Un método cuasi-Newton para ecuaciones polinomiales matriciales

Eduard Mauricio Macías Caicedo

Universidad del Cauca
Facultad de Ciencias Naturales, Exactas y de la Educación
Departamento de Matemáticas
Doctorado en Ciencias Matemáticas
Popayán
Junio de 2023

Un método cuasi-Newton para ecuaciones polinomiales matriciales

Eduard Mauricio Macías Caicedo

Tesis presentada como requisito parcial para optar al título de:
Doctor en Ciencias Matemáticas

Directora
Dra. Rosana Pérez Mera
Profesor de la Universidad del Cauca

Codirector
PhD. Héctor Jairo Martínez Romero
Profesor de la Universidad del Valle

Universidad del Cauca
Facultad de Ciencias Naturales, Exactas y de la Educación
Departamento de Matemáticas
Doctorado en Ciencias Matemáticas
Popayán
Junio de 2023

Dedico esta Tesis principalmente a Dios, por darme la fuerza necesaria para culminar esta meta.

A mis Padres y mi otra familia, por su amor, sus oraciones y motivación para seguir adelante.

Especialmente, a mi amada esposa Juliana y a mi adorado hijo Gonzalito a quienes amo profundamente, por su paciencia, comprensión, amor y por no soltar mi mano en todo este camino.

Juliana: me llenas por dentro para conseguir el equilibrio que me permite dar el máximo de mí, te amo!.

Agradecimientos

A mi Directora de tesis, Dra. Rosana Pérez. Sin usted y sus virtudes, su paciencia y constancia este trabajo no lo hubiese logrado. Sus consejos fueron siempre útiles cuando no salían de mi pensamiento las ideas para escribir lo que hoy he logrado. Usted formó parte importante de esta historia con sus aportes profesionales que la caracterizan. Muchas gracias por sus múltiples palabras de aliento, cuando más las necesité; por estar allí cuando mis horas de trabajo se hacían confusas. Gracias por sus orientaciones.

A mi Codirector, Dr. Héctor Jairo Martínez por sus valiosos aportes y apreciables asesorías que han aportado grandes beneficios para la realización de esta investigación.

Al Mg. Favián Arenas, mi compañero y amigo del grupo de Optimización, por su tiempo, por su valiosa asesoría en aspectos computacionales de este trabajo.

A mis compañeros y amigos del grupo de Optimización de la Universidad del Cauca por su gran apoyo.

Por último, pero no menos importante, agradezco a mi amigo, compadre y compañero de estudio Hevert Vivas. Para mí, el mejor compañero que se puede tener. El ambiente de trabajo creado es simplemente perfecto, y su visión, motivación y optimismo me han ayudado en momentos críticos del Doctorado.

Resumen

En este trabajo de investigación consideramos el problema de resolver una ecuación polinómica matricial, de gran interés por sus numerosas aplicaciones en la ciencia e ingeniería.

En primer lugar, obtenemos la forma explícita del polinomio usado en la globalización de métodos tipo *Newton* que resuelven ese tipo de ecuaciones, el cual sólo se conocía para el caso cuadrático, deducimos una condición necesaria y suficiente para minimizar dicho polinomio en el intervalo $[0, 2]$ y analizamos numéricamente el desempeño del polinomio explícito en un algoritmo *Newton* globalizado.

Por otro lado, proponemos un algoritmo *cuasi-Newton local* para resolver una ecuación polinómica matricial, el cual reduce el costo computacional del método de *Newton* tradicionalmente utilizado para resolver este tipo de ecuaciones. Demostramos que el nuevo algoritmo es local e incluso, cuadráticamente convergente y analizamos su desempeño numérico.

Teniendo en cuenta las ventajas de disponer de un algoritmo global introducimos una estrategia de *búsqueda lineal exacta* en el algoritmo *cuasi-Newton* propuesto y, basados en la función de mérito, proponemos una aproximación de esta y dos algoritmos cuasi-Newton globales para resolver ecuaciones polinómicas matriciales. Para cada algoritmo, demostramos que la estrategia de globalización usada no afecta la convergencia del método *cuasi-Newton* local. Además, analizamos numéricamente el desempeño de los dos algoritmos globales y la ventaja de introducir una búsqueda lineal exacta en el algoritmo local.

Palabras clave: Ecuaciones polinómicas matriciales, método de Newton, búsqueda lineal exacta, polinomio explícito, algoritmo cuasi-Newton, convergencia local, algoritmo cuasi-Newton global.

Abstract

In this research, we consider the problem of solving a matrix polynomial equation of great interest due to its numerous applications in science and engineering.

First, we obtain the explicit form of polynomial used in the globalization of Newton-type methods that solves this equations, which was only known for the quadratic case, we deduce a necessary and sufficient condition to minimize that polynomial in the interval $[0, 2]$ and we analyze numerically the performance of the explicit polynomial in a globalized Newton-type algorithm.

On the other hand, we propose a local quasi-Newton algorithm to solve a matrix polynomial equation, which reduces the computational cost of the Newton method traditionally used to solve this type of equations. We show that this algorithm is locally and even quadratically convergent and we analyze its numerical performance.

Taking into account the advantages of having a global algorithm, we introduce a strategy of *exact line search* in the proposed quasi-Newton algorithm and based on the merit function, we propose an approximation of this and two global quasi-Newton algorithms for solve matrix polynomial equations. For each algorithm, we show that the globalization strategy used does not affect the convergence of the local quasi-Newton method. Furthermore, we numerically analyze the performance of the two global algorithms and the advantage of introducing an exact line search in the local algorithm.

Keywords: Matrix polynomial equations, Newton's method, exact line search, explicit polynomial, quasi-Newton algorithm, local convergence, global quasi-Newton algorithm.

Productos de la investigación

Artículos

[33] *An explicit polynomial to globalize algorithms for solving matrix polynomial equations*, Journal of Computational and Applied Mathematics, **420** (2023), https://doi.org/ 10.1016/j.cam.2022.114806. Macías, E. M., Pérez, R., Martínez, H. J.

[34] *On the local convergence of a quasi-newton method for solving matrix polynomial equations*. Applied Mathematics and Computation, **441** (2023), https://doi.org/10.1016/j. amc.2022.127678. Macías, E. M., Pérez, R., Martínez, H. J.

[32] *Two global quasi-Newton algorithms for solving matrix polynomial equations*. Sometido a evaluación. Macías, E. M., Pérez, R., Martínez, H. J.

Ponencias

– *Un polinomio explícito para globalizar algoritmos que resuelven ecuaciones polinómicas matriciales*. Presentado en MAPI 2, Medellín Colombia, Junio 8–10, 2022.

– *Forma explícita de una función de mérito para globalizar el método de Newton que resuelve ecuaciones polinómicas matriciales*. Presentado en XV-International Fast Workshop on Applied and Computational Mathematics, Trujillo Perú, Enero 6–7, 2022.

Índice general

Índice de tablas

Capítulo 1

Introducción

Una *ecuación polinómica matricial* es de la forma:

$$P(X) = A_m X^m + A_{m-1} X^{m-1} + \cdots + A_0 = 0 \in \mathbb{C}^{n\times n}, \tag{1.1}$$

donde $m, n \in \mathbb{N}$ y $A_m, \ldots, A_0, X \in \mathbb{C}^{n\times n}$.

Entre las numerosas aplicaciones en las que surge la ecuación (1.1) están problemas de análisis dinámico de mecánica estructural y sistemas acústicos [4, 48], simulación de circuitos eléctricos [28], mecánica de fluidos [43], procesamiento de señales [10], problemas de vibración y modelamiento de sistemas mecánicos microelectrónicos [25]. La ecuaciones polinómicas matriciales también surgen en teoría de ecuaciones diferenciales, teoría de sistemas y teoría estocástica, entre otras [1, 2, 7, 25, 26, 27].

Un caso particular de (1.1), de gran interés debido a las variadas aplicaciones en las que aparece, se presenta cuando $m = 2$ y se conoce como *ecuación cuadrática matricial* [11, 20, 21, 22, 31]. Su desarrollo teórico fue motivado en gran parte por el estudio del llamado *problema de los valores propios cuadrático* [19, 21, 31, 52] que en su versión más general, se conoce como el *problema de los valores propios polinomial* [3, 36, 51] y consiste en encontrar números complejos λ y vectores no nulos $\boldsymbol{x} \in \mathbb{C}^n$, tales que

$$P(\lambda)\boldsymbol{x} = \left(\lambda^m A_m + \lambda^{m-1} A_{m-1} + \cdots + A_0\right)\boldsymbol{x} = 0.$$

En este contexto la matriz $P(\lambda)$ se conoce como *matriz-λ* y su importancia teórica y práctica ha sido ampliamente documentada [25, 26, 27].

Respecto a la solución numérica de (1.1), se destaca el método de *Newton* [17, 24,

45], cuya iteración básica, a partir de una matriz inicial X_0, está definida por

$$\begin{aligned} L_{X_k}(S_k) &= -P(X_k), \\ X_{k+1} &= X_k + S_k, \end{aligned} \tag{1.2}$$

donde $L_{X_k}(S_k)$ es la derivada *Fréchet* de P en X_k, en la dirección de S_k [19, 22].

Para que la iteración (1.2) esté bien definida es necesario que la derivada *Fréchet* $L_{X_k}(S_k)$ sea no singular [24]. En el caso particular $m = 2$, la ecuación (1.2) es un caso especial de la ecuación de *Sylvester* generalizada [9] y la estrategia usada para resolverla utiliza la descomposición de *Schur* generalizada [16, 53], la cual tiene un alto costo computacional [21, 22] que conlleva a un alto costo computacional del método de Newton.

Por otra parte, el método de Newton por ser un algoritmo local es ventajoso si la matriz inicial está cerca de la solución, pero pierde su efectividad en otro caso. Una alternativa es introducir una estrategia de globalización [40]. Esto fue hecho para el caso cuadrático ($m = 2$) en [21] mediante una *búsqueda lineal exacta*, la cual no interfiere con la tasa de convergencia del algoritmo de Newton.

En [45], los autores extienden la estrategia de búsqueda lineal exacta al caso polinomial general para determinar el tamaño de paso. Para ello, siguiendo la filosofía de esta estrategia en el caso n dimensional [14], resuelven un problema de minimización cuya función objetivo es una *función de mérito*, en términos de la norma de *Frobenius*, la cual es un polinomio de grado $2m$. Estos autores no presentan la forma explícita de este polinomio, solo mencionan que su minimizador se encuentra en un intervalo apropiado de la forma $[0, k]$, para $k > 1$, sin dar el valor de k. Recomiendan no usar un valor muy grande porque se puede afectar la convergencia del método de Newton.

En [46] los autores manifiestan la dificultad de calcular el polinomio, mencionado en el párrafo anterior, vía su expresión en términos de una norma (es decir sin tenerlo explícitamente) y aún más, minimizarlo a medida que el grado del polinomio aumenta. Como alternativa, proponen otra forma para determinar el tamaño del paso y un algoritmo de Newton "relajado" para resolver un caso particular de una ecuación polinomial matricial que surge con frecuencia en modelos estocásticos.

En el caso de la ecuación cuadrática matricial ($m = 2$), se conoce una forma explícita del polinomio [21] que facilita su minimización y el cálculo del valor de k. Este caso particular, nos condujo al primer problema de investigación: *encontrar la forma explícita del polinomio para el caso polinomial general* ($m > 2$), cuya respuesta presentamos en este documento. En el desarrollo de este problema deducimos no

solo la forma explícita del polinomio, sino también la de su derivada y, una condición necesaria y suficiente para garantizar que el intervalo de minimización es $[0, 2]$. Además, analizamos numéricamente, el desempeño del polinomio explícito en un algoritmo Newton globalizado, comparándolo con los algoritmos de Newton con función de mérito implícita [45] y Newton relajado [46]. Los resultados obtenidos dieron lugar a un primer artículo de investigación [33].

En [31], los autores proponen un algoritmo *cuasi-Newton* para resolver la ecuación cuadrática matricial que reduce el costo computacional del método de Newton [20, 21], tradicionalmente usado para resolver dicha ecuación. Además demuestran que este algoritmo es local y hasta cuadráticamente convergente. Motivados por las bondades de este algoritmo, proponemos uno del mismo tipo para el caso polinomial general, desarrollamos su teoría de convergencia local, demostramos que él es hasta cuadráticamente convergente y analizamos su desempeño numérico, con lo cual damos respuesta a los segundo y cuarto problemas de investigación: *proponer un algoritmo cuasi-Newton para resolver la ecuación polinómica matricial y desarrollar su teoría de convergencia local. Además, hacer un análisis numérico del desempeño del algoritmo cuasi-Newton local*. Los resultados obtenidos dieron lugar a un segundo artículo de investigación [34].

Teniendo en cuenta el buen desempeño del nuevo algoritmo cuasi-Newton local para resolver ecuaciones polinomiales matriciales y con el fin de potenciar sus buenas propiedades de convergencia, introducimos una búsqueda lineal exacta (análoga a la del método de Newton), presentamos una aproximación polinomial a la función de mérito y deducimos una condición suficiente para su intervalo de minimización. Proponemos dos algoritmos cuasi-Newton globales, el primero, usando la función de mérito exacta y el segundo, su aproximación. Para cada algoritmo demostramos que la búsqueda lineal exacta no afecta la convergencia del método cuasi-Newton y presentamos experimentos numéricos de las dos propuestas comparándolas con el método de Newton. Esto permite dar respuesta al tercer y cuarto problemas de investigación: *proponer un algoritmo cuasi-Newton global para resolver ecuaciones polinómicas matriciales usando la estrategia de búsqueda lineal exacta y analizar, si dicha estrategia afecta o no la tasa de convergencia del algoritmo cuasi-Newton local. Hacer un análisis numérico del desempeño del algoritmo cuasi-Newton global.* Los resultados obtenidos dieron lugar a un tercer artículo de investigación [32].

Este documento lo organizamos en seis capítulos, tres de los cuales contienen los resultados centrales de la investigación cuya presentación la hemos estructurado de la siguiente forma: una breve introducción, propuestas algorítmicas, resultados de convergencia y experimentación numérica, cerrando el capítulo con algunos comentarios

finales. La organización del documento es la siguiente:

En el Capítulo **2**, presentamos algunos conceptos y definiciones a manera de preliminares para una mejor comprensión de la investigación.

En el Capítulo **3**, presentamos los problemas de prueba que usamos en la experimentación numérica, algunos de los cuales surgen en aplicaciones de ingeniería y otras ciencias. Además, presentamos el criterio de parada y las matrices iniciales que usamos en los algoritmos a lo largo del documento.

En el Capítulo **4**, presentamos la deducción de la forma explícita de polinomio que define la función de mérito a minimizar cuando se introduce una estrategia de búsqueda lineal exacta en el método de Newton. Este capítulo también contiene la deducción de la derivada del polinomio y de una condición necesaria y suficiente para garantizar que el intervalo de minimización es $[0, 2]$. Además, analizamos numéricamente, el desempeño del polinomio explícito en un algoritmo Newton globalizado, comparándolo con los algoritmos de Newton con función de mérito implícita [45] y Newton relajado [46].

En el Capítulo **5**, proponemos un algoritmo cuasi-Newton local para resolver ecuaciones polinomiales matriciales que generaliza el algoritmo propuesto en [30], desarrollamos su teoría de convergencia local, demostramos que él es hasta cuadráticamente convergente y analizamos su desempeño numérico.

En el Capítulo **6**, proponemos dos algoritmos cuasi-Newton globales, uno usando la función de mérito exacta y el otro una aproximación polinomial a ella, demostramos para cada propuesta que la búsqueda lineal exacta no afecta la convergencia del método cuasi-Newton y presentamos experimentos numéricos de las dos propuestas comparándolas con el método de Newton.

En el Capítulo **7**, presentamos las conclusiones finales y posibles trabajos futuros.

Además, incluimos un Apéndice donde presentamos pruebas numéricas adicionales.

Finalmente, presentamos la bibliografía que usamos en el desarrollo de este trabajo de investigación.

Capítulo 2

Preliminares

En este capítulo incluimos a manera de preliminares algunos conceptos y definiciones que contribuyen a una mejor comprensión del documento investigación. Al finalizar el capítulo introducimos el método de *Newton* para encontrar una solución numérica de la ecuación polinómica matricial, dado que es el método tradicionalmente usado para resolver este tipo de ecuaciones y el cual, estará presente en la parte teórica y numérica de la investigación.

Denotaremos como $\mathbb{C}^{n\times n}$ el conjunto de matrices de orden n con componentes complejas, X_* una solución (solvente) de la ecuación polinomial matricial (1.1), $\|\cdot\|$ una norma de matricial inducida y $\|\cdot\|_F$ la norma de *Frobenius* [16]. Usaremos la igualdad $\|A\|_F^2 = \mathrm{tr}(A^H A)$, para $A \in \mathbb{C}^{n\times n}$, donde $\mathrm{tr}(A)$ denota la *traza* de la matriz A [16]. Denotaremos $N(X, r)$ como la vecindad centrada en X y radio r; $\mathbf{1_{n\times n}}$ y $\mathbf{1}_n$, para una matriz de orden n y un vector columna de orden n con todos sus elementos iguales a 1, respectivamente, I_n la matriz identidad de tamaño $n \times n$.

Por otro lado, usaremos el símbolo *Landau* $O(\cdot)$. Sean $\{\alpha_k\}$ y $\{\beta_k\}$ dos sucesiones en $\mathbb{R}$, con $\alpha_k > 0$, $\beta_k > 0$ para todo $k \in \mathbb{N}$ y $\lim_{x\to\infty} \beta_k = 0$ entonces $\alpha_k = O(\beta_k)$ significa que $\limsup_{x\to\infty} \alpha_k/\beta_k < +\infty$, es decir, existe una constante $c > 0$ tal que $\alpha_k \leq c\,\beta_k$ para todo $k \in \mathbb{N}$.

Definición 2.1. [19] *Sea* $F\colon \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ *una función matricial. La derivada Fréchet de* F *en* $X \in \mathbb{C}^{n\times n}$ *es una aplicación lineal*

$$\begin{aligned} L_X : \ \mathbb{C}^{n\times n} &\longrightarrow \mathbb{C}^{n\times n} \\ S &\longmapsto L_X(S), \end{aligned}$$

tal que para todo $S \in \mathbb{C}^{n\times n}$, $F(X+S) = F(X) + L_X(S) + R(S)$, *con*

$$\lim_{\|S\|\to 0} \frac{\|R(S)\|}{\|S\|} = 0.$$

A continuación presentamos dos resultados de álgebra lineal numérica conocidos como la *descomposición de Schur y descomposición de Schur generalizada* el cual garantiza que cualquier matriz de orden n es unitariamente semejante a una matriz triangular superior cuyos elementos en la diagonal principal son los valores propios de la matriz.

Teorema 2.1. (*Descomposición de Schur* [16]). *Si* $A \in \mathbb{C}^{n\times n}$ *entonces existe una matriz unitaria* $U \in \mathbb{C}^{n\times n}$, *tal que*

$$U^H A U = T = D + N, \tag{2.1}$$

donde $D = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$ *y* $N \in \mathbb{C}^{n\times n}$ *es estrictamente triangular superior.*

Teorema 2.2. (*Descomposición de Schur generalizada* [16].) *Sean* A *y* $B \in \mathbb{C}^{n\times n}$ *entonces existen matrices unitarias* W *y* $Z \in \mathbb{C}^{n\times n}$, *tales que*

$$W^H A Z = T \;\; y \;\; W^H B Z = S,$$

son matrices triangulares superiores.

A continuación presentamos un producto matricial conocido como *producto Kronecker* que surge de manera natural en teoría de grupos y tiene importantes aplicaciones en física de partículas.

Definición 2.2. (*El producto Kronecker* [25]). *Sean* $A = [a_{ij}] \in \mathbb{C}^{m\times n}$ *y* $B = [b_{ij}] \in \mathbb{C}^{p\times q}$, *el producto Kronecker entre* A *y* B, *denotado por* $A \otimes B$, *está definido por:*

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix} \in \mathbb{C}^{mp\times nq}.$$

En la siguiente definición, presentamos un operador que permite expresar una matriz como un vector.

Definición 2.3. (*Operador vec* [25]). *Sea* $A = [A_{*1} \cdots A_{*n}] \in \mathbb{C}^{m\times n}$, *donde* $A_{*j} \in \mathbb{C}^m$, *para* $j = 1, \ldots, n$. *El operador* $vec : \mathbb{C}^{m\times n} \to \mathbb{C}^{mn}$ *está definido por:*

$$vec(A) = \begin{bmatrix} A_{*1} \\ A_{*2} \\ \vdots \\ A_{*n} \end{bmatrix} \in \mathbb{C}^{mn}.$$

El siguiente resultado da una estrecha relación entre el operador *vec* y el producto Kronecker.

Teorema 2.3. [25] *Si* $A \in \mathbb{C}^{m\times m}$, $X \in \mathbb{C}^{m\times n}$ *y* $B \in \mathbb{C}^{n\times n}$ *entonces*

$$vec(AXB) = (B^T \otimes A)vec(X). \tag{2.2}$$

A continuación presentaremos un resultado de álgebra lineal numérica que da una condición suficiente para la no singularidad de una matriz y una cota para su inversa.

Lema 2.1. (*Lema de Banach* [14]). *Sean* $\|\cdot\|$ *una norma matricial inducida en* $\mathbb{C}^{n\times n}$ *y* $A, B \in \mathbb{C}^{n\times n}$. *Si* A *es no singular y* $\|I_n - A^{-1}B\| < 1$ *entonces* B *es no singular y*

$$\left\|B^{-1}\right\| \leq \frac{\|A^{-1}\|}{1 - \|I_n - A^{-1}B\|}.$$

Finalmente respecto a las solución numérica de ecuaciones polinomiales matriciales se destaca el método de *Newton* [17, 24, 45], ampliamente utilizado para resolver este tipo de ecuaciones. Una iteración básica del método de *Newton* [17, 24, 45] para resolver (1.1) es de la forma:

$$\begin{aligned} L_{X_k}(S_k) &= -P(X_k) \\ X_{k+1} &= X_k + S_k, \end{aligned} \tag{2.3}$$

donde

$$L_{X_k}(S_k) = \sum_{i=1}^{m} \left(\sum_{j=0}^{m-i} A_{m-j} X_k^{m-(i+j)} \right) S_k X_k^{i-1}, \tag{2.4}$$

denota la derivada *Fréchet* de P en la dirección de S_k.

Observemos que en cada iteración es necesario encontrar la solución S_k de la ecuación

$$\sum_{i=1}^{m}\left(\sum_{j=0}^{m-i} A_{m-j}X_k^{m-(i+j)}\right)S_k X^{i-1} = -P(X_k), \tag{2.5}$$

para ello, en [17, 45], los autores usan el operador vec (**Definición 2.3**), el producto Kronecker (**Definición 2.2**) y el **Teorema 2.3**, obteniendo la expresión:

$$vec(L_{X_k}(S_k)) = \left\{\sum_{i=1}^{m}\left[\left(X_k^T\right)^{i-1}\otimes\sum_{j=0}^{m-i} A_{m-j}X_k^{m-(i+j)}\right]\right\} vec\,(S_k) = vec(-P(X_k)). \tag{2.6}$$

Usando la notación:

$$\mathbf{L_k} \coloneqq \sum_{i=1}^{m}\left[\left(X_k^T\right)^{i-1}\otimes\sum_{j=0}^{m-i} A_{m-j}X_k^{m-(i+j)}\right],$$

la segunda igualdad de (2.6), se puede expresar como el siguiente sistema de n^2 ecuaciones lineales con n^2 incógnitas (las componentes de la matriz S):

$$\mathbf{L_k}\, vec\,(S_k) = vec(-P(X_k)). \tag{2.7}$$

En [24], los autores demuestran que la no singularidad de la matriz $\mathbf{L_k}$ es una condición necesaria y suficiente para la unicidad en la solución de (2.5) (es decir, el operador L_{X_k} es *regular*) y por tanto, $\text{ínf}_{\|S_k\|=1}\ \|L_{X_k}(S_k)\| = \text{mín}_{\|S_k\|=1}\ \|\mathbf{L_k}\, vec(S_k)\| > 0$.

En la práctica, con el fin de obtener una matriz $\mathbf{L_k}$ que sea *triangular inferior* y con ello, obtener n sistemas de tamaño $n \times n$ se procede de la siguiente forma [45].

Dada $X_k \in \mathbb{C}^{n\times n}$ se encuentra su descomposición de Schur (**Teorema 2.1**),

$$U^H X_k\, U = R, \tag{2.8}$$

donde $U \in \mathbb{C}^{n\times n}$ es una matriz unitaria y $R \in \mathbb{C}^{n\times n}$ triangular superior. Sustituyendo (2.8) en (2.5) y realizando algunas operaciones algebraicas, se obtiene la siguiente ecuación

$$L_{X_k}(S'_k) = B_1 S'_k + B_2 S'_k R + \cdots + B_m S'_k R^{m-1} = C, \tag{2.9}$$

donde $B_i = \sum_{j=0}^{m-i} A_j X_k^{m-j+i}, i = 1, \ldots m$, $S'_k = S_k\, U$ y $C = -P(X_k)\, U$. Aplicando la función vec (**Definición 2.3**) en (2.9), se obtiene

$$vec\,(L_{X_k}(S'_k)) = vec(C) \Leftrightarrow \mathbf{L_k}\, vec(S'_k) = vec(C), \tag{2.10}$$

donde $\mathbf{L_k} \in \mathbb{C}^{n^2 \times n^2}$ es la matriz:

$$\begin{aligned}\mathbf{L_k} &= I \otimes B_1 + R^T \otimes B_2 + (R^T)^2 \otimes B_3 + \cdots (R^T)^{m-1} \otimes B_m \\ &= \sum_{i=1}^{m} (R^T)^{i-1} \otimes B_i = \begin{bmatrix} \mathbf{L}_{1\,1} & & & \\ \mathbf{L}_{2\,1} & \mathbf{L}_{2\,2} & & \\ \vdots & & \ddots & \\ \mathbf{L}_{n\,1} & \cdots & \cdots & \mathbf{L}_{n\,n} \end{bmatrix}\end{aligned}$$

con, $\mathbf{L}_{i\,j} = \sum_{i=1}^{m} [R^{k-1}]_{j\,i} B_k$, una matriz de tamaño $n \times n$. Así, la ecuación (2.10) se puede expresar explícitamente por:

$$\begin{bmatrix} \mathbf{L}_{1\,1} & & & \\ \mathbf{L}_{2\,1} & \mathbf{L}_{2\,2} & & \\ \vdots & & \ddots & \\ \mathbf{L}_{n\,1} & \cdots & \cdots & \mathbf{L}_{n\,n} \end{bmatrix}_{n^2 \times n^2} \begin{bmatrix} \mathbf{s}'_1 \\ \mathbf{s}'_2 \\ \vdots \\ \mathbf{s}'_n \end{bmatrix}_{n^2 \times 1} = \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \\ \vdots \\ \mathbf{c}_n \end{bmatrix}_{n^2 \times 1},$$

equivalentemente, se expresa como n sistemas de ecuaciones lineales, con n incógnitas. En efecto,

$$\begin{aligned} \mathbf{L}_{1\,1}\,\mathbf{s}'_1 &= \mathbf{c}_1 \\ \mathbf{L}_{2\,1}\,\mathbf{s}'_1 + \mathbf{L}_{2\,2}\,\mathbf{s}'_2 &= \mathbf{c}_2 \\ &\vdots \\ \mathbf{L}_{n\,1}\,\mathbf{s}'_1 + \mathbf{L}_{n\,2}\,\mathbf{s}'_2 + \cdots + \mathbf{L}_{n\,n}\,\mathbf{s}'_n &= \mathbf{c}_n. \end{aligned} \tag{2.11}$$

Capítulo 3

Problemas de prueba

En este capítulo presentamos quince problemas de prueba que usamos en la experimentación numérica, algunos de los cuales surgen en aplicaciones de ingeniería y otras ciencias. En cada problema se debe encontrar un solvente de una ecuación polinomial matricial. En la parte final del capítulo presentamos el criterio de parada y las matrices iniciales para los algoritmos que proponemos y presentamos en los Capítulos 4, 5 y 6, respectivamente.

Problema 1 (*Cadenas de Markov*) [46]. Consideramos una ecuación que surge en problemas estocásticos:

$$A_2X^2 + (A_1 - I_m)X + A_0 = 0,$$

donde $A_0 = W + \delta\, I_n$ y $A_1 = A_2 = W$, con W una matriz cuyas componentes de la diagonal principal son ceros y las restantes son constantes no negativas que satisface $(3W + \delta\, I_n)\mathbf{1}_n = \mathbf{1}_n$, con $0 < \delta < 1$. Es importante observar que cuando δ tiende a cero, el problema se vuelve muy inestable [18, 46]. El tamaño del problema es $n = 16$.

Problema 2 (*Problema estocástico*) [47]. Consideremos una ecuación polinomial de grado $m = 6$, con los siguientes tamaños para las matrices de coeficientes: $n = 5$, $n = 50$ y $n = 100$,

$$\begin{cases} A_k = a_kW, & k = 0, 2, 3, 4, 5, \\ A_1 = a_1W - I_n, \\ A_6 = W, \end{cases}$$

donde

$$W = \frac{1}{6200\,(n-1)}\left(\mathbf{1_{n\times n}} - I_n\right) \quad \text{y} \quad \begin{cases} a_0 = 34096, & a_1 = 56, \\ a_2 = 384, & a_3 = 1312, \\ a_4 = 321, & a_5 = 30. \end{cases}$$

Problema 3 (*Sistemas de vibración*) [24, 45]. Consideremos la ecuación polinómica de grado $m = 4$ dada por:

$$X^4 + \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} X^2 + \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ -1 & 0 & 1 \end{bmatrix} X + \begin{bmatrix} -20 & 2 & 1 \\ 2 & -20 & 0 \\ 1 & 0 & -20 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

la cual tiene 4 solventes:

$$S_1 = \begin{bmatrix} 2.056 & -0.113 & -0.149 \\ -0.057 & 2.054 & -0.063 \\ -0.088 & 0.003 & 2.062 \end{bmatrix}, \quad S_2 = \begin{bmatrix} -2.170 & -0.004 & -0.142 \\ -0.048 & 2.170 & -0.049 \\ 0.197 & -2e-05 & -2.169 \end{bmatrix}$$

$$S_3 = \begin{bmatrix} 0.797 & 3.736 & -2.170 \\ 0.067 & 1.673 & 0.137 \\ -1.616 & 4.474 & -0.390 \end{bmatrix}, \quad S_4 = \begin{bmatrix} 0.545 & 0.065 & -2.150 \\ -0.197 & -2.176 & 0.157 \\ -1.941 & -0.055 & -0.365 \end{bmatrix}.$$

Esta ecuación polinómica es el polinomio característico de la ecuación diferencial matricial $Y^{(4)} + A_2Y^{(2)} + A_1Y' + A_0Y = 0$. Tales ecuaciones ocurren en conexión con sistemas de vibración.

Problema 4 (*Ecuación matricial cúbica*) [24, 45]. Sea

$$X^3 + \begin{bmatrix} -6 & 6 \\ -3 & -15 \end{bmatrix} X^2 + \begin{bmatrix} 2 & -42 \\ 21 & 65 \end{bmatrix} X + \begin{bmatrix} 18 & 66 \\ -33 & -81 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix},$$

una ecuación polinomial de grado 3 la cual tiene dos solventes:

$$S_1 = \begin{bmatrix} 4 & -2 \\ 1 & 7 \end{bmatrix} \quad \text{y} \quad S_2 = \begin{bmatrix} 0 & -2 \\ 1 & 3 \end{bmatrix}.$$

Problema 5 (*Ecuación polinómica matricial cúbica*) [13]. Sea

$$X^3 + A_2X^2 + A_1X + A_0 = 0,$$

una ecuación polinómica matricial de grado $m = 3$, donde

$$A_2 = \begin{bmatrix} -11.79104478 & 0.82089552 \\ 1.91044776 & -9.20895522 \end{bmatrix}, \quad A_1 = \begin{bmatrix} 42.34328358 & -10.16417910 \\ -13.43283582 & 25.64179104 \end{bmatrix},$$

$$A_0 = \begin{bmatrix} -50.35820896 & 21.88059701 \\ 19.58208955 & -22.80597015 \end{bmatrix}.$$

Problema 6 (*Ecuación polinómica matricial cúbica*) [41]. Consideremos la ecuación polinómica matricial de grado $m = 3$, dada por:

$$X^3 + A_2X^2 + A_1X + A_0 = 0,$$

donde

$$A_2 = \begin{bmatrix} -12 & 6 \\ -3 & -21 \end{bmatrix}, \quad A_1 = \begin{bmatrix} 38 & -66 \\ 33 & 137 \end{bmatrix}, \quad A_0 = \begin{bmatrix} -18 & 174 \\ -87 & -279 \end{bmatrix}.$$

Problema 7 (*Problema del robot*) [36, 44]. Consideremos un modelo de un robot con motor eléctrico en las juntas dado por el sistema de ecuaciones diferenciales:

$$M(q)q^{(2)} + h(q, q^{(2)}) + K(q - p) = 0, \tag{3.1}$$

$$J\,p^{(2)} + D\,p' - V(q - p) = 0, \tag{3.2}$$

donde la ecuación (3.1) suministra el modelo del robot y la ecuación (3.2) el mecanismo del motor. La matrices K y V son simétricas y definidas positivas, J es una matriz diagonal definida positiva y D es una matriz diagonal semidefinida positiva.

Mediante linealización $\left(h(q, q^{(2)}) = G\,q' + C\,q\right)$ y simplificación $(M(q) = A_4)$ en las ecuaciones del robot (3.1) y (3.2), se obtiene una ecuación de la dinámica del robot dada por:

$$A_4q^{(2)} + G\,q' + (C + K)q - Kp = 0,$$

donde las matrices A_4 es simétrica y definida positiva, G es antisimétrica y C es simétrica. Resolviendo esta ecuación para p y reemplazando en la ecuación (3.2) obtenemos el sistema de ecuaciones polinomial

$$A_4q^{(4)} + A_3q^{(3)} + A_2q^{(2)} + A_1q' + A_0q = 0, \tag{3.3}$$

con,

$$\begin{aligned}
A_3 &= G + KJ^{-1}DK^{-1}A_4,\\
A_2 &= C + K + KJ^{-1}DK^{-1}G + KJ^{-1}VK^{-1}A_4,\\
A_1 &= KJ^{-1}(DK^{-1}C + D + VK^{-1}G),\\
A_0 &= KJ^{-1}VK^{-1}C.
\end{aligned}$$

La ecuación polinomial $A_4X^4 + A_3X_3 + A_2X^2 + A_1X + A_0 = 0$, es el polinomio característico de la ecuación diferencial matricial (3.3).

Problema 8 (*Ecuación diferencial en sistemas de vibración*) [38]. Consideremos la siguiente ecuación matricial diferencial:

$$y^{(3)} + A_2y^{(2)} + A_1y' + A_0y = 0,$$

que surge en conexión con un sistema vibratorio y cuyo polinomio característico asociado es $X^3 + A_2X^2 + A_1X + A_0 = 0$, donde

$$A_2 = \begin{bmatrix} 2.660 & 2.450 & 2.100\\ 0.230 & 1.040 & 0.223\\ 0.600 & 0.756 & 0.658 \end{bmatrix}, \quad A_1 = \begin{bmatrix} -20 & 5 & 0\\ 5 & -20 & 5\\ 0 & 5 & -20 \end{bmatrix}, \quad A_0 = \begin{bmatrix} 1.600 & 1.280 & 2.890\\ 1.280 & 0.840 & 0.413\\ 2.890 & 0.413 & 0.725 \end{bmatrix}.$$

Problema 9 (*Ecuación polinómica matricial de grado* 5) [42]. Consideremos la ecuación polinómica matricial de grado $m = 5$, dada por:

$$X^5 + A_4X^4 + A_3X^3 + A_2X^2 + A_1X + A_0 = 0,$$

donde

$$A_4 = \begin{bmatrix} -20 & 10\\ -5 & -35 \end{bmatrix}, \quad A_3 = \begin{bmatrix} 1200 & -220\\ 110 & 450 \end{bmatrix}, \quad A_2 = \begin{bmatrix} -100 & 1700\\ -850 & -2650 \end{bmatrix},$$

$$A_1 = \begin{bmatrix} -1006 & -5390\\ 2695 & 7079 \end{bmatrix}, \quad A_0 = \begin{bmatrix} 1950 & 5790\\ -2895 & -6735 \end{bmatrix}.$$

Problema 10 (*Problema guía de ondas planas*) [5, 49, 50]. Consideremos una ecuación polinomial de grado $m = 4$ dada por:

$$A_4X^4 + A_3X^3 + A_2X^2 + A_1X + A_0 = 0,$$

la cual surge de una solución en elementos finitos de la ecuación para los modos de una guía de ondas plana utilizando funciones de base lineal por partes $\varphi_i, i = 1, \ldots, 10$. Las matrices de coeficientes se definen por

$$A_1 = \frac{\delta^2}{4}\,\mathrm{diag}(-1, 0, \ldots, 0, 1), \qquad A_3 = \mathrm{diag}(1, 0, \ldots, 0, 1),$$

$$A_0(i,j) = \frac{\delta^4}{16}, \qquad A_2(i,j) = (\varphi_i', \varphi_j') - (q\varphi_i - \varphi_j), \qquad A_4(i,j) = (\varphi_i, \varphi_j).$$

Las matrices A_1 y A_3 son diagonales, mientras que A_0, A_2 y A_4 son tridiagonales. El parámetro δ describe la diferencia en el índice de refracción entre la cubierta y el sustrato de la guía de onda; q es una función utilizada en la derivación de la formulación variacional y es constante en cada capa.

