.2), si D_F es no singular en un entorno de ξ puede obtenerse una iteración de punto fijo definiendo Φ de la siguiente manera

$$\Phi(x) \equiv x - (D_F(x))^{-1} F(x) \tag{3.4}$$

de donde se deduce la iteración de Newton en espacios de matrices para un iterado inicial X_0 *cercano* al cero de F

$$x_{k+1} = x_k - (D_F(x_k))^{-1} F(x_k) \tag{3.5}$$

El algoritmo (3.1) muestra una implementación del esquema iterativo (3.5). En este caso, para evitar el cálculo explícito de la matriz inversa a D_F, es necesario resolver un sistema de ecuaciones lineales en cada iteración.

La convergencia del método de Newton en espacios de matrices queda establecida a partir del siguiente resultado.

Teorema 3.1.1. *Sea $C \subseteq \mathbb{R}^n$ un conjunto abierto y tal que $F : C \to \mathbb{R}^n$ y sea $C_0 \subseteq C$ un conjunto convexo tal que F es diferenciable en C_0 y continua en C. Dado $x_0 \in C_0$, sea $r > 0$ tal que $\{x,\ \|x - x_0\|_2 < r\} \subseteq C_0$ y sean constantes h, α, β y γ tales que $r = \alpha/(1-h)$ y $h = \alpha\beta\gamma/2 < 1$. Si F satisface las siguientes condiciones:*

1. $\|D_F(x) - D_F(y)\|_2 \leq \gamma \|x - y\|_2$ *para todo* $x, y \in C_0$.
2. $(D_F(x))^{-1}$ *existe para todo* $x \in C_0$ *y* $\left\|(D_F(x))^{-1}\right\|_2 \leq \beta$.
3. $\left\|(D_F(x_0))^{-1}F(x_0)\right\|_2 \leq \alpha$.

Entonces:

1. *La sucesión $x_{k+1} = x_k - (D_F(x_k))^{-1}$, $k = 0, 1, \ldots$ está bien definida, y además satisface $x_k \in \{x, \|x - x_0\|_2 < r\}$ para $k = 1, 2, \ldots$.*
2. $\lim_{x\to\infty} x_k = \xi < \infty$ *y* $F(\xi) = 0$.

Demostración. Ver [31] □

Algoritmo 3.1 Método de Newton en varias variables

- **Algoritmo**
  NEWTON_VARIAS_VARIABLES
- **Descripción**
  Calcula el cero de la función F usando el método de Newton en espacios de matrices.
- **Entradas**
  - X_0 Vector columna de n filas, iterado inicial.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - X cero de la función F.

1: **función** NEWTON_VARIAS_VARIABLES(X_0, N_{max}, ϵ)
2: **para** $i \leftarrow 1, \ldots, N_{max}$ **hacer**
3: $v \leftarrow F(X_0)$
4: $M \leftarrow D_F(X_0)$
5: Resolver el sistema $My = v$
6: $X_1 \leftarrow X_0 - y$
7: **si** $\|X_0 - X_1\|_2 / \|X_1\|_2 < \epsilon$ **entonces**
8: $\leftarrow X_1$
9: **fin si**
10: $X_0 \leftarrow X_1$
11: **fin para**
12: **ERROR** (Máximo de iteraciones alcanzado)
13: **fin función**

3.2. El método de Schulz

Sea $A \in \mathbb{R}^{n\times n}$ no singular. En esta sección se construirá la iteración de Schulz como un caso particular del del método de Newton en espacios de matrices.

Lo primero que se necesita es la definir $F : \mathbb{R}^n \to \mathbb{R}^n$ tal que $F(A) = 0$. Una función que satistace esta condición es $F(X) = X^{-1} - A$.

El próximo paso será calcular la derivada de la función F definida anteriormente. Para esto se usará la expansión en *series de Newman* para una matriz, la cual queda establecida en el siguiente teorema

Teorema 3.2.1. *Sea $A \in \mathbb{R}^{n\times n}$ tal que $\|A\|_2 < 1$, entonces, la matriz $I_n - A$ es no singular y además:*

$$(I_n - A)^{-1} = \sum_{k=0}^{\infty} A^k$$

Demostración. Ver [22] □

Con $F = X^{-1} - A$, escójase alguna matriz $B \in \mathbb{R}^{n\times n}$ tal que $\left\|A^{-1}B\right\|_2 \leq \left\|A^{-1}\right\|_2 \|B\|_2 < 1$, esto es, B es una matriz *cercana* a 0. Se desea acotar el diferencial $F(A+B) - F(B) - DB$ para alguna matriz D. En primera instancia, la expresión para $F(A+B) - F(B)$ es

$$\begin{aligned} F(A+B) - F(B) &= \left((A+B)^{-1} - A\right) - \left(A^{-1} - A\right) \\ &= (A+B)^{-1} - A^{-1} \\ &= \left(A\left(I_n + A^{-1}B\right)\right)^{-1} - A^{-1} \\ &= A^{-1}\left(I_n + A^{-1}B\right)^{-1} - A^{-1} \\ &= \left(\left(I_n + A^{-1}B\right)^{-1} - I_n\right) A^{-1} \end{aligned}$$

Sustituyendo la expansión del teorema (3.2.1) en la expresión anterior

$$\begin{aligned} F(B+A) - F(A) &= \left(\sum_{k=0}^{\infty} (-1)^k (A^{-1}B)^k A^{-1} - I_n\right) A^{-1} \\ &= -A^{-1}BA^{-1} + \sum_{k=2}^{\infty} (-1)^k (A^{-1}B)^k A^{-1} \end{aligned}$$

Finalmente, la expresión para el diferencial viene dada por

$$F(A+B) - F(B) - (-A^{-1}BA^{-1}) = \sum_{k=2}^{\infty} (-1)^k (A^{-1}B)^k A^{-1} \tag{3.6}$$

Por otro lado, se obtendrá una cota para la norma del miembro derecho de la ecuación (3.6)

$$\begin{aligned}
\left\|\sum_{k=2}^{\infty}(-1)^k(A^{-1}B)^kA^{-1}\right\|_2 &\leq \left\|\sum_{k=2}^{\infty}(-1)^k(A^{-1}B)^k\right\|_2 \left\|A^{-1}\right\|_2 \\
&\leq \left\|A^{-1}B\right\|_2^2 \left\|A^{-1}\right\|_2 \left\|\sum_{k=0}^{\infty}(-1)^k(A^{-1}B)^k\right\|_2 \\
&\leq \left\|A^{-1}\right\|_2^3 \|B\|_2^2 \underbrace{\sum_{k=0}^{\infty}\|(A^{-1}B)^k\|_2}_{\text{Serie Geométrica}} \qquad (3.7) \\
&= \frac{\left\|A^{-1}\right\|_2^3 \|B\|_2^2}{1-\|A^{-1}B\|_2} \qquad (3.8)
\end{aligned}$$

Así, con la cota anterior

$$\frac{\left\|F(A+B)-F(B)-(-A^{-1}BA^{-1})\right\|_2}{\|B\|_2} \leq \frac{\left\|A^{-1}\right\|_2^3}{1-\|A^{-1}B\|_2}\|B\|_2 \qquad (3.9)$$

Luego, el límite cuando la matriz B tiende a 0 es

$$\lim_{\|B\|_2\to 0}\frac{\left\|F(A+B)-F(B)-(-A^{-1}BA^{-1})\right\|_2}{\|B\|_2} = 0 \qquad (3.10)$$

Lo cual implica que F es diferenciable en $X = A$ y además, por la definición de derivada, $D_F(A)B = -A^{-1}BA^{-1}$. Esta expresión se puede sustituir en la función de iteración del método de Newton, obteniendo:

$$\begin{aligned}
&M_{k+1} = \Phi(M_k) \\
\Rightarrow &M_{k+1} = M_k - (D_F(M_k))^{-1}F(M_k) \\
\Rightarrow &D_F(M_k)(M_{k+1}-M_k)M_k^{-1} = F(M_k) = M_k^{-1} - A \\
\Rightarrow &M_k^{-1}(M_{k+1}-M_k)M_k^{-1} = M_k^{-1} - A \\
\Rightarrow &M_{k+1} - M_k = M_k - M_kAM_k \qquad (3.11)
\end{aligned}$$

De donde se puede obtener la iteración del método de Schulz

$$M_{k+1} = 2M_k - M_kAM_k \qquad (3.12)$$

La condición de convergencia para el método de Schulz se presenta en el siguiente teorema

Teorema 3.2.2. *La sucesión $\{M_k\}$ definida por la ecuación (3.12) converge a A^{-1} si y sólo si $\rho(I - AM_0) < 1$.*

Demostración. Para k = 1,2,... se tiene que

$$\begin{aligned}
I_n - AM_{k+1} =& I_n - A(2M_k - M_kAM_k) \\
=& (AM_k)_2 - 2AM_k + I_n \\
=& (I_n - AM_k)^2
\end{aligned}$$

Con esta relación, puede deducirse indutivamente que

$$\begin{aligned} I_n - AM_k =& (I_n - AM_{k-1})^2 \\ =& ((I_n - AM_{k-2})^2)^2 \\ =& (((I_n - AM_{k-3})^2)^2)^2 \\ =& \cdots \\ =& (I_n - AM_0)^{2^k} \end{aligned}$$

Sea la sucesión $B_k = AM_k$, combinándola con el resultado anterior, se puede reescribir como

$$B_k = I_n - (I_n - B_0)^{2^k} \tag{3.13}$$

Nótese que $lim_{k\to\infty}(I - B_0)^{2^k} = 0$ o equivalentemente, $lim_{k\to\infty}B_k = I_n$ si y solo si $\rho(I - B_0) = \rho(I - AM_0) < 1$.

Por otro lado, $lim_{k\to\infty}B_k = lim_{k\to\infty}AM_k = I_n$ de donde se puede deducir que $lim_{k\to\infty}M_k = A^{-1}$ lo cual concluye la prueba. □

Para cerrar esta sección, se presentará un resultado que da una elección de iterado inicial del método de Schulz que garantiza la convergencia para cualquier matriz A no singular.

Teorema 3.2.3. *Sea $A \in \mathbb{R}^{n\times n}$ y sea $p(x) = a_0 + a_1x + \cdots + a_nx^n$ un polinomio. Si λ es un autovalor de A, entonces $p(\lambda)$ es un autovalor de $P(A)$.*

Demostración. Ver [22] □

Teorema 3.2.4. *La elección del iterado inicial*

$$M_0 = \frac{A^T}{\|A\|_2 \|A^T\|_2}$$

garantiza la convergencia del método de Schulz.

Demostración. Con tal elección de M_0, se tiene que $I_n - AM_0 = I - (1/\|A\|_2\|A^T\|_2)AA^T$. Considerando el polinomio $p(x) = (1 - x)(\|A\|_2\|A^T\|_2)$, puede verificarse que $AA^T = p(I_n - AM_0)$. Así, por el teorema (3.2.3) se cumple que para todo λ, autovalor de $I_n - AM_0$, $p(\lambda) = (1 - \lambda)(\|A\|_2\|A^T\|_2)$ es autovalor de AA^T.

Como A es no singular, AA^T es SPD, por lo cual todos sus autovalores son positivos. En particular, para todo λ, autovalor de $I_n - AM_0$ se cumple

$$(1 - \lambda)(\|A\|_2\|A^T\|_2) > 0 \Rightarrow 1 - \lambda > 0 \Rightarrow \lambda < 1$$

Luego $\rho(I_n - AM_0) < 1$ y se asegura la convergencia del método. □

Usando la el iterado inicial del teorema anterior el algoritmo (3.2) presenta el método de Schulz para una matriz $A \in \mathbb{R}^{n\times n}$

Algoritmo 3.2 Iteración de Schulz

- **Algoritmo**
  IterSchulz
- **Descripción**
  Dada una matriz A, calcula una aproximación a $A^\dagger$ a través del método de Schulz.
- **Entradas**
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - M aproximación a $A^\dagger$.

función IterSchulz(A, ϵ, N_{max})
  $M_0 \leftarrow A^T / \|A\|_2^2$
  para $i \leftarrow 1, \ldots, N_{max}$ **hacer**
    $U \leftarrow M_0 A$
    $V \leftarrow 2I_n - U$
    $M_1 \leftarrow V M_0$
    si $\|M_0 - M_1\|_2 / \|M_1\|_2 < \epsilon$ **entonces**
      $\leftarrow M_1$
    fin si
    $M_0 \leftarrow M_1$
  fin para
  Error (Máximo de iteraciones alcanzado)
fin función

3.3. Propiedades del Método de Schulz para matrices rectangulares

En la sección anterior, se presentó el método de Schulz para matrices cuadradas $A \in \mathbb{R}^{n\times n}$. Este método puede ser aplicado también para matrices rectangulares $A \in \mathbb{R}^{m\times n}$ de rango completo, en cuyo caso, converge a $A^\dagger$, tal como lo establece el siguiente resultado.

Teorema 3.3.1. *Sea $A \in \mathbb{R}^{m\times n}$ de rango completo. La iteración de Schulz (3.12) converge a $A^\dagger$ si y sólo si $\rho(I - AM_0) < 1$.*

Demostración. En primer lugar, es necesario verificar que la expresión de la iteración de Schulz esté bien definida respecto a las dimensiones de la matriz. Para ello es necesario que $M_0 \in \mathbb{R}^{n\times m}$. Se sabe que $A \in \mathbb{R}^{m\times n}$ y $M_0 = A^T/\|A\|_2^2 \in \mathbb{R}^{n\times m}$. Así, para todo $k > 0$ se cumple que $M_k \in \mathbb{R}^{n\times m}$ y por lo tanto $M_kA \in \mathbb{R}^{n\times n}$. Puede observarse que de esta forma también se cumple la relación de recurrencia para $(I_n - AM_k)$ establecida en la prueba del teorema (3.2.2), luego, sustituyendo A^{-1} por $A^\dagger$ en la prueba de dicho teorema se verifica este resultado. □

Para finalizar el capítulo, se probarán dos propiedades relevantes de la iteración de Schulz para el caso de matrices rectangulares: en primer lugar, para cualquier iterado M_k si x es la solución al PMCL asociado a A, entonces se cumple una condición de ortogonalidad sobre $M_k(b-Ax)$ análoga a la condición de ortogonalidad para el PMCL. Este es uno de los resultados teóricos centrales de este trabajo, pues da una indicación, en la teoría, de que la elección de M_k para precondicionar el PMCL es correcta en el sentido que el sistema precondicionado (que será de n ecuaciones con n incógnitas, a diferencia del sistema original que es de m ecuaciones con n incógnitas) tendrá la misma solución del sistema original, esto es la solución al PMCL.

En segundo lugar, se probará que si λ_{k_i} es el i-ésimo autovalor de M_kA, entonces λ_{k_i} es la k-ésima iteración de Newton para hallar el cero de la función $f(x) = 1 - 1/x$ (evidentemente el cero de f es $x = 1$) con iterado inicial λ_{0_i}, el i-ésimo autovalor de la matriz M_0A. Este es el segundo resultado central de este trabajo, pues muestra en teoría, que los iterados M_k tienden a hacer un agrupamiento (*clustering*) de los autovalores del sistema inicial, en torno a $\lambda = 1$, lo cual es una de las propiedades ideales de un buen precondicionador: agrupar el espectro de autovalores de la matriz original en torno a un único valor.

Durante la realización de este trabajo, no se encontró ninguna referencia en la literatura de la demostración de estas propiedades.

Antes de demostrar estos teoremas, se probará un resultado preliminar: que los autovalores de la matriz M_kA para $k = 0, 1, \ldots$ están acotados, tal como se establece a continuación.

Teorema 3.3.2. *Sea $A \in \mathbb{R}^{m\times n}$ de rango completo con $m \geq n$ y sea M_k la k-ésima iteración de Schulz para $k = 0, 1, \ldots$, entonces para todo autovalor λ de M_kA, $0 < \lambda \leq 1$.*

Demostración. Para $k = 0$ se tiene que $M_0 = A^T/\|A\|_2^2$, así $M_0A = A^TA/\|A\|_2^2$. Como A es de rango completo, entonces M_0 es SPD y por consiguiente todos sus autovalores son reales y positivos. Por otro lado, se tiene que para cualquier matriz M, el radio espectral $\rho(M)$ es el ínfimo de todas las normas matriciales (Ver [22]), así

$$\rho(M_0) = \rho\left(\frac{A^TA}{\|A\|_2^2}\right) \leq \left\|\frac{A^TA}{\|A\|_2^2}\right\|_2 < 1$$

Lo cual concluye la prueba para $k = 0$. Ahora, sea $k \geq 1$, con la recurrencia obtenida en la prueba del teorema (3.2.2), se tiene que

$$M_k A = I_n - (I_n - M_0 A)^{2^k}$$

Si se considera el polinomio $p(x) = 1 - (1 - x)^{2^k}$, entonces $M_k A$ se puede reescribir como $M_k A = p(M_0 A)$ y por el teorema (3.2.3) los autovalores de $M_k A$ son de la forma $\Lambda(\lambda) = 1 - (1 - \lambda)^{2^k}$ para λ autovalor de $M_0 A$. En vista de que $0 < \lambda < 1$, se sigue que $0 < \Lambda(\lambda) < 1$, lo cual prueba el teorema para $k \geq 1$. □

Con el teorema anterior, también se puede establecer una cota para los autovalores de la matriz $2I_n - M_k A$, a través del siguiente cololario

Corolario 3.3.1. *Para M_k, $k = 0, 1, \ldots$ se verifica que para todo autovalor λ de $2I_n - M_k A$, $1 < \lambda < 2$.*

Demostración. Basta considerar el polinomio $p(x) = 1 + x$, así $2I_n - M_k A = p(I_n - M_k A)$, como los autovalores de $I_n - M_k A$ están en el intervalo $(0, 1)$ se verifica el resultado. □

Teorema 3.3.3. *Sea $A \in \mathbb{R}^{m \times n}$ de rango completo con $m \geq n$, sea $b \in \mathbb{R}^n$ y sea M_k la k-ésima iteración de Schulz para $k = 0, 1, \ldots$, entonces, x_M es la solución al PMCL asociado a A y b, si y solo si cumple la siguiente condición de ortogonalidad:*

$$M_k(b - Ax_M) = 0, \quad k = 0, 1, \ldots$$

Demostración. La demostración de hará por inducción sobre k. Para $k = 0$ se tiene que $M_0 = A^T/\|A\|_2^2$, por la condición de ortogonalidad del teorema (1.2.1) para el PMCL se satisface

$$M_0(b - Ax_M) = \frac{1}{\|A\|_2^2}\left(A^T(b - Ax)\right) = 0$$

si y solo si x_M es la solución al PMCL, lo cual prueba el resultado en este caso.