Problema 11 (*Sistema de colas*) [6, 37]. Consideremos la ecuación cuadrática matricial,

$$A_2X^2 + A_1X^1 + A_0 = 0,$$

donde las matrices coeficientes son de orden $n = 32$ y todas sus entradas son iguales a $\alpha = (\rho - 1)/(3(n-1))$ para $i \neq j$, $(A_0)_{i,i} = -\rho$, $(A_1)_{i,i} = 1$, $(A_2)_{i,i} = 0$, para $i = 1, \ldots, n$. El parámetro $\rho = 0.99$.

Este problema representa un sistema de colas en un ambiente aleatorio, donde los períodos de desbordes severos se alternan con períodos de llegadas bajas.

Problema 12 (*Ecuación matricial cúbica*) [23]. Consideremos la ecuación polinomial de grado $m = 3$,

$$X^3 + X^2 + X + \begin{bmatrix} -6 & -5 \\ 0 & -6 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

Problema 13 (*Ecuación matricial cúbica*) [23]. Consideremos la ecuación polinomial de grado $m = 3$ dada por

$$X^3 + \begin{bmatrix} 0 & -1 \\ -1 & 1 \end{bmatrix} X^2 + X + \begin{bmatrix} -10 & -7 \\ 4 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

Problema 14 (*Polinomio matricial cúbico hiperbólico*) [39]. Consideremos la ecuación polinomial matricial *hiperbólica* de grado $m = 3$ dada por:

$$A_3X^3 + A_2X^2 + A_1X + A_0 = 0,$$

donde las matrices coeficientes son de orden $n = 10$ y están dadas de la siguiente forma $A_3 = I_n$ y las matrices A_2, A_1, A_0 son matrices esparsas, cada una con aproximadamente 20 elementos no nulos.

La ecuación polinomial matricial dada se dice que es *hiperbólica* si A_3, A_2, A_1 y A_0 son matrices definidas positivas.

Problema 15 (*Ecuación matricial de grado* 5). Consideremos la ecuación polinomial de grado $m = 5$,

$$X^5 + A_4X^4 + A_3X^3 + A_2X^2 + A_1X + A_0 = 0,$$

donde

$$A_4 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \end{bmatrix}, \quad A_3 = \begin{bmatrix} 0 & 0 & 0.01 \\ 0 & 0 & 0 \\ 100 & 0 & 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 & 1000 & 0 \\ 0 & 100 & 1 \\ -1000 & 7090 & 1 \end{bmatrix},$$

$$A_1 = \begin{bmatrix} -20 & 2000 & 1 \\ 2 & -20000 & 0 \\ 0.00061 & 0 & -20 \end{bmatrix}, \quad A_0 = A_4.$$

Finalmente, mencionamos que el criterio de parada que usaremos en los algoritmos es el siguiente:

$$Res(X_k) = \frac{\|P(X_k)\|_F}{\|A_m\|_F \|X_k\|_F^m + \|A_{m-1}\|_F \|X_k\|_F^{m-1} + \cdots + \|A_0\|_F} \leq Tol, \tag{3.4}$$

donde la expresión $Res(X_k)$ una medida relativa del error en $\|P(X_k)\|_F$ definida como en [45, 46].

Adicionalmente, como medidas de control en los algoritmos incluimos $\|P(X_k)\|_F$ y $\|X_{k+1} - X_k\|_F$, con la cual verificamos que en efecto, el algoritmo llega a un solvente del problema.

Para los algoritmos presentados los Capítulo 4 y 5 usamos las matrices iniciales: cero de tamaño nxn, $\alpha\, I_n$ y $10^p I_n$, donde $\alpha, p \in \mathbb{Z}$. En los algoritmos del Capítulo 6, además de las matrices anteriores, generamos matrices aleatorias usando el comando $rand(n)$ de MATLAB®, que permite obtener matrices aleatorias de orden n, cuyas componentes son números aleatorios normalmente distribuidos entre 0 y 1.

Escribimos los códigos de los algoritmos y de las funciones que definen los problemas en MATLAB® [35] y realizamos los experimentos numéricos en un computador Intel (R) Core (TM) i5-3450 de 2.8 GHz.

Capítulo 4

Un polinomio explícito

La solución de ecuaciones polinómicas matriciales mediante algoritmos globales de búsqueda direccional generalmente incluyen una búsqueda lineal exacta como estrategia de globalización que conduce a la minimización de una función de mérito dada en términos de la norma de Frobenius, la cual es un polinomio en una variable, del que no se conocía su forma explícita para el caso polinomial general.

Dada la importancia de conocer esta forma dentro de un proceso de minimización, en este capítulo, para el método de Newton deducimos dicha forma explícita del polinomio, lo cual a su vez, permite obtener la expresión explícita de su derivada. Adicionalmente, obtenemos una condición necesaria y suficiente que garantiza que el intervalo de minimización es $[0, 2]$. Complementamos los desarrollos teóricos con pruebas numéricas preliminares que permiten observar la ventaja de tener el polinomio explícito y el intervalo a minimizar.

4.1. Búsqueda lineal exacta

Para resolver problemas de optimización sin restricciones por métodos descendentes, es común usar la dirección calculada como dirección de búsqueda y tomar un paso en esa dirección que permita definir la siguiente iteración, lo que se conoce como búsqueda lineal haciendo referencia al procedimiento para escoger el "tamaño de paso". Cuando esto se hace minimizando la función de mérito a lo largo de esa direc-

ción, la búsqueda lineal se denomina *exacta* [14]. Una motivación para incorporar una estrategia de globalización como la búsqueda lineal, en el caso del método de Newton, es que, independientemente del problema a resolver, la "bondad" del paso de Newton depende de que la aproximación anterior esté lo suficientemente cerca a la solución. Se espera que con la búsqueda lineal exacta se obtenga un mejor desempeño algorítmico (es decir, podamos llegar a la solución desde aproximaciones iniciales arbitrarias).

En el caso de un campo vectorial $H\colon \mathbb{R}^n \to \mathbb{R}^n$, una estrategia para globalizar el método de Newton que resuelve el sistema de ecuaciones no lineales $H(\boldsymbol{x}) = 0$, consiste en combinarlo con una estrategia global para optimización irrestricta como la descrita en el párrafo anterior [40]. En efecto, independientemente de si el vector $\boldsymbol{x}_k$ está o no próximo a la solución $\boldsymbol{x}_*$ del sistema de ecuaciones no lineales que se quiere resolver, una forma razonable de aceptar la iteración siguiente $\boldsymbol{x}_{k+1}$ es exigir que $\|H(\boldsymbol{x}_{k+1})\|$ sea suficientemente menor que $\|H(\boldsymbol{x}_k)\|$ para alguna norma conveniente. Exigencia que también es hecha cuando se trata de minimizar $\|H(\boldsymbol{x})\|$, con lo cual se llega a un problema de minimización.

Teniendo en cuenta lo anterior, en el caso de la función matricial $P\colon \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$, dada por la ecuación (1.1), tenemos que, si la matriz inicial X_0 no está próxima de la solución X_* del sistema matricial $P(X) = 0$ y S_k es el paso de Newton entonces en el algoritmo global se busca una aproximación siguiente de la forma $X_{k+1} = X_k + tS_k$ para algún valor de t convenientemente elegido, de manera que X_{k+1} sea una aproximación suficientemente aceptable, lo cual significa que buscamos un valor de t tal que $\|P(X_{k+1})\|$ sea menor que $\|P(X_k)\|$ para alguna norma. Para ello, una forma de calcular el valor de t es minimizar $\|P(X_k + tS_k)\|$, para alguna norma conveniente, equivalentemente minimizar la función de mérito $p(t) = \|P(X_k + tS_k)\|^2$. En particular, si tomamos la norma de Frobenius, como en [21, 30] calculamos el valor de t mediante una búsqueda lineal exacta minimizando la función de merito,

$$p(t) = \|P(X_k + tS_k)\|_F^2 . \tag{4.1}$$

4.2. El polinomio $p(t)$.

En esta sección, presentamos en forma detallada y mejorada, algunos resultados de [21, 23, 31, 45] relacionados con la función de mérito p que nos permitieron expresar explícitamente $p(t)$ como un polinomio de grado $2m$ en la variable t.

En [21], los autores muestran cómo incorporar la búsqueda lineal exacta al método de Newton para resolver la ecuación cuadrática matricial $Q(X) = AX^2 + BX + C = 0$.

Expresan explícitamente (4.1) como un polinomio de grado cuatro. Además, muestran que el minimizador de $p(t)$ está en el intervalo $[0, 2]$. En [30], los autores presentan un algoritmo cuasi-Newton global para resolver la ecuación cuadrática matricial, el cual es la globalización del método cuasi-Newton propuesto en [31], mediante una búsqueda lineal exacta como la presentada en [21]. Demuestran que se conservan las buenas propiedades de convergencia del método cuasi-Newton y que el minimizador de $p(t)$ se encuentra en el intervalo $[0, 2]$.

En [45], para el caso general de la función polinómica matricial (1.1), los autores incorporan al método de Newton una búsqueda lineal exacta usando la función de mérito (4.1) y muestran que es un polinomio de grado $2m$ en la variable t, sin dar su forma explícita. Además, mencionan que el minimizador de este polinomio se encuentra en un intervalo $[0, k]$, para $k > 1$. En efecto, ellos consideran la igualdad

$$P(X + tS) = A_m(X + tS)^m + A_{m-1}(X + tS)^{m-1} + \cdots + A_1(X + tS) + A_0 I,$$

y la siguiente relación de equivalencia.

Definición 4.1. *Sean μ y υ dos permutaciones de $1, \ldots, n$ y m un entero fijo dado, con $1 \leq m \leq n$. Definimos una relación, $\sim_m$, de equivalencia entre las permutaciones. Así, si para cada $i = 1, \ldots, n$, se satisface alguna de las siguientes condiciones*

$$\begin{aligned} \mu(i) > m &\Leftrightarrow \upsilon(i) > m \\ \upsilon(i) \leq m &\Leftrightarrow \mu(i) \leq m, \end{aligned}$$

entonces $\mu \sim_m \upsilon$.

Denotamos por P el conjunto de todas las permutaciones de $1, \ldots, k$; P/m el conjunto de las clases de equivalencia para un entero m, $1 \leq m \leq k$, y $[\mu]_m$ cualquier clase de equivalencia que pertenezca al conjunto P/m.

Ejemplo 4.1. *Sea $k = 3$. Las permutaciones de $1, 2, 3$ son las siguientes:*

$$\begin{aligned} \rho_1 &= 1\,2\,3 \\ \rho_2 &= 1\,3\,2 \\ \rho_3 &= 3\,2\,1 \\ \rho_4 &= 2\,1\,3 \\ \rho_5 &= 2\,3\,1 \\ \rho_6 &= 3\,1\,2. \end{aligned}$$

Luego, $P = \{\rho_1, \rho_2, \rho_3, \rho_4, \rho_5, \rho_6\}$.

Para $m = 1$, tenemos $\binom{3}{1} = 3$ clases de equivalencia:

$$[\rho_1]_1 = \{\rho_1, \rho_2\},$$
$$[\rho_3]_1 = \{\rho_3, \rho_5\},$$
$$[\rho_4]_1 = \{\rho_4, \rho_6\}.$$

Luego, $P/1 = \{[\rho_1]_1, [\rho_3]_1, [\rho_4]_1\}$.

Para $m = 2$ tenemos $\binom{3}{2} = 3$ clases de equivalencia:

$$[\rho_1]_2 = \{\rho_1, \rho_4\},$$
$$[\rho_3]_2 = \{\rho_3, \rho_6\},$$
$$[\rho_2]_2 = \{\rho_2, \rho_5\}.$$

Luego, $P/2 = \{[\rho_1]_2, [\rho_3]_2, [\rho_2]_2\}$.

Para $m = 3$ tenemos $\binom{3}{3} = 1$ clase de equivalencia:

$$[\rho_1]_3 = \{\rho_1, \rho_2, \rho_3, \rho_4, \rho_5, \rho_6\}.$$

Luego, $P/3 = \{[\rho_1]_3\}$.

Teniendo en cuenta la relación de equivalencia $\sim_m$, se define a continuación la función Φ_{XS}, como la suma de los productos de todas las permutaciones de X y S [45].

Definición 4.2. *Sean $R_1 = R_2 = \cdots = R_m = X$ y $R_{m+1} = R_{m+2} = \cdots = R_k = S$, matrices de orden n. La función*

$$\Phi^k_{XS}\colon \{0, 1, \ldots, k\} \longrightarrow \mathbb{C}^{n\times n}, \tag{4.2}$$

definida por

$$\Phi^0_{XS}[0] = I;$$
$$\Phi^k_{XS}[m] = \sum_{[\mu]_m \in P/m} R_{[\mu]_m(1)} \ldots R_{[\mu]_m(n)},$$

donde $m = k, \ldots, 1, 0$ y $[\mu]_m(i)$ denota la componente i-ésima de la permutación μ de $\{1, 2, \ldots, k\}$.

Ejemplo 4.2. *En el ejemplo anterior* ($k = 3$)*, para el caso particular* $m = 1$*, de acuerdo con la* **Definición 4.2** *y el* **Ejemplo 4.1**, *tenemos que* $R_1 = X$ *y* $R_2 = R_3 = S$ *entonces*

$$\Phi^3_{XS}[1] = \sum_{[\mu]_1 \in P/1} R_{[\mu]_1(1)} R_{[\mu]_1(2)} R_{[\mu]_1(3)}.$$

Luego, $\Phi^3_{XS}[1] = R_1R_2R_3 + R_3R_2R_1 + R_2R_1R_3 = XSS + SSX + SXS.$

En [45], los autores describen la expansión de $(X + tS)^k$, $k = 1, 2, \ldots, m$, a partir de la función Φ_{XS}, de la siguiente forma

$$\begin{aligned}
(X+tS)^0 &= I = \Phi^0_{XS}[0],\\
(X+tS)^1 &= X + tS = \Phi^1_{XS}[1] + t\,\Phi^1_{XS}[0],\\
(X+tS)^2 &= X^2 + t(XS+SX) + t^2S^2 = \Phi^2_{XS}[2] + t\,\Phi^2_{XS}[1] + t^2\,\Phi^2_{XS}[0],\\
(X+tS)^3 &= X^3 + t(XXS + XSX + SXX) + t^2(XSS + SXS + SSX) + t^3S^3\\
&= \Phi^3_{XS}[3] + t\,\Phi^3_{XS}[2] + t^2\,\Phi^3_{XS}[1] + t^3\,\Phi^3_{XS}[0],\\
&\vdots\\
(X+tS)^m &= \Phi^m_{XS}[m] + t\,\Phi^m_{XS}[m-1] + \cdots + t^{m-1}\,\Phi^m_{XS}[1] + t^m\,\Phi^m_{XS}[0].
\end{aligned}$$

Por lo tanto,

$$(X + tS)^k = \sum_{i=0}^{k} t^i\,\Phi^k_{XS}[k-i];$$

además, por propiedades de la transpuesta hermitiana,

$$\left(\Phi^k_{XS}[i]\right)^H = \left(\sum_{[\mu]_m} R_{[\mu]_m(1)} \ldots R_{[\mu]_m(n)}\right)^H = \Phi^k_{X^HS^H}[i], \tag{4.3}$$

donde H denota la transpuesta hermitiana. Usando (4.3), tenemos que

$$P(X + tS) = \sum_{k=0}^{m}\sum_{i=0}^{k} t^i A_k\,\Phi^k_{XS}[k-i]. \tag{4.4}$$

Extrayendo los dos primeros términos de la doble sumatoria (4.4) y teniendo en cuenta que por el método de Newton (Capítulo 2), $L_X(S) = -P(X)$, donde

$$L_X(S) = \sum_{k=1}^{m} \left\{ \left(\sum_{i=0}^{m-k} A_{m-i} X^{m-(i+k)} \right) S X^{k-1} \right\}, \tag{4.5}$$

se tiene que

$$\begin{aligned} P(X+tS) &= P(X) - tP(X) + \sum_{k=2}^{m} \sum_{i=2}^{k} t^i A_k \Phi_{XS}^k[k-i] \\ &= (1-t)P(X) + \sum_{k=2}^{m} \sum_{i=2}^{k} t^i A_k \Phi_{XS}^k[k-i] \\ &= (1-t)P(X) + \ SUM(m), \end{aligned}$$

donde para simplificar denotamos

$$SUM(m) := \sum_{k=2}^{m} \sum_{i=2}^{k} t^i A_k \Phi_{XS}^k[k-i]. \tag{4.6}$$

Además, la función de mérito (4.1) se puede expresar de la siguiente forma:

$$\begin{aligned} p(t) &= \|P(X+tS)\|_F^2 \\ &= \mathrm{tr}\Big([\ (1-t)P(X) + \ SUM(m)\ ]^H [\ (1-t)P(X) + \ SUM(m)\ ] \Big) \\ &= (1-t)^2 \|P(X)\|_F^2 + \cdots + t^{2m} \|A_m \Phi_{XS}^m[0]\|_F^2 \ , \end{aligned} \tag{4.7}$$

la cual corresponde a un polinomio en la variable t de grado $2m$.

4.3. El polinomio explícito

En esta sección, obtenemos todos los coeficientes de $p(t)$ en forma explícita y con ello, determinamos explícitamente dicho polinomio. En primer lugar, para simplificar la doble sumatoria $SUM(m)$ dada por (4.6), realizamos las siguientes asignaciones para un número natural m dado.

Definición 4.3. *Dado $m \in \mathbb{N}$, definimos las matrices B y B_r, $r = 1, \ldots, q$, donde*

$$q = \frac{m(m-1)}{2} \tag{4.8}$$

por

$$\begin{aligned} B &= P(X), \\ B_1 &= A_2\Phi^2_{XS}[0], \\ B_2 &= A_3\Phi^3_{XS}[1], \\ B_3 &= A_3\Phi^3_{XS}[0], \\ B_4 &= A_4\Phi^4_{XS}[2], \\ B_5 &= A_4\Phi^4_{XS}[1], \\ B_6 &= A_4\Phi^4_{XS}[0], \\ &\vdots \\ B_r &= A_k\Phi^k_{XS}[k-i], \\ &\vdots \\ B_q &= A_m\Phi^m_{XS}[0], \end{aligned} \tag{4.9}$$

donde r, k, i *son los únicos enteros tales que* $r = \frac{(k-2)(k-1)}{2} + (i-1)$, *con* $1 \le r \le q$, $2 \le k \le m$ *y* $2 \le i \le k$.

Además, por la **Definición 4.3**, tenemos que

$$SUM(m) = \sum_{k=2}^{m}\sum_{i=2}^{k} t^i A_k \Phi^k_{XS}[k-i] = \sum_{k=2}^{m}\sum_{i=2}^{k} t^i\, B_{\frac{(k-2)(k-1)}{2}+(i-1).} \tag{4.10}$$

Entonces, de (4.7) y (4.10), tenemos que $p(t)$ se puede expresar por:

$$p(t) = \mathrm{tr}\Big([(1-t)\,B + SUM(m)]^H\, [(1-t)\,B + \ SUM(m)] \Big).$$

Para entender mejor la expresión (4.9), veamos en detalle la correspondencia del índice r de B_r y los índices k, i. Para facilitar, pensemos en los elementos distintos de cero de la triangular inferior de orden $(m-1)$, donde las filas (k) y las columnas (i) están numeradas de 2 a m, como se muestra en la Figura 4.1, y r corresponde a la numeración secuencial por filas de ellos; es decir, el elemento en la posición (k, i) corresponde al número de elementos en las $(k-2)$ filas anteriores $\left[\sum_{j=2}^{k-1}(j-1) = (k-1)(k-2)/2\right]$ más el número de elementos en la fila k incluyendo la posición (k, i) $[i-1]$. En resumen,

$$r = r(k,i) = \frac{(k-2)(k-1)}{2} + i - 1. \tag{4.11}$$

Por otro lado, dado el índice (entero positivo) $r \le q = m(m-1)/2$, podemos encontrar los únicos enteros k, i tales que $r = r(k, i)$ como sigue:

1. Calcule el máximo entero k tal que $(k-2)(k-1)/2 < r$, el cual existe porque $r \leq q$ y es único porque la expresión $(k-2)(k-1)/2$ es estrictamente creciente con k.

2. Halle el entero $i = r-(k-2)(k-1)/2+1$ que es único y positivo por definición; además, es menor o igual a k por la forma en que se eligió el entero k (si $i > k$, k no sería el máximo en el paso 1).

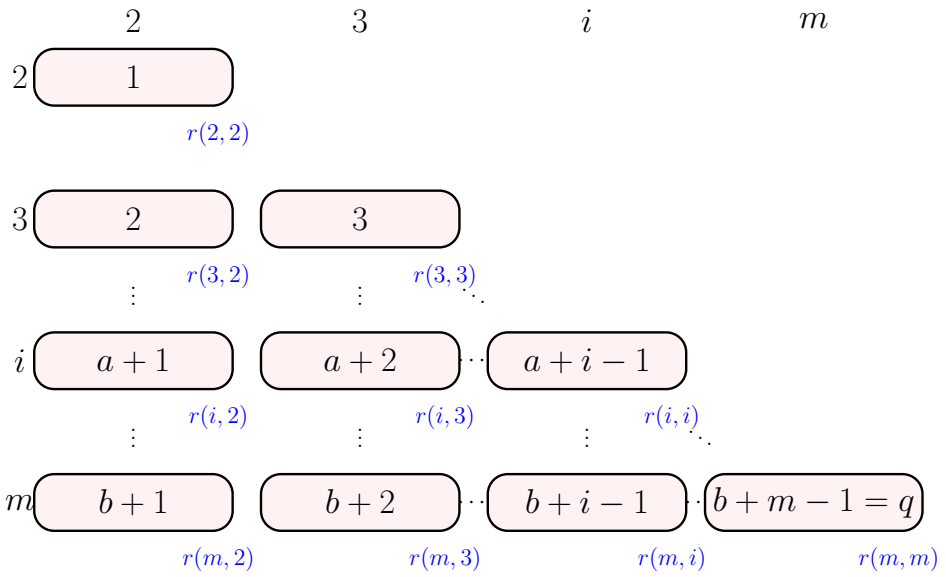

Figura 4.1: Índices de la matriz B_r

En la Figura 4.1,

$$a = \frac{(k-2)(k-1)}{2} \quad \text{y} \quad b = \frac{(m-2)(m-1)}{2}.$$

Observemos que

$$b+m-1 = \frac{(m-2)(m-1)}{2} + m - 1 = (m-1)\frac{[(m-2)+2]}{2} = \frac{(m-1)m}{2} = q.$$

Por otro lado, de (4.7) y (4.10), tenemos que $p(t)$ se puede expresar por

$$p(t) = \operatorname{tr}\left(\left[(1-t)\, B + \sum_{k=2}^{m} \sum_{i=2}^{k} t^i\, B_{r(k,i)} \right]^H \left[(1-t)\, B + \sum_{k=2}^{m} \sum_{i=2}^{k} t^i\, B_{r(k,i)} \right] \right), \qquad (4.12)$$

donde $r(k,i)$ está dado por (4.11).

En el siguiente ejemplo, presentamos dos casos particulares del polinomio (4.12), los cuales serán de gran utilidad para la deducción de los coeficientes del polinomio en el caso general.

Ejemplo 4.3. *Para $m=3$ y $m=5$, expresemos los coeficientes del polinomio (4.12) en términos de la norma de Frobenius.*

Para $m=3$, de (4.12) y algunas manipulaciones algebraicas, tenemos que

$$\begin{aligned} p(t) &= \operatorname{tr}\Big(\left[(1-t)\,B + t^2B_1 + t^2B_2 + t^3B_3\right]^H \left[(1-t)\,B + t^2B_1 + t^2B_2 + t^3B_3\right]\Big) \\ &= \underbrace{\|B\|_F^2}_{c_0}(1-t)^2 + \underbrace{\operatorname{tr}\Big(B^HB_1 + B^HB_2 + B_1^HB + B_2^HB\Big)}_{c_1}(1-t)t^2 \\ &\quad + \underbrace{\operatorname{tr}\Big(B^HB_3 + B_3^HB\Big)}_{c_2}(1-t)t^3 + \underbrace{\operatorname{tr}\Big(B_1^HB_1 + B_1^HB_2 + B_2^HB_1 + B_2^HB_2\Big)}_{c_3}t^4 \\ &\quad + \underbrace{\operatorname{tr}\Big(B_1^HB_3 + B_2^HB_3 + B_3^HB_1 + B_3^HB_2\Big)}_{c_4}t^5 + \underbrace{\|B_3\|_F^2}_{c_5}t^6. \end{aligned}$$

Los coeficientes de $p(t)$ pueden escribirse en términos de la norma de Frobenius, como lo ilustra la Tabla 4.1.

c_i	***valor***
c_0	$\|B\|_F^2$
c_1	$\|B+B_1+B_2\|_F^2 - \|B_1+B_2\|_F^2 - \|B\|_F^2$
c_2	$\|B+B_3\|_F^2 - \|B_3\|_F^2 - \|B\|_F^2$
c_3	$\|B_1+B_2\|_F^2$
c_4	$\|B_1+B_2+B_3\|_F^2 - \|B_1+B_2\|_F^2 - \|B_3\|_F^2$
c_5	$\|B_3\|_F^2$

Tabla 4.1: *Coeficientes de $p(t)$, para $m=3$.*

En forma análoga, para $m = 5$, los coeficientes de $p(t)$ pueden expresarse en términos de la norma de Frobenius, como se aprecia en la Tabla 4.2.

$\boldsymbol{c}_i$	***valor***
$\boldsymbol{c}_0$	$\|B\|_F^2$
$\boldsymbol{c}_1$	$\|B + B_1 + B_2 + B_4 + B_7\|_F^2 - \|B_1 + B_2 + B_4 + B_7\|_F^2 - \|B\|_F^2$
$\boldsymbol{c}_2$	$\|B + B_3 + B_5 + B_8\|_F^2 - \|B_3 + B_5 + B_8\|_F^2 - \|B\|_F^2$
$\boldsymbol{c}_3$	$\|B + B_6 + B_9\|_F^2 - \|B_6 + B_9\|_F^2 - \|B\|_F^2$
$\boldsymbol{c}_4$	$\|B + B_{10}\|_F^2 - \|B_{10}\|_F^2 - \|B\|_F^2$
$\boldsymbol{c}_5$	$\|B_1 + B_2 + B_4 + B_7\|_F^2$
$\boldsymbol{c}_6$	$\|B_1 + B_2 + B_3 + B_4 + B_5 + B_7 + B_8\|_F^2 - \|B_1 + B_2 + B_4 + B_7\|_F^2 - \|B_3 + B_5 + B_8\|_F^2$
$\boldsymbol{c}_7$	$\|B_1 + B_2 + B_4 + B_6 + B_7 + B_9\|_F^2 - \|B_1 + B_2 + B_4 + B_7\|_F^2 - \|B_6 + B_9\|_F^2 + \|B_3 + B_5 + B_8\|_F^2$
$\boldsymbol{c}_8$	$\|B_1 + B_2 + B_4 + B_7 + B_{10}\|_F^2 - \|B_1 + B_2 + B_4 + B_7\|_F^2 - \|B_{10}\|_F^2 + \|B_3 + B_5 + B_6 + B_8 + B_9\|_F^2 - \|B_3 + B_5 + B_8\|_F^2 - \|B_3 + B_9\|_F^2$
$\boldsymbol{c}_9$	$\|B_3 + B_5 + B_8 + B_{10}\|_F^2 - \|B_3 + B_5 + B_8\|_F^2 - \|B_{10}\|_F^2 + \|B_6 + B_9\|_F^2$
$\boldsymbol{c}_{10}$	$\|B_6 + B_9 + B_{10}\|_F^2 - \|B_6 + B_9\|_F^2 - \|B_{10}\|_F^2$
$\boldsymbol{c}_{11}$	$\|B_{10}\|_F^2$

Tabla 4.2: *Coeficientes de $p(t)$, para $m = 5$.*

Después de analizar varios casos particulares, tenemos que el polinomio $p(t)$, de grado $2m$, tiene la forma dada en el siguiente teorema. Para definir exactamente sus coeficientes introducimos una recurrencia para un valor fijo de un entero s, con $2 \leq s \leq m$, por:

$$\alpha_0^s = \frac{(s-1)s}{2}, \qquad \alpha_i^s = [i + (s-2)] + \alpha_{i-1}^s, \qquad i = 1, 2, \ldots, m - s. \tag{4.13}$$

Por otra parte, para compactar algunas expresiones en lo que sigue de este capítulo

usaremos la siguiente notación:

$$\mathcal{S}_p(k) = \sum_{i=0}^{p-k} B_{\alpha_i^k}. \tag{4.14}$$

Teorema 4.1. *En forma explícita, el polinomio $p(t)$ dado en (4.1) está definido por:*

$$p(t) = \boldsymbol{c}_0(1-t)^2 + \boldsymbol{c}_1(1-t)t^2 + \cdots + \boldsymbol{c}_{m-1}(1-t)t^m + \boldsymbol{c}_m t^4 + \cdots + \boldsymbol{c}_{3m-4}t^{2m}, \tag{4.15}$$

donde m es el grado de la ecuación polinómica matricial (1.1), con $\boldsymbol{c}_0 = \|B\|_F^2$, $\boldsymbol{c}_{3m-4} = \|B_q\|_F^2$, con q dado por (4.8), y los coeficientes restantes son diferencias de normas de Frobenius de las matrices B_j definidas en (4.9). Exactamente, estos coeficientes están definidos por

1. *Para los coeficientes $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_{m-1}$, de $(1-t)t^p$, con $p = 2, 3, \ldots, m$, usando (4.13), definimos:*

$$\boldsymbol{c}_{s-1} = \|B + \mathcal{S}_m(s)\|_F^2 - \|\mathcal{S}_m(s)\|_F^2 - \|B\|_F^2, \quad s = 2, \ldots, m. \tag{4.16}$$

2. *Para el coeficiente $\boldsymbol{c}_m$, de t^4, usamos (4.13) para $s = 2$,*

$$\boldsymbol{c}_m = \|\mathcal{S}_m(2)\|_F^2. \tag{4.17}$$

3. *Para los coeficientes $\boldsymbol{c}_{m+1}$ y $\boldsymbol{c}_{m+2}$, de t^5 y t^6 respectivamente, tenemos*

$$\begin{aligned} \boldsymbol{c}_{m+1} &= \|\mathcal{S}_m(2) + \mathcal{S}_m(3)\|_F^2 - \|\mathcal{S}_m(2)\|_F^2 - \|\mathcal{S}_m(3)\|_F^2, \\ \boldsymbol{c}_{m+2} &= \|\mathcal{S}_m(2) + \mathcal{S}_m(4)\|_F^2 - \|\mathcal{S}_m(2)\|_F^2 - \|\mathcal{S}_m(4)\|_F^2 + \|\mathcal{S}_m(3)\|_F^2. \end{aligned} \tag{4.18}$$

4. *Para los coeficientes $\boldsymbol{c}_k$ de $t^{k+(4-m)}$, con $m+3 \le k \le 3m-5$; es decir, para los coeficientes de las potencias $7, 8, \ldots, 2m-1$, consideramos si la potencia $k+(4-m)$ es impar o par. En el primer caso, se puede expresar de la forma $2r+1$, $r \ge 3$ y tenemos las siguientes condiciones:*

   a) *si $m = r+1$,*

$$\boldsymbol{c}_k = \|\mathcal{S}_m(r) + \mathcal{S}_m(r+1)\|_F^2 - \|\mathcal{S}_m(r)\|_F^2 - \|\mathcal{S}_m(r+1)\|_F^2, \tag{4.19}$$

   b) *si $m > r+1$,*

$$\begin{aligned} \boldsymbol{c}_k = {} & \|\mathcal{S}_m(r) + \mathcal{S}_m(r+1)\|_F^2 - \|\mathcal{S}_m(r)\|_F^2 - \|\mathcal{S}_m(r+1)\|_F^2 \\ & + \|\mathcal{S}_m(r-1) + \mathcal{S}_m(r+2)\|_F^2 - \|\mathcal{S}_m(r-1)\|_F^2 \\ & - \|\mathcal{S}_m(r+2)\|_F^2. \end{aligned} \tag{4.20}$$

Análogamente, si la potencia $k+(4-m)$ es par entonces es de la forma $2r$, con $r \geq 4$ y tenemos las mismas condiciones que para el caso anterior:

a) *si $m = r+1$, tenemos*

$$\begin{aligned} \boldsymbol{c}_k = &\|\mathcal{S}_m(r-1) + \mathcal{S}_m(r+1)\|_F^2 - \|\mathcal{S}_m(r-1)\|_F^2 - \|\mathcal{S}_m(r+1)\|_F^2 \\ &+ \|\mathcal{S}_m(r)\|_F^2 , \end{aligned} \tag{4.21}$$

b) *si $m > r+1$, tenemos*

$$\begin{aligned} \boldsymbol{c}_k = &\|\mathcal{S}_m(r-1) + \mathcal{S}_m(r+1)\|_F^2 - \|\mathcal{S}_m(r-1)\|_F^2 - \|\mathcal{S}_m(r+1)\|_F^2 + \|\mathcal{S}_m(r)\|_F^2 \\ &+ \|\mathcal{S}_m(r-2) + \mathcal{S}_m(r+2)\|_F^2 - \|\mathcal{S}_m(r-2)\|_F^2 \\ &- \|\mathcal{S}_m(r+2)\|_F^2 . \end{aligned} \tag{4.22}$$

Demostración. Debemos demostrar que la expresión (4.15) para $p(t)$ coincide con (4.12); es decir,

$$\begin{aligned} p(t) \coloneqq p_m(t) &= \operatorname{tr}\Big(\left[(1-t)\, B + SUM(m)\right]^H \left[(1-t)\, B + SUM(m)\right] \Big) \\ &= \mathbf{c}_0(1-t)^2 + \mathbf{c}_1(1-t)t^2 + \cdots + \mathbf{c}_{m-1}(1-t)t^m + \mathbf{c}_m t^4 + \cdots + \mathbf{c}_{3m-4}t^{2m}. \end{aligned} \tag{4.23}$$

Realizaremos la demostración por inducción matemática sobre m, para $m \geq 2$.

1. Para $m = 2$, tenemos

$$\begin{aligned} p(t) = p_2(t) &= \operatorname{tr}\Big(\left[(1-t)\, B + SUM(2)\right]^H \left[(1-t)\, B + SUM(2)\right] \Big) \\ &= \operatorname{tr}\Big(\left[(1-t)\, B + t^2\, B_1\right]^H \left[(1-t)\, B + t^2\, B_1\right] \Big) \\ &= (1-t)^2 \operatorname{tr}\big(B^H B\big) + (1-t)t^2 \operatorname{tr}\big(B^H B_1 + B_1^H B\big) + t^4 \operatorname{tr}\big(B_1^H B_1\big). \end{aligned} \tag{4.24}$$

Observemos que

$$\begin{aligned} \operatorname{tr}\left(B^H B_1 + B_1^H B\right) &= \operatorname{tr}\left(B^H B + B^H B_1 + B_1^H B + B_1^H B_1 - B_1^H B_1 - B^H B\right) \\ &= \operatorname{tr}\Big(\left[B + B_1\right]^H \left[B + B_1\right]\Big) - \operatorname{tr}\left(B_1^H B_1\right) - \operatorname{tr}\left(B^H B\right) \\ &= \|B + B_1\|_F^2 - \|B_1\|_F^2 - \|B\|_F^2 , \end{aligned} \tag{4.25}$$

usando la recurrencia (4.13), expresamos en forma compacta la expresión (4.25) y obtenemos

$$\operatorname{tr}\left(B^H B_1 + B_1^H B\right) = \|B + \mathcal{S}_2(2)\|_F^2 - \|\mathcal{S}_2(2)\|_F^2 - \|B\|_F^2 , \tag{4.26}$$

reemplazando (4.26) en (4.24),

$$p(t) = p_2(t) = (1-t)^2 \|B\|_F^2 + (1-t)t^2 \left(\|B + \mathcal{S}_2(2)\|_F^2 - \|\mathcal{S}_2(2)\|_F^2 - \|B\|_F^2\right) + t^4 \|B_1\|_F^2 = \mathbf{c}_0(1-t)^2 + \mathbf{c}_1(1-t)t^2 + \mathbf{c}_2 t^4.$$

Por lo tanto, la igualdad (4.23), se cumple para $m = 2$.