Para $k = i$, por hipótesis inductiva se satisface la condición de ortogonalidad $M_i(b - Ax_M) = 0$.

Falta verificar que el resultado se cumple para $k = i + 1$. En este caso, la $(i + 1)$-ésima iteración de Schulz es de la forma

$$M_{i+1} = 2M_i - M_i A M_i = (2I_n - M_i A)\, M_i$$

así

$$M_{i+1}(b - Ax_M) = (2I_n - M_i A)\, M_i(b - Ax_M)$$

Obsérvese que usando la hipótesis inductiva $M_i(b - Ax_M) = 0$ si y solo si x_M es la solución al PMCL, y por consiguiente $(2I_n - M_i A)\, M_i(b - Ax_M) = 0$ si y solo si x_M es la solución al PMCL.

No obstante, para garantizar que no puede haber alguna cancelación trivial del producto $(2I_n - M_i A)$ por algún vector, es necesario verificar que si $b' \in \mathbb{R}^n$ entonces $(2I_n - M_i A)\, b' = 0$ si y solo si $b' = 0$. Esto se cumple cuando la matriz $(2I_n - M_i A)$ es no singular, lo cual

está garantizado pues todos los valores de esta matriz son distintos a 0. En efecto, por el corolario (3.3.1) todos los autovalores de $(2I_n - M_iA)$ se encuentran en el intervalo $(1, 2)$. Luego, haciendo $b' = M_i(b - Ax_M)$ se verifica el teorema para $k = i + 1$, lo cual finaliza la prueba. □

Teorema 3.3.4. *Sea $A \in \mathbb{R}^{m\times n}$ de rango completo con $m \geq n$, sea M_k la k-ésima iteración de Schulz y sea λ_{k_i} el i-ésimo autovalor de la matriz M_kA, entonces λ_{k_i} es igual a la k-ésima iteración del método de Newton en una variable para $f(x) = 1 - 1/x$ con iterado inicial λ_{0_i}, el i-ésimo autovalor de la matriz M_0A, con $M_0 = A^T/\|A\|_2^2$.*

Demostración. Para $f(x) = 1 - 1/x$ se tiene que $f'(x) = 1/x^2$, así, la iteración de Newton para $f(x)$ es de la forma.

$$\begin{aligned} x_{k+1} =& x_k - \frac{f(x_k)}{f'(x_k)} \\ =& x_k - x_k\,(1 - x_k) \\ =& 2x_k - x_k^2 \\ =& 1 - (1 - x_k)^2 \end{aligned}$$

□

Esta iteración puede reescribirse como

$$1 - x_{k+1} = (1 - x_k)^2$$

Y así, puede deducirse inductivamente la siguiente relación de recurrencia

$$\begin{aligned} 1 - x_k =& (1 - x_{k-1})^2 \\ =& \left((1 - x_{k-2})^2\right)^2 \\ =& \left(\left((1 - x_{k-3})^2\right)^2\right)^2 \\ =& \cdots \\ =& 1 - (1 - x_0)^{2^k} \end{aligned}$$

por lo tanto, la iteración de Newton de $f(x)$ es igual a

$$x_k = 1 - (1 - x_0)^{2^k}$$

Por otro lado, para $k \geq 1$, se sabe que $M_kA = I_n - (I_n - M_0A)^{2^k}$. Si se considera el polinomio $p(x) = 1-(1 - x)^{2^k}$, por el teorema (3.2.3), como $p(M_0A) = M_kA$ entonces $p(\lambda_{0_i}) = 1 - (1 - \lambda_{0_i}) = \lambda_{k_i}$, lo cual completa la prueba.

Capítulo 4

El Método de Aceleración de Convergencia Richardson-PR2

Lo supieron los arduos alumnos de Pitágoras
los astros y los hombres vuelven cíclicamente
Jorge Luis Borges - El otro, el mismo

El método de Richardson de primer orden, presentado en la sección (2.2.1) para resolver las ecuaciones normales, fue propuesto originalmente en 1910 (Ver [32]) como un método iterativo para sistemas de ecuaciones lineales de la forma $Ax = b$ donde $A \in \mathbb{R}^{n\times n}$ y $b \in \mathbb{R}^n$. Este método iterativo, obtenido en principio a partir de un *splitting* regular de A, puede generalizarse de la siguiente manera

$$\begin{aligned} x^{(k+1)} &= x^{(k)} + \lambda_k r^{(k)} \\ r^{(k+1)} &= b - Ax^{(k)} \end{aligned} \tag{4.1}$$

Así, el método de Richardson puede generalizarse como un esquema iterativo donde la dirección de búsqueda viene dada por el residual $r^{(k)} = b - Ax^{(k)}$, mientras que el tamaño de paso λ_k, que ahora puede variar en cada iteración, se calcula de distintas formas. Algunas estrategias para la construcción de la sucesión $\{\lambda_k\}$ pueden encontrarse en [44], [43] y [29], entre otros. Hay que hacer notar que la investigación en torno a métodos iterativos basados en la iteración general de Richardson no fue muy prolija a lo largo del siglo XX, debido a que estos métodos se consideraban poco competitivos frente a otras familias de métodos iterativos por algunas características poco favorables, tales como su lenta velocidad de convergencia e inestabilidada numérica (Ver [35])

Sin embargo, en los últimos años, ha resurgido algún interés académico en métodos tipo Richardson, motivado principalmente al hecho de que las técnicas de precondicionamiento relegan a un segundo plano la elección del método iterativo. En la práctica computacional, se ha determinado que si se logra elegir un precondicionador eficiente para un sistema de ecuaciones lineales, la rápida convergencia a su solución se logra independientemente del método iterativo utilizado. Por lo tanto, en ciertos escenarios no merece la pena invertir recursos computacionales en estrategias complejas para generar direcciones de búsqueda, cuando la simple elección del vector residual como dirección de búsqueda puede garantizar una muy buena velocidad de convergencia. En [27] se pueden encontrar varias referencias de investigaciones recientes de

métodos iterativos que usan el vector residual como dirección de búsqueda (también conocidos como *métodos iterativo-residuales*). Otra línea de pensamiento bastante novedosa busca aplicar esquemas iterativo-residuales ideados incialmente para ecuaciones no lineales en espacios de matrices, a la resolución de sistemas de ecuaciones lineales [23].

En este orden de ideas, en lo que resta del capítulo se presentará un esquema iterativo de aceleración de convergencia basado en la iteración de Richardson, el cual fue propuesto en [7]; conocido como *Método de aceleración de Richardson PR2*(Richardson-PR2).

El objetivo es utilizar este método iterativo sobre el PMCL precondicionado con la aproximación de Schultz.

4.1. El esquema de aceleración de convergencia Richardson-PR2

Supóngase que se tiene un método iterativo que sigue el esquema de la ecuación (4.1), el cual genera una sucesión de iterados $\left\{x^{(k)}\right\}$ que converge a la solución del sistema $Ax = b$. Se puede definir una nueva sucesión $\left\{y^{(k)}\right\}$ de iterados asociados al problema inicial, de la siguiente manera

$$\begin{aligned} y^{(k)} &= x^{(k)} - \lambda_k z^{(k)} \\ \rho^{(k)} &= r^{(k)} + \lambda_k A z^{(k)} \end{aligned} \tag{4.2}$$

Donde $\rho^{(k)} = b - Ay^{(k)}$ es el nuevo residual asociado a $y^{(k)}$ y $z^{(k)}$ es una nueva dirección de búsqueda, que en principio, puede elegirse de manera arbitraria. Si se logra que a partir de este esquema, la sucesión $\left\{y^{(k)}\right\}$ converja más rápidamente a la solución del sistema de ecuaciones que la sucesión $\left\{x^{(k)}\right\}$, entonces se dice que el esquema iterativo de la ecuación (4.2) es un *esquema de aceleración de convergencia* del método iterativo (4.1). Para conseguir un esquema de aceleración de convergencia, una elección usual para el tamaño de paso λ_k es el valor que minimice algún funcional asociado a $\rho^{(k)}$ para $k = 1, 2, \ldots$, mientras que $z^{(k)}$ puede escogerse, entre otras formas, como la proyección de $r^{(k)}$ sobre algún subespacio de $\mathbb{R}^n$. En las siguientes secciones se mostrarán las estrategias de elección del tamaño de paso y la dirección de búsqueda en Richardson-PR2.

4.2. Elección del tamaño de paso para el esquema Richardson-PR2

Para el esquema Richardson-PR2, el tamaño de paso λ_k se elige de tal forma que minimice el funcional $F(\lambda_k) \equiv \|\rho^{(k)}\|_2$. El argumento que minimiza al funcional anterior, se obtiene a partir del siguiente teorema

Teorema 4.2.1. *El argumento que minimiza a $F(\lambda_k) \equiv \|\rho^{(k)}\|_2$ para $k = 1, 2, \ldots$ con $\rho^{(k)}$ definido como en la ecuación (4.2) es*

$$\lambda_{k_{min}} = -\frac{\left(Az^{(k)}\right)^T r^{(k)}}{\|Az^{(k)}\|_2^2}$$

Demostración. Ver [20] □

Evidentemente, tal argumento también minimiza a $\|\rho^{(k)}\|_2^2 = \rho^{(k)T}\rho^{(k)}$, así, sustituyendo $\lambda_{k_{min}}$ en la expresión de $\rho^{(k)T}\rho^{(k)}$

$$\begin{aligned}
\min_{\lambda_k}\left(\|\rho^{(k)}\|_2^2\right) &= \rho^{(k)T}\rho^{(k)} \\
&= \left(r^{(k)T} + \lambda_k (Az^{(k)})^T\right)\left(r^{(k)} + \lambda_k (Az^{(k)})\right) \\
&= r^{(k)T}r^{(k)} + 2\lambda_k (Az^{(k)})^T r^{(k)} + \lambda_k^2 (Az^{(k)})^T (Az^{(k)}) \\
&= \|r^{(k)}\|_2^2 - \frac{\left((Az^{(k)})^T r^{(k)}\right)^2}{\left((Az^{(k)})^T (Az^{(k)})\right)^2} \\
&= \|r^{(k)}\|_2^2 - \frac{\left((Az^{(k)})^T r^{(k)}\right)^2}{\|Az^{(k)}\|_2^2 \|r^{(k)}\|_2^2}\|r^{(k)}\|_2^2 \\
&= \left(1 - \cos^2(\theta_k)\right)\|r^{(k)}\|_2^2
\end{aligned}$$

Luego

$$\min_{\lambda_k}\left(\|\rho^{(k)}\|_2^2\right) = \|r^{(k)}\|_2^2 \operatorname{sen}(\theta_k) \tag{4.3}$$

Donde θ_k es el ángulo entre los vectores $r^{(k)}$ y $Az^{(k)}$. Lo anterior implica que con dicha elección de λ_k se cumple que

$$\|\rho^{(k)}\|_2 \leq \|r^{(k)}\|_2$$

Y así, en vista de que $r^{(k)} \to 0$ cuando $k \to \infty$, entonces se deduce que

$$\lim_{k\to\infty} \frac{\|\rho^{(k)}\|_2}{\|r^{(k)}\|_2} = 0 \iff \lim_{k\to\infty} \theta_k = 0 \text{ ó } \lim_{k\to\infty} \theta_k = \pi \tag{4.4}$$

Esto es, para la elección de λ_k del teorema (4.2.1), es condición necesaria y suficiente para la convergencia de $\rho^{(k)}$ que $\theta_k \to 0$ o bien $\theta_k \to \pi$.

Por otro lado, si se define

$$P_k = \frac{Az^{(k)}z^{(k)T}}{\|Az^{(k)}\|_2^2} \tag{4.5}$$

Se tiene que

$$\rho^{(k)} = (I_n - P_k)r^{(k)} \tag{4.6}$$

Obsérvese que $P_k^2 = P_k$ y $P_k^T = P_k$, lo que implica que la matriz $I_n - P_k$ es un proyector ortogonal. Esto permite obtener una interpretación geométrica de la elección del tamaño de paso para el esquema de aceleración Richardson-PR2: en este esquema, el residual $\rho^{(k)}$ es la proyección ortogonal de $r^{(k)}$ sobre $Az^{(k)}$.

4.3. Elección de la dirección de búsqueda para Richardson-PR2

A partir de la condición (4.4) para la convergencia de $\rho^{(k)}$ se deduce que el residual del esquema de aceleración de convergencia tiende a 0 si y sólo si los vectores $r^{(k)}$ y $Az^{(k)}$ son colineales, esto es, $z^{(k)} = \pm\alpha A^{-1}r^{(k)}$ con $\alpha \in \mathbb{R}$ y $\alpha \neq 0$. En particular, se puede elegir $\alpha = 1$.

Sin embargo, tal valor para $z^{(k)}$ no es de utilidad en la práctica, pues se tendría que encarar el problema de obtener A^{-1}. Una alternativa más plausible es sustituir A^{-1} por una sucesión de matrices $\{C_k\}$ tal que $\{C_k\}$ converja a A^{-1} y así, la dirección de búsqueda utilizada, será

$$z^{(k)} = C_k r^{(k)} \tag{4.7}$$

4.4. El método iterativo de Richardson-PR2

Con la dirección de búsqueda y el tamaño de paso hallados tal como se explicó en las dos secciones precedentes, el esquema de aceleración Richardson-PR2 queda de la siguiente manera

$$\begin{aligned} y^{(k)} &= x^{(k)} - \lambda_k C_k r^{(k)} \\ \rho^{(k)} &= r^{(k)} + \lambda_k A C_k r^{(k)} \\ \lambda_k &= -\frac{\left(AC_k r^{(k)}\right)^T r^{(k)}}{\|AC_k r^{(k)}\|_2^2} \end{aligned} \tag{4.8}$$

Como puede observarse, este esquema de aceleración de convergencia, se obtuvo como una combinación de precondicionamiento por la izquierda y proyección. Finalmente, aplicando la estrategia de *reinicio* o *cycling* (Ver [8]), sea $x^{(k+1)} = y^{(k)}$ y $r^{(k+1)} = \rho^{(k)}$, entonces se obtiene el método iterativo Richardson-PR2, el cual, dado un iterado inicial $x^{(0)}$ y haciendo $r^{(0)} = b - Ax^{(0)}$ queda así

$$\begin{aligned} x^{(k+1)} &= x^{(k)} + \lambda_k C_k r^{(k)} \\ r^{(k+1)} &= r^{(k)} - \lambda_k A C_k r^{(k)} \\ \lambda_k &= \frac{\left(AC_k r^{(k)}\right)^T r^{(k)}}{\|AC_k r^{(k)}\|_2^2} \end{aligned} \tag{4.9}$$

Nótese que para $k = 1, 2, \ldots$ el residual puede calcularse en su forma *explícita* $r^{(k)} = b - Ax^{(k)}$, no obstante, en la práctica se prefiere utilizar la fórmula del residual de la ecuación (4.9) la cual se conoce como *residual iterativo* o *residual implícito*. En efecto, el *residual iterativo* suele ser más estable desde el punto de vista numérico que el *residual explícito*, incluso, en la mayoría de los problemas numéricos, sucede que el *residual explícito* puede quedarse *estancado* luego de cierto número de iteraciones, mientras que el *residual iterativo* sigue decreciendo en norma.

Una observación final: Si se hace $C_k = I_n$ para todo k, se obtiene el método iterativo de Richardson *clásico*.

En el algoritmo (4.1) se presenta el método de Richardson-PR2 en su forma *cruda*. Se asume que existe algún procedimiento para obtener el precondicionador C_k en cada iteración, dado que se conoce un precondicionador inicial C_0.

Algoritmo 4.1 Iteración de Richardson-PR2 cruda

- **Algoritmo**
  RichardsonPR2Crudo
- **Descripción**
  Resuelve el sistema $Ax = b$ usando el método iterativo Richardson-PR2.
- **Entradas**
  - A Matriz de n filas por n columnas
  - b Vector columna de n filas.
  - $x^{(0)}$ Vector columna de n filas, iterado inicial.
  - $C^{(0)}$ Matriz de n filas por n columnas, precondicionador inicial.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - x Solución al sistema $Ax = b$.

1: **función** **RichardsonPR2Crudo**$(A, b, x^{(0)}, C_0, N_{max}, \epsilon)$
2: $r^{(0)} \leftarrow b - Ax^{(0)}$
3: **para** $i \leftarrow 0, 1 \ldots, N_{max}$ **hacer**
4: $v \leftarrow C_k r^{(k)}$
5: $u \leftarrow Av$
6: $p_1 \leftarrow u^T r^{(k)}$
7: $p_2 \leftarrow u^T u$
8: $\lambda_k \leftarrow p_1/p_2$
9: $x^{(k+1)} \leftarrow x^{(k)} + \lambda_k v$
10: $r^{(k+1)} \leftarrow r^{(k)} - \lambda_k u$
11: **si** $\|r^{(k+1)}\|_2 < \epsilon$ **entonces**
12: $\leftarrow x^{(k+1)}$
13: **fin si**
14: **fin para**
15: **Error** (Máximo de iteraciones alcanzado)
16: **fin función**

4.5. Algunas estrategias de precondicionamiento para Richardson-PR2

En esta sección se presentarán tres posibles estrategias para la elección de la sucesión de precondicionadores $\{C_k\}$ para la iteración Richardson-PR2; estas son: Precondicionador constante, Precondicionador lineal iteratvo y Precondicionador cuadrático iterativo. Un estudio detallado y referencias bibliográficas de estas estrategias de precondicionamiento pueden encontrarse en [8].

El caso particular de los precondicionadores lineal y cuadrático iterativos son de escaso interés en este trabajo y se presentan con propósitos informativos. Estas dos estrategias de precondicionamiento requieren el cálculo explícito de la sucesión de matrices $\{C_k\}$, algo que para el caso del precondicionamiento del PMCL se hará a través de la sucesión de iteraciones de Schulz.

4.5.1. Precondicionador constante

La estrategia más simple de precondicionamiento para Richardson-PR2 se basa en seleccionar $C_k = C_0$ para $k = 0, 1, 2, \ldots$ una matriz precondicionadora constante. En este caso, lo usual es buscar alguna matriz que se aproxime a A^{-1} y que sea simple de calcular. Dos de las elecciones más usuales para un precondicionador constante son

- $C_k = diag(A)^{-1}$ En este caso ni siquiera es necesario formar explícitamente las matrices C_k, basta con almacenar los valores a_{ii}^{-1} de la inversa de la diagonal de A
- $C_k = L^{-1}$ donde L es la matriz que contiene la porción triangular inferior de A, incluyendo la diagonal principal. Este precondicionador, además de dar una velocidad de convergencia bastante aceptable, no requiere el cálculo explícito de L^{-1}. En efecto, el producto $L^{-1}r^{(k)}$ se puede obtener a través el sistema de ecuaciones $Lx = r^{(k)}$, el cual se puede resolver por una simple sustitución hacia adelante, por ser L una matriz triangular.