2. *Hipótesis inductiva.* Supongamos que la igualdad (4.23) se cumple hasta un $m > 2$, esto es

$$\begin{aligned} p_m(t) &= \operatorname{tr}\Big(\left[(1-t)\,B + SUM(m)\right]^H \left[(1-t)\,B + SUM(m)\right]\Big) \\ &= \mathbf{c}_0(1-t)^2 + \mathbf{c}_1(1-t)t^2 + \mathbf{c}_2(1-t)t^3 + \cdots + \mathbf{c}_{(m-1)}(1-t)t^m + \mathbf{c}_m t^4 \\ &\quad + \mathbf{c}_{(m+1)}t^5 + \mathbf{c}_{(m+2)}t^6 + \cdots + \mathbf{c}_{(3m-4)}t^{2m}, \end{aligned}$$

donde $m+3 \leq k \leq 3m-5$ y los coeficientes están dados por $\mathbf{c}_0 = \|B\|_F^2$, (4.16), (4.17), (4.18), (4.19) o (4.20) y (4.21) o (4.22) y $\mathbf{c}_{3m-4} = \left\|B_{\frac{m(m-1)}{2}}\right\|_F^2$.

3. *Paso de inducción.* Probemos que (4.23) se cumple para $m+1$. Tenemos

$$p_{m+1}(t) = \operatorname{tr}\left[(1-t)\,B + SUM(m+1)\right]^H \left[(1-t)\,B + SUM(m+1)\right]. \quad (4.27)$$

Extrayendo el último término de la doble sumatoria y usando (4.27) y (4.8) en (4.27), tenemos

$$\begin{aligned} p_{m+1}(t) = \operatorname{tr}\Big(&\left[(1-t)\,B + SUM(m) + \left\{t^2 B_{q+1} + \cdots + t^{m+1}B_{q+m}\right\}\right]^H \\ &\left[(1-t)\,B + SUM(m) + \left\{t^2 B_{q+1} + \cdots + t^{m+1}B_{q+m}\right\}\right]\Big). \end{aligned} \quad (4.28)$$

Realizando el producto matricial indicado en (4.28) y aplicando propiedades de la traza, se tiene lo siguiente:

$$\begin{aligned} p_{m+1}(t) = \operatorname{tr}&\Big(\left[(1-t)\,B + SUM(m)\right]^H \left[(1-t)\,B + SUM(m)\right]\Big) \\ &+ \operatorname{tr}\Big(\left[(1-t)\,B + SUM(m)\right]^H \left[t^2 B_{q+1} + \cdots + t^{m+1}B_{q+m}\right] \\ &+ \left[t^2 B_{q+1} + \cdots + t^{m+1}B_{q+m}\right]^H \left[(1-t)\,B + SUM(m)\right] \\ &+ \left[t^2 B_{q+1} + \cdots + t^{m+1}B_{q+m}\right]^H \left[t^2 B_{q+1} + \cdots + t^{m+1}B_{q+m}\right]\Big). \end{aligned} \quad (4.29)$$

Usando la hipótesis inductiva en el primer término de (4.29), obtenemos

$$\begin{aligned} p_{m+1}(t) = p_m(t) &+ \operatorname{tr}\Big(\left[(1-t)\,B + SUM(m)\right]^H \left[t^2 B_{q+1} + \cdots + t^{m+1}B_{q+m}\right] \\ &+ \left[t^2 B_{q+1} + \cdots + t^{m+1}B_{q+m}\right]^H \left[(1-t)\,B + SUM(m)\right] \\ &+ \left[t^2 B_{q+1} + \cdots + t^{m+1}B_{q+m}\right]^H \left[t^2 B_{q+1} + \cdots + t^{m+1}B_{q+m}\right]\Big). \end{aligned} \quad (4.30)$$

Para el segundo sumando de (4.30), usaremos la recurrencia definida en (4.13) en la hipótesis inductiva. Además, teniendo en cuenta que para las potencias impares (b) y pares (p) de t, mayores o iguales que 7 y menores que $2m-1$, existe un entero positivo $r \geq 3$ tal que $b = 2r+1$ y $p = 2(r+1)$; la expresión (4.30) se puede expresar de la siguiente manera:

$$\begin{aligned}
p_{m+1}(t) = p_m(t) + \operatorname{tr}\Big(\big[& (1-t)t^2 \left\{ B^H B_{q+1} + B^H_{q+1} B \right\} + \dots \\
& + (1-t)t^m \left\{ B^H B_{q+(m-1)} + B^H_{q+(m-1)} B \right\} \\
& + (1-t)t^{m+1} \left\{ B^H B_{q+m} + B^H_{q+m} B \right\} \\
& + t^4 \left\{ \left(\mathcal{S}_m(2)\right)^H B_{q+1} + B^H_{q+1} \mathcal{S}_m(2) \right\} + t^5 \left\{ \left(\mathcal{S}_m(2)\right)^H B_{q+2} \right. \\
& \left. + B^H_{q+2} \mathcal{S}_m(2) + \left(\mathcal{S}_m(3)\right)^H B_{q+1} + B^H_{q+1} \left(\mathcal{S}_m(3)\right) \right\} \\
& + t^6 \left\{ \left(\mathcal{S}_m(2)\right)^H B_{q+3} + B^H_{q+3} \mathcal{S}_m(2) + \left(\mathcal{S}_m(3)\right)^H B_{q+2} \right. \\
& \left. + B^H_{q+2} \mathcal{S}_m(3) + \left(\mathcal{S}_m(4)\right)^H B_{q+1} + B^H_{q+1} \mathcal{S}_m(4) \right\} \\
& + t^b \left\{ \left(\mathcal{S}_m(r-1)\right)^H B_{q+r} + B^H_{q+r} \mathcal{S}_m(r-1) \right. \\
& + \left(\mathcal{S}_m(r+2)\right)^H B_{q+(r-2)} + B^H_{q+(r-2)} \mathcal{S}_m(r+2) \\
& + \left(\mathcal{S}_m(r+1)\right)^H B_{q+(r-1)} + B^H_{q+(r-1)} \mathcal{S}_m(r+1) \\
& \left. + \left(\mathcal{S}_m(r)\right)^H B_{q+r} + B^H_{q+r} \mathcal{S}_m(r) \right\} + t^p \left\{ \left(\mathcal{S}_m(r-1)\right)^H B_{q+(r+2)} \right. \\
& + B^H_{q+(r+2)} \mathcal{S}_m(r-1) + \left(\mathcal{S}_m(r)\right)^H B_{q+(r+1)} + B^H_{q+(r+1)} \mathcal{S}_m(r) \\
& + \left(\mathcal{S}_m(r+1)\right)^H B_{q+r} + B^H_{q+r} \mathcal{S}_m(r+1) + \left(\mathcal{S}_m(r+2)\right)^H B_{q+(r-1)} \\
& \left. + B^H_{q+(r-1)} \mathcal{S}_m(r+2) + \left(\mathcal{S}_m(r+3)\right)^H B_{q+(r-2)} + B^H_{q+(r-2)} \mathcal{S}_m(r+3) \right\} \\
& + \dots + t^{2(m+1)} \left\{ B^H_{q+(m+1-1)} B_{q+(m+1-1)} \right\} \big] \Big). \qquad (4.31)
\end{aligned}$$

De (4.31), analizamos los coeficientes de $p_{m+1}(t)$ teniendo en cuenta que los de $p_m(t)$ ya satisfacen la forma explícita y estos se conocen por la hipótesis inductiva. El coeficiente $\mathbf{c}_0$ de $p_{m+1}(t)$ es el coeficiente $\mathbf{c}_0$ de $p_m(t)$, que por hipótesis de inducción es $\mathbf{c}_0 = \|B\|_F^2$. Ahora, consideramos los coeficientes de $p_{m+1}(t)$, $\mathbf{c}_k$, con $1 \leq k \leq m-1$. Observe que cada uno de ellos tiene como primer sumando, el respectivo coeficiente de $p_m(t)$, cuya forma se da en la hipótesis inductiva. Además, usamos propiedades de traza y el hecho de que, para todo $1 \leq w \leq m$, el subíndice $q+w$ se puede expresar de la siguiente manera,

$$q + w = \frac{m(m-1)}{2} + w = \widetilde{q} - (m + 1 - (w+1)),$$

donde $\widetilde{q} = \dfrac{m(m+1)}{2}$.

a) El coeficiente $\mathbf{c}_1$ está dado por:

$$\begin{aligned}
\mathbf{c}_1 &= \underbrace{\|B + \mathcal{S}_m(2)\|_F^2 - \|\mathcal{S}_m(2)\|_F^2 - \|B\|_F^2}_{\text{Hipótesis inductiva}} + \operatorname{tr}\Big(\big[B^H B_{q+1} + B_{q+1}^H B\big]\Big) \\
&= \underbrace{\operatorname{tr}\Big(\big[B^H \mathcal{S}_m(2) + \mathcal{S}_m(2)^H B\big]\Big)}_{\text{Hipótesis indictiva}} + \operatorname{tr}\Big(\big[B^H B_{\widetilde{q}-(m+1-2)} + B_{\widetilde{q}-(m+1-2)}^H B\big]\Big) \\
&= \operatorname{tr}\Big(\big[B^H \mathcal{S}_m(2) + \mathcal{S}_m(2)^H B + B^H B_{\widetilde{q}-(m+1-2)} + B_{\widetilde{q}-(m+1-2)}^H B\big]\Big) \\
&= \operatorname{tr}\Big(\big[B^H \mathcal{S}_{m+1}(2) + \mathcal{S}_{m+1}(2)^H B\big]\Big), \qquad (4.32)
\end{aligned}$$

usando (4.25) en (4.32), tenemos

$$\mathbf{c}_1 = \|B + \mathcal{S}_{m+1}(2)\|_F^2 - \|\mathcal{S}_{m+1}(2)\|_F^2 - \|B\|_F^2 .$$

b) En forma análoga procedemos para encontrar los coeficientes de $\mathbf{c}_2, \ldots, \mathbf{c}_{m-1}$ obteniendo

$$\mathbf{c}_{k-1} = \|B + \mathcal{S}_{m+1}(k)\|_F^2 - \|\mathcal{S}_{m+1}(k)\|_F^2 - \|B\|_F^2 ,$$

donde $3 \leq k \leq m$.

c) Para el coeficiente $\mathbf{c}_m$ de $p_{m+1}(t)$, tenemos

$$\mathbf{c}_m = \operatorname{tr}\Big(\big[B^H B_{q+m} + B_{q+m}^H B\big]\Big),$$

ya que no hay coeficiente respectivo de $p_m(t)$. Así

$$\mathbf{c}_m = \|B + \mathcal{S}_{m+1}(m+1)\|_F^2 - \|\mathcal{S}_{m+1}(m+1)\|_F^2 - \|B\|_F^2 .$$

Para el coeficiente $\mathbf{c}_{m+1}$ de $p_{m+1}(t)$ (correspondiente a t^4) usaremos el coeficiente $\mathbf{c}_m$ de $p_m(t)$ correspondiente al dado por la hipótesis inductiva, esto es

$$\begin{aligned}
\mathbf{c}_{m+1} &= \underbrace{\|\mathcal{S}_m(2)\|_F^2}_{\text{Hipótesis inductiva}} + \operatorname{tr}\Big(\big[\mathcal{S}_m(2)^H B_{q+1} + B_{q+1}^H \mathcal{S}_m(2)\big]\Big) \\
&= \operatorname{tr}\Big(\big[\mathcal{S}_m(2) + B_{\widetilde{q}-(m+1-2)}\big]^H \big[\mathcal{S}_m(2) + B_{\widetilde{q}-(m+1-2)}\big]\Big) \\
&= \operatorname{tr}\Big(\big[\mathcal{S}_{m+1}(2)^H \mathcal{S}_{m+1}(2)\big]\Big) = \|\mathcal{S}_{m+1}(2)\|_F^2 .
\end{aligned}$$

Para el coeficiente $\mathbf{c}_{m+2}$ de $p_{m+1}(t)$ (correspondiente a t^5) procedemos en forma análoga al caso anterior, así

$$\begin{aligned}
\mathbf{c}_{m+2} &= \underbrace{\|\mathcal{S}_m(2)+\mathcal{S}_m(3)\|_F^2 - \|\mathcal{S}_m(2)\|_F^2 - \|\mathcal{S}_m(3)\|_F^2}_{\text{Hipótesis inductiva}} + \operatorname{tr}\Big(\big[\mathcal{S}_m(2)^H B_{q+2} \\
&\qquad + B_{q+2}^H \mathcal{S}_m(2) + \mathcal{S}_m(3)^H B_{q+1} + B_{q+1}^H \mathcal{S}_m(3)\big]\Big) \\
&= \underbrace{\operatorname{tr}\Big(\big[\mathcal{S}_m(2)^H \mathcal{S}_m(3) + \mathcal{S}_m(3)^H \mathcal{S}_m(2)\big]\Big)}_{\text{Hipótesis inductiva}} \\
&\qquad + \operatorname{tr}\Big(\big[\mathcal{S}_m(2)^H B_{\widetilde{q}+(m+1-3)} + B_{\widetilde{q}+(m+1-3)}^H \mathcal{S}_m(2) \\
&\qquad + \mathcal{S}_m(3)^H B_{\widetilde{q}+(m+1-2)} + B_{\widetilde{q}+(m+1-2)}^H \mathcal{S}_m(3)\big]\Big) \\
&= \operatorname{tr}\Big(\big[\mathcal{S}_m(2) + B_{\widetilde{q}+(m+1-2)}\big]^H \big[\mathcal{S}_m(3) + B_{\widetilde{q}+(m+1-3)}\big] \\
&\qquad + \big[\mathcal{S}_m(3) + B_{\widetilde{q}+(m+1-3)}\big]^H \big[\mathcal{S}_m(2) + B_{\widetilde{q}+(m+1-2)}\big]\Big) \\
&= \operatorname{tr}\Big(\big[\mathcal{S}_{m+1}(2)^H \mathcal{S}_{m+1}(3) + \mathcal{S}_{m+1}(3)^H \mathcal{S}_{m+1}(2)\big]\Big) \\
&= \|\mathcal{S}_{m+1}(2)+\mathcal{S}_{m+1}(3)\|_F^2 - \|\mathcal{S}_{m+1}(2)\|_F^2 - \|\mathcal{S}_{m+1}(3)\|_F^2 .
\end{aligned}$$

Entonces para el coeficiente $\mathbf{c}_{m+3}$ de $p_{m+1}(t)$ (correspondiente a t^6) procedemos de manera similar; así,

$$\begin{aligned}
\mathbf{c}_{m+3} &= \underbrace{\|\mathcal{S}_m(2)+\mathcal{S}_m(4)\|_F^2 - \|\mathcal{S}_m(2)\|_F^2 - \|\mathcal{S}_m(4)\|_F^2 + \|\mathcal{S}_m(3)\|_F^2}_{\text{Hipótesis inductiva}} \\
&\qquad + \operatorname{tr}\Big(\big[\mathcal{S}_m(2)^H B_{q+3} + B_{q+3}^H \mathcal{S}_m(2) + \mathcal{S}_m(3)^H B_{q+2} \\
&\qquad + B_{q+2}^H \mathcal{S}_m(3) + \left(\mathcal{S}_m(4)\right)^H B_{q+1} + B_{q+1}^H \mathcal{S}_m(4)\big]\Big) \\
&= \underbrace{\operatorname{tr}\Big(\big[\mathcal{S}_m(2)^H \mathcal{S}_m(4) + \mathcal{S}_m(4)^H \mathcal{S}_m(2) + \mathcal{S}_m(3)^H \mathcal{S}_m(3)\big]\Big)}_{\text{Hipótesis inductiva}} \\
&\qquad + \operatorname{tr}\Big(\big[\mathcal{S}_m(2)^H B_{\widetilde{q}+(m+1-4)} + B_{\widetilde{q}+(m+1-4)}^H \mathcal{S}_m(2) \\
&\qquad + \mathcal{S}_m(4)^H B_{\widetilde{q}+(m+1-2)} + B_{\widetilde{q}+(m+1-2)}^H \mathcal{S}_m(4) \\
&\qquad + \mathcal{S}_m(3)^H B_{\widetilde{q}+(m+1-3)} + B_{\widetilde{q}+(m+1-3)}^H \mathcal{S}_m(3)\big]\Big) \\
&= \operatorname{tr}\Big(\big[\mathcal{S}_{m+1}(2)^H \mathcal{S}_{m+1}(4) + \mathcal{S}_{m+1}(4)^H \mathcal{S}_{m+1}(2) + \mathcal{S}_{m+1}(3)^H \mathcal{S}_{m+1}(3)\big]\Big) \\
&= \|\mathcal{S}_{m+1}(2)+\mathcal{S}_{m+1}(4)\|_F^2 - \|\mathcal{S}_{m+1}(2)\|_F^2 - \|\mathcal{S}_{m+1}(4)\|_F^2 + \|\mathcal{S}_{m+1}(3)\|_F^2 .
\end{aligned}$$

Para los coeficientes $\mathbf{c}_k$ de $p_{m+1}(t)$, $m+4 \leq k \leq 3(m+1)-6$, de $t^{k+(3-m)}$, para las potencias $7, 8, \ldots, 2m$, consideramos dos casos (b impar o p par). En el primer caso, $b = 2r+1$, con $r \geq 4$. Usamos el coeficiente correspondiente de $p_m(t)$ dado en la hipótesis inductiva, por lo tanto

$$\begin{aligned}
\mathbf{c}_k = & \underbrace{\|\mathcal{S}_m(r) + \mathcal{S}_m(r+1)\|_F^2 - \|\mathcal{S}_m(r)\|_F^2 - \|\mathcal{S}_m(r+1)\|_F^2}_{\text{Hipótesis inductiva}} \\
& \underbrace{+ \|\mathcal{S}_m(r-1) + \mathcal{S}_m(r+2)\|_F^2 - \|\mathcal{S}_m(r-1)\|_F^2 - \|\mathcal{S}_m(r+2)\|_F^2}_{\text{Hipótesis inductiva}} \\
& + \operatorname{tr}\Big(\big[\mathcal{S}_m(r-1)^H B_{q+(r+1)} + B_{q+(r+1)}^H \mathcal{S}_m(r-1) \\
& + \mathcal{S}_m(r+2)^H B_{q+(r-2)} + B_{q+(r-2)}^H \mathcal{S}_m(r-2) \\
& + \mathcal{S}_m(r+1)^H B_{q+(r-1)} + B_{q+(r-1)}^H \mathcal{S}_m(r-2) \\
& + \mathcal{S}_m(r)^H B_{q+r} + B_{q+r}^H \mathcal{S}_m(r)\big]\Big) \\
= & \operatorname{tr}\Big(\big[\mathcal{S}_{m+1}(r)^H \mathcal{S}_{m+1}(r+1) + \mathcal{S}_{m+1}(r+1)^H \mathcal{S}_{m+1}(r) \\
& + \mathcal{S}_{m+1}(r-1)^H \mathcal{S}_{m+1}(r+2) + \mathcal{S}_m(r+2)^H \mathcal{S}_m(r-1)\big]\Big) \\
= & \|\mathcal{S}_{m+1}(r) + \mathcal{S}_{m+1}(r+1)\|_F^2 - \|\mathcal{S}_{m+1}(r)\|_F^2 - \|\mathcal{S}_{m+1}(r+1)\|_F^2 \\
& + \|\mathcal{S}_{m+1}(r-1) + \mathcal{S}_{m+1}(r+2)\|_F^2 - \|\mathcal{S}_{m+1}(r-1)\|_F^2 \\
& - \|\mathcal{S}_{m+1}(r+2)\|_F^2 .
\end{aligned}$$

Teniendo en cuenta que si $m = r$ $(m+1 = r+1)$, la suma $\mathcal{S}_{m+1}(r+2)$ es cero, la expresión anterior se puede expresar como

$$\mathbf{c}_k = \|\mathcal{S}_{m+1}(r) + \mathcal{S}_{m+1}(r+1)\|_F^2 - \|\mathcal{S}_{m+1}(r)\|_F^2 - \|\mathcal{S}_{m+1}(r+1)\|_F^2 .$$

Si p es par, $p = 2r$, con $r \geq 4$. En forma análoga a la impar, tenemos

$$\begin{aligned}
\mathbf{c}_k = & \|\mathcal{S}_{m+1}(r) + \mathcal{S}_{m+1}(r+2)\|_F^2 - \|\mathcal{S}_{m+1}(r)\|_F^2 - \|\mathcal{S}_{m+1}(r+2)\|_F^2 \\
& + \|\mathcal{S}_{m+1}(r+1)\|_F^2 + \|\mathcal{S}_{m+1}(r-1) + \mathcal{S}_{m+1}(r+3)\|_F^2 \\
& - \|\mathcal{S}_{m+1}(r-1)\|_F^2 - \|\mathcal{S}_{m+1}(r+2)\|_F^2 .
\end{aligned} \tag{4.33}$$

Teniendo en cuenta que si $m = r+1$ $(m+1 = r+2)$ $\mathcal{S}_{m+1}(r+3)$ se anula y en consecuencia (4.33), se expresa así:

$$\begin{aligned}
\mathbf{c}_k = & \|\mathcal{S}_{m+1}(r) + \mathcal{S}_{m+1}(r+2)\|_F^2 - \|\mathcal{S}_{m+1}(r)\|_F^2 - \|\mathcal{S}_{m+1}(r+2)\|_F^2 \\
& + \|\mathcal{S}_{m+1}(r+1)\|_F^2 .
\end{aligned}$$

Para el coeficiente $\mathbf{c}_k$ de p_{m+1}, con $k = 3(m+1)-5$ (correspondiente a t^{2m+1}), el primer sumando (que corresponde a la hipótesis inductiva) es nulo pues no hay coeficiente respectivo de $p_m(t)$, en este caso como la potencia es impar entonces se puede expresar de la forma $2m+1 = 2r+1$, es decir, $m = r$,

$$\begin{aligned}\mathbf{c}_k &= \operatorname{tr}\Big(\big[\mathcal{S}_{m+1}(r)^H\mathcal{S}_{m+1}(r+1) + \mathcal{S}_{m+1}(r+1)^H\mathcal{S}_{m+1}(r) + B_{q+(r-1)}\big]\Big)\\ &= \|\mathcal{S}_{m+1}(r) + \mathcal{S}_{m+1}(r+1)\|_F^2 - \|\mathcal{S}_{m+1}(r)\|_F^2 - \|\mathcal{S}_{m+1}(r+1)\|_F^2\,.\end{aligned}$$

El último coeficiente, $\mathbf{c}_{3(m+1)-4}$ de p_{m+1} (correspondiente a $t^{2(m+1)}$), se expresa por:

$$\mathbf{c}_{3(m+1)-4} = \operatorname{tr}\Big(B^H_{q+(m+1-1)}B_{q+(m+1-1)}\Big) = \operatorname{tr}\Big(B^H_{\widetilde{q}}B_{\widetilde{q}}\Big) = \|B_{\widetilde{q}}\|_F^2\,.$$

Con el cual se completa la demostración. ■

Por lo tanto, los coeficientes del polinomio (4.15) quedan determinados explícitamente y con ello, tenemos la forma explícita para el polinomio.

Un consecuencia inmediata del **Teorema 4.1** es el cálculo explícito del polinomio derivada $p'(t)$, como lo garantiza el siguiente corolario.

Corolario 4.1. *Sea $p(t)$ el polinomio (4.15). Entonces el polinomio derivada $p'(t)$ está dado por:*

$$p'(t) = -2\boldsymbol{c}_0(1-t) + \sum_{i=1}^{m-1}\left[(i+1) - t\,(i+2)\right]\boldsymbol{c}_i t^i + \sum_{i=1}^{2m-3}(i+3)\;\boldsymbol{c}_{m-1+i}\;t^{3+i-1},$$

donde los coeficientes $\boldsymbol{c}_i, i = 1,\ldots 3m-4$ son definidos explícitamente en el **Teorema 4.1**.

Demostración. Derivando el polinomio (4.15) respecto a t, obtenemos que

$$\begin{aligned}p'(t) &= -2\mathbf{c}_0(1-t) + 2\mathbf{c}_1(1-t)t + 3\mathbf{c}_2(1-t)t^2\cdots + m\,\mathbf{c}_{m-1}(1-t)t^{m-1} +\\ &\quad - \left(\mathbf{c}_1t^2 + \mathbf{c}_2t^3 + \cdots + \mathbf{c}_{m-1}t^m\right) + 4\mathbf{c}_mt^3 + 5\mathbf{c}_{m+1}t^4 + \cdots + 2m\,\mathbf{c}_{3m-4}t^{2m-1},\\ &= -2\mathbf{c}_0(1-t) + \sum_{i=1}^{m-1}(i+1)\,\mathbf{c}_i(1-t)t^i - \sum_{i=1}^{m-1}\mathbf{c}_i\,t^{i+1} + \sum_{i=1}^{2m-3}(i+3)\;\mathbf{c}_{m-1+i}\,t^{3+i-1}\\ &= -2\mathbf{c}_0(1-t) + \sum_{i=1}^{m-1}\left[(i+1) - t\,(i+2)\right]\mathbf{c}_it^i + \sum_{i=1}^{2m-3}(i+3)\;\mathbf{c}_{m-1+i}\,t^{3+i-1},\end{aligned}$$

con lo cual se termina la demostración. ■

4.4. Un criterio para minimizar $p(t)$ en $[0, 2]$

En el caso polinomial general, se ha propuesto minimizar la función de mérito en un intervalo $[0,\ k]$, con $k > 1$ [45]; pero no hay información sobre un valor específico de k. En esta sección obtenemos una condición suficiente que garantiza que un minimizador de $p(t)$ dado por (4.15) pertenece al intervalo $[0, 2]$.

Tenemos que $p'(0) = -2\,\mathbf{c}_0 < 0$. Ya que se requiere minimizar el polinomio $p(t)$, y tenemos $p'(0) < 0$, nos gustaría encontrar un número $k > 0$ tal que $p'(k) > 0$. Así, en el intervalo $[0, k]$ habría un minimizador. Para esto, analicemos $p'(2)$. Evaluando $p'(t)$ en $t = 2$ y realizando algunos cálculos algebraicos, tenemos

$$p'(2) = 2\mathbf{c}_0 - \sum_{i=1}^{m-1} 2^i\ (i+3)\,\mathbf{c}_i + \sum_{i=1}^{2m-3} 2^{i+2}\ (i+3)\ \mathbf{c}_{m-1+i}. \tag{4.34}$$

El siguiente teorema da una expresión equivalente a (4.34) en términos de la norma de Frobenius.

Teorema 4.2. *Sean m el grado del polinomio matricial (1.1). Entonces $p'(2) = p'_m(2)$ se puede expresar por:*

$$2\left[\left\|-B + \sum_{s=2}^{m} 2^{s-2}(s+2)\mathcal{S}_m(s)\right\|_F^2 - \left\|2\left[\sum_{s=3}^{m} 2^{s-3}(s-2)\mathcal{S}_m(s)\right]\right\|_F^2\right], \tag{4.35}$$

donde α_i es la recurrencia definida en (4.13).

Demostración. Debemos demostrar que la expresión (4.34) para $p'(2)$ coincide con (4.35), para todo $m \geq 2$; es decir,

$$\begin{aligned} p'_m(2) &= 2\mathbf{c}_0 - \sum_{i=1}^{m-1} 2^i\ (i+3)\,\mathbf{c}_i + \sum_{i=1}^{2m-3} 2^{i+2}\ (i+3)\ \mathbf{c}_{m-1+i} \\ &= 2\left[\left\|-B + \sum_{s=2}^{m} 2^{s-2}(s+2)\mathcal{S}_m(s)\right\|_F^2 - \left\|2\sum_{s=3}^{m} 2^{s-3}(s-2)\mathcal{S}_m(s)\right\|_F^2\right]. \end{aligned} \tag{4.36}$$

Realizaremos la demostración por inducción matemática sobre m, $m \geq 2$.

1. Para $m=2$, usando la norma de Frobenius y sus propiedades, tenemos

$$\begin{aligned} p_2'(2) &= 2\left[\mathbf{c}_0 - 4\mathbf{c}_1 + 16\mathbf{c}_2\right] \\ &= 2\left[\|B\|_F^2 - 4\left(\|B+B_1\|_F^2 - \|B_1\|_F^2 - \|B\|_F^2\right) + 16\,\|B_1\|_F^2\right] \\ &= 2\left[\operatorname{tr}\left(B^HB\right) - 4\operatorname{tr}\left(B^HB_1 + B_1^HB\right) + 16\operatorname{tr}\left(B_1^HB_1\right)\right], \end{aligned}$$

   equivalentemente $p_2'(2) = 2\,\|B - 4\,B_1\|_F^2$.

   Por lo tanto, (4.35) se cumple para $m=2$.[1]

2. *Hipótesis inductiva.* Supongamos que la igualdad (4.36) se cumple hasta un $m > 2$, (hemos denotado los coeficientes de $p_m(t)$ como $\widehat{\mathbf{c}}_0, \widehat{\mathbf{c}}_1, \ldots, \widehat{\mathbf{c}}_{m-1}$ y $\widehat{\mathbf{c}}_m, \ldots, \widehat{\mathbf{c}}_{3m-4}$); esto es,

$$\begin{aligned} p_m'(2) &= 2\,\widehat{\mathbf{c}}_0 - \sum_{i=1}^{m-1} 2^i\ (i+3)\,\widehat{\mathbf{c}}_i + \sum_{i=1}^{2m-3} 2^{i+2}\ (i+3)\ \widehat{\mathbf{c}}_{m-1+i} \\ &= 2\,\widehat{\mathbf{c}}_0 - \left[8\,\widehat{\mathbf{c}}_1 + \cdots + 2^{m-1}(m+2)\widehat{\mathbf{c}}_{m-1}\right] + 32\,\widehat{\mathbf{c}}_m + \cdots + 2^{2m-1}(2m)\,\widehat{\mathbf{c}}_{3m-4} \\ &= 2\left[\left\|-B + \sum_{s=2}^{m} 2^{s-2}(s+2)\mathcal{S}_m(s)\right\|_F^2 - \left\|2\sum_{s=3}^{m} 2^{s-3}(s-2)\mathcal{S}_m(s)\right\|_F^2\right], \end{aligned}$$

   en forma extendida, se puede expresar de la siguiente manera

$$\begin{aligned} p_m'(2) =& 2\left[\left\|-B + 4\mathcal{S}_m(2) + 10\mathcal{S}_m(3) + \cdots + 2^{m-2}(m+2)\,B_q\right\|_F^2\right. \\ &\left. - \left\|2\left[\mathcal{S}_m(3) + \cdots + 2^{m-3}(m-2)\,B_q\right]\right\|_F^2\right]. \end{aligned} \tag{4.37}$$

3. *Paso de inducción.* Demostremos que (4.36) se cumple para $m+1$. Observe que,

$$\begin{aligned} p_{m+1}'(2) =& 2\,\mathbf{c}_0 - \left[8\,\mathbf{c}_1 + \cdots + 2^{m-1}(m+2)\,\mathbf{c}_{m-1} + 2^m(m+3)\,\mathbf{c}_m\right] + 32\,\mathbf{c}_{m+1} \\ &+ \cdots + 2^{2m-1}(2m)\,\mathbf{c}_{3m-3} + 2^{2m}(2m+1)\,\mathbf{c}_{3m-2} + 2^{2m+1}(2m+2)\,\mathbf{c}_{3m-1}. \end{aligned}$$

   Dado que los coeficientes $\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{m-1}$ y $\mathbf{c}_{m+1}, \ldots, \mathbf{c}_{3m-3}$ del polinomio $p_{m+1}(t)$ se pueden expresar en términos de los coeficientes $\widehat{\mathbf{c}}_0, \widehat{\mathbf{c}}_1, \ldots, \widehat{\mathbf{c}}_{m-1}$ y $\widehat{\mathbf{c}}_m, \ldots, \widehat{\mathbf{c}}_{3m-4}$, de $p_m(t)$ respectivamente (ver demostración del **Teorema 1**), tenemos

[1] La expresión $p_2'(2) = 2\,\|B - 4B_1\|_F^2 \geq 0$, coincide con la expresión encontrada en [21].

$$
\begin{aligned}
p'_{m+1}(2) =& 2\,\widehat{\mathbf{c}}_0 - \big\{8\left[\widehat{\mathbf{c}}_1 + \operatorname{tr}\left(B^H B_{\widetilde{q}-(m+1-2)} + B^H_{\widetilde{q}-(m+1-2)}B\right)\right] + 20\left[\widehat{\mathbf{c}}_2\right.\\
&+ \operatorname{tr}\left(B^H B_{\widetilde{q}-(m+1-3)} + B^H_{\widetilde{q}-(m+1-3)}B\right)\big] + \cdots + 2^{m-1}(m+2)\left[\widehat{\mathbf{c}}_{m-1}\right.\\
&+ \operatorname{tr}\left(B^H B_{\widetilde{q}-(m+1-m)} + B^H_{\widetilde{q}-(m+1-m)}B\right)\big] + 2^m(m+3)\mathbf{c}_m\big\} + 32\left[\widehat{\mathbf{c}}_m\right.\\
&+ \operatorname{tr}\left(\mathcal{S}_m(2)^H B_{\widetilde{q}-(m+1-2)} + B^H_{\widetilde{q}-(m+1-2)}\mathcal{S}_m(2)\right)\big] + \ldots\\
&+ 2^{2m-1}(2m)\left[\widehat{\mathbf{c}}_{3m-4} + \operatorname{tr}\left(\mathcal{S}_m(r)^H B_{\widetilde{q}+(m+1-(r+2))}\right.\right.\\
&+ B^H_{\widetilde{q}+(m+1-(r+2))}\mathcal{S}_m(r) + \mathcal{S}_m(r+2)^H B_{\widetilde{q}+(m+1-r)}\\
&+ B^H_{\widetilde{q}+(m+1-r)}\mathcal{S}_m(r+2) + \mathcal{S}_m(r+1)^H B_{\widetilde{q}+(m+1-(r+1))}\\
&+ B^H_{\widetilde{q}+(m+1-(r+1))}\mathcal{S}_m(r+1) + \mathcal{S}_m(r-1)^H B_{\widetilde{q}+(m+1-(r+3))}\\
&+ B^H_{\widetilde{q}+(m+1-(r+3))}\mathcal{S}_m(r-1) + \mathcal{S}_m(r+3)^H B_{\widetilde{q}+(m+1-(r-1))}\\
&\left.\left.+B^H_{\widetilde{q}+(m+1-(r-1))}\mathcal{S}_m(r+3)\right)\right] + 2^{2m}(2m+1)\mathbf{c}_{3m-2}\\
&+ 2^{2m+1}(2m+2)\mathbf{c}_{3m-1}. \qquad (4.38)
\end{aligned}
$$

usando el lado derecho de (4.37) en (4.38), tenemos que

$$
\begin{aligned}
p'_{m+1}(2) =& p'_m(2) - \big\{8\operatorname{tr}\left(B^H B_{\widetilde{q}-(m+1-2)} + B^H_{\widetilde{q}-(m+1-2)}B\right) + \ldots\\
&+2^{m-1}(m+2)\operatorname{tr}\left(B^H B_{\widetilde{q}-(m+1-m)} + B^H_{\widetilde{q}-(j+1-j)}B\right) + 2^m(m+3)\mathbf{c}_m\big\}\\
&+ 32\operatorname{tr}\left(\mathcal{S}_m(2)^H B_{\widetilde{q}-(m+1-2)} + B^H_{\widetilde{q}-(m+1-2)}\mathcal{S}_m(2)\right) + \ldots\\
&+ 2^{2m}(2m+1)\mathbf{c}_{3m-2} + 2^{2m+1}(2m+2)\mathbf{c}_{3m-1}. \qquad (4.39)
\end{aligned}
$$

Por la hipótesis inductiva, (4.39) se puede expresar como

$$
\begin{aligned}
p'_{m+1}(2) =& 2\left[\left\|-B + 4\mathcal{S}_m(2) + 10\mathcal{S}_m(3) + \cdots + 2^{m-2}(m+2)\,B_q\right\|_F^2\right.\\
&\left.- \left\|2\,\left[\mathcal{S}_m(3) + 4\mathcal{S}_m(4) + \cdots + 2^{m-3}(m-2)\,B_q\right]\right\|_F^2\right]\\
&- \big\{8\operatorname{tr}\left(B^H B_{\widetilde{q}-(m+1-2)} + B^H_{\widetilde{q}-(m+1-2)}B\right) + \ldots\\
&+2^{m-1}(m+2)\operatorname{tr}\left(B^H B_{\widetilde{q}-(m+1-m)} + B^H_{\widetilde{q}-(m+1-m)}B\right) + 2^m(m+3)\mathbf{c}_m\big\}\\
&+ 32\operatorname{tr}\left(\mathcal{S}_m(2)^H B_{\widetilde{q}-(m+1-2)} + B^H_{\widetilde{q}-(m+1-2)}\mathcal{S}_m(2)\right) + \ldots\\
&+ 2^{2m}(2m+1)\mathbf{c}_{3m-2} + 2^{2m+1}(2m+2)\mathbf{c}_{3m-1}.
\end{aligned}
$$

Expresando la norma de Frobenuis en términos de la traza, usando sus propiedades y operaciones algebraicas, obtenemos la siguiente expresión:

$$
\begin{aligned}
p'_{m+1}(2) =& 2\Big\{ \operatorname{tr}\left(B^H B\right) - 4\operatorname{tr}\left(B^H \mathcal{S}_{m+1}(2) + \mathcal{S}_{m+1}(2)^H B\right) \\
& - 10\operatorname{tr}\left(B^H \mathcal{S}_{m+1}(3) + \mathcal{S}_{m+1}(3)^H B\right) - \dots \\
& - 2^{m-2}(m+2)\operatorname{tr}\left(B^H \mathcal{S}_{m+1}(m) + \mathcal{S}_{m+1}(m)^H B\right) \\
& - 2^{m+1-2}(m+1+2)\operatorname{tr}\left(B^H B_{\widetilde{q}} + B_{\widetilde{q}}^H B\right) + 16\operatorname{tr}\left(\mathcal{S}_{m+1}(2)^H \mathcal{S}_{m+1}(2)\right) \\
& + 40\operatorname{tr}\left(\mathcal{S}_{m+1}(2)^H \mathcal{S}_{m+1}(3) + \mathcal{S}_{m+1}(3)^H \mathcal{S}_{m+1}(2)\right) + \dots \\
& + 2^m(m+2)\operatorname{tr}\left(\mathcal{S}_{m+1}(2)^H \mathcal{S}_{m+1}(m) + \mathcal{S}_{m+1}(m)^H \mathcal{S}_{m+1}(2)\right) \\
& + 2^{m+1}(m+1+2)\operatorname{tr}\left(\mathcal{S}_{m+1}(2)^H B_{\widetilde{q}} + B_{\widetilde{q}}^H \mathcal{S}_{m+1}(2)\right) + \\
& + 2^{2m-4}(m+2)^2\operatorname{tr}\left(\mathcal{S}_{m+1}(m)^H \mathcal{S}_{m+1}(m)\right) + 2^{2m-2}(m+3)^2\operatorname{tr}\left(B_{\widetilde{q}}^H B_{\widetilde{q}}\right) \\
& - 4\operatorname{tr}\left(\mathcal{S}_{m+1}(3)^H \mathcal{S}_{m+1}(3)\right) - 16\operatorname{tr}\left(\mathcal{S}_{m+1}(3)^H \mathcal{S}_{m+1}(4)\right. \\
& \left. + \mathcal{S}_{m+1}(4)^H \mathcal{S}_{m+1}(3)\right) - \dots - 2^{m-1}(m-2)\operatorname{tr}\left(\mathcal{S}_{m+1}(3)^H \mathcal{S}_{m+1}(m)\right. \\
& \left. + \mathcal{S}_{m+1}(m)^H \mathcal{S}_{m+1}(3)\right) - 2^m(m-1)\operatorname{tr}\left(\mathcal{S}_{m+1}(3)^H B_{\widetilde{q}}\right. \\
& \left. + B_{\widetilde{q}}^H \mathcal{S}_{m+1}(3)\right) - 64\operatorname{tr}\left(\mathcal{S}_{m+1}(4)^H \mathcal{S}_{m+1}(4)\right) \\
& - 2^{m+1}(m-2)\operatorname{tr}\left(\mathcal{S}_{m+1}(4)^H \mathcal{S}_{m+1}(m) + \mathcal{S}_{m+1}(m)^H \mathcal{S}_{m+1}(4)\right) \\
& - 2^{m+2}(m-1)\operatorname{tr}\left(\mathcal{S}_{m+1}(4)^H B_{\widetilde{q}} + B_{\widetilde{q}}^H \mathcal{S}_{m+1}(4)\right) - \dots \\
& - 2^{2m-4}(m-2)^2\operatorname{tr}\left(\mathcal{S}_{m+1}(m)^H \mathcal{S}_{m+1}(m)\right) - 2^{2m-4}(m-2)^2\operatorname{tr}\left(B_{\widetilde{q}}^H B_{\widetilde{q}}\right)\Big\} \\
=& 2\Big\{ \operatorname{tr}\left(-B + 4\mathcal{S}_{m+1}(2) + 10\mathcal{S}_{m+1}(3) + \dots + 2^{m+1-2}(m+1+2)\, B_{\widetilde{q}}\right)^H \\
& \left(-B + 4\mathcal{S}_{m+1}(2) + 10\mathcal{S}_{m+1}(3) + \dots + 2^{m+1-2}(m+2)\, B_{\widetilde{q}}\right) \\
& - \operatorname{tr}\left(2\,\mathcal{S}_{m+1}(3) + 8\mathcal{S}_{m+1}(4) + \dots + 2^{m+1-2}(m+1-2)\, B_{\widetilde{q}}\right)^H \\
& \left(2\,\mathcal{S}_{m+1}(3) + 8\mathcal{S}_{m+1}(4) + \dots + 2^{m+1-2}(m+1-2)\, B_{\widetilde{q}}\right)\Big\},
\end{aligned}
$$

usando la norma de Frobenius, tenemos

$$
p'_{m+1}(2) = 2\left[\left\| -B + \sum_{s=2}^{m+1} 2^{s-2}(s+2)\mathcal{S}_{m+1}(s)\right\|_F^2 - \left\| 2\sum_{s=3}^{m+1} 2^{s-3}(s-2)\mathcal{S}_{m+1}(s)\right\|_F^2\right].
$$

Luego, se cumple la igualdad (4.35), para $m+1$. ■

A partir de (4.35), tenemos una condición necesaria y suficiente para $p'(2) \geq 0$, lo cual es formalizado en el siguiente corolario.

Corolario 4.2. *$p'(2) \geq 0$ si, y solo si*

$$\left\| -B + \sum_{s=2}^{m} 2^{s-2}(s+2)\mathcal{S}_m(s) \right\|_F^2 \geq \left\| 2 \left[\sum_{s=3}^{m} 2^{s-3}(s-2)\mathcal{S}_m(s) \right] \right\|_F^2 . \tag{4.40}$$

Por lo tanto, la expresión (4.40) da una condición necesaria y suficiente para que un minimizador de $p(t)$ pertenezca al intervalo $[0, 2]$, la cual puede usarse en un algoritmo como criterio para la elección del extremo superior del intervalo donde se buscará el minimizador de $p(t)$.

4.5. Experimentación numérica

En esta sección, consideramos el algoritmo Newton globalizado presentado en [45] y modificamos su búsqueda lineal exacta usando el polinomio explícito (4.15). Además, incluimos su derivada y la condición suficiente dada en el **Corolario 4.2** (llamaremos a este algoritmo *Newton explícito*). Comparamos su desempeño numérico con el algoritmo propuesto en [45] (que llamaremos *Newton*) y con el propuesto en [46] (que llamaremos *Newton relajado*), el cual usa una forma alternativa para calcular el tamaño de paso debido a la dificultad del cálculo de la función de mérito (4.1) y su minimización a medida el grado del polinomio matricial aumenta.

Debido a que los autores en [45] no suministran información sobre el *solver* que usaron para encontrar el minimizador de la función de mérito, usamos el algoritmo de *Newton* con el *solver* $sqp(t, f)$ de MATLAB®, donde el primer parámetro es una aproximación inicial y el segundo, es la función objetivo a minimizar (que en el caso que nos ocupa es la función de mérito).

Para mayor claridad en la lectura del documento, incluimos a continuación los tres algoritmos mencionados.

Algoritmo 1 *Newton explícito*

Entrada: $A_0,\ A_1, \ldots,\ A_m,\ X_0,\ \epsilon = 10^{-5},\ N_{max}$ y $k = 0$.
Salida: Matriz aproximación a X_*.

1: **mientras** $k \leq N_{max},\ Res(X_k) \geq \epsilon$ **hacer**
2: Resolver para S_k, el sistema matricial: $L_{X_k}(S_k) = -P(X_k)$.
Búsqueda lineal.
3: Construir el polinomio $p(t)$.
Aplicar la condición
4: **si** $\left\| -B + \sum_{s=2}^{m} 2^{s-2}(s+2)\mathcal{S}_m(s) \right\|_F^2 \geq \left\| 2\left[\sum_{s=3}^{m} 2^{s-3}(s-2)\mathcal{S}_m(s) \right] \right\|_F^2$ **entonces**
5: Encontrar $t \in [0, 2]$, el minimizador de $p(t)$.
6: **sino**
7: El minimizador no está en el intervalo $[0, 2]$.
8: **fin si**
9: Actualizar $X_{k+1} = X_k + t\, S_k$ y $k = k + 1$.
10: **fin mientras**
11: **Salida** X_*

Observación 1. *En el paso* 7 *del* **Algoritmo 1**, *para encontrar el minimizador fuera del intervalo* $[0, 2]$ *procedemos en forma similar como se hace en el intervalo* $[0, 2]$, *lo cual puede aumentar el costo computacional.*

Algoritmo 2 *Newton*

Entrada: $A_0,\ A_1, \ldots,\ A_m,\ X_0,\ \epsilon = 10^{-5},\ N_{max},\ t_0 = 0$ y $k = 0$.
Salida: Matriz aproximación a X_*.

1: **mientras** $k \leq N_{max},\ Res(X_k) \geq \epsilon$ **hacer**
2: Resolver para S_k, el sistema matricial: $L_{X_k}(S_k) = -P(X_k)$.
Búsqueda lineal.
3: $p(t) = \|P(X_k + t\, S_k)\|^2$.
4: Encontrar $t = sqp(t_0, p(t))$.
5: Actualizar $X_{k+1} = X_k + t\, S_k$ y $k = k + 1$.
6: **fin mientras**
7: **Salida** X_*

Algoritmo 3 *Newton relajado*

Entrada: $A_0,\, A_1, \ldots,\, A_m,\, X_0,\ \epsilon = 10^{-5},\ N_{max}$ y $k = 0$.
Salida: matriz aproximación a X_*.

1: **mientras** $k \leq N_{max},\ Res(X_k) \geq \epsilon$ **hacer**
2: Resolver para S_k, el sistema matricial: $L_{X_k}(S_k) = -P(X_k)$.
3: Actualizar $Z_k = X_k + S_k$.
4: **si** $Res(Z_k) < \epsilon$ **entonces**
5: $X_{k+1} = Z_k$ termina.
6: **sino**
7: Encuentre λ_k tal que

$$\lambda_k = \text{mín}\left\{ \frac{[P(X_k)]_{ij} + [P(X_{k+1})]_{ij}}{[P(X_k)]_{ij}} :\ [P(X_k)]_{ij} > \epsilon,\ i, j \in \{1, 2, \ldots, m\} \right\}$$

8: Actualizar $X_{k+1} = X_k + \lambda_k\, S_k$ y $k = k + 1$.
9: **fin si**
10: **fin mientras**
11: **Salida** X_*

Observación 2. *En el paso* 7 *del* **Algoritmo 3**, *la expresión* $[P(X_k)]_{ij}$ *significa la componente* ij *de la matriz* $P(X_k)$.

Es importante mencionar que el solver $sqp(\cdot\,,\cdot)$ resuelve el problema de minimización mediante programación cuadrática sucesiva y dado que en el algoritmo de Newton no se dispone de la derivada de la función objetivo, el *solver* hace el cálculo interno mediante diferencias finitas [8].

En la experimentación numérica consideramos los **Problemas 1, 2, 3, 4, 7** y **10** descritos en el Capítulo 3, en los cuales es necesario resolver ecuaciones matriciales de grado dos, seis, cuatro, tres, cuatro y cuatro respectivamente.

Para cada uno de los seis problemas comparamos el desempeño de los tres algoritmos en términos de número de iteraciones y tiempo de ejecución. Además, para el algoritmo *Newton explícito* reportamos en cada iteración el número 0, si la condición (4.40) se satisface y el número 1, cuando no se cumple.

A continuación, presentamos los resultados obtenidos en tablas que tiene la siguiente información: para el **Problema 1**, se incluye una columna que corresponde al valor del parámetro (δ), propio del problema, para los tres algoritmos; número de

iteraciones (N); valor de *Res* definido en (3.4); tiempo de ejecución CPU medido en segundos (*tiempo*); para el algoritmo *Newton explícito*, se incluye una columna llamada *con* que indica el número de veces que se infringió la condición (4.40) para encontrar el tamaño de paso en el intervalo $[0, 2]$. Para los demás problemas se omite la columna δ, el número de iteraciones (N) se cambia por (k) que indica cada iteración y se incorpora una nueva columna (*paso*), la cual da el valor del tamaño del paso de Newton en cada iteración.

	Newton explícito				*Newton*			*Newton relajado*		
δ	N	*Res*	*tiempo*	*con*	N	*Res*	*tiempo*	N	*Res*	*tiempo*
10^{-1}	5	1.8389e-06	0.2023	0	5	2.3015e-06	0.2749	5	1.3301e-06	0.1947
10^{-2}	6	1.3760e-05	0.2141	0	6	1.4076e-06	0.3349	6	4.3726e-06	0.2177
10^{-3}	6	6.0669e-06	0.2634	0	6	9.0776e-06	0.3653	6	2.1307e-06	0.2517
10^{-4}	6	6.4222e-06	0.2523	0	6	9.1425e-06	0.3533	6	2.1996e-06	0.2287
10^{-5}	6	6.4525e-06	0.2667	0	6	9.1420e-06	0.3690	6	2.2018e-06	0.2542
10^{-6}	6	6.4555e-06	0.2662	0	6	9.1429e-06	0.3606	6	2.2020e-06	0.2467
10^{-7}	6	6.4557e-06	0.2711	0	6	9.1421e-06	0.3665	6	2.2020e-06	0.2671
10^{-8}	6	6.4558e-06	0.2832	0	6	9.1410e-06	0.3793	6	2.2020e-06	0.2599

Tabla 4.3: Resultados del **Problema 1**, con $n = 16$ y $X_0 = 0$.

En la Tabla 4.3, podemos observar que el comportamiento de los tres algoritmos, respecto al número de iteraciones, es el mismo. Por otro lado, el *Newton relajado* tiene un mejor comportamiento que los otros algoritmos. En este caso, los algoritmos *Newton explícito* y *Newton* son 5 % y 46 % más lentos en términos de tiempo de ejecución, respectivamente. Además, en todos los casos obtuvimos un tamaño de paso en el intervalo $(0, 2]$; es decir, siempre se cumplió la condición.

	Newton explícito				*Newton*			*Newton relajado*		
k	*Res*	*paso*	*tiempo*	*con*	*Res*	*paso*	*tiempo*	*Res*	*paso*	*tiempo*
1	0.024100	1.1401		0	0.024100	1.1401		0.0247	1.1196	
2	2.7051e-03	1.0968		0	2.7051e-03	1.0968		2.9764e-03	1.0845	
3	1.4812e-04	1.0846		0	1.4812e-04	1.0846		1.5403e-04	1.0989	
4	5.6499e-07	1.0030	0.1966	0	5.7175e-07	1.0024	0.2086	8.0877e-07	1.0036	0.0842

Tabla 4.4: Resultados para el **Problema 2**, con $n = 5$ y $X_0 = 0$.

	Newton explícito				*Newton*			*Newton relajado*		
k	*Res*	*paso*	*tiempo*	*con*	*Res*	*paso*	*tiempo*	*Res*	*paso*	*tiempo*
1	4.4047e-03	1.2921		0	4.4047e-03	1.2921		8.1922e-03	1.1462	
2	2.5206e-04	1.0220		0	2.5206e-04	1.0220		1.0979e-03	1.0663	
3	3.4373e-06	1.0363	2.8066	0	3.2982e-06	1.0353	2.9749	3.2062e-05	1.1109	
4								1.6388e-07	1.0000	3.4716

Tabla 4.5: Resultados para el **Problema 2**, con $n = 50$ y $X_0 = 0$.

	Newton explícito				*Newton*			*Newton relajado*		
k	*Res*	*paso*	*tiempo*	*con*	*Res*	*paso*	*tiempo*	*Res*	*paso*	*tiempo*
1	2.3715e-03	1.3212		0	2.3715e-03	1.3212		5.8882e-03	1.1477	
2	9.2850e-05	1.0125		0	9.2850e-05	1.0125		7.9987e-04	1.0654	
3	5.7663e-07	1.0207	17.021	0	5.7718e-07	1.0204	17.382	2.1720e-05	1.1098	
4								1.1142e-07	1.0000	23.128

Tabla 4.6: Resultados para el **Problema 2**, con $n = 100$ y $X_0 = 0$.

Para el **Problema 2**, se considera la matriz inicial $X_0 = 0$ como lo hacen los autores en [47], para el método de *Newton relajado*. En la Tabla 4.4, podemos observar el mismo número de iteraciones para los tres, pero un mejor desempeño en cuanto a tiempo de ejecución para el *Newton relajado*, el cual tiene el 50 % del tiempo del *Newton explícito* y un 60 % del *Newton*. Por otro lado, cuando el tamaño del problema es $n = 50$ tenemos un comportamiento diferente, primero el *Newton relajado* realiza una iteración adicional con respecto a los otros algoritmos; adicionalmente, encontramos un mejor desempeño en el tiempo de ejecución del *Newton explícito* que ahora es un 20 % menor que el *Newton relajado* y un 16 % menor que el de *Newton* (ver Tabla 4.5). En este sentido, en la Tabla 4.6, para $n = 100$ obtenemos un mejor y más notable comportamiento del *Newton explícito* en número de iteraciones y tiempo de ejecución; en particular, el tiempo es 26 % menor que el de *Newton relajado* y un 18 % menor que el tiempo de *Newton*.

Es importante resaltar que para el **Problema 2**, el mejor desempeño del *Newton explícito* se obtiene cuando el tamaño del aumenta. Por otra parte, observamos el mismo tamaño de paso en cada iteración para *Newton explícito* y *Newton*, lo que evidencia que las fórmulas explícitas para el polinomio $p(t)$ y su derivada $p'(t)$ están bien calculadas.

	Newton explícito				*Newton*			*Newton relajado*		
k	*Res*	*paso*	*tiempo*	*con*	*Res*	*paso*	*tiempo*	*Res*	*paso*	*tiempo*
1	0.02114	3.6579		1	0.02114	3.6579		0.1110	1.3164	
2	1.5862e-03	0.9872		0	1.5862e-03	0.9872		0.1107	1.3163	
3	3.1878e-06	1.0078	0.0928	0	3.1878e-06	1.0078	0.1441	1.099	1.3161	
4								0.1062	1.3148	
5								0.0914	1.3081	
6								0.0477	1.2763	
7								4.9446e-03	1.1618	
8								6.4904e-05	1.0180	
9								2.5454e-08	1.0000	0.0459

Tabla 4.7: Resultados para el **Problema 3**, con $n = 3$ y $X_0 = 24I_3$.

En la Tabla 4.7, presentamos los resultados que se obtuvieron para la matriz inicial $X_0 = 24I_3$. Los tres algoritmos convergen al solvente S_1. En cuanto a número de iteraciones, *Newton explícito* y *Newton*, tienen un mejor desempeño, ya que estos convergen al solvente en 3 iteraciones y, el *Newton relajado* lo hace en 9 iteraciones. Por otro lado, observamos que el tiempo de ejecución del *Newton explícito* es aproximadamente un 65 % del usado por *Newton* y el doble del tiempo del *Newton relajado*. Además, observamos que para obtener el *Res* de la segunda iteración de *Newton explícito* y *Newton*, el algoritmo *Newton relajado* requiere nueve iteraciones. En este problema es importante resaltar que se infringe la condición en la primera iteración, con un tamaño de paso mayor que 2 en *Newton explícito* y *Newton*, algo que no ocurre en el *Newton relajado*, en el cual los tamaños de paso se mantienen cercanos a uno.

	Newton explícito				*Newton*			*Newton relajado*		
k	*Res*	*paso*	*tiempo*	*con*	*Res*	*paso*	*tiempo*	*Res*	*paso*	*tiempo*
1	0.0226	3.6402		1	0.0226	3.6402		0.1109	1.3164	
2	2.0274e-03	0.9345		0	2.0274e-03	0.9345		0.1106	1.3163	
3	7.8534e-06	1.0123	0.0914	0	7.8534e-06	1.0123	0.1528	0.1093	1.3159	
4								0.1044	1.3142	
5								0.0865	1.3061	
6								0.0458	1.2770	
7								3.3485e-03	1.1482	
8								2.762e-05	1.0126	
9								5.6014e-08	1.0000	0.0474

Tabla 4.8: Resultados para el **Problema 3**, con $n = 3$ y $X_0 = -24I_3$.

La Tabla 4.8, muestra que los resultados obtenidos por los algoritmos con matriz inicial $X_0 = -24\, I_3$ son similares a los obtenidos para la matriz inicial $X_0 = 24\, I_3$ (Tabla 4.7).

Realizamos otro análisis para este problema como en [45], escogiendo matrices iniciales arbitrarias de la forma

$$X_0 = \begin{bmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{bmatrix}, \qquad -100 \leq x_1, \ldots, x_9 \leq 100.$$

Presentamos los resultados en la Tabla 4.9,que contiene las siguientes columnas: *Iter*, el número de iteraciones; *éxito* indica el número de éxitos con las 100 matrices iniciales en el rango considerado de número convergente de iteraciones y *tiempo* corresponde al tiempo de ejecución promedio de cada éxito en ese rango; se incluye una columna *con* adicional para el algoritmo *Newton explícito* que indica el número de veces que la condición (4.40) no se cumple en ese rango.

	Newton explícito			*Newton*		*Newton relajado*	
Iter	*éxito*	*tiempo*	*con*	*éxito*	*tiempo*	*éxito*	*tiempo*
[0, 30]	70	0.0941	8	66	0.1953	28	0.0570
[31, 50]	9	0.3377	13	2	0.7285	9	0.1358
[51, 100]	5	0.49051	5	8	1.1076	4	0.3451

Tabla 4.9: Resultados para el **Problema 3** con matrices iniciales aleatorias.

Comparando con los resultados presentados en [45], observamos un comportamiento muy similar: se presenta para el *Newton explícito* (84 %), seguido por el método de *Newton* (76 %) y el *Newton relajado* (41 %) destacando que para los tres algoritmos el mayor número de éxitos se presenta en las primeras 30 iteraciones (70 %, 66 %, y 28 %, respectivamente). Además, en cuanto a tiempo de ejecución, el de mejor comportamiento es el *Newton relajado*. Por otro lado, la condición (4.40) en general, es infringida a lo más 13 veces en un total de 9 éxitos en la franja de $[51, 100]$ iteraciones.

	Newton explícito				*Newton*			*Newton relajado*		
k	*Res*	*paso*	*tiempo*	*con*	*Res*	*paso*	*tiempo*	*Res*	*paso*	*tiempo*
1	8.0250e-05	2.9722		1	8.0250e-05	2.9722		0.3031	1.2963	
2	6.1840e-08	0.9989	0.0644	0	6.1371e-08	0.9987	0.1310	0.2706	1.2963	
3								0.2233	1.2962	
4								0.1628	1.2961	
5								0.0989	1.2957	
6								0.0470	1.2901	
7								0.0166	1.1618	
8								4.0897e-03	1.2751	
9								5.1127e-04	1.2182	
10								7.3863e-06	1.0711	0.0338

Tabla 4.10: Resultados para el **Problema 4**, con $n = 2$ y $X_0 = 218I_3$.

En la Tabla 4.10, presentamos los resultados obtenidos. En cuanto a número de iteraciones, tenemos un mejor desempeño de los algoritmos *Newton explícito* y *Newton*, ya que estos convergen al solvente en 2 iteraciones, mientras que el algoritmo *Newton relajado* lo hace en 10. Adicionalmente, se observa que el costo por iteración del algoritmo de *Newton* es casi el doble del *Newton explícito*. Por otro lado, observamos que para obtener el *Res* de la primera iteración de *Newton explícito* y *Newton*, se requiere aproximadamente de 9 iteraciones en el *Newton relajado*. La convergencia en los tres algoritmos es al solvente S_1.

Es importante resaltar que el tamaño de paso en la primera iteración está fuera del intervalo $(0, 2]$ en los algoritmos *Newton explícito* y *Newton*.

	Newton explícito				*Newton*			*Newton relajado*		
k	*Res*	*paso*	*tiempo*	*con*	*Res*	*paso*	*tiempo*	*Res*	*paso*	*tiempo*
1	0.021141	3.6579		1	0.021141	3.6579		0.3585	1.2963	
2	1.5862e-03	0.9872		0	1.5862e-03	0.9872		0.3618	1.2963	
3	3.1878e-06	1.0078	0.0846	0	3.1878e-06	1.0078	0.1428	0.3667	1.2963	
4								0.3724	1.2962	
5								0.3708	1.2961	
6								0.3259	1.2957	
7								0.1610	1.2909	
8								0.0263	1.2909	
9								1.8054e-03	1.221	
10								5.9809e-05	1.1148	
11								1.7586e-07	1.0000	0.0338

Tabla 4.11: Resultados para el **Problema 4**, con $n = 2$ y $X_0 = -218I_3$.

Los resultados obtenidos en la Tabla 4.11 son muy similares a los presentados en la Tabla 4.10. En este caso, observamos que los algoritmos *Newton explícito* y *Newton* convergen a un solvente en 3 iteraciones, mientras que *Newton relajado* lo hace en 11. En términos de tiempo de ejecución, el algoritmo *Newton explícito* usa aproximadamente la mitad del tiempo de *Newton* y el doble de *Newton relajado*. El comportamiento del tamaño del paso de los algoritmos *Newton explícito* y *Newton* es el mismo en cada iteración, notándose que en la primera iteración está fuera del intervalo $(0, 2]$, lo cual no ocurre en el *Newton relajado*, el cual para alcanzar el valor de *Res* obtenido en la segunda iteración del *Newton explícito* y *Newton*, el *Newton relajado* requiere de 11 iteraciones. Es importante observar que los tres algoritmos convergen al solvente S_2.

	Newton explícito				*Newton*			*Newton relajado*		
k	*Res*	*paso*	*tiempo*	*con*	*Res*	*paso*	*tiempo*	*Res*	*paso*	*tiempo*
1	5.1371e-03	0.0576		0	5.1371e-03	0.0576		0.1011	1.3164	
2	1.4130e-03	2.1014		1	1.4130e-03	2.1014		0.1010	1.3164	
3	5.2013e-04	0.8782		0	5.2013e-04	0.8782		0.1008	1.3164	
4	6.5282e-05	0.9889		0	6.5282e-05	0.9889		0.1006	1.3164	
5	5.6437e-07	0.2865	0.2188	0	5.6438e-07	0.2865	0.2969	0.1001	1.3164	
6								0.0991	1.3164	
7								0.0970	1.3162	
8								0.0925	1.3154	
9								0.0860	1.3106	
10								0.0751	1.2852	
11								0.0600	1.1069	
12								0.0960	1.3162	
13								0.0870	1.3157	
14								0.0696	1.4165	
15								2.8122e-06	2	0.5781

Tabla 4.12: Resultados para el **Problema 10**, con $n = 10$, $\delta = 0.2$ y $X_0 = -I_n$.

En la Tabla 4.12 presentamos los resultados obtenidos para $X_0 = -I_{10}$. Los tres algoritmos convergen. Tenemos un mejor desempeño de los algoritmos *Newton explícito* y *Newton* respecto al número de iteraciones, ya que convergen a un solvente en 5 iteraciones, y el algoritmo *Newton relajado* lo hace en 15 iteraciones. Con respecto al tiempo CPU, observamos que el del algoritmo *Newton explícito* es de aproximadamente 74 % del tiempo utilizado por *Newton*, y el 38 % del *Newton relajado*. Adicionalmente, para este problema es importante resaltar que en el *Newton explícito*, la condición (4.40) no se cumple en la segunda iteración.

	Newton explícito				*Newton*			*Newton relajado*		
k	*Res*	*paso*	*tiempo*	*con*	*Res*	*paso*	*tiempo*	*Res*	*paso*	*tiempo*
1	0.8521	0.0439		0	0.8521	0.0439		–	–	
2	0.5512	0.2345		0	0.5512	0.2345		–	–	
3	0.5552	0.0481		0	0.5552	0.0481		–	–	
4	5.0282e-03	0.9343		0	5.0282e-03	0.9343				
5	3.0208e-03	0.5879		0	3.0208e-03	0.5879		–	–	
6	1.0029e-03	0.9326		0	1.0029e-03	0.9326		–	–	
7	8.6862e-07	0.9990	0.06250	0	8.6864e-07	0.9990	0.1094	–	–	

Tabla 4.13: Resultados para el **Problema 7**, con $n = 2$ y $X_0 = -10^{-2} I_n$.

Podemos observar de la Tabla 4.13 que el algoritmo *Newton relajado* no converge para la matriz inicial considerada, mientras que los algoritmos *Newton explícito* y *Newton* convergen en el mismo número de iteraciones . Con respecto al tiempo de CPU, observamos que el algoritmo *Newton explícito* usa 58 % del tiempo del algoritmo *Newton*.

4.6. Comentarios finales del capítulo

En los algoritmos globales tipo Newton para resolver una ecuación polinomial matricial, la búsqueda lineal exacta se realiza mediante una función de mérito que es un polinomio de grado dos veces mayor que el de la ecuación. En [21, 30, 45], este polinomio no aparece explícitamente para $m > 2$, donde m es el grado de la ecuación, y tampoco es explícito el intervalo de minimización. En este capítulo presentamos la forma explícita de ese polinomio y su derivada, y damos una condición necesaria y suficiente que garantiza que el intervalo de minimización es $[0, 2]$

Complementamos el análisis teórico con una comparación numérica del método Newton con búsqueda lineal, usando polinomio explícito (*Newton explícito*), usando función de mérito no explícita [45] (*Newton*) y el método propuesto en [46] (*Newton relajado*) que usa una forma alternativa para encontrar el tamaño del paso.

Los resultados numéricos nos permiten observar que independientemente del problema, el tamaño del paso siempre tiende al paso de Newton puro; el método *Newton explícito* funciona bien, ya que siempre toma el mismo número de iteraciones que los

métodos de Newton, pero se requiere menos tiempo. Además, la comparación entre el método *Newton explícito* y *Newton* confirma que las fórmulas explícitas presentadas para el polinomio y su derivada están bien calculadas, lo que se refleja en la igualdad en los tamaños de paso obtenidos. Por otro lado, el tiempo de ejecución de los dos algoritmos (el mejor es *Newton explícito*) muestra la ventaja de tener explícitamente la función de mérito al calcular la longitud del paso en el proceso de globalización.

Obviamente, la forma de calcular la longitud del paso del método *Newton relajado* es simple y tiene un bajo costo computacional; por esto, cuando funciona (calcula una buena longitud de paso), el tiempo de ejecución del algoritmo es menor que el de los otros algoritmos, pero los resultados numéricos muestran que muchas veces esta longitud de paso no es la adecuada, lo cual podemos ver en **Problema 3**, donde el algoritmo requiere 9 iteraciones para converger contra 3 de los otros dos algoritmos, y **Problema 10**, el algoritmo requiere 15 iteraciones para converger contra 5 de los otros dos algoritmos.

Además de lo anterior, los resultados presentados en la Tabla 4.9 muestran la efectividad en el cálculo de la longitud del paso (método de globalización) de los métodos *Newton explícito* y *Newton* en comparación con *Newton relajado*, que representa menos de la mitad de los aciertos en términos de convergencia global debido a la ineficiencia de su procedimiento en el cálculo de la longitud de paso.

En resumen, tener explícito el polinomio que define la función de mérito resulta ser una gran ventaja, ya que mantiene la efectividad global del método de Newton con menos tiempo de cómputo. Los resultados preliminares de las pruebas numéricas que realizamos confirman el buen desempeño del algoritmo *Newton* con búsqueda lineal exacta con una función de mérito explícita.

Capítulo 5

Un método cuasi-Newton local

En el método de Newton para resolver ecuaciones polinomiales matriciales involucra la solución de un sistema matricial de ecuaciones, para el cual se utiliza la descomposición de Schur en cada iteración haciéndolo costoso computacionalmente.

En este capítulo, proponemos un algoritmo cuasi-Newton para resolver ecuaciones polinomiales matriciales, donde el anterior sistema es reemplazado por un sistema matricial más fácil de resolver, el cual puede verse como una generalización del algoritmo del mismo tipo para resolver la ecuación cuadrática matricial propuesta en [31]. El algoritmo propuesto reduce el costo computacional del método de Newton tradicionalmente utilizado para resolver este tipo de ecuaciones. Demostramos que el algoritmo es local y hasta cuadráticamente convergente. En la parte final del capítulo, presentamos experimentos numéricos que ratifican los resultados teóricos desarrollados.

Es de anotar que el sistema propuesto en el algoritmo cuasi-Newton coincide con el sistema que resuelve en el método de Newton cuando las matriz inicial conmuta con las matrices coeficientes de la ecuación polinómica y la iteración del método de Newton está bien definida.

5.1. Algoritmo y resultados de convergencia

Como motivación de nuestro algoritmo, observemos que si en la iteración de Newton (2.3) tenemos conmutatividad entre las matrices A_j, X_k^j y S_k, para $j = 0, \ldots, m$, obtenemos una nueva iteración que llamaremos cuasi-Newton, dada por:

$$\begin{aligned} \mathcal{B}_m(X_k)S_k &= -P(X_k) \\ X_{k+1} &= X_k + S_k, \end{aligned} \tag{5.1}$$

donde $\mathcal{B}_m(X_k) = m\,A_m\,X_k^{m-1} + (m-1)\,A_{m-1}\,X_k^{m-2} + \cdots + A_1$, puede verse como una extensión al caso matricial de la derivada clásica de una función polinomial escalar. Así, siguiendo la filosofía de los métodos cuasi-Newton, aproximamos $L_{X_k}(S_k)$ definido en (2.4) por la matriz $\mathcal{B}_m(X_k)S_k$.

Ahora, nuestro interés es mostrar que la iteración (5.1) está bien definida y analizar su convergencia. Para ello, asumimos las siguientes hipótesis generales. Las primeras dos son análogas a las usados para demostrar convergencia local de algoritmos cuasi-Newton en el caso vectorial [14]. La tercera hipótesis puede verse como una versión matricial de la condición tipo *Dennis-Moré* [14], para convergencia superlineal de algoritmos cuasi-Newton.

H1. Existe $X_* \in \mathbb{C}^{n\times n}$ tal que $P(X_*) = 0$.

H2. La matriz $\mathcal{B}_m(X_*)$ es no singular y β es la norma de su inversa; es decir,

$$\left\|\mathcal{B}_m(X_*)^{-1}\right\| = \beta.$$

H3. Existe un $\theta \in [0, 1]$ tal que

$$\lim_{\|S\|\to 0} \frac{\|L_{X_*}(S) - \mathcal{B}_m(X_*)\,S\|}{\|S\|^{1+\theta}} = 0.$$

Equivalentemente, existe $\theta \in [0, 1]$, tal que, para todo $\gamma > 0$ existe $\epsilon_0 > 0$ tal que

$$\frac{\|L_{X_*}(S) - \mathcal{B}_m(X_*)\,S\|}{\|S\|^{1+\theta}} < \gamma,$$

siempre que $\|S\| < \epsilon_0$.

El siguiente es un lema técnico que permite acotar la norma de la diferencia de dos potencias de matrices. Este lema será útil en la demostración del **Teorema 5.1**.