4.5.2. Precondicionador lineal iterativo

Si se conoce un *splitting* regular de A de la forma $A = M - N$, donde M es una matriz no singular, se puede definir un esquema iterativo para C_k. Sustituyendo $A = M - N$ en la relación $I_n = AA^{-1}$ se obtiene

$$A^{-1} = (M^{-1}N)A^{-1} + M^{-1} \tag{4.10}$$

Lo cual se puede convertir en el procedimiento iterativo

$$C_{k+1} = (M^{-1}N)C_k + M^{-1} \tag{4.11}$$

Con C_0 una matriz arbitraria. Esta sucesión converge linealmente a A^{-1}.

4.5.3. Precondicionador cuadrático iterativo

Si las matrices de la sucesión $\{C_k\}$ se obtienen a través de una relación recursiva de la forma

$$C_{k+1} = C_k U_k + D_k \tag{4.12}$$

Donde $\{U_k\}$ y $\{D_k\}$ son un par de sucesiones de matrices arbitrarias. Considérese la *matriz residual* R_k definida como

$$R_k = I_n - AC_k \tag{4.13}$$

Entonces, puede hacerse un refinamiento iterativo de la sucesión (4.12) haciendo $D_k = C_k$ y $U_k = R_k$ y así se obtiene una nueva relación de recurrencia para las matrices C_k

$$C_{k+1} = C_k R_k + C_k = C_k \left(I_n + R_k\right) \tag{4.14}$$

La cual converge cuadráticamente a A^{-1}.

Capítulo 5

Precondicionamiento del PMCL por la Aproximación del Método de Schultz

La programación es una de las disciplinas más complejas en las matemáticas aplicadas. Es preferible que los malos matemáticos se queden haciendo matemáticas puras

Edsger W. Dijkstra

En este capítulo se presentarán los algoritmos para la resolución del PMCL a través del método ierativo Richardson-PR2 , usando como precondicionador la aproximación a $A^{\dagger}$ obtenida por el método de Schultz. Se proponen las siguientes variantes sobre el método Richardson-PR2:

- *Algoritmo 1:* Precondicionamiento por la izquierda explícito para la matriz A, usando M_j para algún $j > 1$. De esta forma se resolverá el sistema $M_j A = M_j b$ usando Richardson-PR2 con $C_k = I_n$ para $k = 0, 1, 2, \ldots$ la k-ésima iteración de Richardson-PR2, lo cual es equivalente al método de Richardson de primer orden.
- *Algoritmo 2:* Se extenderá el método Richardson-PR2 para iterar directamente sobre la matriz rectangular A, siendo $C_k = M_k$ para $k = 0, 1, 2, \ldots$

En la próxima sección se analizará cómo la matriz M_k, mejora el condicionamiento del PMCL. Posteriormente se presentarán los algoritmos para el cálculo del producto matriz-matriz $M_k A$ y el producto matriz-vector $M_k b$. El resto del capítulo muestra los algoritmos mencionados en el párrafo anterior.

5.1. M_k como precondicionador al PMCL

Al utilizar la matriz M_k para el precondicionamiento por la izquierda del PMCL, se tiene que el número de condición asociado al PMCL dado por la definición (2.0.4) se convierte en

$$\kappa_{LS}(M_k A, M_k b) = \kappa(M_k A) + \kappa^2(M_k A)\frac{\|M_k(b - Ax_{LS})\|_2}{\|M_k A\|_2 \|x_{LS}\|_2} \tag{5.1}$$

donde x_{LS} es la solución al PMCL.

Obsérvese que por el teorema (3.3.3) $A^T(b - M_k x_{LS}) = 0$. Por lo tanto, se deduce que el precondicionamiento por la izquierda del PMCL, usando la matriz M_k elimina el término de orden $\kappa^2(A)$ en el número de condición del PMCL. Así, finalmente, el número de condición del PMCL precondicionado por M_k es igual a

$$\kappa_{LS}(M_k A, M_k b) = \kappa(M_k A) \tag{5.2}$$

Luego, en vista de que $M_k \to A^\dagger$, a partir de cierto valor de k, se garantiza que $\kappa_{LS}(M_k, M_k b) \approx 1$.

Lo anterior muestra que los iterados de Schultz introducen una gran mejora en el condicionamiento del PMCL, garantizando que cualquier método iterativo pueda converger rápidamente a la solución x_{LS}.

5.2. Cálculo de $M_k A$ y $M_k b$

Como se vio en la demostración del teorema (3.2.2), el iterado de Schultz M_k satisface la siguiente relación de recurrencia

$$I_n - M_k A = (I - M_0 A)^{2^k} \tag{5.3}$$

De donde se obtiene

$$M_k A = I_n - (I - M_0 A)^{2^k} \tag{5.4}$$

Esta relación de recurrencia puede utilizarse en el cálculo de $M_k A$, observando que dada una matriz $C \in \mathbb{R}^{n\times n}$ la potencia C^p con $p \in \mathbb{N}$ y $p = 2^k$ para algún $k \geq 0$ puede expresarse de forma recursiva como sigue

$$C^p = \begin{cases} C^{\frac{p}{2}} C^{\frac{p}{2}} = \left(C^{\frac{p}{2}}\right)^2 & \text{si } p > 1 \\ C & \text{si } p = 1 \\ I_n & \text{si } p = 0 \end{cases} \tag{5.5}$$

De esta forma, la potencia C^{2^k} se puede calcular iterativamente realizando k productos matriz-matriz. En el algoritmo (5.1) se presenta el procedimiento para el cálculo de dicha potencia.

Usando la estrategia del algoritmo (5.1), el algoritmo (5.2) calcula la matriz $M_k A$ para $A \in r_m n$ y $k \geq 1$

Para el cálculo del producto matriz-vector $M_k b$ con $b \in \mathbb{R}^m$, debe observarse que a partir de iteración de Schulz se tiene, para $k \geq 1$

$$\begin{aligned} M_k b &= (2M_{k-1} - M_{k-1} A M_{k-1})\, b \\ &= 2M_{k-1} b - (M_{k-1} A)\, M_{k-1} \end{aligned} \tag{5.6}$$

Sustituyendo el valor de $M_{k-1} A$ obtenido a partir de la ecuación (3.13) se obtiene

Algoritmo 5.1 Cálculo de la potencia A^{2^k} para una matriz A

- **Algoritmo**
  POTENCIA
- **Descripción**
  Dada una matriz cuadrada A, calcula la potencia A^{2^k} para $k > 0$ entero.
- **Entradas**
  - A Matriz de n filas por n columnas.
- **Salidas**
  - P Matriz, potencia 2^k de A.

1: **función** **POTENCIA**(A, k)
2: **si** $k = 0$ **entonces**
3: $\leftarrow I_n$
4: **fin si**
5: $P \leftarrow A$
6: **para** $i \leftarrow 0, 1 \ldots, k$ **hacer**
7: $P \leftarrow P^2$
8: **fin para**
9: $\leftarrow P$
10: **fin función**

Algoritmo 5.2 Cálculo de la matriz M_kA

- **Algoritmo**
  PrecondMkA
- **Descripción**
  Dada una matriz A rectangular, calcula la matriz M_kA donde M_k es la k-ésima iteración de Schulz, para $k > 0$.
- **Entradas**
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - k Entero positivo.
- **Salidas**
  - Matriz M_kA.

1: **función PrecondMkA**(A, k)
2: $P \leftarrow I_n - \left(A^TA/\|A\|_2^2\right)$
3: **para** $i \leftarrow 0, 1 \ldots, k$ **hacer**
4: $P \leftarrow P^2$
5: **fin para**
6: $\leftarrow I_n - P$
7: **fin función**

$$\begin{aligned} M_kb &= 2M_{k-1}b - \left(I_n - (I_n - M_0A)^{2^{k-1}}\right) M_{k-1}b \\ &= M_{k-1}b + (I_n - M_0A)^{2^{k-1}} b \end{aligned} \tag{5.7}$$

Finalmente, combinando la ecuación (5.6) con el algoritmo (5.2), se puede hallar un nuevo algoritmo que calcula de una sola vez la matriz M_kA y el vector M_kb. Basta con calcular el vector M_0b antes de entrar en el ciclo principal del algoritmo, y así, en la k-ésima iteración se tendrá el producto matriz vector $M_{k-1}b$ necesario para el cálculo de M_kb. Esta estrategia se ilustra en el algoritmo (5.3).

5.3. Algoritmo PMCL-Richardson-PR2 1

Para este algoritmo se precondiciona de manera explícita el PMCL y se resuelve el sistema precondicionado de n ecuaciones con n incógnitas

$$M_kA = M_kb$$

usando el esquema de aceleracion Richardson-PR2 con $C_k = I_n$ para $k = 0, 1, \ldots$ lo cual es equivalente al método de Richardson de primer orden. En el algoritmo (5.4) se presenta esta estrategia.

Algoritmo 5.3 Cálculo de la matriz M_kA y el vector M_kb

- **Algoritmo**
  PrecondMkAb
- **Descripción**
  Dada una matriz A rectangular y un vector b, calcula la matriz M_kA y el vector M_kb donde M_k es la k-ésima iteración de Schulz, para $k > 0$.
- **Entradas**
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - b Vector columna de m filas.
  - k Entero positivo.
- **Salidas**
  - Matriz M_kA.
  - Vector M_kb.

1: **función PrecondMkAb**(A, b, k)
2: $M \leftarrow A^T/\|A\|_2^2$
3: $v \leftarrow Mb$
4: $P \leftarrow I_n - MA$
5: **para** $i = 1, \ldots k$ **hacer**
6: $v \leftarrow v + Pv$
7: $P \leftarrow P^2$
8: **fin para**
9: $\leftarrow [I_n - P, v]$
10: **fin función**

Algoritmo 5.4 Algoritmo PMCL-Richardson-PR2 1

- **Algoritmo**
  PCMLRichardsonPR2_1
- **Descripción**
  Resuelve el PMCL asociado a A y b a través del método Richardson-PR2, usando como precondicionador una matriz M_k.
- **Entradas**
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - b Vector columna de m filas.
  - $x^{(0)}$ Vector columna de n filas, iterado inicial.
  - k Entero positivo que indica el número de iteraciones de Schulz para generar el precondicionador.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - x Solución al sistema $Ax = b$.

1: **función** **PCMLRichardsonPR2_1**$(A, b, x^{(0)}, k, N_{max}, \epsilon)$
2: $[b', C] \leftarrow$ **PrecondMkAb**(A)
3: $r^{(0)} \leftarrow b' - Cx^{(0)}$
4: **para** $i \leftarrow 0, 1, \ldots, N_{max}$ **hacer**
5: $u \leftarrow Cr^{(k)}$
6: $p_1 \leftarrow u^T r^{(k)}$
7: $p_2 \leftarrow u^T u$
8: $\lambda_k \leftarrow p_1/p_2$
9: $x^{(k+1)} \leftarrow x^{(k)} + \lambda_k r^{(k)}$
10: $r^{(k+1)} \leftarrow r^{(k)} - \lambda_k u$
11: **si** $\|r^{(k+1)}\|_2 < \epsilon$ **entonces**
12: $\leftarrow x^{(k+1)}$
13: **fin si**
14: **fin para**
15: **fin función**

5.4. Algoritmo PMCL-Richardson-PR2 2

El algoritmo *PMCL-Richardson-PR2 2* es una adaptación del esquema de aceleración Richardson-PR2 para una matriz rectangular $A \in \mathbb{R}^{m\times n}$ con $m > n$. De la ecuación (4.9) se observa que en una iteración de Richardson-PR2 es necesario realizar los siguientes productos matriz-vector:

- $C_k r^{(k)}$.
- $AC_k r^{(k)}$.

Si la matriz $A \in \mathbb{R}^{m\times n}$ es rectangular deben verificarse que los productos anteriores están bien definidos. Como $r^{(k)} = b - Ax^{(k)}$ para $k = 0, 1, \ldots$, entonces $r^{(k)} \in \mathbb{R}^m$; por otro lado, para que el producto $A\left(C_k r^k\right)$ esté bien definido, es necesario que $C_k r^{(k)}$ sea un vector en $\mathbb{R}^n$, lo cual se cumple si $C_k \in \mathbb{R}^{n\times m}$. Así, si se tiene una sucesión $\{C_k\}$ de matrices en $\mathbb{R}^{n\times m}$, entonces la iteración de Richardson-PR2 queda bien definida para una matriz rectangular. Por otro lado, para el caso de matrices cuadradas, en el esquema de aceleración Richardson-PR2 se escoge una sucesión de matrices C_k que converja a A^{-1}, luego, una elección *heurística* para el caso rectangular, puede ser una sucesión de matrices C_k que converja a $A^{\dagger}$.

A partir de lo anterior, se propone la elección de $C_k = M_k$ donde M_k es el k-ésimo iterado del método de Schulz para A, con lo cual se define una extensión del esquema de aceleración Richardson-PR2 para matrices rectangulares.

Para construir el algoritmo es necesario buscar una expresión para calcular $C_{k+1}r^{(k+1)} = M_{k+1}r^{(k+1)}$ en función de los valores de la iteración k para $k = 0, 1, \ldots$.Esta expresión puede hallarse combinando la igualdad $r^{(k+1)} = r^{(k)} - \lambda_k A M_k r^{(k+1)}$, con la ecuación (5.4) siendo $P_k \equiv (I_n - M_0 A)^{2^k}$, de donde se obtiene

$$\begin{aligned} M_{k+1}r^{(k+1)} &= M_k r^{(k+1)} + P_k M_k r^{(k+1)} \\ &= M_k\left(r^{(k)} - \lambda_k A M_k r^{(k)}\right) + P_k M_k\left(r^{(k)} - \lambda_k A M_k r^{(k)}\right) \\ &= M_k r^{(k)} - \lambda_k M_k r^{(k)} + P_k M_k r^{(k)} + \lambda_k P_k^2 M_k r^{(k)} \\ &= (1-\lambda_k) M_k r^{(k)} + P_k M_k r^{(k)} + \lambda_k P_k^2 M_k r^{(k)} \end{aligned} \tag{5.8}$$

Obsérvese que $P_{k+1} = P_k^2$, y por lo tanto se obtiene una relación de recurrencia que permite calcular $M_{k+1}r^{(k+1)}$ en función de los valores obtenidos en la iteración k.

Utilizando la relación de recurrencia de la ecuación (5.8) en el algoritmo del esquema de aceleración Richardson-PR2 en su forma *cruda*, se obtiene una extensión para matrices rectangulares usando $\{M_k\}$ como sucesión de precondicionadores. La implementación de este esquema de aceleracion Richardson-PR2 extendido se presenta en el algoritmo (5.5). Nótese que para este algoritmo, no se puede usar el valor de la norma del residual como condición de parada, pues en el caso del PMCL, no se garantiza que el residual converja a 0. No obstante, la condición de ortogonalidad de las soluciones del PMCL ofrece una condición para detener la iteración, que es la norma del vector $A^T r^{(k)}$, la cual sí converge a 0 cuando $x^{(k)}$ converge a la solución.

Algoritmo 5.5 Algoritmo PMCL-Richardson-PR2 2

- **Algoritmo**
  PCMLRichardsonPR2_2
- **Descripción**
  Resuelve el PMCL asociado a A y b a través del método Richardson-PR2, usando como precondicionador una matriz M_k.
- **Entradas**
  - A Matriz de rango completo de m filas por n columnas con $m > n$.
  - b Vector columna de m filas.
  - $x^{(0)}$ Vector columna de n filas, iterado inicial.
  - N_{max} Entero positivo. Número máximo de iteraciones.
  - ϵ Real positivo. Tolerancia mínima para detener la iteración.
- **Salidas**
  - x Solución al sistema $Ax = b$.

función PCMLRichardsonPR2_2$(A, b, x^{(0)}, N_{max}, \epsilon)$
  $r^{(0)} \leftarrow b - Ax^{(0)}$
  $M \leftarrow A^T / \|A\|_2^2$
  $d \leftarrow Mr^{(0)}$
  $C \leftarrow I_n - MA$
  para $i \leftarrow 0, 1 \ldots, N_{max}$ **hacer**
    $u \leftarrow Ad$
    $p_1 \leftarrow u^T r^{(k)}$
    $p_2 \leftarrow u^T u$
    $\lambda_k \leftarrow p_1/p_2$
    $x^{(k+1)} \leftarrow x^{(k)} + \lambda_k d$
    $r^{(k+1)} \leftarrow r^{(k)} - \lambda_k u$
    si $\|A^T r^{(k+1)}\|_2 < \epsilon$ **entonces**
      $\leftarrow x^{(k+1)}$
    fin si
    $d' \leftarrow (1 - \lambda_k)\, d + Cd$
    $C \leftarrow C^2$
    $d \leftarrow d' + \lambda_k Cd$
  fin para
  Error (Máximo de iteraciones alcanzado)
fin función

Capítulo 6

Experimentación Numérica

Un algoritmo debe verse para creerse

Donald Knuth

Si no fallas al menos el noventa porciento de las veces,
entonces no estás pensando en algo realmente grande

Alan Kay

En este capítulo se presentarán los resultados de varios experimentos numéricos realizados sobre la técnica de precondicionamiento para el PMCL desarrollada a lo largo de este trabajo. Las pruebas numéricas se dividen en dos grupos.

En primer lugar, se estudiarán las propiedades de las matrices M_k como precondicionadores. En los resultados numéricos se verificará que cuando $k \to \infty$, la matriz M_k hace que los autovalores de $M_k A$ se agrupe en torno a $x = 1$, tal como lo sugiere el teorema (3.3.4). Así mismo, se mostrará que los autovalores de $M_k A$ se corresponden a los iterados del método de Newton en una variable para hallar el cero de la función $f(x) = 1 - 1/x$ usando como iterados iniciales los autovalores de $A^T A / \|A\|_2^2$, tal como lo establece dicho teorema. Adicionalmente, se analizará el comportamiento de $\kappa(M_k A)$ en función de k.

En segundo lugar, se harán experimentos numéricos de los algoritmos planteados en el capítulo 5 para la resolución del PMCL. Se analizará el número de iteraciones y tiempo de ejecución para el sistema precondicionado, para diversos valores de M_k tanto en matrices bien condicionadas, como en matrices mal condicionadas. Además, estos métodos se compararán con el esquema de aceleración de convergencia Richardson-PR2 aplicado directamente a las ecuaciones normales, como representante de los *métodos clásicos* para la resolución del PMCL.

Tanto el código fuente de la implementación de todos los experimentos numéricos, así como los archivos de datos contentivos de las matrices de prueba, se encontrarán en el material suplementario de este trabajo.