Lema 5.1. *Sean* $m \in \mathbb{N}$, $m \geq 2$; $A, B \in \mathbb{C}^{n\times n}$ *y* $\|\cdot\|$ *una norma matricial inducida en* $\mathbb{C}^{n\times n}$. *Entonces*

$$\|A^m - B^m\| \leq \|A-B\| \left[(\|A+B\| + 2\|B\|) (\|A+B\| + \|B\|)^{m-2} + \sum_{j=2}^{m-1} \|B^j\| (\|A+B\| + \|B\|)^{m-(j+1)} \right]. \tag{5.2}$$

Demostración. La demostración la haremos por inducción sobre m.

1. Para $m = 2$, tenemos que

$$A^2 - B^2 = (A-B)(A+B) + (B-A)B + B(A-B),$$

así, $\|A^2 - B^2\| \leq \|A-B\| [\|A+B\| + 2\|B\|]$. Luego, (5.2) se satisface para $m = 2$.

2. *Hipótesis inductiva.* Supongamos que (5.2) se satisface para $m = k$; es decir,

$$\|A^k - B^k\| \leq \|A-B\| \left[(\|A+B\| + 2\|B\|) (\|A+B\| + \|B\|)^{k-2} + \sum_{j=2}^{k-1} \|B^j\| (\|A+B\| + \|B\|)^{k-(j+1)} \right].$$

3. *Paso de inducción.* Demostremos que (5.2) se cumple para $m+1$. Procediendo en forma similar al caso $m = 2$, tenemos

$$\begin{aligned} A^{k+1} - B^{k+1} &= (A^k - B^k)(A+B) + B^{k+1} - A^kB + B^kA - B^{k+1} \\ &= (A^k - B^k)(A+B) + (B^k - A^k)B + B^k(A-B), \end{aligned}$$

luego, $\|A^{k+1} - B^{k+1}\| \leq \|A^k - B^k\| [\|A+B\| + \|B\|] + \|A-B\| \|B^k\|$. Usando la hipótesis inductiva y realizando operaciones algebraicas, obtenemos

$$\begin{aligned}
\left\|A^{k+1}-B^{k+1}\right\| &\leq \|A-B\| \left[(\|A+B\|+2\|B\|)(\|A+B\|+\|B\|)^{k-2}\right. \\
&\quad \left.+\sum_{j=2}^{k-1}\left\|B^j\right\|(\|A+B\|+\|B\|)^{k-(j+1)}\right](\|A+B\|+\|B\|) \\
&\quad +\|A-B\|\left\|B^k\right\| \\
&\leq \|A-B\| \left[(\|A+B\|+2\|B\|)(\|A+B\|+\|B\|)^{(k+1)-2}\right. \\
&\quad \left.+\sum_{j=2}^{k-1}\left\|B^j\right\|(\|A+B\|+\|B\|)^{(k+1)-(j+1)}\right]+\|A-B\|\left\|B^k\right\| \\
&\leq \|A-B\| \left[(\|A+B\|+2\|B\|)(\|A+B\|+\|B\|)^{(k+1)-2}\right. \\
&\quad \left.+\sum_{j=2}^{(k+1)-1}\left\|B^j\right\|(\|A+B\|+\|B\|)^{(k+1)-(j+1)}\right].
\end{aligned}$$

Por lo tanto, se satisface (5.2) para $k+1$. ■

El siguiente teorema garantiza que existe una vecindad del solvente X_* de la ecuación polinomial matricial (1.1), tal que para cada matriz Z en esa vecindad, la matriz $\mathcal{B}_m(Z)$ es no singular y la norma de su inversa está acotada.

Teorema 5.1. *Sea $P\colon \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ una función polinómica matricial de grado $m \geq 2$, $P(X) = A_m X^m + A_{m-1}X^{m-1} + \cdots + A_0$ que satisface las hipótesis* **H1** *y* **H2**. *Existe una constante positiva ϵ que depende de m, tal que si $\|Z - X_*\| < \epsilon$ entonces la matriz $\mathcal{B}_m(Z)$ es no singular y $\|\mathcal{B}_m(Z)^{-1}\| \leq 2\beta$.*

Demostración. Sea $\epsilon > 0$, [1] tal que

$$\beta\left[\sum_{i=0}^{m-3}(m-i)\|A_{m-i}\|\left[(\epsilon+4\|X_*\|)(\epsilon+3\|X_*\|)^{m-(3+i)}\right.\right. \\ \left.\left.+\sum_{j=2}^{m-(2+i)}\left\|X_*^j\right\|(\epsilon+3\|X_*\|)^{m-(j+2+i)}\right]+2\|A_2\|\right]\epsilon \leq \frac{1}{2}. \tag{5.3}$$

[1] Observe que (5.3) puede expresarse como $g(\epsilon) \leq \frac{1}{2}$, así la existencia de ϵ está garantizada ya que $g(0) \leq \frac{1}{2}$ y g es una función creciente para $\epsilon > 0$ ($g'(\epsilon) > 0$).

Supongamos que $\|Z - X_*\| < \epsilon$. Entonces

$$\begin{aligned}\left\|\mathcal{B}_m(X_*)^{-1}\mathcal{B}_m(Z) - I_n\right\| &= \left\|\mathcal{B}_m(X_*)^{-1}\left[\mathcal{B}_m(Z) - \mathcal{B}_m(X_*)\right]\right\| \\ &\leq \left\|\mathcal{B}_m(X_*)^{-1}\right\| \left\|\mathcal{B}_m(Z) - \mathcal{B}_m(X_*)\right\| \\ &\leq \left\|\mathcal{B}_m(X_*)^{-1}\right\| \left[\sum_{j=0}^{m-2} \left\|(m-j)A_{m-j}(Z^{m-(j+1)} - X_*^{m-(j+1)})\right\|\right].\end{aligned}$$

Por **H2** y usando el primer término de la sumatoria, tenemos que

$$\begin{aligned}&\left\|\mathcal{B}_m(X_*)^{-1}\mathcal{B}_m(Z) - I_n\right\| \\ &\quad\leq \beta\left[m\left\|A_m\right\|\left\|Z^m - X_*^m\right\| + \sum_{j=1}^{m-2}(m-j)\left\|A_{m-j}\right\|\left\|Z^{m-(j+1)} - X_*^{m-(j+1)}\right\|\right].\end{aligned}$$

Usando el **Lema 5.1**, tenemos

$$\begin{aligned}&\left\|\mathcal{B}_m(X_*)^{-1}\mathcal{B}_m(Z) - I_n\right\| \\ &\quad\leq \beta\left[m\left\|A_m\right\|\left\|Z - X_*\right\|\left[\left(\|Z + X_*\| + 2\left\|X_*\right\|\right)\left(\|Z + X_*\| + \|X_*\|\right)^{m-3}\right.\right. \\ &\qquad\left.+ \sum_{j=2}^{m-2}\left\|X_*^j\right\|\left(\|Z + X_*\| + \|X_*\|\right)^{m-1-(j+1)}\right] \\ &\qquad+ 2\left\|Z - X_*\right\|\left[\sum_{i=0}^{m-4}(m-1-i)\left\|A_{m-1-i}\right\|\left[\left(\|Z + X_*\|\right.\right.\right. \\ &\qquad+ 2\left\|X_*\right\|\left)\left(\|Z + X_*\| + \|X_*\|\right)^{(m-2)-(2+i)}\right. \\ &\qquad\left.\left.\left.+ \sum_{j=2}^{(m-2)-(1+i)}\left\|X_*^j\right\|\left(\|Z + X_*\| + \|X_*\|\right)^{(m-2)-(j+1+i)}\right] + 2\left\|A_2\right\|\right]\right] \\ &\quad\leq \beta\left[\sum_{i=0}^{m-3}(m-i)\left\|A_{m-i}\right\|\left[\left(\|Z + X_*\| + 2\left\|X_*\right\|\right)\left(\|Z + X_*\| + \|X_*\|\right)^{(m)-(3+i)}\right.\right. \\ &\qquad\left.\left.+ \sum_{j=2}^{m-(2+i)}\left\|X_*^j\right\|\left(\|Z + X_*\| + \|X_*\|\right)^{m-(j+2+i)}\right] + 2\left\|A_2\right\|\right]\left\|Z - X_*\right\| \\ &\quad\leq \beta\left[\sum_{i=0}^{m-3}(m-i)\left\|A_{m-i}\right\|\left[\left(\|Z - X_*\| + 4\left\|X_*\right\|\right)\left(\|Z - X_*\| + 3\left\|X_*\right\|\right)^{(m)-(3+i)}\right.\right. \\ &\qquad\left.\left.+ \sum_{j=2}^{m-(2+i)}\left\|X_*^j\right\|\left(\|Z - X_*\| + 3\left\|X_*\right\|\right)^{m-(j+2+i)}\right] + 2\left\|A_2\right\|\right]\left\|Z - X_*\right\|.\end{aligned}$$

Así,

$$\left\|\mathcal{B}_m(X_*)^{-1}\mathcal{B}_m(Z) - I_n\right\| \leq \beta \left[\sum_{i=0}^{m-3}(m-i)\left\|A_{m-i}\right\| \left[\left(\epsilon + 4\left\|X_*\right\|\right)\left(\epsilon + 3\left\|X_*\right\|\right)^{m-(3+i)} + \sum_{j=2}^{m-(2+i)} \left\|X_*^j\right\| \left(\epsilon + 3\left\|X_*\right\|\right)^{m-(j+2+i)}\right] + 2\left\|A_2\right\|\right] \epsilon \leq \frac{1}{2}.$$

Luego, $1 - \|\mathcal{B}_m(X_*)^{-1}\mathcal{B}_m(Z) - I_n\| \geq 1/2$. Entonces por el **Lema 2.1** la matriz $\mathcal{B}_m(Z)$ es no singular y está acotada por 2β. En efecto,

$$\left\|\mathcal{B}_m(Z)^{-1}\right\| \leq \frac{\|\mathcal{B}_m(X_*)^{-1}\|}{1 - \|\mathcal{B}_m(X_*)^{-1}\mathcal{B}_m(Z) - I_n\|} \leq \frac{\|\mathcal{B}_m(X_*)^{-1}\|}{\frac{1}{2}} = 2\beta.$$

Lo cual completa la demostración. ■

Los dos lemas siguientes son resultados técnicos que usaremos en las demostraciones de los teoremas de convergencia del algoritmo propuesto. En los lemas consideramos $m \geq 3$ porque para $m = 2$, T_m es la matriz nula.

Lema 5.2. *Sean* $P\colon \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ *una función polinómica matricial de grado* $m \geq 3$, $X_* \in \mathbb{C}^{n\times n}$ *tal que* $P(X_*) = 0$, $\mathcal{B}_m(X) = mA_mX^{m-1} + (m-1)A_{m-1}X^{m-2} + \cdots + A_1$ *y*

$$T_m = \sum_{i=0}^{m-3} A_{m-i}\left[(m-2-i)\left(X^{m-i} - X_*^{m-i} - X_*^{m-i-1}X\right) + 2\left(\sum_{j=1}^{m-i-1} X_*^{m-i-j-1}XX_*^j\right) - (m-i)X^{m-i-1}X_*\right].$$

Entonces

$$[P(X) - P(X_*) - L_{X_*}(X - X_*)] + [\mathcal{B}_m(X_*)(X - X_*) - L_{X_*}(X - X_*)] + T_m = H_m,$$

donde $H_m = \sum_{i=0}^{m-2}(m-1-i)A_{m-i}X^{m-i} - A_0 - \mathcal{B}_m(X)X_*$.

Demostración. Para una función polinomial matricial P de grado m, tenemos

$$\begin{aligned} P(X) &= A_mX^m + A_{m-1}X^{m-1} + \cdots + A_0, \\ P(X_*) &= A_mX_*^m + A_{m-1}X_*^{m-1} + \cdots + A_0, \\ \mathcal{B}_m(X) &= mA_mX^{m-1} + (m-1)A_{m-1}X^{m-2} + \cdots + A_1, \end{aligned}$$

$$
\begin{aligned}
L_{X_*}(X-X_*) &= \sum_{i=1}^{m}\left[\sum_{j=0}^{m-i} A_{m-j}X^{m-i-j}\right](X-X_*)X^{i-1} \\
&= A_m\left(-mX_*^m + X_*^{m-1}X + X_*^{m-2}XX_* + \cdots + X_*XX_*^{m-2} + XX_*^{m-1}\right) \\
&\quad + \cdots + A_3\left(-3X_*^3 + X_*^2X + X_*XX_* + XX_*^2\right) \\
&\quad + A_2\left(-2X_*^2 + X_*X + XX_*\right) + A_1\left(X-X_*\right), \\
\mathcal{B}_m(X_*)\left(X-X_*\right) &= \left(mA_mX_*^{m-1} + \cdots + A_1\right)(X-X_*) \\
&= \left(mA_mX_*^{m-1}X + (m-1)A_{m-1}X_*^{m-2}X + \cdots + 2A_2X_*X + A_1X\right) \\
&\quad - \left(mA_mX_*^m + (m-1)A_{m-1}X_*^{m-1} + \cdots + 3A_3X_*^3 + 2A_2X_*^2 + A_1X_*\right)
\end{aligned}
$$

$$
\begin{aligned}
T_m =& A_m\left[(m-2)\left(X^m - X_*^m - X_*^{m-1}\right) + 2\left(X_*^{m-2}XX_* + \cdots + XX_*^{m-1}\right) - mX^{m-1}X_*\right] \\
&+ \cdots + A_3\left[X^3 - X_*^3 - X_*^2X + 2(X_*XX_* + XX_*^2) - 3X^2X_*\right].
\end{aligned}
$$

Teniendo en cuenta lo anterior y usando algunas manipulaciones algebraicas, tenemos

$$
\begin{aligned}
&P(X) - P(X_*) - L_{X_*}(X-X_*) + \mathcal{B}_m(X_*)(X-X_*) - L_{X_*}(X-X_*) + T_m \\
&= A_m\left[X^m - X_*^m + 2mX_*^m - 2X_*^{m-1}X - \cdots - 2X_*XX_*^{m-2} - 2XX_*^{m-1} + mX_*^{m-1}\right. \\
&\quad \left. -mX_*^m + (m-2)\left(X^m - X_*^m - X_*^{m-1}\right) + 2\left(X_*^{m-2}XX_* + \cdots + XX_*^{m-1}\right) - mX^{m-1}X_*\right] \\
&\quad + \cdots + A_3\left[X^3 - X_*^3 + 6X_*^3 - 2X_*^2X - 2X_*XX_* - 2XX_*^2 + 3X_*^2X - 3X_*^3 + X^3 - X_*^3\right. \\
&\quad \left. -X_*^2X + 2X_*XX_* + 2XX_*^2 - 3X^2X_*\right] + A_2\left[X^2 - X_*^2 + 4X_*^2 - 2X_*X - 2XX_*\right. \\
&\quad \left. +2X_*X - 2X_*^2\right] + A_1\left[X - X_* - 2X + 2X_* + X - X_*\right] \\
&= A_m\left[(m-1)X^m + X_*^m - mX^{m-1}X_*\right] + \cdots + A_3\left[2X^3 + X_*^3 - 3X^2X_*\right] \\
&\quad + A_2\left[X^2 + X_*^2 - 2XX_*\right].
\end{aligned}
$$

Sumando y restando A_1X_* y A_0, y usando $P(X_*) = 0$, tenemos

$$
\begin{aligned}
&P(X) - P(X_*) - L_{X_*}(X-X_*) + \mathcal{B}_m(X_*)(X-X_*) - L_{X_*}(X-X_*) + T_m \\
&= A_m\left[(m-1)X^m - mX^{m-1}X_*\right] + \cdots + A_3\left[2X^3 - 3X^2X_*\right] + A_2\left[X^2 - 2XX_*\right] \\
&\quad - A_1X_* - A_0 + \left(A_mX_*^m + \cdots + A_3X_*^3 + A_2X_*^2 + A_1X_* + A_0\right) \\
&= A_m\left[(m-1)X^m - mX^{m-1}X_*\right] + \cdots + A_3\left[2X^3 - 3X^2X_*\right] + A_2\left[X^2 - 2XX_*\right] \\
&\quad - A_1X_* - A_0 + (P(X_*)) \\
&= (m-2)A_mX^m + \cdots + 2A_3X^3 + A_2X^2 - A_0 - \left(3A_3X^2 + A_2X + A_1\right)X_* \\
&= \sum_{i=0}^{m-2}(m-1-i)A_{m-i}X^{m-i} - A_0 - \mathcal{B}_m(X_*)X_* = H_m,
\end{aligned}
$$

lo cual completa la demostración. ■

El siguiente lema da otra expresión y una cota para la matriz T_m del **Lema 5.2**.

Lema 5.3. *Sean* $P\colon \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ *una función polinómica matricial de grado* $m \geq 3$ *y*

$$T_m = \sum_{i=0}^{m-3} A_{m-i}\Bigg[(m-2-i)\left(X^{m-i} - X_*^{m-i} - X_*^{m-i-1}X\right) + 2\left(\sum_{j=1}^{m-i-1} X_*^{m-i-j-1}XX_*^j\right) - (m-i)X^{m-i-1}X_*\Bigg].$$

Entonces

$$T_m = \sum_{i=0}^{m-3} A_{m-i}\Bigg[(m-2-i)\left(X^{m-i-1} - X_*^{m-i-1}\right)(X-X_*) - 2\left(\sum_{j=1}^{m-i-3}\left(X^{m-i-j-1} - X_*^{m-i-j-1}\right)(X-X_*)X_*^j\right) - 2(X-X_*)^2 X_*^{m-i-2}\Bigg], \tag{5.4}$$

y

$$\begin{aligned}\|T_m\| \leq \Bigg\{\sum_{i=0}^{m-3}\|A_{m-j}\|\Bigg[&(m-2-i)\Big[(E+4\|X_*\|)(E+3\|X_*\|)^{m-i-3} \\ &+ \sum_{j=2}^{m-i-2}\|X_*\|^j(E+3\|X_*\|)^{m-i-j-2}\Big] \\ &+ 2\Bigg[\|X_*\|^{m-i-2} + \sum_{k=1}^{m-i-3}\Big[(E+4\|X_*\|)(E+3\|X_*\|)^{m-i-k-3} \\ &+ \sum_{j=2}^{m-i-k-3}\|X_*\|^j(E+3\|X_*\|)^{m-i-j-k-3}\Big]\|X_*\|^k\Bigg]\Bigg]\Bigg\}E^2,\end{aligned} \tag{5.5}$$

donde $E = \|X - X_*\|$.

Demostración. La demostración la haremos por inducción sobre m.

1. Para $m = 3$, tenemos $P(X) = A_3X^3 + A_2X^2 + A_1X + A_0$. Partiendo de T_3, reorganizando las matrices y factorizando a la derecha X; y X_*, obtenemos

$$\begin{aligned}T_3 &= A_3\left[X^3 - X_*^3 - X_*^2X + 2\left(X_*XX_* + XX_*^2\right) - 3X^2X_*\right] \\ &= A_3\left[\left(X^2 - X_*^2\right)(X-X_*) - 2(X-X_*)^2X_*\right].\end{aligned} \tag{5.6}$$

Luego,

$$T_3 = \sum_{i=0}^{3-3} A_{3-i}\Bigg[(3-2-i)\left(X^{3-i-1}-X_*^{3-i-1}\right)(X-X_*) \\ -2\left(\sum_{j=1}^{3-i-3}\left(X^{3-i-j-1}-X_*^{3-i-j-1}\right)(X-X_*)\,X_*^j\right)-2\,(X-X_*)^2\,X_*^{3-i-2}\Bigg].$$

Por otro lado, usando la norma matricial inducida, la desigualdad triangular y el **Lema 5.1**, tenemos

$$\begin{aligned}\|T_3\| &\leq \|A_3\|\left[\left\|X^2-X_*^2\right\|\|X-X_*\|+2\|X-X_*\|^2\|X_*\|\right] \\ &\leq \|A_3\|\left[\|X-X_*\|+4\|X_*\|+2\|X_*\|\right]\|X-X_*\|^2.\end{aligned}$$

En consecuencia, (5.4) y (5.5) se cumplen para $m=3$.

2. *Hipótesis inductiva.* Supongamos que (5.4) y (5.5) se cumplen para $m=l-1$; es decir,

$$\begin{aligned}T_{l-1} = \sum_{i=0}^{l-1-3} A_{l-1-i}\Bigg[&(l-1-2-i)\left(X^{l-1-i-1}-X_*^{l-1-i-1}\right)(X-X_*) \\ &-2\left(\sum_{j=1}^{l-1-i-3}\left(X^{l-1-i-j-1}-X_*^{l-1-i-j-1}\right)(X-X_*)\,X_*^j\right) \\ &-2\,(X-X_*)^2\,X_*^{l-1-i-2}\Bigg].\end{aligned} \tag{5.7}$$

$$\begin{aligned}\|T_{l-1}\| \leq \Bigg\{\sum_{i=0}^{l-1-3}\|A_{l-1-j}\|\Bigg[&(l-1-2-i)\Big[(E+4\|X_*\|)(E+3\|X_*\|)^{l-1-i-3} \\ &+\sum_{j=2}^{l-1-i-2}\|X_*\|^j(E+3\|X_*\|)^{l-1-i-j-2}\Big] \\ &+2\Bigg[\|X_*\|^{l-1-i-2}+\sum_{k=1}^{l-1-i-3}\Big[(E+4\|X_*\|)(E+3\|X_*\|)^{l-1-i-k-3} \\ &+\sum_{j=2}^{l-1-i-k-3}\|X_*\|^j(E+3\|X_*\|)^{l-1-i-j-k-3}\Big]\|X_*\|^k\Bigg]\Bigg]\Bigg\}E^2,\end{aligned} \tag{5.8}$$

donde $E=\|X-X_*\|$.

3. *Paso de inducción.* Demostraremos que (5.4) y (5.5) se cumplen para $m = l$.

$$\begin{aligned} T_l =& \sum_{i=0}^{l-3} A_{l-i} \Bigg[(l-2-i)\left(X^{l-i} - X_*^{l-i} - X_*^{l-i-1}X\right) \\ &+ 2\left(\sum_{j=1}^{l-i-1} X_*^{l-i-j-1} X X_*^j\right) - (l-i)X^{l-i-1}X_* \Bigg] \\ =& \underbrace{A_l\left[(l-2)X^l - (l-2)X_*^l - (l-2)X_*^{l-1} + \cdots + 2XX_*^{l-1} - lX^{l-1}X_*\right]}_{C} + T_{l-1} \\ =& C + T_{l-1}. \end{aligned} \tag{5.9}$$

Mediante un proceso análogo al que se hizo en (5.6), el primer término del lado derecho se puede expresar de la siguiente forma

$$\begin{aligned} C =& A_l \Bigg[(l-2)\left[X^{l-1} - X_*^{l-1}\right](X - X_*) - 2\left[\sum_{j=1}^{l-3}\left(X^{(l-1)-j} - X_*^{(l-1)-j}\right)(X - X_*)X^j\right] \\ &- 2(X - X_*)^2 X_*^{l-2}\Bigg]. \end{aligned} \tag{5.10}$$

Usando (5.10) y la hipótesis inductiva (5.7) en (5.9), tenemos

$$\begin{aligned} T_l =& \sum_{i=0}^{l-3} A_{l-i} \Bigg[(l-2-i)\left(X^{l-i-1} - X_*^{l-i-1}\right)(X - X_*) \\ &- 2\left(\sum_{j=1}^{l-i-3}\left(X^{l-i-j-1} - X_*^{l-i-j-1}\right)(X - X_*)X_*^j\right) - 2(X - X_*)^2 X_*^{l-i-2}\Bigg]. \end{aligned}$$

Por otro lado, usando la norma matricial inducida, la desigualdad triangular en (5.9) y teniendo en cuenta (5.10), tenemos

$$\|T_l\| \leq \|C\| + \|T_{l-1}\|.$$

Aplicando la desigualdad triangular, el **Lema 5.1** y operaciones algebraicas al

primer término del lado derecho de la desigualdad anterior, obtenemos

$$\begin{aligned}\|C\| \leq \|A_l\| \Bigg[& (l-2)\Big[(E+4\|X_*\|)(E++3\|X_*\|)^{l-3} \\ & + \sum_{j=2}^{l-2} \|X_*\|^j (E+3\|X_*\|)^{l-j-2}\Big] \\ & + 2\Bigg[\|X_*\|^{l-2} + \sum_{k=1}^{l-3}\Big[(E+4\|X_*\|)(E+3\|X_*\|)^{l-k-3} \\ & + \sum_{j=2}^{l-k-3} \|X_*\|^j (E+3\|X_*\|)^{l-j-k-3}\Big]\|X_*\|^k\Bigg]\Bigg] E^2. \end{aligned} \tag{5.11}$$

Usando (5.11), la hipótesis inductiva (5.8) y realizando operaciones algebraicas, tenemos

$$\begin{aligned}\|T_l\| \leq \Bigg\{ & \sum_{i=0}^{l-3} \|A_{l-j}\| \Bigg[(l-2-i)\Big[(E+4\|X_*\|)(E+3\|X_*\|)^{l-i-3} \\ & + \sum_{j=2}^{l-i-2} \|X_*\|^j (E+3\|X_*\|)^{l-i-j-2}\Big] \\ & + 2\Bigg[\|X_*\|^{l-i-2} + \sum_{k=1}^{l-i-3}\Big[(E+4\|X_*\|)(E+3\|X_*\|)^{l-i-k-3} \\ & + \sum_{j=2}^{l-i-k-3} \|X_*\|^j (E+3\|X_*\|)^{l-i-j-k-3}\Big]\|X_*\|^k\Bigg]\Bigg]\Bigg\} E^2. \end{aligned}$$

Por lo tanto, (5.5) se cumple para $m = l$ y se completa la demostración. ■

Previo al siguiente lema, recordamos que para una función polinómica matricial P de grado m, tenemos que $P(X+S) = P(X) + L_X(S) + R_m(S)$, con

$$R_m(S) = \sum_{i=0}^{m-2} \left[\sum_{j=2}^{m-i} A_{m-i}\, \Phi_{X\,S}^{m-i}\,[m-i-j] \right],$$

donde la función $\Phi_{X\,S}$ dada por la **Definición 4.2**, es la suma de los productos de todas las permutaciones repetidas de X y S, en particular, $\Phi_{X\,S}^{k}\,[n]$ es la suma de los productos de todas las permutaciones de X y S en las que X aparece n veces y S, $k-n$ veces, con $0 \leq n \leq k$.

El siguiente resultado da una cota para $\|R_m(S)\|$ que usaremos en la pruebas de convergencia del algoritmo cuasi-Newton que proponemos.

Lema 5.4. *Sean* $P\colon \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ *una función polinómica matricial de grado* $m \geq 2$, $X, X_* \in \mathbb{C}^{n\times n}$ *y* $R_m(S) = \sum_{i=0}^{m-2} \left[\sum_{j=2}^{m-i} A_{m-i}\, \Phi^{m-i}_{X_* S}[m-i-j]\right]$ *donde* $S = X - X_*$. *Entonces*

$$\|R_m(S)\| \leq \left\{\sum_{i=0}^{m-3} \|A_{m-i}\| \left[\sum_{w=0}^{m-i-2} \binom{m-i}{w} \|X_*\|^w \|S\|^{m-i-w-2}\right] + \|A_2\|\right\} \|S\|^2 . \quad (5.12)$$

Demostración. La demostración la haremos por inducción sobre m.

1. Para $m = 2$, tenemos $R_2(S) = A_2\left[\Phi^2_{X_* S}[0]\right] = A_2\left[S^2\right]$. Usando propiedades de la norma inducida, tenemos

   $$\|R_2(S)\| \leq \|A_2\| \|S\|^2 .$$

   Por lo tanto, (5.12) se cumple para $m = 2$.

2. *Hipótesis inductiva.* Supongamos que (5.12) se cumple para $m = l - 1$; es decir,

   $$\|R_{l-1}(S)\| \leq \left\{\sum_{i=0}^{l-4} \|A_{l-1-i}\| \left[\sum_{w=0}^{l-1-i-2} \binom{l-1-i}{w} \|X_*\|^w \|S\|^{l-1-i-w-2}\right] + \|A_2\|\right\} \|S\|^2 .$$

3. *Paso de inducción.* Demostraremos que (5.12) se cumple para $m = l$.

$$\begin{aligned} R_l(S) &= \sum_{i=0}^{l-2} \left[\sum_{j=2}^{l-i} A_{l-i}\, \Phi^{l-i}_{X_* S}[l-i-j]\right] \\ &= A_l\left[\Phi^l_{X_* S}[l-2] + \cdots + \Phi^l_{X_* S}[0]\right] + \sum_{i=1}^{l-2} \left[\sum_{j=2}^{l-i} A_{l-i}\, \Phi^{l-i}_{X_* S}[l-i-j]\right] . \quad (5.13) \end{aligned}$$

Podemos observar que el segundo término de la (5.13) mediante un cambio en los índices de las sumatorias y en término general, es equivalente a la siguiente igualdad

$$\begin{aligned} \sum_{i=1}^{l-2} \left[\sum_{j=2}^{l-i} A_{l-i}\, \Phi^{l-i}_{X_* S}[l-i-j]\right] &= \sum_{i=0}^{l-1-2} \left[\sum_{j=2}^{l-1-i} A_{l-1-i}\, \Phi^{l-1-i}_{X_* S}[l-1-i-j]\right] \\ &= R_{l-1}(S), \quad (5.14) \end{aligned}$$

luego reemplazando (5.14) en (5.13) tenemos

$$R_l(S) = \underbrace{A_l\left[\Phi^l_{X_*S}[l-2] + \Phi^l_{X_*S}[l-3] + \cdots + \Phi^l_{X_*S}[0]\right]}_{D} + R_{l-1}(S) = D + R_{l-1}(S).$$

Por otro lado, usando la norma matricial inducida y la desigualdad triangular, obtenemos

$$\|R_l(S)\| \leq \|D\| + \|R_{l-1}(S)\|. \tag{5.15}$$

A partir de la definición de Φ_{XS} (**Definición 4.2**) y después de algunas manipulaciones algebraicas aplicadas al primer término del lado derecho de (5.15), obtenemos

$$\begin{aligned}
\|D\| &\leq \|A_l\| \left[\left\|\Phi^l_{X_*S}[l-2]\right\| + \left\|\Phi^l_{X_*S}[l-3]\right\| + \cdots + \left\|\Phi^l_{X_*S}[0]\right\|\right] \\
&\leq \|A_l\| \left[\binom{l}{l-2}\|X_*\|^{l-2}\|S\|^2 + \binom{l}{l-3}\|X_*\|^{l-3}\|S\|^3 + \cdots + \binom{l}{0}\|S\|^l\right] \\
&= \|A_l\| \left[\binom{l}{0}\|S\|^l + \cdots + \binom{l}{l-3}\|X_*\|^{l-3}\|S\|^3 + \binom{l}{l-2}\|X_*\|^{l-2}\|S\|^2\right] \\
&= \|A_l\| \sum_{w=0}^{l-2}\binom{l}{w}\|X_*\|^w\|S\|^{l-w} \\
&= \left\{\|A_l\| \sum_{w=0}^{l-2}\binom{l}{w}\|X_*\|^w\|S\|^{l-w-2}\right\}\|S\|^2.
\end{aligned} \tag{5.16}$$

Finalmente, usando (5.16), la hipótesis inductiva en (5.15) y realizando operaciones algebraicas, obtenemos

$$\begin{aligned}
\|R_l(S)\| &\leq \left\{\|A_l\| \sum_{w=0}^{l-2}\binom{l}{w}\|X_*\|^w\|S\|^{l-w-2}\right\}\|S\|^2 \\
&\quad + \left\{\sum_{i=0}^{l-4}\|A_{l-1-i}\|\left[\sum_{w=0}^{l-1-i-2}\binom{l-1-i}{w}\|X_*\|^w\|S\|^{l-1-i-w-2}\right] + \|A_2\|\right\}\|S\|^2 \\
&\leq \left\{\sum_{i=0}^{l-3}\|A_{l-i}\|\left[\sum_{w=0}^{l-i-2}\binom{l-i}{w}\|X_*\|^w\|S\|^{l-i-w-2}\right] + \|A_2\|\right\}\|S\|^2.
\end{aligned}$$

Por lo tanto, (5.12) se cumple para $m = l$. Esto completa la demostración. ■

El siguiente teorema garantiza que cuando el parámetro θ (hipótesis **H3**) pertenece al intervalo $[0,1]$, la iteración cuasi-Newton (5.1) está bien definida y converge linealmente a un solvente X_*.

Teorema 5.2. *Sea* $P\colon \mathbb{C}^{n\times n} \longrightarrow \mathbb{C}^{n\times n}$ *una función polinómica matricial de grado* $m \geq 2$ *que satisface las hipótesis* **H1** *a* **H3.** *Existen constantes* $\epsilon_3 > 0$ *y* $\eta \in [0,1)$ *tal que, si* $\|X_0 - X_*\| < \epsilon_3$ *entonces para todo* $k = 0, 1, 2, \ldots$

$$\mathcal{B}_m(X_k), \quad \textit{es no singular.} \tag{5.17}$$

Además,

$$\|X_{k+1} - X_*\| \leq \eta \|X_k - X_*\|, \ \textit{para} \ \theta \in [0,1]. \tag{5.18}$$

Demostración. Sea $P(X) = A_m X^m + \cdots + A_1 X + A_0$, con $m \geq 2$. La demostración la haremos por inducción sobre k. Sea $\eta \in [0,1)$ cualquiera. Sean $\gamma \leq \dfrac{\eta}{8\beta}$ y ϵ_0 el correspondiente a γ dado por **H3**. Definimos

$$\epsilon_3 = \min\{1, \epsilon_0, \epsilon_1, \epsilon_2\}, \tag{5.19}$$

donde ϵ_1 está dado por el **Teorema 5.1** (ver (5.3)) y $\epsilon_2 > 0$ [2] tal que

$$\begin{aligned}
&\left\{ \sum_{i=0}^{m-3} \|A_{m-j}\| \left[(m-2-i) \left[(\epsilon_2 + 4\|X_*\|)(\epsilon_2 + 3\|X_*\|)^{m-i-3} \right.\right.\right. \\
&\qquad \left. + \sum_{j=2}^{m-i-2} \|X_*\|^j (\epsilon_2 + 3\|X_*\|)^{m-i-j-2} \right] \\
&\qquad +2 \left[\|X_*\|^{m-i-2} + \sum_{k=1}^{m-i-3} \left[(\epsilon_2 + 4\|X_*\|)(\epsilon_2 + 3\|X_*\|)^{m-i-k-3} \right.\right. \\
&\qquad \left.\left. + \sum_{j=2}^{m-i-k-3} \|X_*\|^j (\epsilon_2 + 3\|X_*\|)^{m-i-j-k-3} \right] \|X_*\|^k \right] \\
&\qquad \left.\left. + \sum_{w=0}^{m-i-2} \binom{m-i}{w} \|X_*\|^w \epsilon_2^{m-i-w-2} \right] \right] + \|A_2\| \Bigg\} \epsilon_2 \leq \frac{3\eta}{8\beta}.
\end{aligned} \tag{5.20}$$

[2] Observe que (5.20), puede ser expresado por $g(\epsilon) \leq \frac{3\eta}{8\beta}$, así la existencia de $\epsilon_2 > 0$ se garantiza ya que $g(0) \leq \frac{3\eta}{8\beta}$ y g es una función creciente para $\epsilon_2 > 0$ ($g'(\epsilon_2) > 0$).