6.1. Plataforma computacional

La plataforma computacional utilizada para la realización de los experimentos numéricos fue la siguiente:

- *Equipo:* Computador con procesador INTEL© Core 2 DUO@1.8 GHz con 2Gb. de Memoria RAM.
- *Sistema Operativo:* Microsoft Windows© Vista Home Premium.
- *Lenguaje y Entorno de programación:* MATLAB© R2008.a (Versión 7.6.0) para Windows©.

6.2. Matrices de prueba

En esta sección se presentarán los grupos de matrices que serán utilizados en los experimentos numéricos. Se harán pruebas numéricas para matrices densas y para matrices *sparse*. Aún cuando el método tiene su campo de aplicación en matrices densas, debido a que la iteración de Schulz tiende a *rellenar* las matrices *sparse* luego de unas pocas iteraciones, se verificará, que aún aquellas, se observa que el precondicionador acelera notablemente la convergencia.

6.2.1. Matrices densas de prueba

Las matrices densas a utilizar, se generan con diversas funciones provistas por MATLAB©y son las siguientes.

Matriz $NormRand(\mu, \sigma, m, n)$ Esta es una matriz densa de dimensión $m \times n$ con valores aleatorios distribuídos normalmente con promedio μ y desviación estándar σ. Esta matriz se genera con ayuda de la función `randn` de MATLAB©, a través de la siguiente fórmula:

$$randn(m, m). * \sigma + \mu$$

El condicionamiento de estas matrices depende, en el caso general, de la desviación estándar elegida. Para valores muy pequeños de la desviación estándar, los valores singulares tieden a agruparse en torno a valores cercanos a 0, por lo cual la matriz generada es muy mal condicionada, mientras que para valores grandes de la desviación estándar, la matriz tiene un número de condición pequeño.

Matriz $RandSingVal((\sigma_1, \ldots, \sigma_n), m)$ La matriz $RandSingVal$ es una matriz densa de $m \times n$ con valores singulares $\sigma_1, \sigma_2, \ldots \sigma_n$ positivos con valores aleatorios uniformemente distribuidos. Para generar la matriz, se hacen los siguientes pasos, basados en la idea de la descomposición en valores singulares

- Sean $U \in \mathbb{R}^{m \times m}$ y $V \in \mathbb{R}^{n \times n}$ matrices densas con valores aleatorios uniformemente distribuidos, la cual se genera con la función `rand` de MATLAB©.
- Sean Q_1, R_1 y Q_2, R_2 las matrices de las descomposiciones QR de U y V respectivamente, las cuales se obtienen
  vía la función `qr` de MATLAB©.
- La matriz final viene dada por la expresión
  $Q_1 diag(\sigma_1, \sigma_2, \ldots, \sigma_n, 0, \ldots, 0) Q_2^T$

El condicionamiento de estas matrices depende de la distribución de sus valores singulares. Si los valores singulares están agrupados en torno a unos pocos valores, la matriz estará bien condicionada, mientras que si los valores singulares se encuentran muy dispersos, o bien se tienen un grupo de valores singulares muy pequeños y otro grupo de valores singulares muy grandes la matriz estará mal condicionada.

Matriz $RandSVD(m,n)$ La matriz $RandSVD$ es una de las matrices de prueba de MATLAB©, obtenidas a través de la función `gallery`. Esta es una matriz densa de $m \times n$ con n valores singulares distintos, distribuídos geométricamente entre 0 y 1. Estas matrices son muy mal condicionadas con el número de condición del orden de $1/\sqrt{\epsilon}$ donde ϵ es el valor *epsilon* del computador.

Matriz $Vandermonde((v_1, \dots v_m), n)$ La matriz de Vandermonde de orden n para un conjunto de valores reales $v_1, v_2, \dots v_n$ es una matriz, cuya i-ésima fila tiene la forma $(1, v_i, v_i^2, \dots, v_i^{n-1})$ para $i = 1, 2, \dots n$. Esta matriz tiene varias aplicaciones en estadística y otras áreas de ciencias aplicadas, usándose principalmente para representar modelos de regresión polinomial. Es una matriz muy mal condicionada.

6.2.2. Matrices *sparse* de prueba

Las matrices sparse de prueba, se tomaron del conjunto de matrices de problemas de mínimos cuadrados de la *Colección de matrices sparse Harwell-Boeing* (Ver [9]). Más información sobre el formato de las matrices Harwell-Boeing se puede obtener en `http://math.nist.gov/MatrixMarket/formats.html`. Para este trabajo se utilizaron las matrices en formato MATLAB©, las cuales se pueden obtener en `http://www.cise.ufl.edu/research/sparse/mat/HB/`.

A continuación se dará el código de las matrices de la colección Harwell-Boeing utilizadas en este trabajo

- *Matriz ILLC1033 del grupo LSQ* Matriz de 1033×320 mal condicionada.
- *Matriz ILLC185 del grupo LSQ* Matriz de 1850×712 mal condicionada.
- *Matriz WELL1033 del grupo LSQ* Matriz de 1033×320 bien condicionada.
- *Matriz WELL1850 del grupo LSQ* Matriz de 1850×712 bien condicionada.

6.3. Estudio numérico de M_k como precondicionador

En esta sección se presentarán algunos experimentos numéricos que permitirán verificar en la práctica que M_k es un buen condicionador para el PMCL. Los experimentos se dividirán en dos grupos: en el primer grupo de experimentos se analizará el comportamiento del número de condición $\kappa(M_k A)$ en función de k donde k es el k-ésimo iterado de Schulz para A. En el segundo grupo de experimento se analizará el comportamiento de los autovalores de $M_k A$, y se verificará la propiedad de *clustering* que tiene la aproximación a $A^\dagger$. En este último grupo de experimentos se usarán matrices con pocas columnas, con fines de que sea fácil visualizar las gráficas del espectro de $M_k A$ en función de k.

6.3.1. Grupo de Experimentos A

Este grupo de experimentos, se divide en 3 subgrupos de matrices de la siguiente manera.

1. **Subgrupo de matrices A**
   - $DA11 = NormRand(100, 5, 200, 20)$. Matriz de 200×20 con $\kappa(DA11) = 1{,}26 \times 10^2$
   - $DA12 = NormRand(300, 50, 500, 400)$. Matriz de 500×400 con $\kappa(DA12) = 1{,}18 \times 10^3$
   - $DA13 = RandSingVal((5, 10, 20, \ldots 500), 300)$. Matriz de 300×51 con $\kappa(DA13) = 0{,}99 \times 10^2$
2. **Subgrupo de matrices B**
   - $DB11 = RandSVD(500, 10)$. Matriz de 500×10 con $\kappa(DB11) = 6{,}71 \times 10^7$
   - $DB12 = NormRand(300, 50, 500, 400)$. Matriz de 500×400 con $\kappa(DB12) = 4{,}50 \times 10^9$
   - $DB13 = RandSingVal((100, 500, 5000, 50000, 500000, 5000000), 100)$. Matriz de 100×6 con $\kappa(DB13) = 5 \times 10^4$
3. **Subgrupo de matrices C**
   - $DC11 = WELL1033$. Matriz de 1033×320 con $\kappa(DC11) = 1{,}66 \times 10^7$
   - $DC12 = WELL1850$. Matriz de 1850×712 con $\kappa(DC12) = 1{,}13 \times 10^2$
   - $DC13 = ILL1033$. Matriz de 1033×320 con $\kappa(DC13) = 1{,}8 \times 10^4$
   - $DC14 = ILL1850$. Matriz de 1850×712 con $\kappa(DC14) = 1{,}40 \times 10^3$

Para las matrices anteriores se compararán los valores de $\kappa(M_k A)$ en función de k (agregando inicialmente el valor de $\kappa(A)$) con el comportamiento del error en la iteración de Schulz.

Las figuras (6.1), (6.2) y (6.3) muestran gráficamente el comportamiento del error en la iteración de Schulz para las matrices de los grupos A, B y C respectivamente, mientras que las figuras (6.4), (6.5) y (6.6) muestran el valor del número de condición de las matrices $M_k A$ en función de k, para cada subgrupo. Además, el primer punto de dichas gráficas corresponde a $\kappa(A)$.

A partir de dichos resultados pueden determinarse algunas características del precondicionador dado por M_k. En primer lugar nótese que en las primeras iteraciones del método de Schulz, se evidencia que M_k *empeora* el condicionamiento del sistema original; la razón de este hecho es que el iterado inicial del método de Schulz es de la forma $\alpha A^T A$, lo cual implica que $\kappa(M_0 A) \approx \kappa(A)^2$, así, en el caso de matrices muy mal condicionadas, se tiene un aumento considerable del número de condición en el sistema precondicionado. Este resultado puede observarse fácilmente en las gráfica $\kappa(M_k A)$ vs. k: en todos los casos se observa un *salto* grande entre los dos primeros puntos de esta (recuérdese que el primer punto de la gráfica corresponde a $\kappa(A)$). Este hecho puede suponer alguna desventaja, pues se tiene una cierta cantidad de iteraciones de Schulz que van a ser *inútiles* para el precondicionamiento; en efecto, sólo después de algún valor de $k > 1$ es que se conseguirá que las matrices $M_k A$ estén mejor condicionadas que la matriz original A. No obstante, una vez que se alcanza tal valor de k las

siguientes iteraciones de Schulz hacen que el número de condición converja con gran velocidad a 1.

En segundo lugar, el comportamiento del número de condición, presenta una correspondencia con la convergencia del método de Schulz. En las gráficas de error de la iteración de Schulz, puede observarse, de manera análoga a lo anterior, que en las primeras iteraciones el error puede ser incluso creciente, o bien tener un comportamiento errático: mantenerse más o menos constante o fluctuar, hasta que se alcanza cierto valor de k en el cual, en unas pocas iteraciones (en algunos casos 2 o 3) el método converge *explosivamente*. Y precisamente en ese punto donde inicia la rápida convergencia de la iteración de Schulz, también se acelera la convergencia del número de condición del sistema precondicionado M_kA hacia 1.

Como ejemplo, considérese la figura (6.1) correspondiente a la convergencia de Schulz para las matrices del subgrupo A: En particular, obsérvese el patrón de convergencia de la matriz $DA11$ (línea roja), el cual es un patrón típico de convergencia del método de Schulz en una matriz bien condicionada. En las primeras 7 iteraciones el método parece diverger, posteriormente entre las iteraciones 8 y 15 el error de los iterados se estabiliza en torno a un valor de error, y entre la iteración 16 y la iteración 19, el error decae desde aproximadamente 10^{-1} hasta 10^{-8}, es decir, alrededor de siete órdenes de magnitud, superando inclusive el valor de tolerancia dado como entrada en el algoritmo que era de 10^{-5}. Ahora obsérvese la figura 6.4 correspondiente a la gráfica $\kappa(M_kA)$ vs. k; como para la gráfica anterior, considérese el comportamiento del número de condición para el caso de la matriz $DA11$ (línea roja). Puede observarse que en las 7 primeras iteraciones, el número de condición de M_kA se mantiene por encima del número de condición original de A, y justo luego de la iteración 8, que se corresponde a la iteración en la cual el error en el método de Schulz deja de ser creciente, el número de condición de M_kA es mejor que el número de condición de A. Además se puede observar en dicha gráfica, que la pendiente a partir de la iteración 15 se hace mayor, lo cual indica una mayor velocidad de convergencia del número de condición, justo en la misma iteración en donde el método de Schulz comienza su rápida convergencia. En el resto de los experimentos de este grupo se observa un comportamiento semejante.

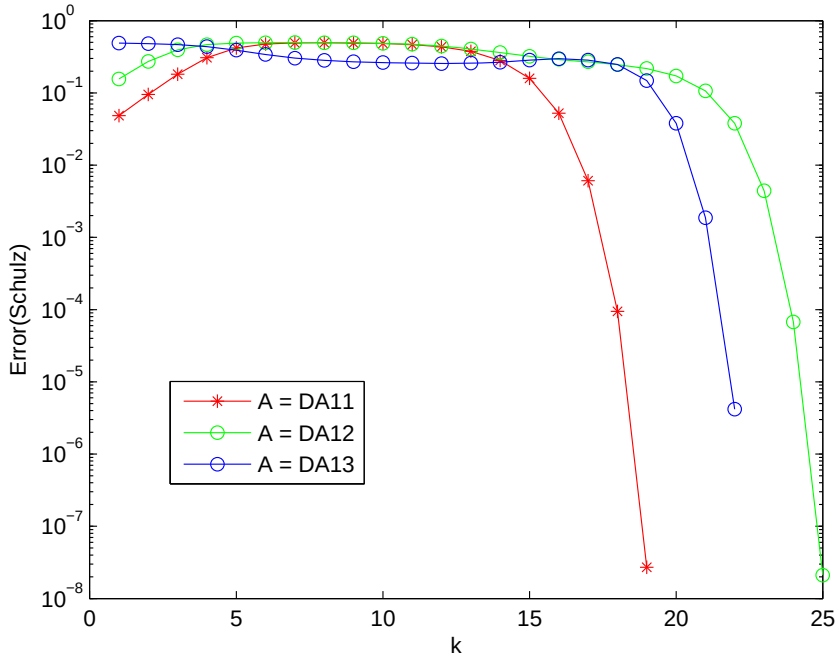

Figura 6.1: Comportamiento de la iteración de Schulz para las matrices del subgrupo A

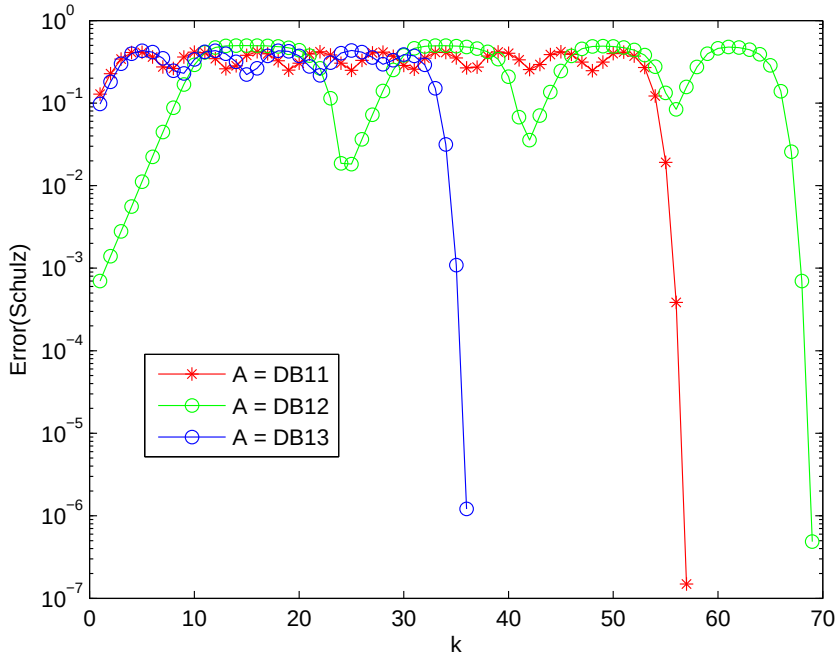

Figura 6.2: Comportamiento de la iteración de Schulz para las matrices del subgrupo B

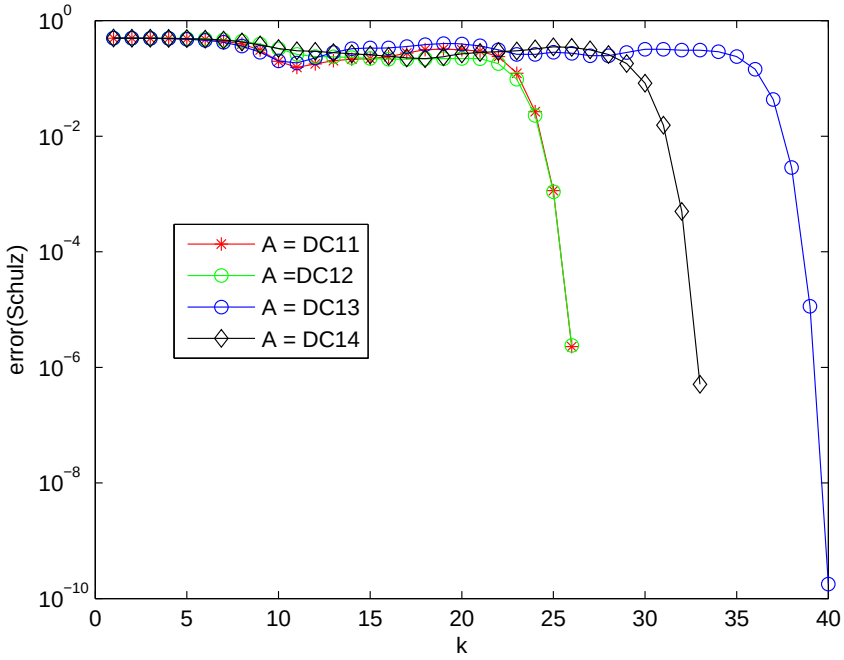

Figura 6.3: Comportamiento de la iteración de Schulz para las matrices del subgrupo C

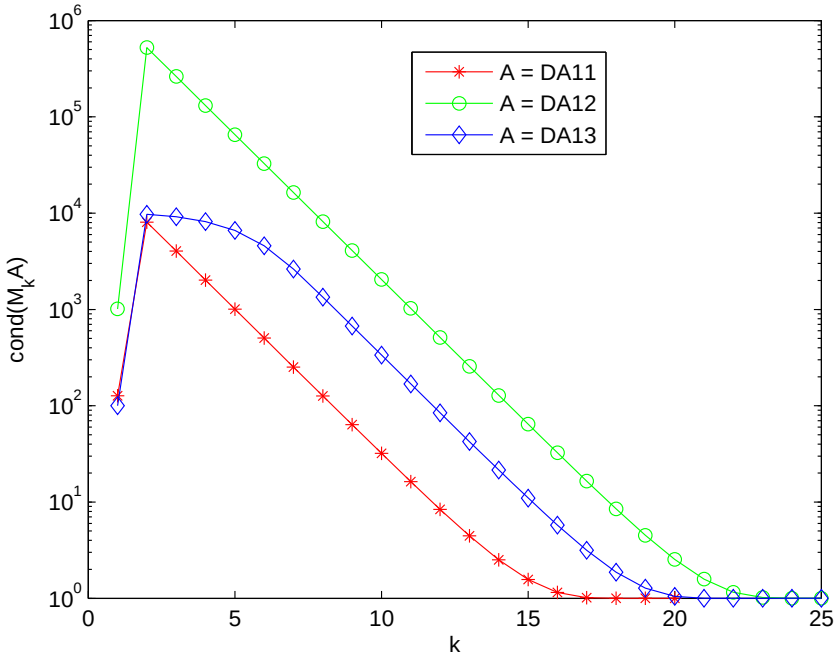

Figura 6.4: Gráfico $M_k A$ en función de k para las matrices del subgrupo A

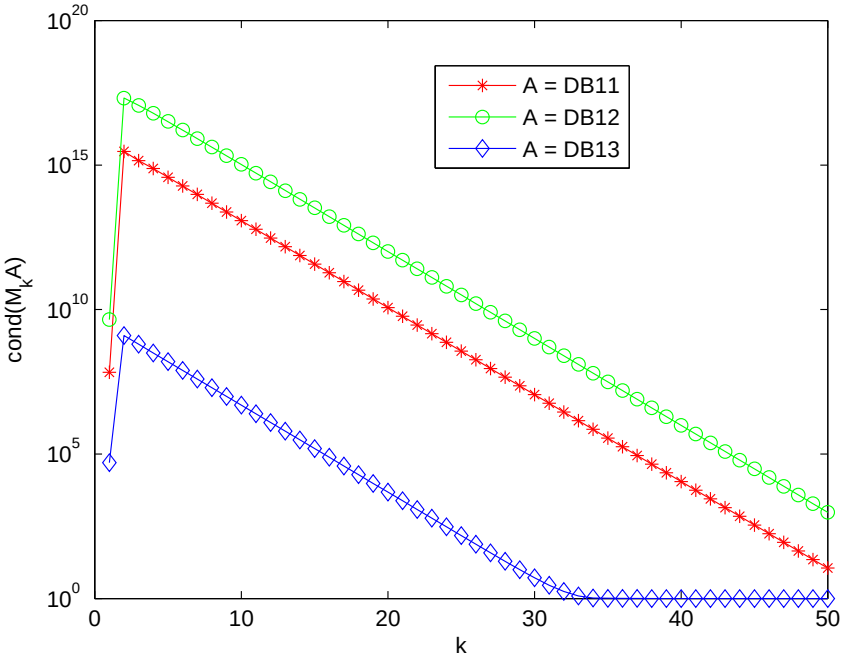

Figura 6.5: Gráfico $M_k A$ en función de k para las matrices del subgrupo B

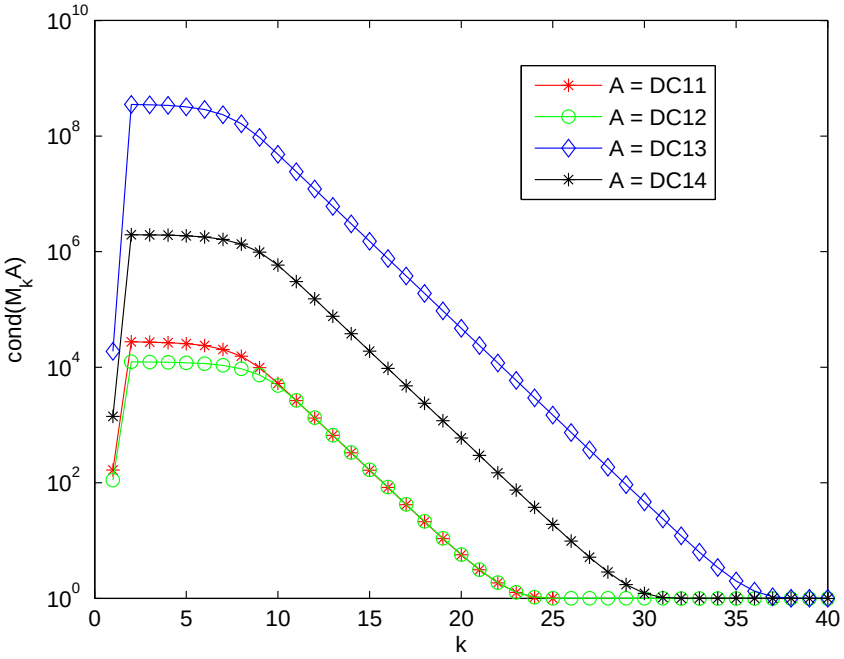

Figura 6.6: Gráfico $M_k A$ en función de k para las matrices del subgrupo C

6.3.2. Grupo de Experimentos B

En este grupo de experimentos, se utilizaron las siguientes matrices densas:

- $DA21 = NormRand(10, 4, 500, 5)$. Matriz de 500×5 con $\kappa(DA21) = 6{,}53$
- $DA22 = NormRand(5, 0{,}1, 20, 5)$. Matriz de 20×5 con $\kappa(DA22) = 4{,}97 \times 10^2$
- $DA23 = RandSVD(20, 5)$. Matriz de 20×5 con $\kappa(DA23) = 6{,}71 \times 10^7$
- $DA24 = Vandermonde((5, 10, 15, \ldots 400), 4)$. Matriz de 40×4 con $\kappa(DA24) = 1{,}03 \times 10^8$

Para cada una de estas matrices se realizó la iteración de Schulz, para obtener las sucesiones de autovalores de las matrices M_kA y se realizó la iteración de Newton en una variable de la función $f(x) = 1 - 1/x$, tomando como iterado inicial los autovalores de la matriz M_0A. Los autovalores de las matrices fueron calculados con la función `eig` de MATLAB©.

El comportamiento del espectro de autovalores aproximados vía los iterados de Newton en una variable M_kA se sumariza en los cuadros (6.1), (6.2), (6.3) y (6.4) para $A = DA21$, $A = DA22$, $A = DA23$ y $A = DA24$ respectivmente. En todos los casos, el error relativo de la la iteración de Newton respecto a los autovalores de las matrices M_kA es menor a 10^2. Por otro lado las figuras (6.7), (6.8), (6.9) y (6.10) muestran gráficamente el comportamiento de los autovalores de las matrices precondicionadas M_kA.

A partir de las gráficas, puede observarse, que de forma análoga al comportamiento del número de condición de M_kA en función d k, los autovalores mantienen la tendencia de converger de manera estable hacia 1; convergiendo de manera lenta en las primeras iteraciones, y luego de cierto valor de $k > 1$ se acelera notablemente la convergencia y el agrupamiento de todos los autovalores en torno a $x = 1$. Esto confirma el resultado teórico del teorema (3.3.4) el cual establecía como consecuencia, que los precondicionadores M_k *comprimen* el espectro de valores singulares de la matriz original en torno a 1, lo cual verifica finalmente que los iterados de Schulz son un buen precondicionador para el PMCL.

Además, en los experimentos se observó una propiedad adicional sobre el espectro de autovalores de las matrices M_kA: en todos los experimentos, para el iterado inicial M_0 siempre se tiene un autovalor muy cercano a 1 y otro mucho menor a 1 en la matriz M_0A. En efecto como $M_0 = A^T/\|A\|_2^2$ entonces todos los autovalores de M_0A son menores a 1. Por otro lado se sabe que el radio espectral de una matriz es una cota inferior de todas las normas matriciales, por lo tanto, si para una matriz se tiene que $\rho(A) \approx \|A\|_2$, entonces la matriz $A^TA/\|A\|_2^2$ siempre tendrá un autovalor muy cercano a 1, el cual va a ser justamente $\rho(A^TA)$.

k	λ_1	λ_2	λ_3	λ_4	λ_5
1	$8,91 \times 10^{-01}$	$3,32 \times 10^{-02}$	$2,08 \times 10^{-02}$	$2,87 \times 10^{-02}$	$2,62 \times 10^{-02}$
2	$9,88 \times 10^{-01}$	$6,53 \times 10^{-02}$	$4,12 \times 10^{-02}$	$5,65 \times 10^{-02}$	$5,18 \times 10^{-02}$
3	$1,00 \times 10^{+00}$	$1,26 \times 10^{-01}$	$8,08 \times 10^{-02}$	$1,10 \times 10^{-01}$	$1,01 \times 10^{-01}$
4	$1,00 \times 10^{+00}$	$2,37 \times 10^{-01}$	$1,55 \times 10^{-01}$	$2,07 \times 10^{-01}$	$1,92 \times 10^{-01}$
4	$1,00 \times 10^{+00}$	$4,17 \times 10^{-01}$	$2,86 \times 10^{-01}$	$3,72 \times 10^{-01}$	$3,46 \times 10^{-01}$
5	$1,00 \times 10^{+00}$	$6,61 \times 10^{-01}$	$4,90 \times 10^{-01}$	$6,06 \times 10^{-01}$	$5,73 \times 10^{-01}$
6	$1,00 \times 10^{+00}$	$8,85 \times 10^{-01}$	$7,40 \times 10^{-01}$	$8,44 \times 10^{-01}$	$8,18 \times 10^{-01}$
7	$1,00 \times 10^{+00}$	$9,87 \times 10^{-01}$	$9,32 \times 10^{-01}$	$9,76 \times 10^{-01}$	$9,67 \times 10^{-01}$
8	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$9,95 \times 10^{-01}$	$9,99 \times 10^{-01}$	$9,99 \times 10^{-01}$
9	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$
10	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$
11	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$
12	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$
13	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$
14	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$

Cuadro 6.1: Aproximación a los autovalores de las matrices M_kA para $A = DA21$, obtenidos a través de la iteración de Newton de $f(x) = 1 - 1/x$

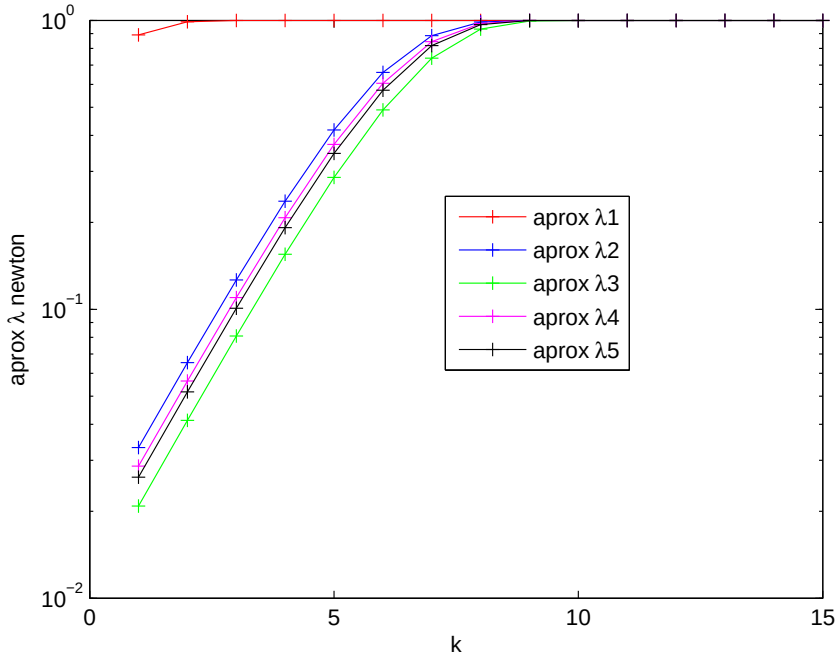

Figura 6.7: Gráfica del espectro de autovalores de la matriz M_kA en función de k para $A = DA21$

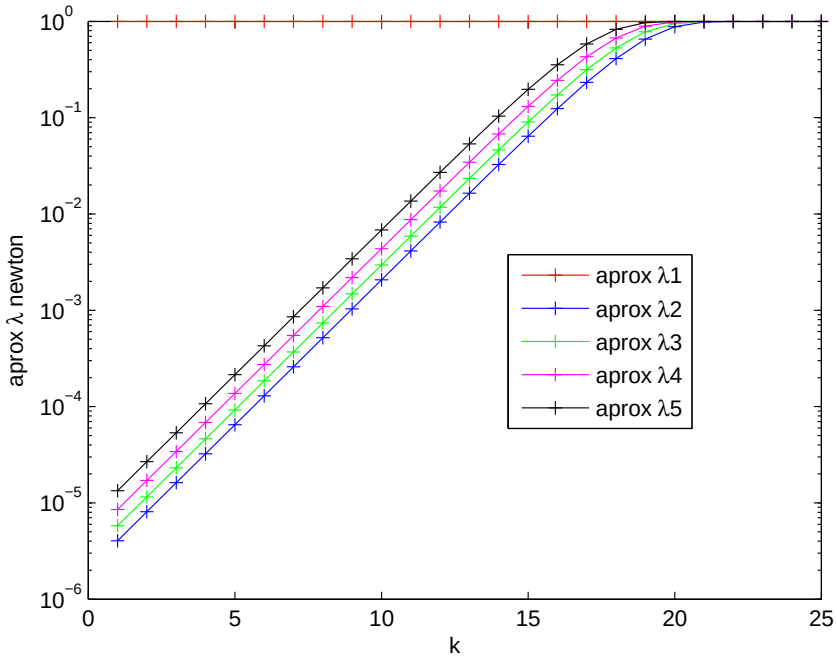

Figura 6.8: Gráfica del espectro de autovalores de la matriz $M_k A$ en función de k para $A = DA22$

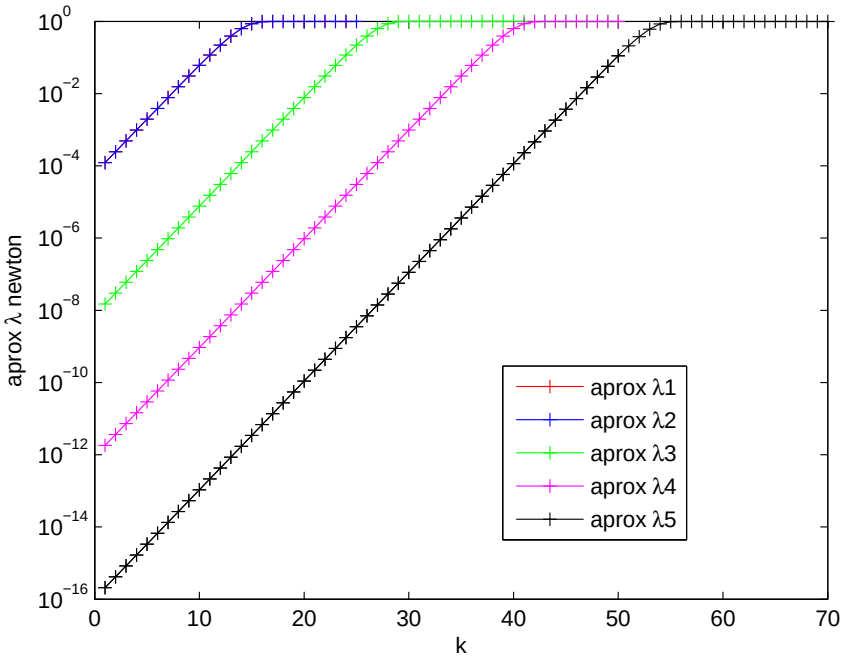

Figura 6.9: Gráfica del espectro de autovalores de la matriz $M_k A$ en función de k para $A = DA23$

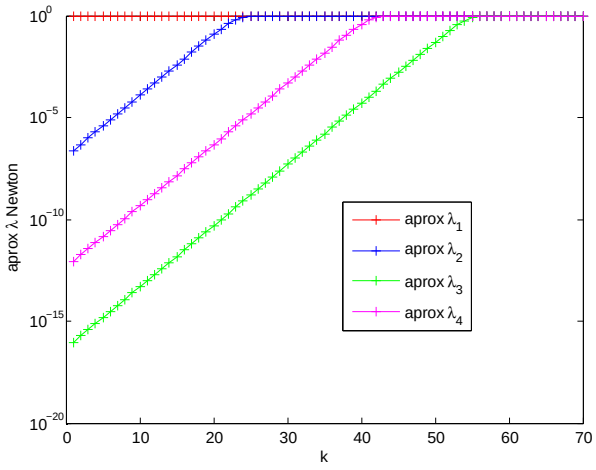

Figura 6.10: Gráfica del espectro de autovalores de la matriz $M_k A$ en función de k para $A = DA24$

k	λ_1	λ_2	λ_3	λ_4	λ_5
1	$9,90 \times 10^{-01}$	$4,05 \times 10^{-06}$	$5,78 \times 10^{-06}$	$8,55 \times 10^{-06}$	$1,34 \times 10^{-05}$
2	$9,90 \times 10^{-01}$	$8,09 \times 10^{-06}$	$1,16 \times 10^{-05}$	$1,71 \times 10^{-05}$	$2,68 \times 10^{-05}$
3	$1,00 \times 10^{+00}$	$1,62 \times 10^{-05}$	$2,31 \times 10^{-05}$	$3,42 \times 10^{-05}$	$5,35 \times 10^{-05}$
4	$1,00 \times 10^{+00}$	$3,24 \times 10^{-05}$	$4,62 \times 10^{-05}$	$6,84 \times 10^{-05}$	$1,07 \times 10^{-04}$
4	$1,00 \times 10^{+00}$	$6,47 \times 10^{-05}$	$9,24 \times 10^{-05}$	$1,37 \times 10^{-04}$	$2,14 \times 10^{-04}$
5	$1,00 \times 10^{+00}$	$1,29 \times 10^{-04}$	$1,85 \times 10^{-04}$	$2,74 \times 10^{-04}$	$4,28 \times 10^{-04}$
6	$1,00 \times 10^{+00}$	$2,59 \times 10^{-04}$	$3,70 \times 10^{-04}$	$5,47 \times 10^{-04}$	$8,56 \times 10^{-04}$
7	$1,00 \times 10^{+00}$	$5,18 \times 10^{-04}$	$7,39 \times 10^{-04}$	$1,09 \times 10^{-03}$	$1,71 \times 10^{-03}$
8	$1,00 \times 10^{+00}$	$1,04 \times 10^{-03}$	$1,48 \times 10^{-03}$	$2,19 \times 10^{-03}$	$3,42 \times 10^{-03}$
9	$1,00 \times 10^{+00}$	$2,07 \times 10^{-03}$	$2,95 \times 10^{-03}$	$4,37 \times 10^{-03}$	$6,83 \times 10^{-03}$
10	$1,00 \times 10^{+00}$	$4,13 \times 10^{-03}$	$5,90 \times 10^{-03}$	$8,72 \times 10^{-03}$	$1,36 \times 10^{-02}$
11	$1,00 \times 10^{+00}$	$8,25 \times 10^{-03}$	$1,18 \times 10^{-02}$	$1,74 \times 10^{-02}$	$2,70 \times 10^{-02}$
12	$1,00 \times 10^{+00}$	$1,64 \times 10^{-02}$	$2,34 \times 10^{-02}$	$3,44 \times 10^{-02}$	$5,33 \times 10^{-02}$
13	$1,00 \times 10^{+00}$	$3,26 \times 10^{-02}$	$4,62 \times 10^{-02}$	$6,77 \times 10^{-02}$	$1,04 \times 10^{-01}$
14	$1,00 \times 10^{+00}$	$6,41 \times 10^{-02}$	$9,03 \times 10^{-02}$	$1,31 \times 10^{-01}$	$1,97 \times 10^{-01}$
16	$1,00 \times 10^{+00}$	$1,24 \times 10^{-01}$	$1,72 \times 10^{-01}$	$2,44 \times 10^{-01}$	$3,55 \times 10^{-01}$
17	$1,00 \times 10^{+00}$	$2,33 \times 10^{-01}$	$3,15 \times 10^{-01}$	$4,29 \times 10^{-01}$	$5,84 \times 10^{-01}$
18	$1,00 \times 10^{+00}$	$4,12 \times 10^{-01}$	$5,31 \times 10^{-01}$	$6,74 \times 10^{-01}$	$8,27 \times 10^{-01}$
19	$1,00 \times 10^{+00}$	$6,54 \times 10^{-01}$	$7,80 \times 10^{-01}$	$8,94 \times 10^{-01}$	$9,70 \times 10^{-01}$
20	$1,00 \times 10^{+00}$	$8,80 \times 10^{-01}$	$9,52 \times 10^{-01}$	$9,89 \times 10^{-01}$	$9,99 \times 10^{-01}$
21	$1,00 \times 10^{+00}$	$9,86 \times 10^{-01}$	$9,98 \times 10^{-01}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$
22	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$
23	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$
24	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$
25	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$