1. Para $k = 0$. Por hipótesis tenemos que $\|X_0 - X_*\| < \epsilon_3$ y teniendo en cuenta (5.19), $\epsilon_3 \leq \epsilon_1$, se tiene que $\|X_0 - X_*\| < \epsilon_1$. El **Teorema 5.1** garantiza que la matriz $\mathcal{B}_m(X_0)$ es no singular y además, $\|(\mathcal{B}_m(X_0))^{-1}\| \leq 2\beta$, con lo cual se prueba (5.17). En consecuencia, X_1 está bien definido y de (5.1) se tiene

$$\begin{aligned} X_1 &= \mathcal{B}_m(X_0)^{-1}\left[\mathcal{B}_m(X_0)X_0 - P(X_0)\right] \\ &= \mathcal{B}_m(X_0)^{-1}\left[(m-1)A_m X_0^m + \cdots + A_2 X_0^2 - A_0\right]. \end{aligned}$$

Así,

$$\begin{aligned} X_1 - X_* &= (\mathcal{B}_m(X_0))^{-1}\left[(m-1)A_m X_0^m + \cdots + A_2 X_0^2 - A_0 - \mathcal{B}_m(X_0)X_*\right] \\ &= (\mathcal{B}_m(X_0))^{-1}\left[\sum_{i=0}^{m-2}(m-1-i)A_{m-i}X_0^{m-i} - A_0 - \mathcal{B}_m(X_0)X_*\right]. \end{aligned}$$

Reemplazando el segundo factor de la desigualdad anterior usando **Lema 5.2,** tenemos

$$\begin{aligned} X_1 - X_* = (\mathcal{B}_m(X_0))^{-1}&\left[[P(X_0) - P(X_*) - L_{X_*}(X_0 - X_*)]\right. \\ &\left. + [\mathcal{B}_m(X_*)(X_0 - X_*) - L_{X_*}(X_0 - X_*)] + T_m\right]. \end{aligned}$$

Como P es Fréchet diferenciable en X_*, existe el operador L_{X_*} que para toda $S \in \mathbb{C}^{n\times n}$ satisface $P(X_* + S) = P(X_*) + L_{X_*}(S) + R_m(S)$. Usando la hipótesis **H3**, la desigualdad de triangular y (5.19) (en particular $\|X_0 - X_*\| < 1$) entonces $\|X_0 - X_*\|^{1+\theta} < \|X_0 - X_*\|$, tenemos

$$\begin{aligned} \|X_1 - X_*\| &\leq \left\|\mathcal{B}_m(X_0)^{-1}\right\|\left[\|P(X_0) - P(X_*) - L_{X_*}(X_0 - X_*)\|\right. \\ &\qquad \left. + \|\mathcal{B}_m(X_0)(X_0 - X_*) - L_{X_*}(X_0 - X_*)\| + \|T_m\|\right] \\ &\leq \left\|\mathcal{B}_m(X_0)^{-1}\right\|\left[\|R_m(X_0 - X_*)\| + \gamma\|X_0 - X_*\|^{1+\theta} + \|T_m\|\right] \\ &\leq \left\|\mathcal{B}_m(X_0)^{-1}\right\|\left[\|T_m\| + \|R_m(X_0 - X_*)\| + \gamma\|X_0 - X_*\|\right]. \end{aligned}$$

Por los **Lemas 5.3** y **5.4**, obtenemos

$$\begin{aligned}\|X_1 - X_*\| \leq 2\beta\Bigg[\Bigg\{\sum_{i=0}^{m-3}\|A_{m-j}\|\Bigg[(m-2-i)\Big[(E_0+4\|X_*\|)(E_0+3\|X_*\|)^{m-i-3} \\ + \sum_{j=2}^{m-i-2}\|X_*\|^j(E_0+3\|X_*\|)^{m-i-j-2}\Big] \\ + 2\Bigg[\|X_*\|^{m-i-2} + \sum_{k=1}^{m-i-3}\Big[(E_0+4\|X_*\|)(E_0+3\|X_*\|)^{m-i-k-3} \\ + \sum_{j=2}^{m-i-k-3}\|X_*\|^j(E_0+3\|X_*\|)^{m-i-j-k-3}\Big]\|X_*\|^k\Bigg]\Bigg]\Bigg\}E_0 \\ + \Bigg\{\sum_{i=0}^{m-3}\|A_{m-i}\|\Bigg[\sum_{w=0}^{m-i-2}\binom{m-i}{w}\|X_*\|^w E_0^{m-i-w-2}\Bigg] \\ + \|A_2\|\Bigg\}E_0 + \gamma\Bigg]E_0,\end{aligned}$$

donde $E_0 = \|X_0 - X_*\|$. Usando $\|X_0 - X_*\| < \epsilon_3$ y $\epsilon_3 \leq \epsilon_2$, en la desigualdad anterior, tenemos

$$\begin{aligned}\|X_1 - X_*\| < 2\beta\Bigg[\Bigg\{\sum_{i=0}^{m-3}\|A_{m-j}\|\Bigg[(m-2-i)\Big[(\epsilon_3+4\|X_*\|)(\epsilon_3+3\|X_*\|)^{m-i-3} \\ + \sum_{j=2}^{m-i-2}\|X_*\|^j(\epsilon_3+3\|X_*\|)^{m-i-j-2}\Big] \\ + 2\Big[\|X_*\|^{m-i-2} + \sum_{k=1}^{m-i-3}\Big[(\epsilon_3+4\|X_*\|)(\epsilon_3+3\|X_*\|)^{m-i-k-3} \\ + \sum_{j=2}^{m-i-k-3}\|X_*\|^j(\epsilon_3+3\|X_*\|)^{m-i-j-k-3}\Big]\|X_*\|^k\Big] \\ + \sum_{w=0}^{m-i-2}\binom{m-i}{w}\|X_*\|^w\epsilon_3^{m-i-w-2}\Bigg] + \|A_2\|\Bigg\}\epsilon_3 + \gamma\Bigg]\|X_0 - X_*\|,\end{aligned}$$

entonces

$$\|X_1 - X_*\| \leq 2\beta\left[\frac{\eta}{8\beta} + \frac{3\eta}{8\beta}\right]\|X_0 - X_*\| = \eta\|X_0 - X_*\|,$$

lo cual prueba que $\|X_1 - X_*\| < \epsilon_3$.

2. *Hipótesis inductiva.* Supongamos que (5.17) y (5.18) se satisfacen para $k = l-1$; es decir

$$\mathcal{B}_m(X_{l-1}) \text{ es no singular y} \tag{5.21}$$

$$\|X_l - X_*\| \leq \eta \|X_{l-1} - X_*\|. \tag{5.22}$$

3. *Paso de inducción.* Probemos que (5.17) y (5.18), se cumplen para $k = l$.

   Por (5.21), se tiene que X_l está bien definido. Además de (5.22) y teniendo en cuenta que $\eta^l < 1$, puesto que $0 \leq \eta < 1$, tenemos

$$\|X_l - X_*\| \leq \eta^l \|X_0 - X_*\| < \epsilon_3.$$

   Luego, $\|X_l - X_*\| < \epsilon_1$. Por el **Teorema 5.1,** la matriz $B_m(X_l)$ es no singular y $\|\mathcal{B}_m(X_{l-1})^{-1}\| \leq 2\beta$. Por otro lado, teniendo en cuenta que X_l está bien definido y mediante un proceso análogo al utilizado para acotar $\|X_1 - X_*\|$, se demuestra que

$$\|X_{l+1} - X_*\| \leq \eta \|X_l - X_*\|.$$

   Esto completa la demostración. ■

El siguiente teorema garantiza que la iteración (5.1) está bien definida y converge superlinealmente a un solvente X_*, si $\theta \in (0, 1)$ y cuadráticamente, si $\theta = 1$.

Teorema 5.3. *Sea* $P\colon \mathbb{C}^{n\times n} \longrightarrow \mathbb{C}^{n\times n}$ *una función polinómica matricial de grado* $m \geq 2$ *que satisface las hipótesis* **H1** *a* **H3.** *Existen constantes positivas* ϵ_4 *y* μ *tales que, si* $\|X_0 - X_*\| < \epsilon_4$ *entonces para todo* $k = 0, 1, 2, \ldots$

$$\mathcal{B}_m(X_k), \quad \textit{es no singular, y} \tag{5.23}$$

$$\|X_{k+1} - X_*\| < \mu \|X_k - X_*\|^{1+\theta}, \quad \textit{para } 0 < \theta \leq 1. \tag{5.24}$$

Demostración. Sea $P(X) = A_m X^m + \cdots + A_1 X + A_0$, con $m \geq 2$. La demostración la haremos por inducción sobre k.

Sea γ y el correspondiente ϵ_0 (ver **H3**) tales que $\gamma \leq \dfrac{1}{(12\beta)^{1-\theta}}$. Definimos

$$\epsilon_4 = \text{mín}\left\{1, \epsilon_0, \epsilon_1, \frac{\epsilon_2}{3}\right\}, \tag{5.25}$$

donde ϵ_1 está dado por el **Teorema 5.1** (ver (5.3)) y $\epsilon_2 > 0$ [3] es tal que

$$\left\{\sum_{i=0}^{m-3}\|A_{m-j}\|\left[(m-2-i)\Big[(\epsilon_2+4\|X_*\|)(\epsilon_2+3\|X_*\|)^{m-i-3}\right.\right.$$
$$+\sum_{j=2}^{m-i-2}\|X_*\|^j(\epsilon_2+3\|X_*\|)^{m-i-j-2}\Big]$$
$$+2\left[\|X_*\|^{m-i-2}+\sum_{k=1}^{m-i-3}\Big[(\epsilon_2+4\|X_*\|)(\epsilon_2+3\|X_*\|)^{m-i-k-3}\right.$$
$$\left.\left.+\sum_{j=2}^{m-i-k-3}\|X_*\|^j(\epsilon_2+3\|X_*\|)^{m-i-j-k-3}\Big]\|X_*\|^k\right]$$
$$\left.+\sum_{w=0}^{m-i-2}\binom{m-i}{w}\|X_*\|^w\epsilon_2^{m-i-w-2}\right]+\|A_2\|\Bigg\}\epsilon_2\leq\frac{1}{4\beta}. \tag{5.26}$$

Por otro lado, observamos que $\epsilon_4 < \epsilon_3$, donde ϵ_3 viene dado por (5.19). Esta desigualdad garantiza por **Teorema 5.2** que la sucesión $\{X_k\}$ converge al menos linealmente a X_* y $\|X_k - X_*\| < \epsilon_4$, para todo k.

1. Para $k = 0$. Por hipótesis, $\|X_0 - X_*\| < \epsilon_4$, y por (5.25) $\epsilon_4 \leq \epsilon_1$ entonces $\|X_0 - X_*\| < \epsilon_1$. El **Teorema 5.1** garantiza que la matriz $B(X_0)$ es no singular y además $\|(B(X_0))^{-1}\| \leq 2\beta$, con lo cual se prueba (5.23). En consecuencia, X_1 está bien definido y de (5.1), se tiene

$$\begin{aligned} X_1 &= \mathcal{B}_m(X_0)^{-1}\left[B(X_0)X_0 - P(X_0)\right] \\ &= \mathcal{B}_m(X_0)^{-1}\left[(m-1)A_mX_0^m + \cdots + A_2X_0^2 - A_0\right]. \end{aligned}$$

Así,

$$\begin{aligned} X_1 - X_* &= \mathcal{B}_m(X_0)^{-1}\left[(m-1)A_mX_0^m + \cdots + A_2X_0^2 - A_0\right] - X_* \\ &= \mathcal{B}_m(X_0)^{-1}\left[\sum_{i=0}^{m-2}(m-1-i)A_{m-i}X_0^{m-i} - A_0 - \mathcal{B}_m(X_0)X_*\right]. \end{aligned}$$

Por el **Lema 5.2**, tenemos que el segundo factor de la ecuación anterior se puede expresar de la siguiente manera,

$$\begin{aligned} X_1 - X_* &= \mathcal{B}_m(X_0)^{-1}\left[[P(X_0) - P(X_*) - L_{X_*}(X_0 - X_*)]\right. \\ &\quad \left. + [\mathcal{B}_m(X_*)(X_0 - X_*) - L_{X_*}(X_0 - X_*)] + T_m\right]. \end{aligned} \tag{5.27}$$

[3] Observe que (5.26), puede ser expresado por $g(\epsilon) \leq \frac{1}{4\beta}$, así la existencia de $\epsilon_2 > 0$ se garantiza ya que $g(0) \leq \frac{1}{4\beta}$ y g es una función creciente para $\epsilon_2 > 0$ ($g'(\epsilon) > 0$).

Como P es Fréchet diferenciable en X_*, existe el operador L_{X_*} tal que para toda $S \in \mathbb{C}^{n\times n}$

$$P(X_* + S) = P(X_*) + L_{X_*}(S) + R_m(S). \tag{5.28}$$

Luego, usando (5.27), (5.28), **H3** y la desigualdad triangular, tenemos

$$\begin{aligned}\|X_1 - X_*\| &\leq \left\|\mathcal{B}_m(X_0)^{-1}\right\| [\|P(X_0) - P(X_*) - L_{X_*}(X_0 - X_*)\| \\ &\quad + \|\mathcal{B}_m(X_0)(X_0 - X_*) - L_{X_*}(X_0 - X_*)\| + \|T_m\|] \\ &\leq \left\|\mathcal{B}_m(X_0)^{-1}\right\| \left[\|R_m(X_0 - X_*)\| + \gamma \|X_0 - X_*\|^{1+\theta} + \|T_m\|\right] \\ &= \left\|\mathcal{B}_m(X_0)^{-1}\right\| \left[\|T_m\| + \|R_m(X_0 - X_*)\| + \gamma \|X_0 - X_*\|^{1+\theta}\right].\end{aligned}$$

Usando los **Lemas 5.3** y **5.4**, obtenemos

$$\begin{aligned}\|X_1 - X_*\| \leq 2\beta &\left[\left\{\sum_{i=0}^{m-3} \|A_{m-j}\| \left[(m-2-i)\Big[(E_0 + 4\|X_*\|)(E_0\right.\right.\right. \\ &\left.\left.\left. +3\|X_*\|)^{m-i-3} + \sum_{j=2}^{m-i-2} \|X_*\|^j (E_0 + 3\|X_*\|)^{m-i-j-2}\Big]\right.\right. \\ &+2\left[\|X_*\|^{m-i-2} + \sum_{k=1}^{m-i-3}\Big[(E_0+4\|X_*\|)(E_0 + 3\|X_*\|)^{m-i-k-3}\right. \\ &\left.\left.\left. + \sum_{j=2}^{m-i-k-3} \|X_*\|^j (E_0 + 3\|X_*\|)^{m-i-j-k-3}\Big]\|X_*\|^k\right]\right]\right\} E_0 \\ &+\left\{\sum_{i=0}^{m-3} \|A_{m-i}\| \left[\sum_{w=0}^{m-i-2} \binom{m-i}{w} \|X_*\|^w E_0^{m-i-w-2}\right]\right. \\ &\left.+ \|A_2\|\right\} E_0 + \gamma \|X_0 - X_*\|^\theta \Bigg] E_0,\end{aligned} \tag{5.29}$$

donde $E_0 = \|X_0 - X_*\|$. Por otro lado, dado que $\|X_1 - X_*\| < \epsilon_4$. Ahora, factorizando $\|X_0 - X_*\|^{1+\theta}$ en (5.29) y usando (5.25), tenemos

$$\begin{aligned}\|X_1 - X_*\| &< 2\beta \left[\epsilon_2^{1-\theta} + \gamma\right] \|X_0 - X_*\|^{1+\theta} \\ &< 2\beta \left[\left(\frac{1}{12\beta}\right)^{1-\theta} + \frac{1}{(12\beta)^{1-\theta}}\right] \|X_0 - X_*\|^{1+\theta} \\ &\leq \frac{1}{3}(12\beta)^\theta \|X_0 - X_*\|^{1+\theta} = \mu \|X_0 - X_*\|^{1+\theta},\end{aligned}$$

donde $\mu = \dfrac{1}{3}(12\beta)^\theta > 0$.

2. *Hipótesis inductiva.* Supongamos que (5.23) y (5.24), se satisfacen para $k = l-1$; esto es

$$\begin{aligned} &\mathcal{B}_m(X_{l-1}) \text{ es no singular y} \\ &\|X_l - X_*\| < \mu \|X_{l-1} - X_*\|^{1+\theta}. \end{aligned} \tag{5.30}$$

3. *Paso de inducción:* Probemos que (5.23) y (5.24), se cumplen para $k = l$. Por (5.30), X_l está bien definido y de $\|X_l - X_*\| < \epsilon_4$. Así, $\|X_l - X_*\| < \epsilon_2$. Usando el **Teorema 5.1**, tenemos que la matriz $B(X_l)$ es no singular y además $\|(B_m(X_l))^{-1}\| \leq 2\beta$.

   Por otro lado, teniendo en cuenta que X_l está bien definido y mediante un proceso análogo al utilizado para acotar $\|X_1 - X_*\|$, se demuestra que,

$$\|X_{l+1} - X_*\| < \mu \|X_l - X_*\|^{1+\theta},$$

   donde $\mu = \dfrac{1}{3}(12\beta)^{\theta} > 0$. Lo cual completa la demostración. ■

Antes de finalizar esta sección, presentamos un resultado que garantiza que las iteraciones de *Newton* (2.3) y cuasi-*Newton* (5.1) coinciden bajo ciertas condiciones de conmutatividad. Para evitar confusiones cambiaremos la notación en la iteración de *Newton* así,

$$\begin{aligned} L_{Y_k}(F_k) &= -P(Y_k) \\ Y_{k+1} &= Y_k + F_k. \end{aligned}$$

El siguiente lema extiende el resultado del **Lema 3.3** en [31] a funciones polinomiales matriciales de grado $m \geq 2$.

Lema 5.5. *Sea* $P\colon \mathbb{C}^{n\times n} \longrightarrow \mathbb{C}^{n\times n}$ *una función polinómica matricial de grado* $m \geq 2$. *Consideremos la iteraciones* (2.3) *y* (5.1) *que están dadas respectivamente por:*

$$\begin{aligned} L_{Y_k}(F_k) &= -P(Y_k) \\ Y_{k+1} &= Y_k + F_k. \end{aligned} \qquad\qquad \begin{aligned} \mathcal{B}_m(X_k)S_k &= -P(X_k) \\ X_{k+1} &= X_k + S_k. \end{aligned}$$

Si la matriz inicial $Y_0 = X_0$ *conmuta con las matrices* $A_0, \ldots, A_m$, *las matrices* $A_0, \ldots, A_m$, *conmutan entre si, y la iteración de Newton* (2.3) *está bien definida, entonces la matriz* Y_k *conmuta con las matrices* $A_0, \ldots, A_m$, *y* $Y_k = X_k$, *para todo* $k =, 1, 2, \ldots$

Demostración. Probaremos por inducción sobre k en forma conjunta que

$$\begin{aligned} F_k &= S_k, \\ Y_k &= X_k, \quad k = 1, 2, \ldots \end{aligned}$$

Primero mostramos que $F_0 = S_0$ y $Y_1 = X_1$. Por hipótesis tenemos que $A_j Y_0 = Y_0 A_j$ para $j = 0, 1, \ldots, m$. Como consecuencia directa de lo anterior,

$$Y_0\, \mathcal{B}_m(Y_0) = \mathcal{B}_m(Y_0)\, Y_0. \tag{5.31}$$

Recordemos que la iteración cuasi-Newton (5.1) está bien definida (**Teorema 5.1**) entonces la matriz $\mathcal{B}_m(Y_0)^{-1}$ existe. De (5.31), para $j = 0, 1, \ldots, m$, tenemos

$$\mathcal{B}_m(Y_0)^{-1} Y_0 = Y_0 \mathcal{B}_m(Y_0)^{-1}. \tag{5.32}$$

De la iteración cuasi-Newton y la hipótesis ($X_0 = Y_0$), tenemos

$$S_0 = -\mathcal{B}_m(X_0)^{-1}\, P(X_0) = -\mathcal{B}_m(Y_0)^{-1}\, P(Y_0).$$

Usando (5.32), tenemos

$$\begin{aligned} S_0 Y_0 &= -\mathcal{B}_m(Y_0)^{-1} P(Y_0) Y_0 = -\mathcal{B}_m(Y_0)^{-1} Y_0 P(Y_0) \\ &= -Y_0 \left[\mathcal{B}_m(Y_0)^{-1} P(Y_0)\right] = Y_0 S_0 \end{aligned} \tag{5.33}$$

Usando (5.33) y realizando operaciones algebraicas, tenemos

$$\begin{aligned} L_{Y_0}(S_0) &= \left[A_m Y_0^{m-1} + \cdots + A_1\right] S_0 + \left[A_m Y_0^{m-2} + \cdots + A_2\right] Y_0 S_0 \\ &\quad + \cdots + \left[A_m Y_0 + A_{m-1}\right] S_0^{m-2} F_0 + A_m Y_0^{m-1} S_0 \\ &= \left[A_m Y_0^{m-1} + \cdots + A_1\right] S_0 + \left[A_m Y_0^{m-1} + \cdots + A_2 Y_0\right] S_0 \\ &\quad + \cdots + \left[A_m Y_0^{m-1} + A_{m-1} Y_0^{m-2}\right] F_0 + A_m Y_0^{m-1} S_0 \\ &= \left[m A_m Y_0^{m-1} + (m-1) A_{m-1} Y_0^{m-2} + \cdots + 2A_2 Y_0 + A_1\right] S_0 \\ &= \left[\mathcal{B}_m(Y_0)\right] S_0 = \mathcal{B}_m(Y_0) \left[-\mathcal{B}_m(Y_0)^{-1} P(Y_0)\right] = -P(Y_0), \end{aligned}$$

entonces S_0 es una solución de (2.3). Luego, $S_0 = F_0$.

En consecuencia, $Y_1 = Y_0 + F_0 = X_0 + S_0 = X_1$.

La prueba del paso de inducción se realiza en forma idéntica. . ■

Observación 3. *Para una matriz inicial X_0 que satisfaga las hipótesis del* **Lema 5.5** *y que se encuentre en la región de convergencia dada por el* **Teorema 5.3** *tendríamos que la sucesión generada por (5.1) converge cuadráticamente a un solvente de X_*. En este caso, $L_{X_*}(S) = \mathcal{B}_m(X_*)S$ y la hipótesis* **H3** *se cumple para $\theta = 1$.*

5.2. Experimentación numérica

En esta sección, analizamos numéricamente el comportamiento local del algoritmo propuesto (5.1), el cual llamaremos *cuasi-Newton*. Para ello, comparamos con el algoritmo tipo *Newton* propuesto en [24]. A continuación, presentamos los dos algoritmos mencionados.

Algoritmo 4 *Cuasi-Newton*

Entrada: $A_0,\ A_1, \ldots,\ A_m,\ X_0,\ \epsilon = 10^{-5},\ N_{\text{máx}}$ y $k = 0$.
Salida: Matriz aproximación a X_*.
1: **mientras** $k \leq N_{max},\ Res(X_k) \geq \epsilon$ **hacer**
2: Resolver para S_k, el sistema matricial: $\mathcal{B}_m(X_k)S_k = -P(X_k)$.
3: Actualizar $X_{k+1} = X_k + S_k$ y $k = k + 1$.
4: **fin mientras**
5: **Salida** X_*

Observación 4. *En el Paso 2,* $\mathcal{B}_m(X_k)S_k$ *es la aproximación al operador derivada de Fréchet* $L_{X_k}(S_k)$.

Algoritmo 5 *Newton*

Entrada: $A_0,\ A_1, \ldots,\ A_m,\ X_0,\ \epsilon = 10^{-5},\ N_{\text{máx}}$ y $k = 0$.
Salida: Matriz aproximación a X_*.
1: **mientras** $k \leq N_{max},\ Res(X_k) \geq \epsilon$ **hacer**
2: Resolver para S_k, el sistema matricial: $L_{X_k}(S_k) = -P(X_k)$.
3: Actualizar $X_{k+1} = X_k + S_k$ y $k = k + 1$.
4: **fin mientras**
5: **Salida** X_*

Observación 5. *En el Paso 2, para encontrar* S_k*, procedemos de manera análoga como en* [24, 45] *usando la descomposición de Schur* (**Teorema 2.1**) *y la noción del producto Kronecker* (**Definición 2.2**) *y de la función vec* (**Definición 2.3**) *como en* [25].

Observación 6. *En los dos algoritmos, la expresión* $Res(X_k)$ *es una medida relativa del error en* $\|P(X_k)\|_F$ *definida en el Capítulo 3.*

Para la experimentación numérica, consideramos los **Problemas 2, 3, 8, 9** y **15** descritos en el Capítulo 3, en los cuales es necesario resolver ecuaciones matriciales de grado seis, cuatro, tres, cinco y cinco, respectivamente.

Para cada problema, comparamos el desempeño local de los dos algoritmos en cuanto a número de iteraciones y tiempo CPU de ejecución. Además, analizamos numéricamente el orden de convergencia del algoritmo propuesto con cada uno de los problemas.

5.2.1. Desempeño local

En esta sección presentamos el análisis del desempeño local del algoritmo propuesto (*cuasi-Newton*). Para ello, comparamos con el algoritmo de *Newton* respecto a número de iteraciones y tiempo CPU de ejecución, para cada problema. Presentamos los resultados obtenidos en tablas que contienen la siguiente información: matriz inicial (X_0); número de iteraciones (N); valor de *Res*; tiempo CPU en segundos (*tiempo*).

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*time*	N	*Res*	*tiempo*
0	6	3.9092e-07	4.7259e-03	6	3.9092e-07	5.9180e-02
I_n	7	3.9092e-07	5.1980e-03	7	3.9092e-07	6.6406e-02
$10^{-1}I_n$	6	2.8376e-07	5.1460e-03	6	2.8376e-07	5.2603e-02
$10^{-2}I_n$	6	3.8217e-07	3.6159e-03	6	3.8217e-07	5.3347e-02
$10I_n$	13	1.6046e-06	6.7761e-03	13	1.6046e-06	1.1988e-01
10^2I_n	24	9.0956e-07	1.1160e-02	24	9.0956e-07	2.1463e-01

Tabla 5.1: Resultados para el **Problema 2**, $n = 5$.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	6	1.7126e-07	1.1440e-02	6	1.7126e-07	4.4407e+00
I_n	7	1.7126e-07	1.6563e-02	7	1.7126e-07	5.0828e+00
$10^{-1}I_n$	6	1.2432e-07	1.7497e-02	6	1.2432e-07	4.2018e+00
$10^{-2}I_n$	6	1.6743e-07	1.3489e-02	6	1.6743e-07	4.4007e+00
$10I_n$	13	6.9568e-07	2.4403e-02	13	6.9568e-07	8.5198e+00
10^2I_n	24	3.9435e-07	6.2500e-02	24	3.9435e-07	1.7567e+01

Tabla 5.2: Resultados para el **Problema 2**, $n = 50$.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	6	1.2609e-07	3.1250e-01	6	1.2609e-07	3.8104e+01
I_n	7	1.2609e-07	1.8750e-01	7	1.2609e-07	3.9888e+01
$10^{-1}I_n$	6	9.1526e-08	1.7188e-01	6	9.1526e-08	3.6004e+01
$10^{-2}I_n$	6	1.2327e-07	2.8125e-01	6	1.2327e-07	4.1938e+01
$10I_n$	13	5.1184e-07	4.2188e-01	13	5.1184e-07	6.4984e+01
10^2I_n	24	2.9013e-07	1.0156e+00	24	2.9013e-07	2.0761e+02

Tabla 5.3: Resultados para el **Problema 2**, $n = 100$.

En las Tablas 5.1, 5.2 y 5.3, podemos observar el mismo número de iteraciones y los mismos valores de *Res* para ambos algoritmos, lo cual confirma el resultado del **Lema 5.5,** ya que las matrices A_k, $k = 0, 1, \ldots, 5$ conmutan con la matriz X_0. Además, el algoritmo *cuasi-Newton* gasta menos tiempo CPU, lo que se hace evidente cuando aumenta el tamaño del problema; el tiempo del algoritmo *cuasi-Newton* respecto al algoritmo de *Newton* es 6 %, cuando $n = 5$; 0.3 % para $n = 50$ y 0.7 % para $n = 100$.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
I_n	10	2.0143e-06	2.9740e-03	9	3.4518e-09	3.1250e-02
$-24I_n$	13	4.0000e-06	3.1250e-03	13	7.6208e-10	3.3450e-02
$10^{-1}I_n$	23	4.1827e-06	7.5271e-03	20	1.3205e-07	6.2500e-02
$10I_n$	10	5.8076e-06	3.8719e-03	10	2.6873e-09	4.6875e-02
$24I_n$	13	6.9023e-06	4.9021e-03	13	4.3816e-09	3.1250e-02
10^2I_n	18	6.4723e-06	6.1131e-03	18	2.6993e-09	6.2500e-02
10^3I_n	26	6.5330e-06	8.0202e-03	26	2.8315e-09	7.8125e-02

Tabla 5.4: Resultados para el **Problema 3**.

En la Tabla 5.4 presentamos los resultados obtenidos para las diferentes matrices iniciales. Los dos algoritmos convergen al solvente S_1. Los algoritmos presentan un comportamiento muy similar con respecto al número de iteraciones. De otro lado, observamos que el tiempo CPU del algoritmo *cuasi-Newton* es aproximadamente el 10 % del tiempo que emplea el algoritmo de *Newton.*

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	6	2.2005e-07	3.0889e-03	4	9.7266e-08	1.1260e-02
I_n	9	4.7186e-07	3.9980e-03	6	1.2685e-10	3125e-02
$10^{-1}I_n$	6	1.1041e-06	2.9469e-03	4	1.0213e-08	1.185e-02
$-10^{1}I_n$	13	6.7813e-08	4.9469e-03	7	6.0166e-12	3.1250e-02
$-10^{2}I_n$	21	6.1475e-08	1.003e-02	13	6.5070e-11	1.5625e-02

Tabla 5.5: Resultados para el **Problema 8**.

En la Tabla 5.5 podemos observar que el algoritmo de *Newton* tiene un número menor de iteraciones que el algoritmo *cuasi-Newton,* mientras que el algoritmo *cuasi-Newton* gasta un menor tiempo CPU, lo cual se puede evidenciar en el peor caso del algoritmo *cuasi-Newton* (para la matriz inicial $X_0 = -10^2 I_n$), donde el algoritmo *cuasi-Newton* utiliza el 64 % del tiempo del algoritmo de *Newton.*

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	6	5.0743e-06	3.80oe-03	6	5.0743e-06	1.453e-02
I_n	7	4.3894e-16	8.750e-03	7	1.8560e-17	2.084e-02
$10^{2}I_n$	18	2.2987e-06	1.820e-02	18	2.2987e-06	3.490e-02
$10^{-2}I_n$	6	4.8211e-06	4.684e-03	6	4.8211e-06	1.535e-02

Tabla 5.6: Resultados para el **Problema 9**.

De la Tabla 5.6 podemos observar que los dos algoritmos tienen un comportamiento muy similar en número de iteraciones, pero en este caso no se satisface el **Lema 5.5**. Con respecto al tiempo CPU, se observa un mejor desempeño del algoritmo *cuasi-Newton*, que es aproximadamente del 42 % del tiempo que usa el algoritmo de *Newton.* En este problema, observamos que los valores de *Res* para el algoritmo de *cuasi-Newton* son ligeramente menores que los dados por el algoritmo de *Newton*; es decir, el algoritmo *cuasi-Newton* da una mejor aproximación a la solución.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	22	3.8718e-07	9.6231e-03	6	8.1019e-08	2.0456e-02
$10^1 I_n$	58	7.7966e-08	5.6107e-02	24	9.1964e-10	7.8125e-02
$10^{-1} I_n$	18	1.4685e-06	1.2775e-02	6	1.6096e-09	2.3127e-02
$10^{-2} I_n$	16	2.2772e-06	1.1120e-02	6	7.2670e-09	2.0385e-02
$10^{-5} I_n$	19	5.8521e-07	1.1618e-02	6	8.0853e-08	2.2118e-02

Tabla 5.7: Resultados para el **Problema 15**.

Para este problema, el algoritmo *cuasi-Newton* emplea un número de iteraciones que supera las de *Newton* en todos los casos; no obstante, el algoritmo de *Newton* no logra mejorar el tiempo CPU que usa el *cuasi-Newton,* ya que en el peor de los casos, este utiliza solo el 72 % del que emplea *Newton*. Por otro lado, de las Tablas 5.4, 5.5 y 5.7, observamos que para el algoritmo de *Newton,* los valores de *Res* son menores que los del *cuasi-Newton* lo que da una mejor aproximación a la solución.

5.2.2. Orden de convergencia.

En esta sección, analizamos numéricamente el orden de convergencia del algoritmo propuesto. Para el experimento, usamos los cinco problemas descritos anteriormente y definimos la función $f_\theta \colon \mathbb{N}_0 \longrightarrow \mathbb{R}^+ \cup \{0\}$ por

$$f_\theta(k) = \frac{E_{k+1}}{E_k^{1+\theta}}, \quad \text{donde} \quad E_k = \|X_k - X_*\|_F \,,$$

la cual está relacionada con las definiciones de convergencia lineal, superlineal y cuadrática, que determinaremos en cada problema.

Los resultados obtenidos en cada caso, los presentamos en tablas que contienen el número de iteraciones (k) usadas por el algoritmo propuesto junto con los valores de f_0 y f_1 para cada k.

En las Tablas 5.10 y 5.12, observamos que $f_0(k)$ parece estar acotada y $f_1(k)$ crece sin cota, lo cual significa, para los **Problemas 8** y **15**, que el algoritmo *cuasi-Newton* converge linealmente.

k	$f_0(k)$	$f_1(k)$
0	0.4155	0.1781
1	0.2581	0.2663
2	0.3161	1.2640
3	0.2101	2.6571
4	0.0701	2.2198

Tabla 5.8: **Problema 2**

k	$f_0(k)$	$f_1(k)$
0	2.9541	1.6014
1	0.6110	0.1121
2	0.5472	0.1644
3	0.4462	0.2449
4	0.2901	0.3569
5	0.1022	0.4334
6	0.0317	1.3152
7	0.1527	199.72
8	0.0016	1363.6

Tabla 5.9: **Problema 3**

k	$f_0(k)$	$f_1(k)$
0	0.0655	0.0196
1	0.0129	0.5878
2	0.1628	574.35
3	0.1734	3758.4

Tabla 5.10: **Problema 8**

k	$f_0(k)$	$f_1(k)$
0	0.6358	0.1700
1	0.5446	0.2290
2	0.3690	0.2849
3	0.0467	0.3068
4	0.0125	0.2845
5	0.0008	0.2142

Tabla 5.11: **Problema 9**

k	$f_0(k)$	$f_1(k)$
0	0.9643	4.4105
1	0.6475	3.0712
2	0.6429	4.7092
3	0.6453	7.3518
4	0.6454	11.394
5	0.6430	17.589
6	0.6375	27.119
7	0.6274	41.871
8	0.6101	64.888
9	0.5796	101.051
10	0.5224	157.134
11	0.3974	228.848

Tabla 5.12: **Problema 15**

En la Tabla 5.9, podemos observar que $f_0(k)$ converge a cero y $f_1(k)$ crece sin cota, lo cual significa que, para el **Problema 3**, el algoritmo *cuasi-Newton* convergerá a lo más superlinealmente.

Para los **Problemas 2** y **9**, los resultados de las Tablas 5.8 y 5.11 muestran que $f_0(k)$ converge a cero y $f_1(k)$ parece estar acotado, lo cual significa que el algoritmo *cuasi-Newton* converge cuadráticamente. En particular, en el **Problema 2**, los resultados numéricos confirman esta convergencia, debido a que se cumple el **Lema 5.5.** Por otro lado, en el **Problema 9**, a pesar de no satisfacerse el **Lema 5.5**, el algoritmo propuesto puede converger cuadráticamente.

5.3. Comentarios finales del capítulo

En este capítulo, proponemos un algoritmo *cuasi-Newton* para resolver la ecuación polinómica matricial, el cual reduce el alto costo computacional del algoritmo *Newton* que tradicionalmente es utilizado para resolver esta ecuación. Demostramos que bajo ciertas hipótesis el algoritmo propuesto es local y alcanza hasta convergencia cuadrática.

Numéricamente, analizamos y comparamos su desempeño local con el algoritmo *Newton* usando 5 problemas. Los resultados obtenidos muestran un comportamiento similar de los dos algoritmos para los problemas **2, 3** y **9**, respecto al número de iteraciones; en los problemas restantes este número aumenta, como sucede en el caso vectorial. En contraste con lo anterior, el algoritmo *cuasi-Newton* da un menor tiempo CPU en todos los problemas; en particular, en el peor de los casos, el algoritmo *cuasi-Newton* usa el 64 % (**Problema 8**) y el 74 % (**Problema 15**) del tiempo que consume el algoritmo *Newton,* respectivamente. En el mejor caso, el algoritmo *cuasi-Newton* usa el 0.3 % (**Problema 2** para $n = 50$) del tiempo que consume el algoritmo *Newton.* Adicionalmente, se observa que el porcentaje de tiempo CPU del algoritmo *cuasi-Newton* disminuye comparado con el de *Newton* cuando el tamaño del problema aumenta (**Problema 2**).

La diferencia en los tiempos CPU de los algoritmos *Newton* y *cuasi-Newton*, observada en la experimentación numérica, se debe a la forma como estos calculan la matriz S_k. El primero, por la forma de la ecuación que define a S_k, debe usar la descomposición de Schur de la matriz X_k; mientras que el segundo, lo hace resolviendo un sistema matricial.

Por otro lado, realizamos un experimento numérico sobre la tasa de convergencia. Experimentalmente obtuvimos convergencia lineal en los **Problemas 8** y **15**, superlineal en el **Problema 3** y cuadrática en los **Problemas 2** y **9**.

En los **Problemas 2** y **9**, observamos igual comportamiento en cuanto a número de iteraciones de los dos algoritmos, lo cual en el caso particular del **Problema 2**, se debe a que las iteraciones *Newton* y *cuasi-Newton* son iguales en virtud de que se satisface el **Lema 5.5,** mientras que en el **Problema 9** dicho lema no se cumple.

Los resultados preliminares de las pruebas numéricas que se realizaron confirman el buen rendimiento del algoritmo *cuasi-Newton*. Consideramos que sería interesante incluir una estrategia de globalización, analizar el efecto de esta en la convergencia del algoritmo y el desempeño numérico del algoritmo global.

Capítulo 6

Dos algoritmos cuasi-Newton globales

En este capítulo globalizamos el algoritmo cuasi-Newton propuesto en Capítulo 5, introduciendo una búsqueda lineal exacta, proponemos una aproximación polinomial a la función de mérito y deducimos una condición suficiente para su intervalo de minimización. Usamos la función de mérito exacta y su aproximación para proponer dos algoritmos cuasi-Newton globales para resolver ecuaciones polinómicas matriciales. Para cada algoritmo, demostramos que la búsqueda lineal exacta no afecta la convergencia del método cuasi-Newton. Además, presentamos pruebas numéricas comparativas de las propuestas algorítmicas en las que también comparamos con el método de Newton.

6.1. Introduciendo una búsqueda lineal

El algoritmo cuasi-Newton (**Algoritmo 4**) presenta un buen desempeño local (Capítulo 5) y es una buena alternativa al alto costo computacional del método de Newton, que utiliza la descomposición de Schur (**Teoremas 2.1 2.2**). Sin embargo, debido a su naturaleza local, sus buenas propiedades de convergencia dependen de la aproximación inicial. Para mejorar la convergencia de este algoritmo desde puntos de partida arbitrarios consideramos inicialmente una estrategia de búsqueda lineal exacta análoga a la utilizada para globalizar el método de Newton en [33, 45] la siguiente iteración es de la forma $X_{k+1} = X_k + tS_k$, donde S_k es el paso cuasi-Newton y t se

calcula mediante una búsqueda lineal exacta que minimiza la función de mérito:

$$m(t) = \|P(X_k + tS_k)\|_F^2 . \tag{6.1}$$

En el caso particular del método de Newton (Capítulo 4), $m(t)$ es un polinomio de grado $2m$ [45], cuya forma explícita fue presentada recientemente en [33]. La forma explícita del polinomio a minimizar facilita el proceso de búsqueda lineal y ayuda a conjeturar sobre el valor de k para el cual el minimizador (tamaño de paso) pertenece al intervalo $[0, k]$. Además facilita la evaluación del polinomio y su derivada, lo cual es muy útil en el proceso de minimización.

En el caso del método cuasi-Newton, la dificultad del cálculo numérico de la función de mérito (6.1) es la misma que para el método de Newton: la evaluación de la función polinomial matricial en la iteración y su minimización a medida que el grado del polinomio aumenta.

Teniendo en cuenta lo anterior y con el fin de proponer una alternativa para el cálculo numérico de $m(t)$, analizamos a continuación la expresión $P(X + tS)$.

De la definición de la función polinomial matricial P, tenemos que

$$P(X + tS) = A_m(X + tS)^m + \cdots + A_1(X + tS) + A_0 I.$$

En [45], se demostró que

$$\begin{aligned} P(X + tS) &= \sum_{k=0}^{m} \sum_{i=0}^{k} t^i A_k \Phi_{XS}^k[k - i] \\ &= P(X) + tL_X(S) + \sum_{k=2}^{m} \sum_{i=2}^{k} t^i A_k \Phi_{XS}^k[k - i], \end{aligned} \tag{6.2}$$

donde $L_X(S)$ está dada por (4.5) y la función Φ_{XS} está definida como la suma de los productos de todas las permutaciones de X y S (ver Capítulo 2).

Dado que el método cuasi-Newton usa una aproximación a $L_X(S)$ concluimos de (6.2) que en general no es posible encontrar una expresión exacta para la función de mérito (6.1), tal como se da para el método de Newton. Como alternativa, presentamos una aproximación polinomial a (6.1) y calculamos su forma explícita.

El lado derecho de (6.2) se puede aproximar mediante la expresión:

$$P(X) + t\,\mathcal{B}_m(X)S + \sum_{k=2}^{m} \sum_{i=2}^{k} t^i A_k \Phi_{XS}^k[k - i],$$

que usando la igualdad (método cuasi-Newton Capítulo 5) $\mathcal{B}_m(X)S = -P(X)$, es igual a la expresión:

$$(1-t)P(X) + \mathcal{U}_m,$$

donde $\mathcal{U}_m = \sum_{k=2}^{m}\sum_{i=2}^{k} t^i A_k \Phi_{XS}^k[k-i]$, con lo cual tenemos la siguiente aproximación:

$$P(X+tS) \approx (1-t)P(X) + \mathcal{U}_m \equiv P_1(X+tS), \tag{6.3}$$

y definimos la siguiente aproximación a la función de mérito (6.1),

$$q(t) = \|P_1(X_k + tS_k)\|_F^2\,, \tag{6.4}$$

cuya forma explícita se obtiene mediante manipulaciones algebraicas análogas a las realizadas en el Capítulo 4, para el método de Newton. Esta forma explícita se representa en el **Teorema 6.1** para el cual usamos la recurrencia (4.13) y la notación (4.14).

Teorema 6.1. *El polinomio $q(t)$ está definido explícitamente por:*

$$q(t) = \widehat{\boldsymbol{c}}_0(1-t)^2 + \widehat{\boldsymbol{c}}_1(1-t)t^2 + \cdots + \widehat{\boldsymbol{c}}_{m-1}(1-t)t^m + \widehat{\boldsymbol{c}}_m t^4 + \cdots + \widehat{\boldsymbol{c}}_{3m-4}t^{2m}, \tag{6.5}$$

donde m es el grado del polinomio matricial (1.1), $\widehat{\boldsymbol{c}}_0 = \|B\|_F^2$, $\widehat{\boldsymbol{c}}_{3m-4} = \|B_q\|_F^2$, con $q = m(m-1)/2$, y los coeficientes restantes son diferencias de normas de Frobenius de las matrices B_j definidas en (4.9).

1. *Para los coeficientes $\widehat{\boldsymbol{c}}_1, \ldots, \widehat{\boldsymbol{c}}_{m-1}$, de $(1-t)t^p$, con $p = 2, 3, \ldots, m$, usando (4.13) y (4.14), definimos:*

$$\boldsymbol{c}_{s-1} = \|B + \mathcal{S}_m(s)\|_F^2 - \|\mathcal{S}_m(s)\|_F^2 - \|B\|_F^2\,, \quad s = 2, \ldots, m.$$

2. *Para el coeficiente $\widehat{\boldsymbol{c}}_m$, de t^4, usamos (4.13) para $s = 2$, $\widehat{\boldsymbol{c}}_m = \|\mathcal{S}_m(2)\|_F^2$.*

3. *Para los coeficientes $\widehat{\boldsymbol{c}}_{m+1}$ y $\widehat{\boldsymbol{c}}_{m+2}$, de t^5 y t^6 respectivamente, tenemos*

$$\begin{aligned}
\widehat{\boldsymbol{c}}_{m+1} &= \|\mathcal{S}_m(2) + \mathcal{S}_m(3)\|_F^2 - \|\mathcal{S}_m(2)\|_F^2 - \|\mathcal{S}_m(3)\|_F^2\,,\\
\widehat{\boldsymbol{c}}_{m+2} &= \|\mathcal{S}_m(2) + \mathcal{S}_m(4)\|_F^2 - \|\mathcal{S}_m(2)\|_F^2 - \|\mathcal{S}_m(4)\|_F^2 + \|\mathcal{S}_m(3)\|_F^2\,.
\end{aligned}$$

4. *Para los coeficientes $\widehat{\boldsymbol{c}}_k$ de $t^{k+(4-m)}$, con $m+3 \le k \le 3m-5$; es decir, para los coeficientes de las potencias $7, 8, \ldots, 2m-1$, consideramos si la potencia $k+(4-m)$ es impar o par. En el primer caso, se puede expresar de la forma $2r+1$, $r \ge 3$ y tenemos las siguientes condiciones:*

a) *si* $m = r + 1$,

$$\widehat{\boldsymbol{c}}_k = \|\mathcal{S}_m(r) + \mathcal{S}_m(r+1)\|_F^2 - \|\mathcal{S}_m(r)\|_F^2 - \|\mathcal{S}_m(r+1)\|_F^2 \,,$$

b) *si* $m > r + 1$,

$$\begin{aligned}\widehat{\boldsymbol{c}}_k = &\|\mathcal{S}_m(r) + \mathcal{S}_m(r+1)\|_F^2 - \|\mathcal{S}_m(r)\|_F^2 - \|\mathcal{S}_m(r+1)\|_F^2 \\ &+ \|\mathcal{S}_m(r-1) + \mathcal{S}_m(r+2)\|_F^2 - \|\mathcal{S}_m(r-1)\|_F^2 - \|\mathcal{S}_m(r+2)\|_F^2 \,.\end{aligned}$$

Análogamente, si la potencia $k + (4 - m)$ *es par entonces es de la forma* $2r$*, con* $r \geq 4$ *y tenemos las mismas condiciones que para el caso anterior:*

a) *si* $m = r + 1$, *tenemos*

$$\widehat{\boldsymbol{c}}_k = \|\mathcal{S}_m(r-1) + \mathcal{S}_m(r+1)\|_F^2 - \|\mathcal{S}_m(r-1)\|_F^2 - \|\mathcal{S}_m(r+1)\|_F^2 + \|\mathcal{S}_m(r)\|_F^2 \,,$$

b) *si* $m > r + 1$, *tenemos*

$$\begin{aligned}\widehat{\boldsymbol{c}}_k = &\|\mathcal{S}_m(r-1) + \mathcal{S}_m(r+1)\|_F^2 - \|\mathcal{S}_m(r-1)\|_F^2 - \|\mathcal{S}_m(r+1)\|_F^2 + \|\mathcal{S}_m(r)\|_F^2 \\ &+ \|\mathcal{S}_m(r-2) + \mathcal{S}_m(r+2)\|_F^2 - \|\mathcal{S}_m(r-2)\|_F^2 - \|\mathcal{S}_m(r+2)\|_F^2 \,.\end{aligned}$$

Demostración. Es análoga a la demostración del **Teorema 4.1**. ■

Aunque la forma del polinomio $q(t)$ dada por (6.5) es la misma que la del polinomio presentado en el Capítulo 4, para el método de Newton, sus coeficientes no son necesariamente los mismos ya que finalmente dependen de la matriz S.

Una consecuencia inmediata del **Teorema 6.1** es el cálculo explícito de la derivada de $q(t)$ que está garantizado por el siguiente corolario.

Corolario 6.1. *Sea* $q(t)$ *el polinomio dado por* (6.5)*. Entonces su derivada está dada por:*

$$q'(t) = -2\widehat{\boldsymbol{c}}_0(1-t) + \sum_{i=1}^{m-1} \left[(i+1) - t\,(i+2)\right] \widehat{\boldsymbol{c}}_i t^i + \sum_{i=1}^{2m-3} (i+3)\ \widehat{\boldsymbol{c}}_{m-1+i}\ t^{2+i}.$$

Demostración. Es análoga a la demostración del **Corolario 4.1**. ■

El siguiente resultado da una condición suficiente para minimizar el polinomio $q(t)$ en el intervalo $[0, 2]$.

Corolario 6.2. *$q'(2) \geq 0$ si, y solo si*

$$\left\| -B + \sum_{s=2}^{m} 2^{s-2}(s+2)\mathcal{S}_m(s) \right\|_F^2 \geq \left\| 2 \left[\sum_{s=3}^{m} 2^{s-3}(s-2)\mathcal{S}_m(s) \right] \right\|_F^2 . \tag{6.6}$$

Demostración. Es análoga a la demostración del **Corolario 4.2**. ■

6.2. Propuestas algorítmicas

A continuación presentamos dos algoritmos cuasi-Newton globales para resolver la ecuación polinomial matricial (1.1). El primero, que llamaremos algoritmo *cuasi-Newton implícito* usa la función de mérito exacta $m(t)$, dada por (6.1), y un *solver* para encontrar su minimizador. El segundo, que llamaremos algoritmo *cuasi-Newton explícito* utiliza la aproximación polinomial $q(t)$, dada por (6.4), como función de mérito y la condición (6.6) para su minimización.

Algoritmo 6 *cuasi-Newton implícito*

Entrada: $A_0, A_1, \ldots, A_m$, X_0, $\epsilon = 10^{-5}$, N_{max} y $k = 0$.
Salida: Matriz aproximación a X_*.

1: **mientras** $k \leq N_{max}$, $Res(X_k) \geq \epsilon$ **hacer**
2: Resolver para S_k, el sistema matricial: $\mathcal{B}_m(X_k)\, S_k = -P(X_k)$.
Búsqueda lineal.
3: $m(t) = \|P(X_k + t\, S_k)\|^2$.
4: Encontrar $t = sqp(t_0, m(t))$.
5: Actualizar $X_{k+1} = X_k + t\, S_k$ y $k = k + 1$.
6: **fin mientras**
7: **Salida** X_*

Algoritmo 7 *cuasi-Newton explícito*

Entrada: $A_0,\, A_1, \ldots,\, A_m,\, X_0,\ \epsilon = 10^{-5},\ N_{max}$ y $k = 0$.
Salida: Matriz aproximación a X_*.

mientras $k \leq N_{max},\ Res(X_k) \geq \epsilon$ **hacer**
  Resolver para S_k, el sistema matricial: $\mathcal{B}_m(X_k)\, S_k = -P(X_k)$.
  Búsqueda lineal.
  Construimos el polinomio $q(t)$.
  Aplicamos el criterio
  si $q'(2) > 0$ **entonces**
    Encontramos $t \in [0, 2]$, el minimizador de $q(t)$.
  sino
    Encontramos el minimizador de $q(t)$ fuera de $[0, 2]$.
  fin si
  Actualizar $X_{k+1} = X_k + t\, S_k$ y $k = k + 1$.
fin mientras
Salida X_*

Observación 7. *La búsqueda lineal exacta en el* **Algoritmo 7** *usa el polinomio explícito dado en el* **Teorema 6.1**. *Además, incluye la condición suficiente* (6.2) *para minimizar este polinomio.*

6.3. Análisis de convergencia

En esta sección, para cada propuesta algorítmica, demostramos que la estrategia de búsqueda lineal exacta no afecta la tasa de convergencia del método cuasi-Newton (5.1). Para ello, usaremos las hipótesis generales **H1, H2** y **H3**, y una hipótesis adicional [29]:

H4. El operador L es localmente *Lipschitz* continuo en $N(X_*, r) \subset \mathbb{C}^{n\times n}$. Es decir, existe una constante positiva γ tal que

$$\|L_X(S) - L_{X_*}(S)\| \leq \gamma \, \|X - X_*\| \, \|S\| \,, \tag{6.7}$$

para todo $X \in N(X_*, r)$ y $S \in \mathbb{C}^{n\times n}$.

Comenzamos con un lema técnico que da un límite superior para la doble sumatoria en (6.3) que será útil en la demostración de los teoremas centrales de esta sección.

Lema 6.1. *Sean* $P\colon \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ *una función polinómica matricial de grado* $m \geq 2$, X, $S \in \mathbb{C}^{n\times n}$ *y*

$$\mathcal{U}_m = \sum_{k=2}^{m}\sum_{i=2}^{k} t^i A_k\, \Phi_{XS}^k[k-i] = \sum_{i=0}^{m-2}\left[\sum_{j=2}^{m-i} t^j A_{m-i}\, \Phi_{X\,S}^{m-i}\,[m-i-j]\right]. \tag{6.8}$$

Entonces

$$\|\mathcal{U}_m\| \;\leq\; \left\{\sum_{i=0}^{m-3}\|A_{m-i}\|\left[\sum_{w=0}^{m-i-2}\binom{m-i}{w} t^{m-i-w}\,\|X\|^w\|S\|^{m-i-w-2}\right]+t^2\|A_2\|\right\}\|S\|^2. \tag{6.9}$$

Demostración. La demostración es análoga a la del **Lema 5.4**. Sea $\|\cdot\|$ una norma matricial inducida en $\mathbb{C}^{n\times n}$. La demostración la haremos por inducción sobre m.

1. Para $m = 2$, tenemos $\mathcal{U}_2 = A_2\left[t^2\Phi_{X\,S}^2\,[0]\right] = t^2 A_2\left[S^2\right]$. Así,

$$\|\mathcal{U}_2\| \;=\; t^2\,\|A_2\|\,\left\|S^2\right\| \leq t^2\,\|A_2\|\,\|S\|^2.$$

   Por lo tanto, (6.9) se cumple para $m = 2$.

   Hipótesis inductiva. Supongamos que (6.9) se cumple para $m = l-1$,

$$\|\mathcal{U}_{l-1}\| \;\leq\; \left\{\sum_{i=0}^{l-4}\|A_{l-1-i}\|\left[\sum_{w=0}^{l-1-i-2}\binom{l-1-i}{w} t^{l-1-i-w}\,\|X\|^w\,\|S\|^{l-1-i-w-2}\right]\right.$$
$$\left. +\; t^2\,\|A_2\|\right\}\|S\|^2. \tag{6.10}$$

2. *Paso de inducción.* Demostremos que (6.9) se cumple para $m = l$.

$$\begin{aligned}\mathcal{U}_l =& \sum_{i=0}^{l-2}\left[\sum_{j=2}^{l-i} t^j A_{l-i}\,\Phi_{X\,S}^{l-i}\,[l-i-j]\right]\\ =& A_l\left[t^2\Phi_{X\,S}^l\,[l-2]+\cdots+t^l\Phi_{X\,S}^l\,[0]\right]+\sum_{i=1}^{l-2}\left[\sum_{j=2}^{l-i} t^j A_{l-i}\,\Phi_{X\,S}^{l-i}\,[l-i-j]\right].\end{aligned}$$

   Haciendo un cambio de índices, la doble suma en la igualdad anterior equivale a la siguiente expresión:

$$\mathcal{U}_{l-1} = \sum_{i=0}^{l-1-2}\left[\sum_{j=2}^{l-1-i} t^j A_{l-1-i}\,\Phi_{X\,S}^{l-1-i}\,[l-1-i-j]\right], \tag{6.11}$$

luego

$$\mathcal{U}_l = \underbrace{A_l \left[t^2 \Phi^l_{X\,S}\,[l-2] + \cdots + t^l \Phi^l_{X\,S}\,[0]\right]}_{E} + \mathcal{U}_{l-1} = E + \mathcal{U}_{l-1},$$

y usando la desigualdad triangular, obtenemos

$$\|\mathcal{U}_l\| \leq \|E\| + \|\mathcal{U}_{l-1}\|\,. \tag{6.12}$$

Por otro lado, teniendo en cuenta la definición de la función $\Phi_{X\,S}$ (**Definición 4.2**), usando la desigualdad triangular y algunas manipulaciones algebraicas sobre el primer término del lado derecho de la desigualdad (6.12), obtenemos

$$\begin{aligned}
\|E\| &\leq \|A_l\| \left(t^2 \left\|\Phi^l_{X\,S}\,[l-2]\right\| + \cdots + t^l \left\|\Phi^l_{X\,S}\,[0]\right\|\right) \\
&\leq \|A_l\| \left(\binom{l}{l-2} t^2 \|X\|^{l-2} \|S\|^2 + \cdots + \binom{l}{0} t^l \|S\|^l\right) \\
&= \|A_l\| \left(\binom{l}{0} t^l \|S\|^l + \cdots + \binom{l}{l-2} t^2 \|X\|^{l-2} \|S\|^2\right) \\
&= \|A_l\| \sum_{w=0}^{l-2} \binom{l}{w} t^{l-w} \|X\|^w \|S\|^{l-w} \\
&= \left\{\|A_l\| \sum_{w=0}^{l-2} \binom{l}{w} t^{l-w} \|X\|^w \|S\|^{l-w-2}\right\} \|S\|^2\,.
\end{aligned}$$

Finalmente, usando la desigualdad anterior, la *hipótesis inductiva* (6.10) en (6.12) y algunas manipulaciones algebraicas, obtenemos

$$\begin{aligned}
\|\mathcal{U}_l\| &\leq \left\{\|A_l\| \sum_{w=0}^{l-2} \binom{l}{w} t^{l-w} \|X\|^w \|S\|^{l-w-2}\right\} \|S\|^2 \\
&\quad + \left\{\sum_{i=0}^{l-4} \|A_{l-1-i}\| \left[\sum_{w=0}^{l-1-i-2} \binom{l-1-i}{w} t^{l-1-i-w} \|X\|^w \|S\|^{l-1-i-w-2}\right]\right. \\
&\quad \left. + t^2 \|A_2\|\right\} \|S\|^2 \\
&\leq \left\{\sum_{i=0}^{l-3} \|A_{l-i}\| \left[\sum_{w=0}^{l-i-2} \binom{l-i}{w} t^{l-i-w} \|X\|^w \|S\|^{l-i-w-2}\right] + t^2 \|A_2\|\right\} \|S\|^2.
\end{aligned}$$

Por lo tanto, (6.9) se cumple para $m = l$. Esto completa la demostración. ■

El siguiente es un lema técnico que permite acotar la norma de la diferencia entre las $L_X(S)$ y $B_m(X)S$. Este resultado será útil en la demostración de los **Teoremas 6.2** y **6.3**.

Lema 6.2. *Sea $P\colon \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ una función polinómica matricial de grado $m \geq 2$, que satisface las hipótesis* **H1, H3** *y* **H4***; X, $S \in \mathbb{C}^{n\times n}$ y*

$$E_{XS} = L_X(S) - B_m(X)S. \tag{6.13}$$

Entonces existen constantes positivas γ, α y φ, tales que

$$\|E_{XS}\| \leq \gamma \|X - X_*\| \|S\| + \alpha \|S\|^{1+\theta} + \varphi \|X - X_*\| \|S\|. \tag{6.14}$$

Demostración. Sea

$$\begin{aligned}\|E_{XS}\| &= \|L_X(S) - B_m(X)S\| \\ &= \|L_X(S) - L_{X_*}(S) + L_{X_*}(S) - \mathcal{B}_m(X_*)S + \mathcal{B}_m(X_*)S - \mathcal{B}_m(X)S\|,\end{aligned}$$

usando la desigualdad triangular y las hipótesis **H3** y **H4**, tenemos que existen constantes positivas γ y α, tales que

$$\begin{aligned}\|E_{XS}\| &\leq \|L_X(S) - L_{X_*}(S)\| + \|L_{X_*}(S) - \mathcal{B}_m(X_*)S\| + \|\mathcal{B}_m(X_*)S - \mathcal{B}_m(X)S\| \\ &\leq \|(L_X - L_{X_*})\, S\| \|S\| + \alpha \|S\|^{1+\theta} + \|\mathcal{B}_m(X_*) - \mathcal{B}_m(X)\| \|S\| \\ &\leq \gamma \|X - X_*\| \|S\| + \alpha \|S\|^{1+\theta} + \|\mathcal{B}_m(X_*) - \mathcal{B}_m(X)\| \|S\|. \end{aligned} \tag{6.15}$$

Por otra parte, dado que B_m es una función polinomial matricial, ella es derivable y Lipschitz continua entonces existe una constante positiva $\varphi > 0$, tal que

$$\|\mathcal{B}_m(X_*) - \mathcal{B}_m(X)\| \leq \varphi \ \|X_* - X\|. \tag{6.16}$$

Entonces usando (6.16) en (6.15), obtenemos

$$\|E_{XS}\| \leq \gamma \|X - X_*\| \|S\| + \alpha \|S\|^{1+\theta} + \varphi \|X - X_*\|. \qquad \blacksquare$$

El siguiente lema técnico y su demostración serán de utilidad en las demostraciones de los Teoremas **6.2** y **6.3**.

Lema 6.3. *Sea $P\colon \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ una función polinómica matricial que satisface la hipótesis* **H1**. *Para el método cuasi-Newton (5.1) suponga que X_k está en la región de convergencia de orden $1+\theta$, $\theta \in [0,1]$ a X_* y sea $\Delta_k = X_k - X_*$. Entonces*

$$\|P(X_k)\| = O\left(\|\Delta_k\|\right). \tag{6.17}$$

Demostración. Como X_k está en la región de convergencia de orden $1+\theta$, $\theta \in [0,1]$, a X_* entonces por los **Teoremas 5.2** y **5.3**, para $\mu > 0$ y $\theta \in [0,1]$, se tiene que $\|X_{k+1} - X_*\| < \mu \|X_k - X_*\|^{1+\theta}$. Equivalentemente,

$$\|\Delta_{k+1}\| = O\left(\|\Delta_k\|^{1+\theta}\right), \tag{6.18}$$

Dado que X_k está dentro de la región de convergencia de X_*,

$$\|\Delta_k\|^{1+\theta} < \|\Delta_k\|, \tag{6.19}$$

para algún $\theta \in [0,1]$. Además, $S_k = X_{k+1} - X_k = \Delta_{k+1} - \Delta_k$. De esta igualdad, de (6.18) y (6.19), tenemos

$$\|S_k\| = O\left(\|\Delta_k\|\right). \tag{6.20}$$

Para la función polinómica matricial P de grado m,

$$\begin{aligned} P(X+S) &= P(X) + L_X(S) + \sum_{k=2}^{m}\sum_{i=2}^{k} A_k \Phi_{XS}^k[k-i] \\ &= P(X) + \mathcal{B}_m(X)S + E_{XS} + \overline{\mathcal{U}}_m, \end{aligned}$$

donde $E_{X\,S} = L_X(S) - \mathcal{B}_m(X)S$ y $\overline{\mathcal{U}}_m = \sum_{k=2}^{m}\sum_{i=2}^{k} A_k \Phi_{XS}^k[k-i]$. Por el **Lema 6.1**, con $t=1$, obtenemos

$$\left\|\overline{\mathcal{U}}_m\right\| \leq \left\{\sum_{k=2}^{m} \|A_k\| \left[\sum_{i=2}^{k} \binom{k}{i} \|X\|^{k-i} \|S\|^{i-2}\right]\right\} \|S\|^2,$$

luego $\left\|\overline{\mathcal{U}}_m\right\| = O\left(\|S\|^2\right)$. Por el **Lema 6.2**, existen constantes positivas γ, α y φ, tales que

$$\|E_{XS}\| \leq \gamma \|X - X_*\| \|S\| + \alpha \|S\|^{1+\theta} + \varphi \|X - X_*\| \|S\|. \tag{6.21}$$

De otro lado,

$$\begin{aligned} P(X_k) = P(X_* + \Delta_k) &= P(X_*) + \mathcal{B}_m(X_*)\Delta_k + E_{X_*\,\Delta_k} + O\left(\|\Delta_k\|^2\right) \\ &= \mathcal{B}_m(X_*)\Delta_k + E_{X_*\,\Delta_k} + O\left(\|\Delta_k\|^2\right). \end{aligned} \tag{6.22}$$

Por (6.21), tenemos

$$\|E_{X_*\,\Delta_k}\| \leq \gamma \|\Delta_k\|^2 + \alpha \|\Delta_k\|^{1+\theta} + \varphi \|\Delta_k\|^2 = O\left(\|\Delta_k\|^{1+\theta}\right). \tag{6.23}$$

Además, $\|\mathcal{B}_m(X_*)\Delta_k\| = O(\|\Delta_k\|)$. Por lo tanto, usando (6.22) y (6.23), obtenemos

$$\|P(X_k)\| = O(\|\Delta_k\|).$$

En forma análoga, $\|P(X_{k+1})\| = O(\|\Delta_{k+1}\|)$ y usando (6.18), concluimos que

$$\|P(X_{k+1})\| = O\left(\|\Delta_k\|^{1+\theta}\right). \tag{6.24}$$

Con lo cual se termina la demostración. ■

El siguiente teorema es uno de los resultados centrales del capítulo, el cual garantiza que la estrategia de búsqueda lineal exacta utilizada en el **Algoritmo 6** no interfiere con la convergencia del método *cuasi-Newton* (5.1).

Teorema 6.2. *Sea $P\colon \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ una función polinómica matricial que satisface las hipótesis* **H1** *a* **H4**. *Para el método cuasi-Newton* (5.1) *suponga que X_k está en la región de convergencia de orden $1+\theta$, $\theta \in [0,1]$ a X_*, y la sucesión de tamaños de paso $\{t_k^*\}$ generada por el* **Algoritmo 6** *es acotada. Entonces la estrategia de búsqueda lineal exacta usada en el* **Algoritmo 6** *preserva las propiedades de convergencia del método cuasi-Newton* (5.1).

Demostración. Sean $X_{k+1} = X_k + S_k$ y $\overline{X}_{k+1} = X_k + t_k^* S_k$ las iteraciones cuasi-Newton y cuasi-Newton con búsqueda lineal exacta, esta última generada por el **Algoritmo 6**.

De (6.1) y (6.2), tenemos

$$m(t) = \|P(X_k + t\,S_k)\|^2 = \left\|P(X_k) + tL_{X_k(S_k)} + \mathcal{U}_m\right\|^2.$$

Ahora, de (5.1), concluimos que

$$m(t) = \|(1-t)P(X_k) + tE_{X_k S_k} + \mathcal{U}_m\|^2, \tag{6.25}$$

Además, sabemos que el tamaño de paso t_k^* es el minimizador global del polinomio $m(t)$; en consecuencia,

$$\|P(X_k + t_k^*\,S_k)\| \le \|P(X_k + S_k)\| = \|P(X_{k+1})\|. \tag{6.26}$$

Usando (6.18), (6.22), (6.24) y (6.26) en (6.25), tenemos

$$\|(1-t_k^*)\,P(X_k) + t_k^* E_{X_k S_k} + \mathcal{U}_m\| = \|P(X_k + t_k^* S_k)\| \le \|P(X_{k+1})\| = O\left(\|\Delta_k\|^{1+\theta}\right).$$

Entonces,

$$\|(1-t_k^*)\,P(X_k)+t_k^*E_{X_kS_k}+\mathcal{U}_m\| \;=\; O\left(\|\Delta_k\|^{1+\theta}\right). \tag{6.27}$$

De (6.27), la notación de Landau (Capítulo 2), la desigualdad triangular inversa y el **Lema 6.1,** tenemos que existen constantes positivas α_1 and α_2, tales que

$$\begin{aligned}|1-t_k^*|\,\|P(X_k)\|-|t_k^*|\,\|E_{X_kS_k}\|-\|\mathcal{U}_m\| &\le \alpha_1\,\|\Delta_k\|^{1+\theta}\\ |1-t_k^*|\,\|P(X_k)\|-|t_k^*|\,\|E_{X_kS_k}\| &\le \alpha_1\,\|\Delta_k\|^{1+\theta}+\alpha_2\,\|\Delta_k\|^2. \end{aligned}\tag{6.28}$$

Usando (6.23) teniendo en cuenta que por hipótesis la sucesión $\{t_k^*\}$ es acotada, existen constantes positivas c y α_3, tales que $|t_k^*|\,\|E_{X_kS_k}\| \le c\,\alpha_3\,\|\Delta_k\|^{1+\theta}$. Entonces

$$|1-t_k^*|\,\|P(X_k)\| \le \alpha_1\,\|\Delta_k\|^{1+\theta}+\alpha_2\,\|\Delta_k\|^2+c\,\alpha_3\,\|\Delta_k\|^{1+\theta} \le \alpha\,\|\Delta_k\|^{1+\theta}, \tag{6.29}$$

donde α es una constante positiva. De (6.17),

$$\|P(X_k)\| \le \rho\,\|\Delta_k\|, \tag{6.30}$$

para alguna constante positiva ρ.

Dado que la matriz $\mathcal{B}_m(X_*)$ es no singular (hipótesis **H1**) entonces $\|P(X_k)\| \neq 0$. Así, por (6.30), tenemos que

$$\frac{1}{\|P(X_k)\|} \ge \frac{1}{\rho\,\|\Delta_k\|}.$$

Multiplicando (6.29) por $\dfrac{1}{\rho\,\|\Delta_k\|}$, tenemos

$$|1-t_k^*|\tau \;\le\; \frac{\alpha}{\rho}\,\|\Delta_k\|^{\theta},$$

donde $\tau = \dfrac{\|P(X_k)\|}{\rho\,\|\Delta_k\|} \le 1$. Así, $|1-t_k^*| \le \alpha_4\left(\|\Delta_k\|^{\theta}\right)$ con $\alpha_4 = \dfrac{\alpha}{\tau\,\rho}$. Entonces,

$$|1-t_k^*| = O\left(\|\Delta_k\|^{\theta}\right). \tag{6.31}$$

Usando las iteraciones cuasi-Newton y cuasi-Newton con búsqueda lineal exacta, tenemos

$$\overline{X}_{k+1}-X_* = \overline{X}_{k+1}-X_{k+1}+X_{k+1}-X_* = (1-t_k^*)S_k+\Delta_{k+1}.$$

Finalmente, por la desigualdad triangular, (6.18), (6.20) y (6.31), obtenemos

$$\left\|\overline{X}_{k+1}-X_*\right\| \le |1-t_k^*|\,\|S_k\|+\|\Delta_{k+1}\| = O\left(\|\Delta_k\|^{1+\theta}\right),$$

por lo tanto, $\overline{X}_k$ converge a X_*, con orden $1+\theta$. ■

El siguiente es el segundo resultado central de este capítulo, el cual es análogo al anterior y garantiza que la estrategia de búsqueda lineal exacta utilizada en **Algoritmo 7** no interfiere con la convergencia del método cuasi-Newton (5.1).

Teorema 6.3. *Sea* $P\colon \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ *una función polinómica matricial que satisface las hipótesis* **H1** *a* **H4**. *Para el método cuasi-Newton* (5.1) *suponga que* X_k *está en la región de convergencia de orden* $1+\theta$, $\theta \in [0,1]$ *a* X_*, *y que la sucesión de tamaños de paso* $\{t_k^*\}$ *generada por el* **Algoritmo 7** *es acotada. Entonces la estrategia de búsqueda lineal exacta usada en el* **Algoritmo 7** *preserva las propiedades de convergencia del método cuasi-Newton* (5.1).

Demostración. Sean $X_{k+1} = X_k + S_k$ y $\overline{X}_{k+1} = X_k + t_k^* S_k$, las iteraciones cuasi-Newton y cuasi-Newton con búsqueda lineal exacta, esta última generada por el **Algoritmo 7**.