Cuadro 6.2: Aproximación a los autovalores de las matrices $M_k A$ para $A = DA22$, obtenidos a través de la iteración de Newton de $f(x) = 1 - 1/x$

k	λ_1	λ_2	λ_3	λ_4	λ_5
1	$1,22 \times 10^{-04}$	$1,22 \times 10^{-04}$	$1,49 \times 10^{-08}$	$1,82 \times 10^{-12}$	$0,00 \times 10^{+00}$
2	$2,44 \times 10^{-04}$	$2,44 \times 10^{-04}$	$2,98 \times 10^{-08}$	$3,64 \times 10^{-12}$	$0,00 \times 10^{+00}$
3	$4,88 \times 10^{-04}$	$4,88 \times 10^{-04}$	$5,96 \times 10^{-08}$	$7,28 \times 10^{-12}$	$1,00 \times 10^{-15}$
4	$9,76 \times 10^{-04}$	$9,76 \times 10^{-04}$	$1,19 \times 10^{-07}$	$1,46 \times 10^{-11}$	$2,00 \times 10^{-15}$
5	$1,95 \times 10^{-03}$	$1,95 \times 10^{-03}$	$2,38 \times 10^{-07}$	$2,91 \times 10^{-11}$	$3,00 \times 10^{-15}$
6	$3,90 \times 10^{-03}$	$3,90 \times 10^{-03}$	$4,77 \times 10^{-07}$	$5,82 \times 10^{-11}$	$7,00 \times 10^{-15}$
7	$7,78 \times 10^{-03}$	$7,78 \times 10^{-03}$	$9,54 \times 10^{-07}$	$1,16 \times 10^{-10}$	$1,30 \times 10^{-14}$
8	$1,55 \times 10^{-02}$	$1,55 \times 10^{-02}$	$1,91 \times 10^{-06}$	$2,33 \times 10^{-10}$	$2,70 \times 10^{-14}$
9	$3,08 \times 10^{-02}$	$3,08 \times 10^{-02}$	$3,81 \times 10^{-06}$	$4,66 \times 10^{-10}$	$5,40 \times 10^{-14}$
10	$6,06 \times 10^{-02}$	$6,06 \times 10^{-02}$	$7,63 \times 10^{-06}$	$9,31 \times 10^{-10}$	$1,07 \times 10^{-13}$
11	$1,17 \times 10^{-01}$	$1,17 \times 10^{-01}$	$1,53 \times 10^{-05}$	$1,86 \times 10^{-09}$	$2,14 \times 10^{-13}$
12	$2,21 \times 10^{-01}$	$2,21 \times 10^{-01}$	$3,05 \times 10^{-05}$	$3,72 \times 10^{-09}$	$4,29 \times 10^{-13}$
13	$3,93 \times 10^{-01}$	$3,93 \times 10^{-01}$	$6,10 \times 10^{-05}$	$7,45 \times 10^{-09}$	$8,58 \times 10^{-13}$
14	$6,32 \times 10^{-01}$	$6,32 \times 10^{-01}$	$1,22 \times 10^{-04}$	$1,49 \times 10^{-08}$	$1,72 \times 10^{-12}$
15	$8,65 \times 10^{-01}$	$8,65 \times 10^{-01}$	$2,44 \times 10^{-04}$	$2,98 \times 10^{-08}$	$3,43 \times 10^{-12}$
16	$9,82 \times 10^{-01}$	$9,82 \times 10^{-01}$	$4,88 \times 10^{-04}$	$5,96 \times 10^{-08}$	$6,86 \times 10^{-12}$
17	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$9,76 \times 10^{-04}$	$1,19 \times 10^{-07}$	$1,37 \times 10^{-11}$
18	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,95 \times 10^{-03}$	$2,38 \times 10^{-07}$	$2,74 \times 10^{-11}$
19	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$3,90 \times 10^{-03}$	$4,77 \times 10^{-07}$	$5,49 \times 10^{-11}$
20	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$7,78 \times 10^{-03}$	$9,54 \times 10^{-07}$	$1,10 \times 10^{-10}$
21	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,55 \times 10^{-02}$	$1,91 \times 10^{-06}$	$2,20 \times 10^{-10}$
22	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$3,08 \times 10^{-02}$	$3,81 \times 10^{-06}$	$4,39 \times 10^{-10}$
23	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$6,06 \times 10^{-02}$	$7,63 \times 10^{-06}$	$8,78 \times 10^{-10}$
24	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,17 \times 10^{-01}$	$1,53 \times 10^{-05}$	$1,76 \times 10^{-09}$
25	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$2,21 \times 10^{-01}$	$3,05 \times 10^{-05}$	$3,51 \times 10^{-09}$

Cuadro 6.3: Aproximación a los autovalores de las matrices $M_k A$ para $A = DA23$, obtenidos a través de la iteración de Newton de $f(x) = 1 - 1/x$

k	λ_1	λ_2	λ_3	λ_4
1	$9,90 \times 10^{-01}$	$2,40 \times 10^{-07}$	$0,00 \times 10^{+00}$	$8,89 \times 10^{-13}$
2	$9,90 \times 10^{-01}$	$4,80 \times 10^{-07}$	$0,00 \times 10^{+00}$	$1,78 \times 10^{-12}$
3	$9,90 \times 10^{-01}$	$9,60 \times 10^{-07}$	$0,00 \times 10^{+00}$	$3,56 \times 10^{-12}$
4	$1,00 \times 10^{+00}$	$1,92 \times 10^{-06}$	$1,00 \times 10^{-15}$	$7,11 \times 10^{-12}$
5	$1,00 \times 10^{+00}$	$3,84 \times 10^{-06}$	$1,00 \times 10^{-15}$	$1,42 \times 10^{-11}$
6	$1,00 \times 10^{+00}$	$7,68 \times 10^{-06}$	$3,00 \times 10^{-15}$	$2,85 \times 10^{-11}$
7	$1,00 \times 10^{+00}$	$1,54 \times 10^{-05}$	$6,00 \times 10^{-15}$	$5,69 \times 10^{-11}$
8	$1,00 \times 10^{+00}$	$3,07 \times 10^{-05}$	$1,20 \times 10^{-14}$	$1,14 \times 10^{-10}$
9	$1,00 \times 10^{+00}$	$6,14 \times 10^{-05}$	$2,40 \times 10^{-14}$	$2,28 \times 10^{-10}$
10	$1,00 \times 10^{+00}$	$1,23 \times 10^{-04}$	$4,80 \times 10^{-14}$	$4,55 \times 10^{-10}$
11	$1,00 \times 10^{+00}$	$2,46 \times 10^{-04}$	$9,50 \times 10^{-14}$	$9,10 \times 10^{-10}$
12	$1,00 \times 10^{+00}$	$4,91 \times 10^{-04}$	$1,90 \times 10^{-13}$	$1,82 \times 10^{-09}$
13	$1,00 \times 10^{+00}$	$9,82 \times 10^{-04}$	$3,81 \times 10^{-13}$	$3,64 \times 10^{-09}$
14	$1,00 \times 10^{+00}$	$1,96 \times 10^{-03}$	$7,62 \times 10^{-13}$	$7,28 \times 10^{-09}$
15	$1,00 \times 10^{+00}$	$3,92 \times 10^{-03}$	$1,52 \times 10^{-12}$	$1,46 \times 10^{-08}$
16	$1,00 \times 10^{+00}$	$7,83 \times 10^{-03}$	$3,05 \times 10^{-12}$	$2,91 \times 10^{-08}$
17	$1,00 \times 10^{+00}$	$1,56 \times 10^{-02}$	$6,09 \times 10^{-12}$	$5,83 \times 10^{-08}$
18	$1,00 \times 10^{+00}$	$3,10 \times 10^{-02}$	$1,22 \times 10^{-11}$	$1,17 \times 10^{-07}$
19	$1,00 \times 10^{+00}$	$6,10 \times 10^{-02}$	$2,44 \times 10^{-11}$	$2,33 \times 10^{-07}$
20	$1,00 \times 10^{+00}$	$1,18 \times 10^{-01}$	$4,87 \times 10^{-11}$	$4,66 \times 10^{-07}$
21	$1,00 \times 10^{+00}$	$2,22 \times 10^{-01}$	$9,75 \times 10^{-11}$	$9,32 \times 10^{-07}$
22	$1,00 \times 10^{+00}$	$3,95 \times 10^{-01}$	$1,95 \times 10^{-10}$	$1,86 \times 10^{-06}$
23	$1,00 \times 10^{+00}$	$6,35 \times 10^{-01}$	$3,90 \times 10^{-10}$	$3,73 \times 10^{-06}$
24	$1,00 \times 10^{+00}$	$8,66 \times 10^{-01}$	$7,80 \times 10^{-10}$	$7,46 \times 10^{-06}$
25	$1,00 \times 10^{+00}$	$9,82 \times 10^{-01}$	$1,56 \times 10^{-09}$	$1,49 \times 10^{-05}$
26	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$3,12 \times 10^{-09}$	$2,98 \times 10^{-05}$
27	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$6,24 \times 10^{-09}$	$5,97 \times 10^{-05}$
28	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$1,25 \times 10^{-08}$	$1,19 \times 10^{-04}$
29	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$2,50 \times 10^{-08}$	$2,39 \times 10^{-04}$
30	$1,00 \times 10^{+00}$	$1,00 \times 10^{+00}$	$4,99 \times 10^{-08}$	$4,77 \times 10^{-04}$

Cuadro 6.4: Aproximación a los autovalores de las matrices $M_k A$ para $A = DA24$, obtenidos a través de la iteración de Newton de $f(x) = 1 - 1/x$

6.4. Experimentos numéricos de la resolución del PMCL

En este segundo conjunto de experimentos, se mostrarán resultados de diversas ejecuciones de los algoritmos (5.4) y (5.5) desarrollados en el capítulo 5 tanto para matrices densas, como para matrices sparse.

Se presentarán resultados comparativos tanto de las iteraciones del método de Richardson-PR2, como del tiempo de ejecución de cada uno de los algoritmos para todas las matrices de prueba. Además se comparará con la aceleración de Richardson-PR2 aplicada a las ecuaciones normales, lo cual es equivalente al método iterativo de Richardson de primer orden.

6.4.1. Formato de los experimentos numéricos de la resolución del PMCL

La experimentación se dividirá en los siguientes dos grupos de matrices.

- *Grupo de matrices A:* Matrices densas, se ejecutará el algoritmo para 5 matrices de prueba generadas con las funciones especificadas al inicio del capítulo.
- *Grupo de matrices B:* Matrices *sparse*. Se utilizarán las matrices de la colección Harwell-Boeing presentadas anteriormente.

Para cada matriz en los grupos se harán varias ejecuciones de los algoritmos, y se mostrará un cuadro comparativo con los siguientes resultados

- *Alg:* Algoritmo utilizado, *alg1* corresponde al algoritmo (5.4), *alg2* corresponde al algoritmo (5.5) y *alg3* corresponde al algoritmo (4.1) aplicado a las ecuaciones normales.
- k_{sch}*:* para *alg1* este valor indica el número de iteraciones de Schulz realizadas para generar el precondicionador M_k.
- n_{it}*:* Número de iteraciones realizadas por el método. Se denota con ∞ en el caso de que el algoritmo no haya alcanzado la convergencia luego del número máximo de iteraciones establecido.
- t_{pre}*:* Tiempo para generar la matriz precondicionadora M_k en segundos para *alg1*.
- t_{rich}*:* Tiempo que tarda la iteración de Richardson-PR2 en segundos.
- t_{tot}*:* Tiempo total de ejecución del algoritmo en segundos.
- *err:* Error relativo del resultado de la ejecución del algoritmo, respecto una solución de referencia al PMCL para cada matriz, calculado con las funciones de MATLAB©..

En todas las matrices del grupo A, para la resolución de $Ax = b$ se toma $b = (1, 1, 1, \ldots, 1)^T$ con la dimensión adecuada para cada caso. Para los casos de estudio del grupo B, se usa el vector b proveído por la misma colección Harwell-Boeing. En ambos grupos, se utiliza como iterado inicial el vector $x_0 = (1, 1, 1, \ldots 1)^T$ con la dimensión adecuada para cada caso.

En todos los casos de prueba, se ejecutó la iteración con una tolerancia $\epsilon = 10^{-7}$ y una cantidad máxima de iteraciones $M_{max} = 300$.

Para la medición de los tiempos de ejecución se toma el tiempo promedio de 1000 ejecuciones de cada algoritmo.

La solución de referencia utilizada en ambos grupos, se obtiene aplicando el operador `backslash` (\) de MATLAB©. En el caso de un sistema sobredeterminado $Ax = b$, la función $A\backslash b$ retorna la solución al PMCL asociado a A y b la cual se calcula a través de una variante de la descomposición en valores singulares (SVD).

Para el caso del algoritmo (5.5) se observó que en la práctica, el valor de $\|A^T r^{(k))}\|_2$ se estanca, aun cuando los iterados $x^{(k)}$ siguen convergiendo, por lo tanto, para los experimentos, se utilizó como condición de parada para dicho algoritmo, el valor $\|r^{(k+1)} - r^{(k)}\|_2$. Esta elección de condición de parada, en la práctica funcionó para conseguir el resultado con la tolerancia deseada en todos los casos.

6.4.2. Grupo de Experimentos A

Las matrices de prueba de este grupo son las siguientes:

- $DD11 = NormRand(500, 300, 1500, 300)$. Matriz de 1500×300 con $\kappa(DD11) = 15{,}61$
- $DD12 = NormRand(4500, 800, 2000, 800)$. Matriz de 2000×800 con $\kappa(DD12) = 1{,}16 \times 10^3$
- $DD13 = RandSingVal((10, 20, \ldots 5000), 1500)$. Matriz de 1500×500 con $\kappa(DD13) = 4{,}99 \times 10^2$
- $DD14 = RandSingVal(100, 50)$. Matriz de 100×50 con $\kappa(DD14) = 6{,}71 \times 10^7$
- $DD15 = Vandermonde((1/2, 1, 3/2, \ldots 20), 5)$. Matriz de 40×5 con $\kappa(DD15) = 3{,}88 \times 10^7$

Las figuras (6.11) a la (6.4.2) muestran el comportamiento del residual de las iteraciones de Richardson-PR2 para cada matriz. En cada gráfica se muestra el valor del residual en función de la iteración, i, para cada uno de los algoritmos. En el caso de *alg1* se muestra el comportamiento del método para distintos valores de M_k. Los resultados se resumen en los cuadros (6.5) al (6.9). En general, para todas las matrices, precondicionando con M_k para k menor a los valores reportados, no mejora la convergencia del método.

A partir de estos resultados, se observa, que en principio, el precondicionador tiene una desventaja en lo relativo al alto tiempo que se requiere para el precondicionamiento del sistema, el cual es mucho mayor al tiempo empleado por el método iterativo Richardson-PR2, llegando a ser en algunos casos hasta 200 veces mayor.

No obstante, para las matrices de prueba utilizadas, no se obtuvo convergencia del método Richardson-PR2 aplicado a las ecuaciones normales, para el número máximo de iteraciones usando en todos los experimentos ($N_{MAX} = 300$), más aún en todos los casos, para las ecuaciones normales, el error relativo es mayor a 1 respecto a la solución de referencia obtenida con las funciones de MATLAB©, lo cual implica que para la resolución del PMCL el precondicionamiento es un paso necesario aun cuando este sea de un alto costo computacional.

Respecto a la mejora de la convergencia, para el caso del algoritmo 1 (precondicionamiento explícito), puede observarse que la sucesión de precondicionadores M_k ofrecen una gran mejora a la convergencia del PMCL originales, una vez que se consigue el valor de k para el cual el método de Schulz inicia su rápida convergencia. A partir de cierto valor de k *crítico* que depende de la matriz, se observa que cada precondicionador M_k reduce *drásticamente* el número

de iteraciones que se necesitan para la convergencia. Por ejemplo, en el caso de la matriz $DD11$, una matriz bien condicionada, para $k = 5$, el sistema precondicionado con M_k requiere de 106 iteraciones, y para $k = 9$ ya el sistema precondicionado con M_k requiere de apenas 12 iteraciones, casi 10 veces menos. Igualmente el comportamiento se observa, incluso en matrices muy mal condicionadas. Por ejemplo, para la matriz $DD14$ el sistema precondicionado con M_k para $k = 50$ necesita de 98 iteraciones para la convergencia, y con $k = 55$, apenas 5 iteraciones de Schulz más, hacen que el sistema precondicionado requiera de apenas 6 iteraciones para la convergencia. Este mismo comportamiento se puede observar en las gráficas del residual en función de las iterciones para todas las matrices de prueba.

El algoritmo 2, en el cual se utilizan las matrices M_k como sucesión de precondicionadores en el esquema de aceleración de Richardson-PR2, se puede observar un mayor paralelismo del comportamiento del residual en cada iteración del esquema de aceleración de Richardson-PR2, respecto al comportamiento de la convergencia en el método de Shculz, esto es, en las primeras iteraciones, se observa que el residual alcanza valores muy altos, llegando a ser para algunas matrices del orden de 10^{10}, lo cual se debe a ese empeoramiento inicial del número de condición de la matriz que produce la iteración de Schulz. Para el caso de matrices bien condicionadas, como son las matrices $DD11$ y $DD12$, en las primeras iteraciones hay un descenso fuerte del residual, pero aun siendo mucho mayor a 1, mientras que en las matrices mal condicionadas como $DD14$ y $DD15$ se observa que en las primeras iteraciones se mantiene más estable el residual. Posteriormente unas pocas iteraciones en donde hay un aparente estancamiento del residual, y finalmente en las últimas 2 o 3 iteraciones, hay una caída muy rápida del residual, llegando incluso a un nivel varios órdenes menor a la tolerancia escogida para el algoritmo.

Sin embargo, para el caso del algoritmo 2, el tiempo de ejecución es mucho mayor que el utilizado con la combinación de precondicionamiento directo más iteración de Richardson del algoritmo 1, por lo cual hace que no aporte nada significativo la elección del algoritmo 2.

Alg	k_{sch}	n_{it}	$t_{pre} \times 10^3$	$t_{rich} \times 10^3$	$t_{tot} \times 10^3$	err
alg1	5	106	110,93	17,71	128,64	4,65
alg1	7	36	124,08	5,66	129,74	$6{,}6 \times 10^{-3}$
alg1	9	12	146,14	1,76	147,90	$1{,}1 \times 10^{-3}$
alg1	12	3	181,19	0,44	181,63	$9{,}7 \times 10^{-5}$
alg2	N/A	16	N/A	264,45	264,45	$2{,}2 \times 10^{-8}$
alg3	N/A	237	N/A	54,04	54,04	$1{,}3 \times 10^{-5}$

Cuadro 6.5: Resultados de la experimentación numérica para la resolución del PMCL para la matriz $A = DD11$.

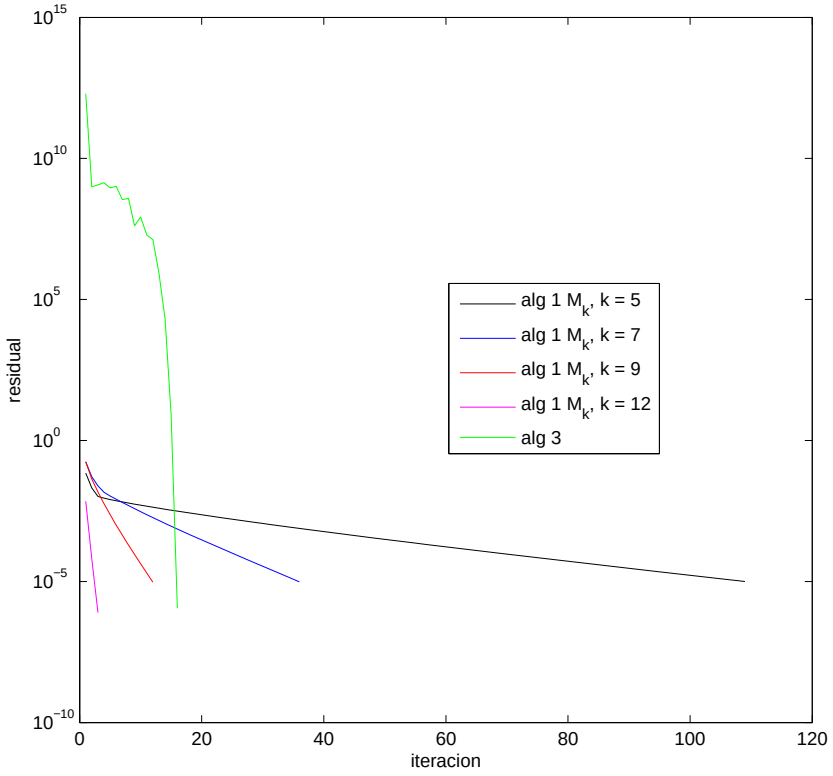

Figura 6.11: Convergencia de los algoritmos para la resolución del PMCL para $A = DD11$

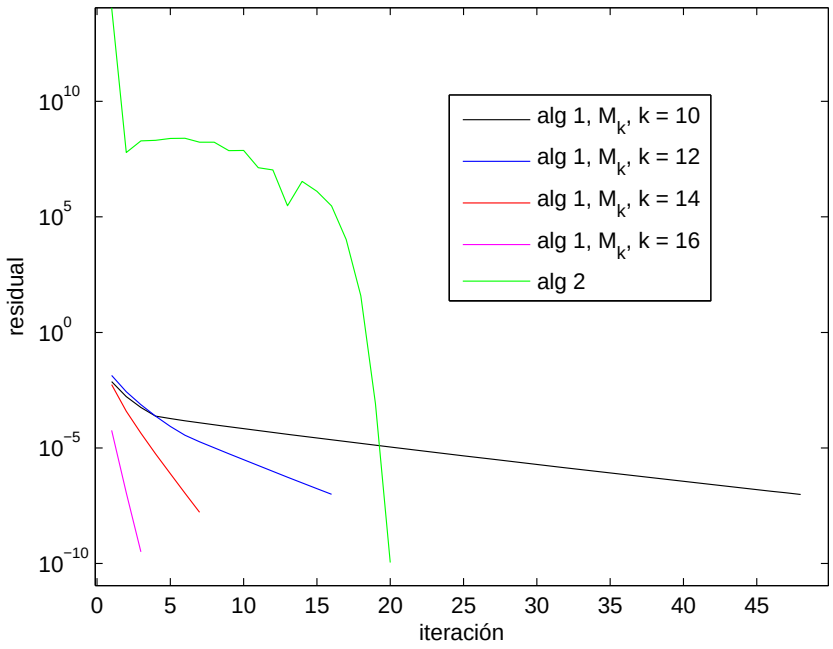

Figura 6.12: Convergencia de los algoritmos para la resolución del PMCL para $A = DD12$

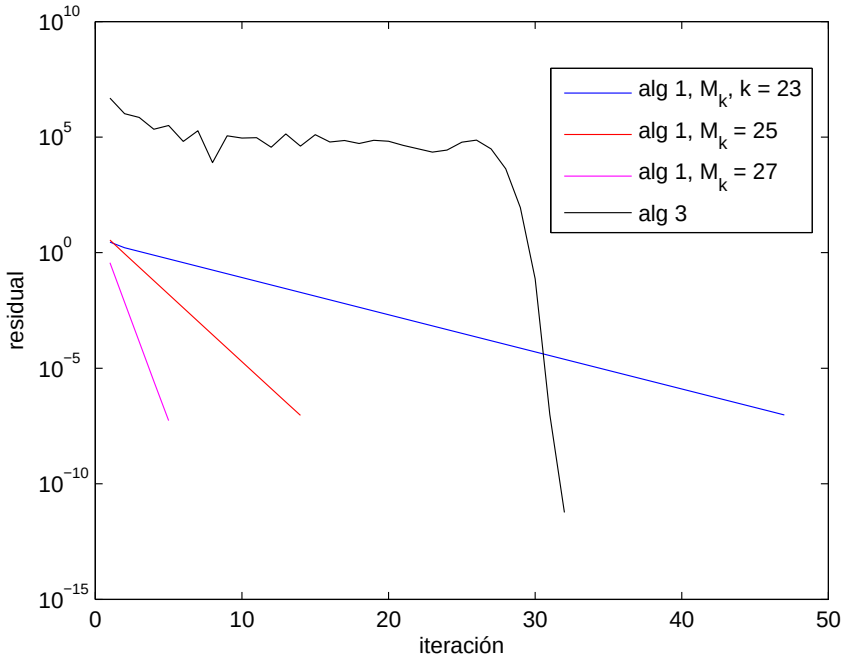

Figura 6.13: Convergencia de los algoritmos para la resolución del PMCL para $A = DD13$

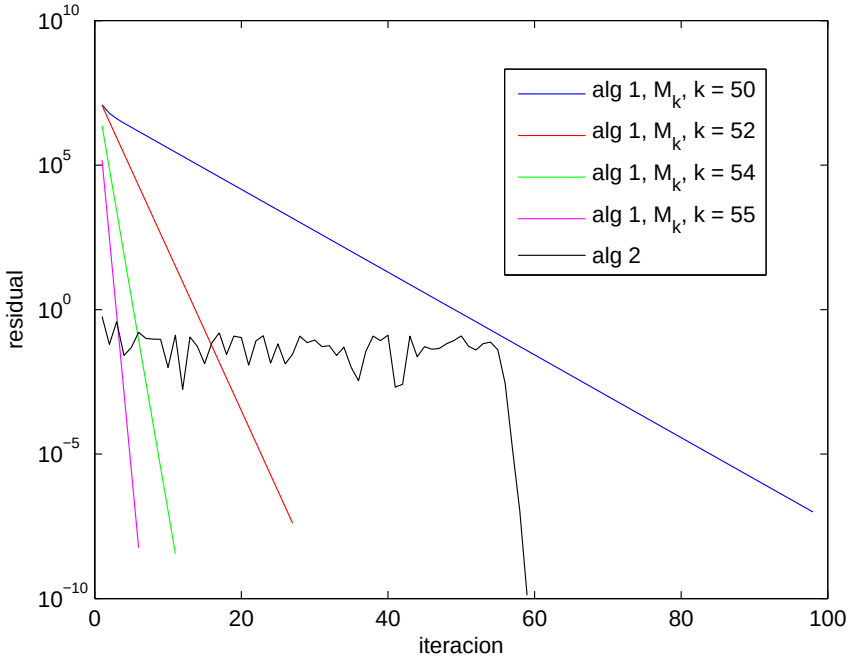

Figura 6.14: Convergencia de los algoritmos para la resolución del PMCL para $A = DD14$

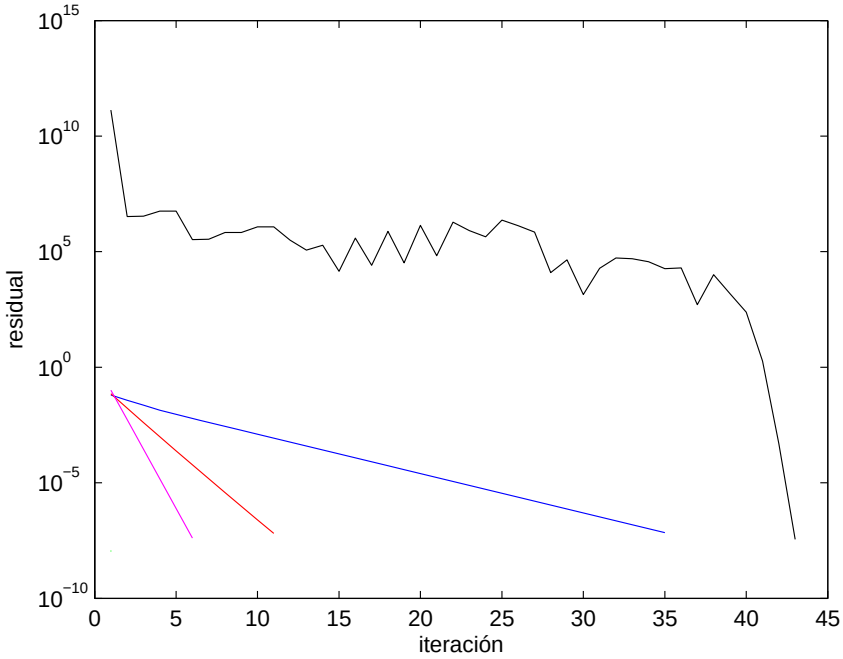

Figura 6.15: Convergencia de los algoritmos para la resolución del PMCL para $A = DD15$

Alg	k_{sch}	n_{it}	$t_{pre} \times 10^3$	$t_{rich} \times 10^3$	$t_{tot} \times 10^3$	err
alg1	7	101	63,73	5,18	68,91	0,55
alg1	10	48	66,24	2,48	68,72	0,07
alg1	12	16	71,19	0,86	72,05	0,01
alg1	14	7	76,09	0,42	76,51	$1{,}2 \times 10^{-3}$
alg1	16	3	80,45	0,21	80,66	$1{,}9 \times 10^{-5}$
alg2	N/A	20	N/A	119,83	119,83	$6{,}9 \times 10^{-8}$
alg3	N/A	∞	N/A	54,48	54,48	$> \times 10^2$

Cuadro 6.6: Resultados de la experimentación numérica para la resolución del PMCL para la matriz $A = DD12$.

Alg	k_{sch}	n_{it}	$t_{pre} \times 10^3$	$t_{rich} \times 10^3$	$t_{tot} \times 10^3$	err
alg1	21	165	3013,75	40,90	3054,65	$5{,}8 \times 10^{-7}$
alg1	23	47	3078,32	11,63	3089,95	$1{,}4 \times 10^{-7}$
alg1	25	14	3356,05	4,04	3360,09	$3{,}6 \times 10^{-8}$
alg1	27	5	3562,03	1,33	3563,36	$1{,}5 \times 10^{-8}$
alg2	N/A	32	N/A	4931,93	4931,93	$1{,}1 \times 10^{-9}$
alg3	N/A	∞	N/A	54,48	54,48	5,6

Cuadro 6.7: Resultados de la experimentación numérica para la resolución del PMCL para la matriz $A = DD13$.

Alg	k_{sch}	n_{it}	$t_{pre} \times 10^3$	$t_{rich} \times 10^3$	$t_{tot} \times 10^3$	err
alg1	50	98	20,05	2,34	22,39	0,18
alg1	52	27	21,54	0,84	22,38	0,18
alg1	54	11	23,06	0,32	23,38	0,18
alg1	55	6	24,34	0,11	24,45	0,18
alg2	N/A	59	N/A	16,89	19,89	0,18
alg3	N/A	∞	N/A	7,23	7,23	5,6

Cuadro 6.8: Resultados de la experimentación numérica para la resolución del PMCL para la matriz $A = DD14$.

Alg	k_{sch}	n_{it}	t_{pre}	t_{rich}	t_{tot}	err
alg1	34	35	0,65	0,26	0,91	$2{,}4 \times 10^{-6}$
alg1	35	11	0,79	0,21	1,00	$2{,}2 \times 10^{-6}$
alg1	36	6	0,81	0,14	0,95	$1{,}9 \times 10^{-6}$
alg2	N/A	43	N/A	1,71	1,71	$7{,}2 \times 10^{-7}$
alg3	N/A	∞	N/A	5,42	5,42	$> 10 \times 10^2$

Cuadro 6.9: Resultados de la experimentación numérica para la resolución del PMCL para la matriz $A = DD15$.

6.4.3. Grupo de Experimentos B

Las matrices de prueba de este grupo son las siguientes:

- $DD21 = WELL1033$. Matriz de 1033×320 con $\kappa(DD21) = 1{,}66 \times 10^7$
- $DD22 = WELL1850$. Matriz de 1850×712 con $\kappa(DD22) = 1{,}13 \times 10^2$
- $DD23 = ILL1033$. Matriz de 1033×320 con $\kappa(DD23) = 1{,}8 \times 10^4$
- $DD24 = ILL1850$. Matriz de 1850×712 con $\kappa(DD24) = 1{,}40 \times 10^3$

Las figuras (6.16) a la (6.4.3) muestran el comportamiento del residual de las iteraciones de Richardson-PR2 para cada matriz. En cada gráfica se muestra el valor del residual en función de la iteración, i, para cada uno de los algoritmos. En el caso de *alg1* se muestra el comportamiento del método para distintos valores de M_k. Los resultados se resumen en los cuadros (6.10) al (6.13).

Para el caso de matrices *sparse*, se puede observar un comportamiento análogo al caso de matrices densas en cuanto a la mejora de la convergencia del método de Richardson-PR2 con el precondicionamiento como en el algoritmo 1, así como para el algoritmo 2. No obstante, puede verse que la diferencia de tiempo entre la generación del precondicionador es mayor en el caso general. Esto se debe a que en el caso de matrices *sparse* evidentemente, la iteración de Richardson-PR2 es más rápida pues se puede aprovechar el hecho de que no es necesario construir explícitamente, la matriz $A^T A$ para el caso de las ecuaciones normales, mientras que el precondicionamiento con M_k hace que la matriz se vuelva densa.

Para el grupo de prueba de matrices *sparse* igualmente se observa que el algoritmo 2 no ofrece una mejora sustancial al precondicionaiento explícito del algoritmo 1.

Alg	k_{sch}	n_{it}	$t_{pre} \times 10^3$	$t_{rich} \times 10^3$	$t_{tot} \times 10^3$	err
alg1	18	120	211,28	12,04	223,32	$1{,}3 \times 10^{-10}$
alg1	20	33	227,04	3,26	230,30	$5{,}7 \times 10^{-11}$
alg1	22	11	254,64	1,15	255,79	$4{,}2 \times 10^{-11}$
alg1	24	4	284,69	0,48	285,17	$3{,}9 \times 10^{-11}$
alg2	N/A	27	N/A	360,34	360,34	$2{,}6 \times 10^{-11}$
alg3	N/A	∞	N/A	319,52	319,52	1,5

Cuadro 6.10: Resultados de la experimentación numérica para la resolución del PMCL para la matriz $A = DD21$.

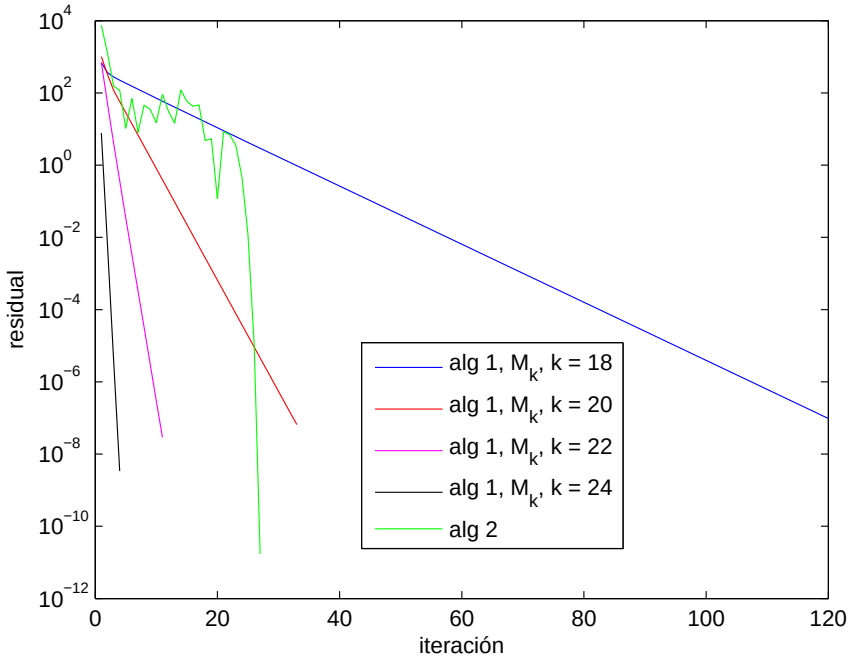

Figura 6.16: Convergencia de los algoritmos para la resolución del PMCL para $A = DD21$

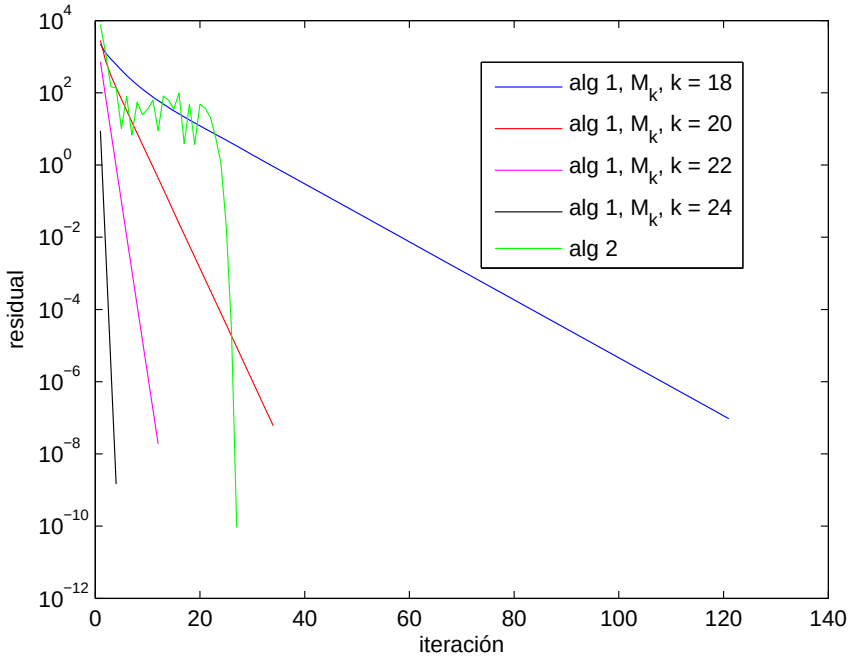

Figura 6.17: Convergencia de los algoritmos para la resolución del PMCL para $A = DD22$

Alg	k_{sch}	n_{it}	$t_{pre} \times 10^3$	$t_{rich} \times 10^3$	$t_{tot} \times 10^3$	err
alg1	18	121	2127,39	63,84	2191,23	$1{,}5 \times 10^{-10}$
alg1	20	34	2516,00	24,65	2540,65	$6{,}8 \times 10^{-11}$
alg1	22	12	2799,13	7,81	2806,94	$6{,}1 \times 10^{-11}$
alg1	24	4	2856,82	2,90	2859,72	$6{,}1 \times 10^{-11}$
alg2	N/A	27	N/A	3508,61	3508,61	$3{,}8 \times 10^{-11}$
alg3	N/A	∞	N/A	347,72	347,72	0,7

Cuadro 6.11: Resultados de la experimentación numérica para la resolución del PMCL para la matriz $A = DD22$.

Alg	k_{sch}	n_{it}	$t_{pre} \times 10^3$	$t_{rich} \times 10^3$	$t_{tot} \times 10^3$	err
alg1	30	165	361,22	2,43	363,65	$1{,}5 \times 10^{-7}$
alg1	32	67	408,54	6,68	415,22	$1{,}5 \times 10^{-7}$
alg1	34	11	413,59	2,25	415,84	$1{,}5 \times 10^{-8}$
alg1	36	6	429,62	0,70	430,32	$1{,}5 \times 10^{-8}$
alg2	N/A	40	N/A	592,08	592,08	$1{,}1 \times 10^{-9}$
alg3	N/A	∞	N/A	57,64	57,64	0,9

Cuadro 6.12: Resultados de la experimentación numérica para la resolución del PMCL para la matriz $A = DD23$.

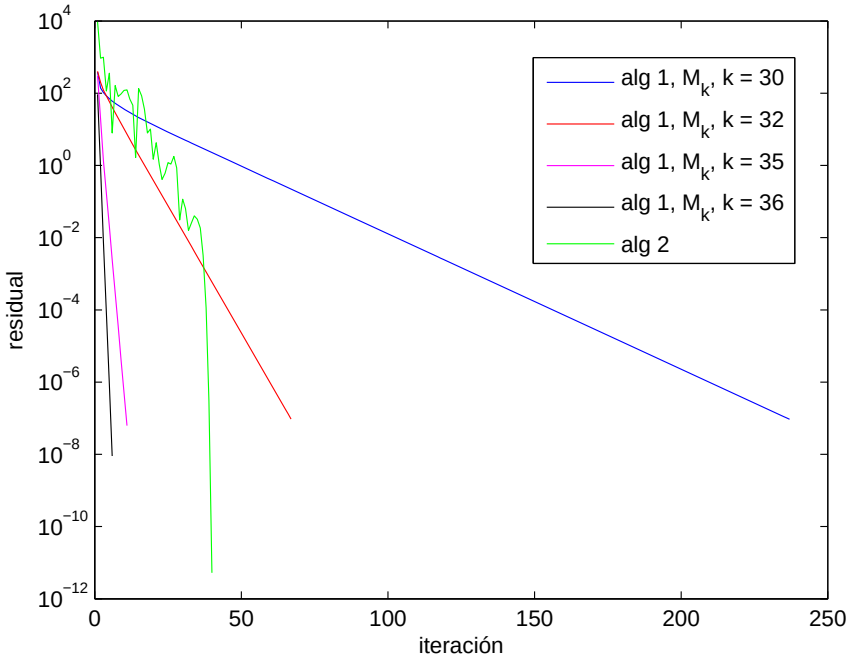

Figura 6.18: Convergencia de los algoritmos para la resolución del PMCL para $A = DD23$

Alg	k_{sch}	n_{it}	$t_{pre} \times 10^3$	$t_{rich} \times 10^3$	$t_{tot} \times 10^3$	err
alg1	18	181	2870,18	108,84	2929,02	$9{,}5 \times 10^{-10}$
alg1	20	54	2931,49	30,91	2962,40	$9{,}5 \times 10^{-10}$
alg1	22	18	3034,97	10,27	3045,24	$9{,}5 \times 10^{-10}$
alg1	24	6	3216,58	4,08	3219,66	$9{,}5 \times 10^{-10}$
alg2	N/A	34	N/A	4088,09	4088,09	$9{,}5 \times 10^{-10}$
alg3	N/A	∞	N/A	314,21	314,21	0,79

Cuadro 6.13: Resultados de la experimentación numérica para la resolución del PMCL para la matriz $A = DD24$.

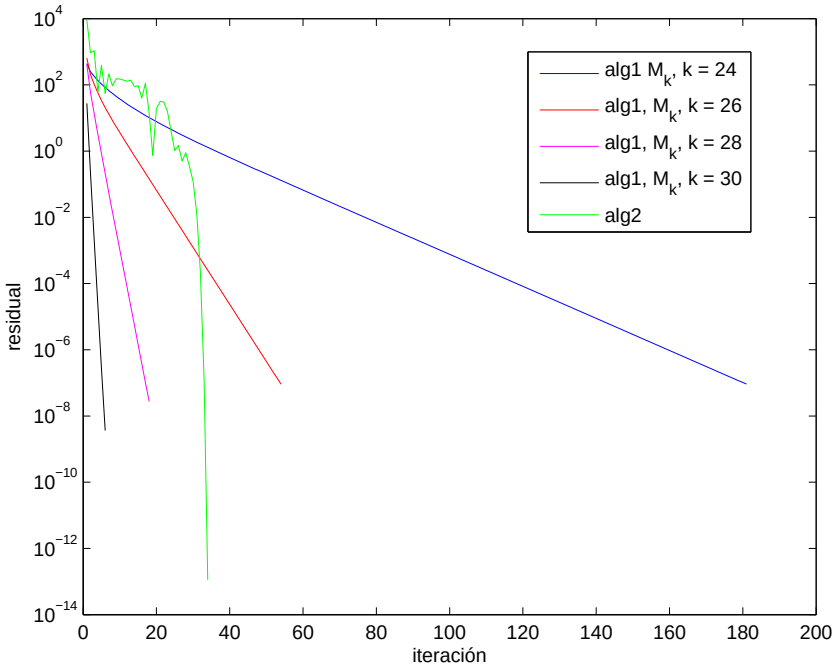

Figura 6.19: Convergencia de los algoritmos para la resolución del PMCL para $A = DD24$

Capítulo 7

Conclusiones

Ten siempre a Itaca en tu mente.
Llegar allí es tu destino.
Mas no apresures nunca el viaje.
Mejor que dure muchos años
y atracar, viejo ya, en la isla,
enriquecido de cuanto ganaste en el camino
sin aguantar a que Itaca te enriquezca.
Itaca te brindó tan hermoso viaje.
Sin ella no habrías emprendido el camino.
Pero no tiene ya nada que darte.
Aunque la halles pobre, Itaca no te ha engañado.
Así, sabio como te has vuelto, con tanta experiencia,
entenderás ya qué significan las Itacas.

Constantino Kavafis

En este último capítulo se presentarán algunas conclusiones obtenidas luego de la realización del trabajo y a partir de la experimentación numérica.

El principal aporte del trabajo, es el estudio y aplicación del precondicionamiento a sistemas lineales en matrices rectangulares. Si bien la tendencia actual del análisis numérico en el área de resolución de sistemas lineales es la búsqueda de técnicas de precondicionamiento y precondicionadores para el caso general; en la literatura no se encontró ninguna referencia a precondicionadores rectangulares en el contexto del PMCL. En efecto, y tal como se vio en el capitulo 2, los precondicionadores *genericos* como la factorización de Cholesky incompleta o la factorización LU incompleta, se aplican al sistema de ecuaciones normales, el cual es un sistema de n ecuaciones con n incógnitas. Sin embargo, para el caso de las ecuaciones normales, estos precondicionadores presentan serios problemas de estabilidad numérica (Ver [34]) Por consiguiente, esta nueva idea de un precondicionador rectangular que no presenta problemas de estabilidad numérica de precondicionadores *clásicos* aplicados a las ecuaciones normales adquiere relevancia, y se deja como un punto de inicio para la investigación de una nueva línea de precondicionadores rectangulares para el PMCL.

Por otro lado, se planteó una extensión de un método iterativo, que fue concebido en principio para matrices cuadradas, como lo es el esquema de acelereación de convergencia Richardson-PR22, a matrices rectangulares. A pesar de que los resultados de la convergencia de este método extendido no presentaron una mejora sustancial en tiempo para las matrices de

prueba, debido a que no se consiguió una expresión que eliminara los productos matriz-matriz asociados a la iteración de Schulz en el algoritmo (5.5) (apenas se logro reducir el cálculo de dos productos matriz-matriz a uno solo) sí se observó una mejora interesante en cuanto a la convergencia a la solución del problema: el método produce inicialmente una serie de iteraciones en apariencia *inútiles*, pero finalmente, al conseguir corregir el condicionamiento, el método alcanza una convergencia muy acelerada en pocas iteraciones, usualmente en 2 o 3 iteraciones, el residual se reduce en varios órdenes de magnitud.

Respecto al precondicionamiento explícito al PMCL propuesto en el algoritmo (5.4), se observa que la aproximación a la matriz pseudoinversa obtenida por el método de Schulz es un precondicionador muy efectivo, aun cuando su cálculo es costoso en tiempo, debido, nuevamente a la necesidad de efectuar productos matriz-matriz. No obstante, aun cuando el tiempo invertido en precondicionar el problema sea varias veces mayor al tiempo empleado por el método iterativo, debe observarse, que para el caso de las ecuaciones normales, no se observó convergencia para ninguna de las matrices de prueba utilizadas al intentar resolver el sistema de ecuaciones normales sin precondicionar. Más aun, se observó que en dicho sistema, el método de aceleración de Richardson-PR2 tiende a *estancarse* luego de algunas iteraciones.

De lo anterior puede concluirse que a pesar de que el precondicionamiento de un sistema lineal puede implicar un alto costo computacional, en muchas ocasiones es un paso indispensable para obtener la solución a dicho sistema; máxime en el caso de problemas que son mal condicionados y típicamente inestables como el caso de las ecuaciones normales aplicadas al PMCL. Además, el precondicionamiento evita la necesidad de elegir métodos iterativos que tengan asociado un alto costo computacional para el cálculo de direcciones de búsqueda en cada iteración, tal como los métodos de proyección. En efecto, el método iterativo utilizado en este trabajo: el esquema de aceleración de convergencia Richardson-PR2, utiliza como dirección de búsqueda en la iteración $n+1$ el vector residual de la iteración n escalado con un valor real *adecuado*, con lo cual, se puede utilizar la mayor parte de la capacidad computacional en la obtención de un *buen* precondicionador, que generalmente garantizará la convergencia a la solución correcta del problema.

Bibliografía

[1] Akritas, A.G., Malaschonok, G.I., y Vigklas, P.S. The SVD-Fundamental theorem of linear algebra. *Nonlinear Analysis: Modelling and Control Vol. 11 N^o 2*, 2006.

[2] Barrett, R., Berry, M., Chan, T. F., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., y Van der Vorst, H. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition.* SIAM, Philadelphia, PA, 1994.

[3] Benzi, M. Preconditioning techniques for large linear systems: a survey. *Journal of computational physics*, páginas 418–477, 2002.

[4] Benzi, M., Giraud, L., y All, G. Sparse approximate inverse preconditioning for dense linear systems arising in computational electromagnetics. *Numerical Algorithms*, 16:1–15, 1997.

[5] Björck, A. *Numerical methods for least squares problem.* Society for Industrial and Applied Mathematics, 1996.

[6] Bourden, R. y Douglas-Faires, J. *Análisis Numérico.* International Thompson Editores, sexta edición, 1998.

[7] Brezinski, C. Variations on Richardson's method and acceleration. *Bull. Soc. Math. Belg.*, páginas 33–44, 1996.

[8] Brezinski, C. *Projection methods for systems of equations.* Elsevier Science Publishers B. V., 1997.

[9] Duff, I., Grimes, R., y Lewis, J. Users' guide for the Harwell-Boeing sparse matrix collection (Release I). Informe técnico, European Centre for Rresearch and Advanced Training in Scientific Computation (CERFACS), 1992.

[10] Forsman, K., Gropp, W., Kettunen, L., Levine, D., y Salonen, J. Solution of dense systems of linear equations arising from integral equation formulations. *Antennas and propagation magazine*, 37:96–100, 1995.

[11] Freud, R. W., Golub, G. H., y Nachtigal, N. Iterative solution of linear systems. *Acta Numerica*, 1992.

[12] Golub, G. Numerical methods for solving least squares problems. *Numer. Math*, 7:206–216, 1965.

[13] Golub, G., Klema, V., y Stewart, G. Rank degeneracy and least squares problems. Informe técnico, Computer Sciences Department, Stanford University, 1976.

[14] Golub, G. y Plemmons, R. Large scale geodetic least squares adjustment by dissection and orthogonal decomposition. Informe técnico, Computer Sciences Department, Stanford University, 1979.

[15] Golub, G. y Van Loan, C. *Matrix Calculations*. Johns Hopkings University Press, tercera edición, 1996.

[16] Golub, G. y Varga, R. Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and the second order Richardson iterative methods part I. *Numer. Math*, 3:147–156, 1961.

[17] Golub, G. y Varga, R. Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and the second order Richardson iterative methods part II. *Numer. Math*, 3:157–168, 1961.

[18] Grossman, S. *Álgebra Lineal*. Mac Graw Hill Latinoamericana, quinta edición, 1999.

[19] Herón, B., Issard-Rorch, F., y Picard, C. *Analyse numérique: Exercises et problèmes corrigés*. Dunod, París, 1999.

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

[21] Kariya, T. y Kurata, H. *Generalized Least Squares*. John Wiley and Sons Ltd., 2004.

[22] Kincaid, D. y Cheney, W. *Numerical Analysis, Mathematics of scientific computing*. Brooks/Cole Publishing Company, 1991.

[23] La Cruz, W. y Raydán, M. Residual iterative schemes for large-scale nonsymmetric positive definite linear systems. *Computational and applied mathematics*, 49:151–173, 2008.

[24] Leach, S. Singular value decomposition, a primer. Informe técnico, Department of Computer Science, Brown University, Providence, USA, 1997.

[25] Lin, C. y Saigal, R. An incomplete Cholesky factorization for dense matrices. *Applied Numerical Mathematics*, 536:536–558, 2000.

[26] Marsden, J. y Hoffman, M. *Elementary Classical Analisys*. W. H. Freeman, 1974.

[27] Molina, B. y Raydán, M. *Métodos iterativos tipo Krylov para sistemas lineales*. Centro de Estudios Avanzados, Instituto Venezolano de Investigaciones Científicas (IVIC) Caracas - Venezuela, 2004. ISBN 980-261-078-X.

[28] Nievergelt, Y. A tutorial history of least squares with applications to astronomy and geodesy. *J. Comput. Appl. Math.*, 121(1–2):37–72, 2000.

[29] Opfer, G. y Schober, G. Richardson's iteration for nonsymmetric matrices. *Linear Algebra Appl*, 58:343–361, 1984.

[30] Ortega, J. *Numerical Analysis: A second course.* Classics in applied mathematics. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1990.

[31] Ortega, J. y Rheinboldt, W. *Iterative solution of nonlinear equations in several variables.* Classics in applied mathematics. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[32] Richardson, L.F. The approximate arithmethical solution by finite differences of physical problems involving differential equations, with application to the stress in a masonry dam. *Philos. Trans. Roy. Soc. London Series A*, páginas 307–357, 1910.

[33] Saad, Y. *Iterative methods for sparse linear systems.* Y. Saad, segunda edición, 2000.

[34] Saad, Y. y Sosonkina, M. Enhanced preconditioners for large sparse least squares problem. Informe técnico, Minnesota Supercomputer Institute, 2001.

[35] Saad, Y. y Van der Vorst, H.A. Iterative solutions of linear systems in the 20th century. Informe técnico, Minnesota Supercomputer Institute, 1999.

[36] Schmidt, E. y Stewart, G. W. On the early history of singular value decomposition. Informe técnico, Institute of Advanced Computer Studies, University of Maryland, 1992.

[37] Schulz, G. Iterative berechung der reziproken matrix. *ZAMM - Zeitschrift für Angewandte Mathematik und Mechanik*, páginas 57–59, 1933.

[38] Sheynin, O. On the history of the principle of least squares. *Archive for history of exact sciences*, 49:39–54, 1993.

[39] Stoer, J. y Bulirsch, R. *Introduction to numerical analysis.* Springer-Verlag, 1993.

[40] Wolberg, J. *Data analysis using the method of the least squares.* Springer-Verlag, 2006.

[41] Wolf, P. y Ghilani, C. *Adjustment Computations: Statistics and Least Squares in Surveying and GIS.* John Wiley and Sons Ltd., 1997.

[42] Yan, Y. Sparse preconditioned iterative methods for dense linear systems. *SIAM J. Sci. Comp*, 15:1190–1200, 1994.

[43] Young, D.M. On Richardson's method for solving linear systems with positive definite matrices. *J. Math Phys*, 32:243–255, 1954.

[44] Young, D.M. *Iterative solution of large linear systems.* Academic Press, New York, 1971.