De (6.4), tenemos

$$q(t) = \|P_1(X_k + t\,S_k)\|^2 = \|(1-t)P(X_k) + \mathcal{U}_m\|^2 . \tag{6.32}$$

De otro lado, recordemos que el tamaño de paso t_k^* es el minimizador global del polinomio $q(t)$, en consecuencia,

$$\|P_1(X_k + t_k^*\,S_k)\| \le \|P_1(X_k + S_k)\| . \tag{6.33}$$

Así, realizando algunas manipulaciones algebraicas, tenemos

$$\begin{aligned}
\|P_1(X_k + t_k^*\,S_k) + t_k^* E_{X_k S_k}\| &\le \|P_1(X_k + S_k) + t_k^* E_{X_k S_k}\| \\
&\le \|P_1(X_k + S_k) + E_{X_k S_k} + (t_k^* - 1)E_{X_k S_k}\| \\
&= \|P(X_k + S_k) + (t_k^* - 1)E_{X_k S_k}\| \\
&\le \|P(X_k + S_k)\| + |t_k^* - 1|\,\|E_{X_k S_k}\| \\
&= \|P(X_{k+1})\| + |t_k^* - 1|\,\|E_{X_k S_k}\| .
\end{aligned}$$

Por (6.24), existe $\alpha_1 > 0$ tal que $\|P(X_{k+1})\| \le \alpha_1 \|\Delta_k\|^{1+\theta}$. Por otra parte, usando (6.23) y teniendo en cuenta que por hipótesis la sucesión $\{t_k^*\}$ es acotada, existen constantes positivas c y α_2 tales que

$$|t_k^* - 1|\,\|E_{X_k S_k}\| \le c\,\alpha_2 \|\Delta_k\|^{1+\theta} .$$

Entonces

$$\|P_1(X_k + t_k^*\,S_k) + t_k^* E_{X_k S_k}\| \le \alpha_1 \|\Delta_k\|^{1+\theta} + c\,\alpha_2 \|\Delta_k\|^{1+\theta} = O\left(\|\Delta_k\|^{1+\theta}\right). \tag{6.34}$$

Dado que $P_1(X_k + t_k^* S_k) + t_k^* E_{X_k S_k} = (1 - t_k^*)P(X_k) + \mathcal{U}_m + t_k^* E_{X_k S_k}$; usando (6.34), la notación Landau $O(\cdot)$ (Capítulo 2), la desigualdad triangular inversa y el **Lema 6.1,** tenemos que existen constantes positivas α_3, y α_4, tales que

$$|1 - t_k^*| \, \|P(X_k)\| - |t_k^*| \, \|E_{X_k S_k}\| - \|\mathcal{U}_m\| \leq \alpha_3 \, \|\Delta_k\|^{1+\theta}$$
$$|1 - t_k^*| \, \|P(X_k)\| - |t_k^*| \, \|E_{X_k S_k}\| \leq \alpha_3 \, \|\Delta_k\|^{1+\theta} + \alpha_4 \, \|\Delta_k\|^2 ,$$

usando (6.23) y teniendo en cuenta que $\{t_k^*\}$ es acotada, $|t_k^*| \, \|E_{X_k S_k}\| = O\left(\|\Delta_k\|^{1+\theta}\right)$, equivalentemente, $|t_k^*| \, \|E_{X_k S_k}\| \leq \alpha_5 \, \|\Delta_k\|^{1+\theta}$, para algún $\alpha_5 > 0$. Entonces

$$\begin{aligned} |1 - t_k^*| \, \|P(X_k)\| &\leq \alpha_3 \, \|\Delta_k\|^{1+\theta} + \alpha_4 \, \|\Delta_k\|^2 + \alpha_5 \, \|\Delta_k\|^{1+\theta} \\ &\leq \alpha \, \|\Delta_k\|^{1+\theta} , \end{aligned} \tag{6.35}$$

donde α es una constante positiva. De (6.17),

$$\|P(X_k)\| \leq \rho \, \|\Delta_k\| , \tag{6.36}$$

para alguna constante positiva ρ.

Dado que la matriz $\mathcal{B}_m(X_*)$ es no singular (hipótesis **H1**) entonces $\|P(X_k)\| \neq 0$. Tomando recíprocos en (6.36), obtenemos

$$\frac{1}{\|P(X_k)\|} \geq \frac{1}{\rho \, \|\Delta_k\|}.$$

Multiplicando (6.35) por $\dfrac{1}{\rho \, \|\Delta_k\|}$, tenemos

$$|1 - t_k^*| \tau \leq \frac{\alpha}{\rho} \, \|\Delta_k\|^{\theta} ,$$

donde $\tau = \dfrac{\|P(X_k)\|}{\rho \, \|\Delta_k\|} \leq 1$. Then $|1 - t_k^*| \leq \alpha_6 \left(\|\Delta_k\|^{\theta}\right)$ con $\alpha_6 = \dfrac{\alpha}{\tau \, \rho}$. Entonces,

$$|1 - t_k^*| = O\left(\|\Delta_k\|^{\theta}\right). \tag{6.37}$$

Usando las iteraciones *cuasi-Newton* y *cuasi-Newton* con búsqueda lineal exacta, tenemos

$$\overline{X}_{k+1} - X_* = \overline{X}_{k+1} - X_{k+1} + X_{k+1} - X_* = (1 - t_k^*)S_k + \Delta_{k+1}.$$

Por la desigualdad triangular (6.18), (6.20), y (6.37), obtenemos

$$\left\|\overline{X}_{k+1} - X_*\right\| \leq |1 - t_k^*| \, \|S_k\| + \|\Delta_{k+1}\| = O\left(\|\Delta_k\|^{1+\theta}\right),$$

por lo tanto, $\overline{X}_k$ converge a X_*, con orden $1 + \theta$. ■

Es importante mencionar que las propiedades de convergencia global del método cuasi-Newton con búsqueda lineal exacta se pueden obtener de la teoría estándar [14] [15], ya que al resolver el sistema matricial $P(X) = 0$ por un método cuasi Newton haciendo búsqueda lineal exacta con función de mérito $F(X) = \|P_1(X)\|_F^2$ estamos, en realidad, resolviendo un sistema vectorial $f(x) = 0$, donde $f : \mathbb{R}^{n^2} \to \mathbb{R}^{n^2}$, mediante una búsqueda lineal exacta con función de mérito $F(x) = f(x)^T f(x)$. Recordemos que en el caso vectorial, la teoría está hecha bajo el supuesto que se satisfaga la condición Dennis-Moré [12]. En nuestro caso, tenemos una hipótesis análoga a esta condición: el supuesto **H3** que en otras palabras nos dice que la aproximación debe converger al operador, en la dirección de la matriz S.

6.4. Experimentación numérica

En esta sección, analizamos el desempeño numérico de los algoritmos *cuasi-Newton implícito* y *cuasi-Newton explícito* propuestos en la **Sección 6.2**, los cuales denotaremos como Algoritmos IQN y EQN, respectivamente. Nuestro interés es analizar numéricamente el beneficio de introducir una búsqueda lineal en el algoritmo cuasi-Newton (5.1) y el desempeño global de las dos nuevas propuestas algorítmicas globales.

Inicialmente comparamos el algoritmo cuasi-Newton local propuesto en la **Sección 5.2** (que denotaremos algoritmo QN) con los nuevos algoritmos cuasi-Newton globales IQN y EQN. A continuación, comparamos los algoritmos IQN y EQN con el algoritmo Newton explícito presentado en Capítulo 4, **Sección 4.5** (que denotaremos algoritmo EN), en términos de número de iteraciones, tiempo de CPU, a su vez, analizamos si la condición suficiente (6.6) se cumple o no, lo cual informamos en las tablas de resultados con los números 1 o 0, respectivamente.

Para la experimentación numérica, consideramos los **Problemas 1, 2, 3, 4, 7, 8, 10** y **9** descritos en el Capítulo 3, en los cuales es necesario resolver ecuaciones matriciales de grado dos, seis, cuatro, tres, cuatro, tres, cuatro y cinco, respectivamente.

A continuación, presentamos los resultados en dos tablas que contienen la siguiente información: para el **Problema 1**, en la primera tabla, el valor del parámetro δ para los dos algoritmos, las otras columnas contienen el número de iteraciones para el algoritmo cuasi-Newton sin y con búsqueda lineal exacta. La segunda tabla presenta los resultados obtenidos de los algoritmos globalizados, con la información presentada en las siguientes columnas: la matriz inicial (X_0), el número de iteraciones (N) y Tiempo de CPU (*time*) para cada algoritmo. Para los otros problemas, la columna

(δ), se cambia a (X_0) indicando la matriz inicial.

δ	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
10^{-1}	6	5	5
10^{-2}	8	6	6
10^{-3}	8	6	6
10^{-4}	8	6	6
10^{-5}	8	6	6
10^{-6}	8	6	6
10^{-7}	8	6	6
10^{-8}	8	6	6

Tabla 6.1: Número de iteraciones para la convergencia del **Problema 1**, $X_0 = 0$.

En la Tabla 6.1 presentamos el número de iteraciones de los algoritmos cuasi-Newton sin y con búsqueda lineal. Para todos los valores de δ observamos una disminución en el número de iteraciones cuando se aplica búsqueda lineal en el algoritmo, en particular, para $\delta = 10^{-1}$ se obtiene una disminución menor, mientras que en los demás casos esta disminución es constante.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
δ	N	*time*	*con*	N	*time*	N	*time*	con
10^{-1}	5	0.0148	0	5	0.0937	5	0.1653	0
10^{-2}	6	0.0156	0	6	0.0954	6	0.1982	0
10^{-3}	6	0.0158	0	6	0.0938	6	0.1975	0
10^{-4}	6	0.0157	0	6	0.0917	6	0.1980	0
10^{-5}	6	0.0156	0	6	0.0908	6	0.2188	0
10^{-6}	6	0.0173	0	6	0.0875	6	0.2050	0
10^{-7}	6	0.0165	0	6	0.0904	6	0.1905	0
10^{-8}	6	0.0186	0	6	0.0937	6	0.1819	0

Tabla 6.2: Resultados para **Problema 1**, con $X_0 = 0$.

En la Tabla 6.2 observamos el mismo número de iteraciones en los tres algoritmos. Por otra parte, los algoritmos EQN e IQN usan menos tiempo de CPU que el algoritmo

EN, ellos utilizan aproximadamente 9 % y 56 % del tiempo de CPU del algoritmo EN. En gneral, el algoritmo EQN muestra un mejor comportamiento que los otros dos algoritmos.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
0	6	4	5
$-I_n$	7	5	5
$10^1 I_n$	13	5	6
$10^3 I_n$	37	6	6
$10^4 I_n$	48	8	7

Tabla 6.3: Número de iteraciones para la convergencia del **Problema 2**, con $n = 5$.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
0	6	4	4
$-I_n$	7	4	5
$10^1 I_n$	6	7	7
$10^3 I_n$	37	9	6
$10^4 I_n$	48	9	9

Tabla 6.4: Número de iteraciones para la convergencia del **Problema 2**, con $n = 50$.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
0	6	4	4
I_n	7	5	5
$10^1 I_n$	13	7	7
$10^3 I_n$	37	8	5
$10^4 I_n$	48	9	9

Tabla 6.5: Número de iteraciones para la convergencia del **Problema 2**, con $n = 100$.

En las Tablas 6.3, 6.4 y 6.5, el número de iteraciones en ambos métodos depende de X_0 y aumenta a medida que el múltiplo escalar de I_n, aumenta. Por otro lado, el porcentaje de eficiencia de la búsqueda lineal depende del tamaño del problema, es decir 62 % y 74 % para $n = 5$, 69 % y 72 % para $n = 50$ y 69 % y 72 % para $n = 100$ usando los algoritmos EQN e IQN, respectivamente.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
0	4	0.0337	0	5	0.0856	5	0.0937	0
$-I_n$	5	0.0347	0	5	0.0901	6	0.1170	0
$10^1 I_n$	5	0.0368	1	6	0.1150	6	0.1244	1
$10^3 I_n$	6	0.0452	3	6	0.1875	6	0.1905	3
$10^4 I_n$	8	0.0473	4	7	0.2188	9	0.2303	4

Tabla 6.6: Resultados para **Problema 2**, con $n = 5$.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
0	4	0.0566	0	4	0.0781	4	2.5469	0
$-I_n$	4	0.0751	0	5	0.1419	5	3.1562	0
$10^1 I_n$	6	0.0866	1	7	0.2188	7	3.8906	1
$10^3 I_n$	9	0.1719	2	6	0.2610	8	4.4688	4
$10^4 I_n$	9	0.1870	4	9	0.4988	9	4.4875	2

Tabla 6.7: Resultados para **Problema 2**, con $n = 50$.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
0	4	0.3053	0	4	0.4427	4	14.9840	0
I_n	5	0.3302	0	5	0.5672	5	16.8590	0
$10^1 I_n$	7	0.4345	1	7	0.6145	8	21.7660	1
$10^3 I_n$	8	0.9305	3	5	1.2102	9	31.3590	3
$10^4 I_n$	9	1.3906	4	9	2.744	10	33.8590	4

Tabla 6.8: Resultados para **Problema 2**, con $n = 100$.

En las Tablas 6.6, 6.7 y 6.8 podemos observar un número similar de iteraciones en los tres algoritmos, debido a que en este caso se cumple la hipótesis de conmutatividad del **Lema 6** en [34]. En cuanto al tiempo de CPU, los algoritmos EQN e IQN tienen

un tiempo similar; en particular el algoritmo EQN utiliza, en el peor de los casos 40 % del tiempo del algoritmo IQN, para el tamaño $n = 5$. Cuando el tamaño del problema es $n = 50$, encontramos un mejor rendimiento en el tiempo de ejecución del algoritmo EQN que ahora es 4 % del tiempo del algoritmo EN (Tabla 6.7). Para $n = 100$, mejora el rendimiento del algoritmo EQN en cuanto a tiempo de ejecución, ya que en este caso utiliza 2 % del tiempo de ejecución del algoritmo EN. Es importante señalar que para este problema el rendimiento de los algoritmos EQN y IQN aumenta cuando aumenta el tamaño del problema.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
I_n	10	6	6
$-24I_n$	13	4	5
$10^1 I_n$	10	4	5
$10^3 I_n$	26	5	4
$10^5 I_n$	42	6	5
$10^{18} I_n$	–	13	13
$10^{20} I_n$	–	15	15

Tabla 6.9: Número de iteraciones para la convergencia del **Problema 3**.

En la Tabla 6.9, podemos ver que el número de iteraciones del algoritmo cuasi-Newton con búsqueda lineal exacta disminuye en comparación con las del algoritmo cuasi-Newton sin búsqueda lineal exacta, dicha disminución está entre 60 % y 85 %. Además, observamos la convergencia en los algoritmos EQN y IQN en todos los casos.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
I_n	6	0.0278	0	6	0.0774	5	0.0312	0
$-24I_n$	4	0.0187	1	5	0.0889	4	0.0193	1
$10^1 I_n$	4	0.0197	1	5	0.0798	4	0.0197	1
$10^3 I_n$	5	0.0227	2	4	0.1294	5	0.0278	2
$10^5 I_n$	6	0.0268	3	5	0.1562	6	0.0302	3
$10^{18} I_n$	13	0.0432	9	13	0.2589	14	0.0753	9
$10^{20} I_n$	15	0.0445	11	15	0.2698	15	0.0780	11

Tabla 6.10: Resultados para **Problema 3**.

De la Tabla 6.10 podemos observar que los tres algoritmos convergen en un número similar de iteraciones. Por otro lado, observamos que el algoritmo EQN usa en tiempo de CPU aproximadamente el 90 % del tiempo que usa el algoritmo EN y el 22 % del tiempo que usa el algoritmo IQN. Adicionalmente, para este problema es importante resaltar que en los algoritmos EQN y EN, la condición (6.6) no se cumple el mismo número de veces.

Hicimos otro análisis para este problema, análogo al hecho en [45]. Para ello, elegimos 100 matrices iniciales aleatorias de la forma:

$$X_0 = \begin{bmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{bmatrix}, \text{ donde } -100 \leq x_1, \ldots, x_9 \leq 100.$$

Los resultados se muestran en la Tabla 6.11 que tiene las siguientes columnas: *Intervalo* de iteraciones; *Éxitos* indica el número de aciertos, con las 100 matrices iniciales en el intervalo considerado y *tiempo* que indica el tiempo medio de acierto en el intervalo.

	Algoritmo QN		Algoritmo EN		Algoritmo EQN	
Intervalo	*Éxitos*	*tiempo*	*Éxitos*	*tiempo*	*Éxitos*	*tiempo*
[0, 30]	11	0.0054	70	0.0941	42	0.0789
[31, 50]	31	0.0102	8	0.3377	24	0.2140
[51, 100]	24	0.0157	5	0.4905	10	0.3282

Tabla 6.11: Resultados para **Problema 3**, con X_0 aleatorias.

Comparando los resultados con los presentados en [45], observamos un comportamiento muy similar: el mayor número de aciertos para los algoritmos se encuentra en el intervalo $[0, 30]$, con un alto rendimiento del algoritmo EN, en el que se encuentran los 70 % de aciertos, mientras que el algoritmo EQN logra 42 %, de aciertos y el algoritmo QN solo logra 11 % es decir, el algoritmo EN tiene un mejor comportamiento. En general, los algoritmos EN y EQN tienen más éxito, ambos logran un rendimiento de 83 % en comparación con 76 % para el algoritmo EQN y 61 % para el algoritmo QN. Por otro lado, el algoritmo QN tiene un mejor comportamiento con respecto al tiempo de ejecución.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
$-10^1 I_n$	9	3	4
$-10^2 I_n$	14	2	3
$10^5 I_n$	32	2	2
$10^8 I_n$	48	4	3
$10^{28} I_n$	–	10	11
$10^{30} I_n$	–	11	11

Tabla 6.12: Número de iteraciones para la convergencia del **Problema 4**.

En la Tabla 6.12 observamos una disminución de 90 % en promedio en el número de iteraciones cuando aplicamos búsqueda lineal exacta en el algoritmo cuasi-Newton en los casos en que los algoritmos convergen. Podemos observar convergencia en para los algoritmos con búsqueda lineal en todos los casos.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
$-10^1 I_n$	3	0.0135	1	3	0.0773	4	0.0156	1
$-10^2 I_n$	2	0.0108	1	3	0.0790	3	0.0127	1
$10^5 I_n$	2	0.0117	2	2	0.1146	3	0.0124	2
$10^8 I_n$	4	0.0164	3	3	0.1833	4	0.0152	3
$10^{28} I_n$	10	0.0257	9	10	0.2346	19	0.0680	11
$10^{30} I_n$	11	0.0265	11	11	0.2463	20	0.0690	12

Tabla 6.13: Resultados para el **Problema 4**.

En la Tabla 6.13 observamos que los tres algoritmos tienen un rendimiento similar para todas las matrices iniciales en términos de número de iteraciones. Por otro lado, el algoritmo EQN consume aproximadamente 93 % del tiempo de CPU que usa el algoritmo EN y 12 % del que usa el algoritmo IQN. En este problema, los algoritmos EQN y EN no cumplen la condición (6.6) el mismo número de veces, en cada caso.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
$-10^{-3}I_n$	26	15	6
$-10^{-2}I_n$	26	15	5
I_n	49	12	6
$-10^{2}I_n$	70	12	7
$10^{2}I_n$	66	23	15
$10^{15}I_n$	–	28	31
$10^{16}I_n$	–	30	32

Tabla 6.14: Número de iteraciones para la convergencia del **Problema 7**.

En la Tabla 6.14, podemos ver que cuando hay convergencia en los algoritmos, el número de iteraciones del algoritmo cuasi-Newton con búsqueda lineal exacta disminuye en comparación con las del algoritmo cuasi-Newton sin búsqueda lineal exacta, dicha disminución está entre 68 % y 83 % utilizando los algoritmos EQN e IQN, respectivamente.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
$-10^{-3}I_n$	15	0.0403	0	6	0.1139	11	0.0494	0
$-10^{-2}I_n$	15	0.0407	0	10	0.1534	11	0.0508	0
I_n	12	0.0382	0	6	0.0975	6	0.0421	0
$-10^{2}I_n$	12	0.0401	1	7	0.1786	7	0.0405	1
$10^{2}I_n$	23	0.0634	1	15	0.2386	15	0.0650	2
$10^{15}I_n$	28	0.0770	2	31	0.4752	19	0.0820	3
$10^{16}I_n$	30	0.0809	2	32	0.4833	20	0.0843	3

Tabla 6.15: Resultados para el **Problema 7**.

En la Tabla 6.15, observamos un buen desempeño del algoritmo IQN en términos de número de iteraciones. Por otro lado, en términos de tiempo de CPU observamos que el algoritmo EQN utiliza aproximadamente 90 % del tiempo de CPU del algoritmo EN y 29 % del tiempo de CPU del algoritmo IQN. En este problema observamos un buen desempeño de los algoritmos IQN y EQN.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
$-10^1 I_n$	13	6	7
$-10^3 I_n$	21	11	5
$10^5 I_n$	44	12	14
$10^7 I_n$	124	9	16
$10^{12} I_n$	–	13	16
$10^{15} I_n$	–	14	16

Tabla 6.16: Número de iteraciones para la convergencia del **Problema 8**.

Al aplicar la estrategia de búsqueda lineal exacta al algoritmo cuasi-Newton encontramos una disminución en el número de iteraciones que en exceso está entre 81 % a 74 % aproximadamente, lo cual se puede observar en la Tabla 6.16.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
$-10^1 I_n$	6	0.0167	0	7	0.0852	4	0.0170	0
$-10^3 I_n$	11	0.0241	1	5	0.1340	5	0.0254	0
$10^5 I_n$	12	0.0264	2	14	0.3204	6	0.0260	2
$10^7 I_n$	9	0.0225	2	16	0.3336	6	0.0264	3
$10^{12} I_n$	13	0.0296	4	16	0.3328	9	0.0395	4
$10^{15} I_n$	14	0.03058	5	16	0.3425	10	0.0411	5

Tabla 6.17: Resultados para el **Problema 8**.

En la Tabla 6.17 podemos ver que el algoritmo EN usa un menor número de iteraciones que los otros algoritmos. Por otro lado, en términos de tiempo de CPU, el algoritmo EQN funciona mejor, puesto que utiliza aproximadamente 95 % del tiempo que usa el algoritmo EN y 11 % del tiempo que usa el algoritmo IQN.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
I_n	21	11	10
$10^1 I_n$	30	11	13
$10^2 I_n$	37	10	14
$10^7 I_n$	67	14	15
$10^8 I_n$	–	18	17
$10^{10} I_n$	–	18	19

Tabla 6.18: Número de iteraciones para la convergencia del **Problema 10**.

En la Tabla 6.18, observamos una pequeña disminución en el número de iteraciones que es de aproximadamente 65 % en promedio cuando aplicamos la búsqueda lineal exacta en el algoritmo cuasi-Newton. Por otro lado, observamos convergencia en los algoritmos EQN y IQN par todos los casos.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
I_n	11	0.0458	1	10	0.1838	5	0.1406	1
$10^1 I_n$	11	0.0474	1	13	0.2671	6	0.1642	1
$10^2 I_n$	10	0.0493	2	14	0.3121	7	0.2114	2
$10^7 I_n$	14	0.0508	4	15	0.5002	9	0.3304	2
$10^8 I_n$	18	0.0890	5	17	0.5489	11	0.4010	5
$10^{10} I_n$	18	0.0889	5	19	0.5505	16	0.5824	5

Tabla 6.19: Resultados para el **Problem 10**.

De la Tabla 6.19, podemos observar un comportamiento similar al de los otros problemas, es decir, el algoritmo EN tiene un mejor desempeño con respecto al número de iteraciones. Por otro lado, con respecto al tiempo de CPU, el algoritmo EQN usa aproximadamente 23 % del tiempo que usa el algoritmo EN y 15 % del tiempo que usa el algoritmo IQN.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
0	6	4	5
I_n	5	2	2
$-10^1 I_n$	10	4	5
$-10^2 I_n$	20	3	4
$-10^4 I_n$	39	4	4
$-10^{15} I_n$	–	14	14
$10^{15} I_n$	–	12	13

Tabla 6.20: Número de iteraciones para la convergencia del **Problema 9**.

En la Tabla 6.20, observamos una disminución en el número de iteraciones de aproximadamente 82 % en promedio cuando aplicamos la búsqueda lineal exacta en el algoritmo cuasi-Newton.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
0	4	0.0324	1	5	0.1017	5	0.0347	1
I_n	2	0.0166	1	2	0.0616	2	0.0170	1
$-10^1 I_n$	4	0.0337	1	5	0.1333	5	0.0334	1
$-10^2 I_n$	3	0.0239	2	4	0.1318	4	0.0244	2
$10^4 I_n$	4	0.0299	3	4	0.1668	5	0.0342	3
$-10^{15} I_n$	14	0.1192	5	14	0.4640	13	0.1256	5
$10^{15} I_n$	12	0.1033	6	13	0.4106	12	0.1089	7

Tabla 6.21: Resultados para la convergencia del **Problema 9**.

De la Tabla 6.21, observamos un comportamiento similar en los tres algoritmos con respecto al número de iteraciones. Por otro lado, con respecto al tiempo de CPU, el algoritmo EQN usa aproximadamente 75 % del tiempo que usa el algoritmo EN y 18 % del tiempo que usa el algoritmo IQN.

6.5. Comentarios finales del capítulo

En este capítulo globalizamos el algoritmo cuasi-Newton propuesto en el Capítulo 5, introduciendo una búsqueda lineal exacta, proponemos una aproximación polino-

mial a la función de mérito y deducimos una condición suficiente para su intervalo de minimización. Usamos la función de mérito exacta y su aproximación para proponer dos algoritmos cuasi-Newton globales para resolver ecuaciones polinómicas matriciales.

Para cada una de las propuestas algorítmicas, demostramos que la búsqueda lineal exacta preserva la convergencia local del método cuasi-Newton.

Analizamos y comparamos el desempeño numérico de los dos algoritmos propuestos, primero con un algoritmo cuasi-Newton local para ver el efecto de introducir una búsqueda lineal y segundo, con un algoritmo *explícito de Newton* global para analizar su desempeño global. Las pruebas numéricas muestran la ventaja, en cuanto al número de iteraciones, de incluir una estrategia de búsqueda lineal exacta, ya que estas se reducen significativamente mejorando la convergencia local.

Con respecto a los algoritmos globales *cuasi-Newton explícito* y *cuasi-Newton implícito*, observamos un rendimiento similar en número de iteraciones, pero en tiempo de CPU, el algoritmo *cuasi-Newton explícito* se desempeña mejor, ya que usa menos tiempo, particularmente, en **Problemas 3, 7, 9** y **10** el algoritmo de *cuasi-Newton explícito* usa menos de la mitad del tiempo de CPU del algoritmo de *cuasi-Newton implícito*.

Por otro lado, el algoritmo *explícito de Newton* usa pocas iteraciones, pero más tiempo de CPU con todos los problemas; en particular, usa más del doble del tiempo de CPU del algoritmo *cuasi-Newton explícito* en **Problemas 1, 2** y **10**. Además, en el último problema, observamos que el tiempo de CPU del algoritmo *explícito de Newton* aumenta significativamente a medida que aumenta el tamaño del problema.

La diferencia en los tiempos de CPU de los algoritmos *Newton explícito*, *cuasi-Newton implícito* y *cuasi-Newton explícito* observados en la experimentación numérica tal vez se deba al cálculo de S_k y a la forma de función de mérito en los tres algoritmos.

Capítulo 7

Conclusiones y trabajos futuros

Con el propósito de resolver la ecuación polinomial matricial usando algoritmos tipo Newton, en este trabajo de investigación deducimos la forma explícita de la función de mérito que resulta al introducir una búsqueda lineal exacta en el método de Newton (un polinomio de grado $2m$), así como de su derivada y presentamos una condición necesaria y suficiente para su minimización en el intervalo $[0, 2]$. Además, como alternativa al alto costo computacional del método de Newton, tradicionalmente usado para resolver ecuaciones polinomiales, proponemos un algoritmo cuasi-Newton local y, para mejorar sus propiedades de convergencia presentamos dos propuestas de globalización de este algoritmo.

Con el fin de analizar numéricamente el desempeño del polinomio explícito en conexión con el método de Newton, lo implementamos y comparamos su desempeño con el presentado en [45] (usando función de mérito no explícita) y el propuesto en [46] (*Newton relajado*). Los resultados permiten observar que independientemente del problema, el tamaño del paso siempre tiende al paso de Newton puro; el método *Newton explícito* funciona bien, ya que siempre toma el mismo número de iteraciones que los métodos de Newton, pero se requiere menos tiempo CPU. Además, se confirma que las fórmulas explícitas presentadas para el polinomio y su derivada están bien calculadas, lo que se refleja en la igualdad de los tamaños de paso obtenidos. Por otro lado, el tiempo de ejecución de los dos algoritmos (el mejor es *Newton explícito*) muestra la ventaja de tener explícitamente la función de mérito al calcular la longitud del paso en el proceso de globalización.

Por otra parte, para el algoritmo *cuasi-Newton* local propuesto para resolver la

ecuación polinómica matricial demostramos que, bajo ciertas hipótesis, converge local y hasta cuadráticamente a una solución del problema. Su desempeño numérico frente al de Newton muestra un número de iteraciones ligeramente mayor pero, un tiempo CPU menor, en todos los problemas.

Proponemos dos algoritmos cuasi-Newton globales para resolver ecuaciones polinómicas matriciales. El primero, usando la función de mérito exacta y el segundo, una aproximación. Para cada una de las propuestas algorítmicas demostramos que la búsqueda lineal exacta no afecta la tasa de convergencia del método cuasi-Newton local. Con estos algoritmos globalizados, las pruebas numéricas muestran la ventaja, en cuanto a número de iteraciones, de incluir una estrategia de búsqueda lineal exacta, ya que este se reduce significativamente mejorando las propiedades de convergencia del método.

Es importante mencionar que en esta investigación se abordaron todos los problemas propuestos en el proyecto de tesis doctoral que dio lugar a la misma, dando cumplimiento así a los objetivos planteados. Adicionalmente, se propusieron dos algoritmos: uno tipo Newton explícito, el cual se comparó numéricamente con el tradicional y con el Newton relajado y, otro cuasi-Newton global, del cual se demostró que la estrategia de globalización usada no afecta la tasa de convergencia y mejora las propiedades de convergencia del algoritmo local.

Para finalizar, mencionamos a continuación algunos problemas que podrían dar lugar a trabajos futuros en el tema objeto de esta investigación.

1. Incorporar en la función de mérito la hipótesis de conmutatividad y analizar su desempeño teórico y numérico.
2. Relacionar el problema de los valores propios polinomial con la solución numérica de una ecuación polinómica matricial.
3. Estudiar una estrategia de globalización diferente a la búsqueda lineal exacta para la solución numérica de una ecuación polinómica matricial.
4. Extender la teoría del método de Newton inexacto a ecuaciones polinómicas matriciales.

Apéndice A

Pruebas numéricas adicionales

En este apéndice incluimos las pruebas numéricas correspondientes a los problemas de prueba (Capítulo 3) que no se consideraron el los capítulos 4, 5 y 6. Recordamos que los problemas de prueba fueron 15 en total; no obstante, en los capítulos mencionados sólo se presentan aquellos que se incluyeron en los artículos de investigación publicados o sometidos. Cabe mencionar que el desempeño de los algoritmos propuestos con los problemas que incluimos a continuación es similar al que se presenta con los otros problemas. Por esta razón no incluimos comentarios adicionales respecto a las tablas presentadas.

Para mayor claridad en la lectura, este apéndice está organizado en tres secciones, cada una de ellas indica con el nombre del capítulo al que corresponden.

A.1. Un polinomio explícito

	Newton relajado			*Newton explícito*				*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*	*con*	N	*Res*	*tiempo*
0	40	7.8899e-06	0.0781	6	4.4842e-08	0.0625	0	6	4.4842e-08	0.0625
I	31	6.3789e-06	0.0781	7	6.8553e-08	0.0781	0	6	1.8144e-07	0.1875
$10rand_n$	–	–	–	10	4.9476e-07	0.0781	0	7	1.2331e-06	0.1562
$rand_n$	45	1.7312e-06	0.0781	6	1.0716e-07	0.0625	0	6	1.0716e-07	0.0937

Tabla A.1: Resultados para el **Problema 5**.

	Newton relajado			*Newton explícito*				*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*	*con*	N	*Res*	*tiempo*
$10^{-2}I$	12	5.8264e-06	0.01562	4	7.7488e-06	0.0342	2	4	7.7269e-06	0.0468
I	14	2.5853e-06	0.0266	4	8.4811e-06	0.0343	0	4	8.4811e-06	0.0787
$10^{3}I$	19	1.9443e-06	0.0428	3	1.4366e-06	0.0312	0	2	6.9893e-06	0.0937
$10^{5}I$	25	1.3131e-06	0.0449	3	2.1387e-07	0.0156	0	2	4.8748e-07	0.1094

Tabla A.2: Resultados para el **Problema 6**.

	Newton relajado			*Newton explícito*				*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*	*con*	N	*Res*	*tiempo*
0	–	–	–	12	9.8890e-08	0.0994	0	11	1.4809e-06	0.1532
$10I$	–	–	–	11	1.5013e-06	0.0997	0	–	–	–
$10^{3}I$	–	–	–	13	2.1112e-07	0.1177	0	6	1.6417e-06	0.1580
$-10^{-2}I$	13	2.8199e-08	0.0357	8	8.6862e-07	0.0873	8	14	8.6864e-07	0.1008

Tabla A.3: Resultados para el **Problema 7**.

	Newton relajado			*Newton explícito*				*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*	*con*	N	*Res*	*tiempo*
0	3	1.4620e-09	0.0212	4	8.1468e-11	0.0608	0	4	6.1225e-10	0.1008
I	4	3.6166e-07	0.0235	5	1.1857e-08	0.0628	0	5	1.1030e-08	0.1099
rand(n)	–	–	–	6	5.9093e-06	0.0684	0	6	5.9093e-06	0.1248
-10I	12	9.0389e-08	0.0464	4	8.0464e-06	0.0597	0	4	8.0464e-06	0.1002

Tabla A.4: Resultados para el **Problema 8**.

	Newton relajado			*Newton explícito*				*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*	*con*	N	*Res*	*tiempo*
0	7	1.8469e-06	0.0312	5	2.8771e-07	0.0975	1	5	2.8771e-07	0.1094
I	5	2.0641e-06	0.0312	2	4.4910e-16	0.0312	0	2	4.0666e-11	0.0468
$10^{2}I$	–	–	–	4	3.0991e-07	0.0781	0	3	8.1763e-09	0.1250
$10^{-2}I$	7	1.7783e-06	0.0312	5	2.7362e-07	0.0937	0	5	2.7362e-07	0.0937

Tabla A.5: Resultados para el **Problema 9**.

	Newton relajado			*Newton explícito*				*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*	*con*	N	*Res*	*tiempo*
0	–	–	–	2	1.7542e-10	0.2730	0	2	6.5584e-09	0.2969
$10^1 I$	–	–	–	2	1.3497e-06	0.2656	0	2	1.3498e-06	0.2969
$10^4 I$	–	–	–	6	1.4659e-08	0.6010	0	6	1.4659e-08	0.6194

Tabla A.6: Resultados para el **Problema 11**.

	Newton relajado			*Newton explícito*				*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*	*con*	N	*Res*	*tiempo*
0	14	1.2471e-07	0.0470	5	1.0377e-09	0.0567	0	5	6.4564e-09	0.1047
I	10	2.4892e-10	0.0374	4	3.1738e-08	0.0483	0	4	3.3123e-08	0.0848
$-I$	13	2.0062e-07	0.0494	5	3.1804e-08	0.0488	0	5	3.1804e-08	0.0845
$10I$	6	2.2582e-09	0.0240	4	2.6019e-10	0.0494	0	4	4.5004e-10	0.0892

Tabla A.7: Resultados para el **Problema 12**.

	Newton relajado			*Newton explícito*				*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*	*con*	N	*Res*	*tiempo*
0	–	–	–	9	8.6004e-11	0.0731	0	7	7.5226e-09	0.0953
2I	3	8.5079e-07	0.0124	4	3.2438e-10	0.0371	0	4	1.8497e-08	0.0650
I	3	1.1764e-06	0.0161	4	1.7859e-08	0.0487	0	5	6.8923e-10	0.0953
10I	12	3.6194e-07	0.0427	5	6.4895e-10	0.0601	1	5	6.8923e-10	0.0953

Tabla A.8: Resultados para el **Problema 13**.

	Newton relajado			*Newton explícito*				*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*	*con*	N	*Res*	*tiempo*
$10^{-2} I$	–	–	–	4	5.1663e-07	0.0937	1	4	8.9077e-06	0.1406
I	–	–	–	7	6.6735e-06	0.1250	1	7	6.6735e-06	0.2278
$10^2 I$	13	2.7638e-06	0.2188	8	4.9191e-06	0.1406	1	8	4.9191e-06	0.2656
$-10^2 I$	–	–	–	7	8.3579e-06	0.1406	1	7	8.3579e-06	0.2288

Tabla A.9: Resultados para el **Problema 14**.

	Newton relajado			*Newton explícito*				*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*	*con*	N	*Res*	*tiempo*
$10I$	5	3.4806e-06	0.0368	5	1.5407e-06	0.1001	0	5	1.5407e-06	0.1216
$10^{-1}I$	1	1.0055e-09	0.0128	2	8.6164e-10	0.0508	0	2	8.6164e-10	0.0532

Tabla A.10: Resultados para el **Problema 15**.

A.2. Un método cuasi-Newton local

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	9	1.0814e-06	3.2520e-03	9	1.0814e-06	0.2188
I_n	4	1.3914e-06	2.4440e-03	4	1.9256e-12	0.1250
$10^1 I_n$	12	1.3571e-06	3.9110e-03	12	1.3571e-06	0.3121
$10^{-1} I_n$	9	8.7550e-07	3.3710e-03	9	8.7550e-07	0.2245
$10^5 I_n$	26	6.2491e-07	6.7260e-03	26	6.2473e-07	0.5964

Tabla A.11: Resultados para el **Problema 1** con $\delta = 10^{-4}$.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	7	1.5866e-13	3.0720e-03	7	1.5861e-13	9.8319e-03
I_n	6	1.0440e-12	2.8422e-03	6	1.0438e-12	9.4788e-03
$10^1 I_n$	8	1.0491e-10	3.4301e-03	8	1.0491e-10	0.011855
$10^{-1} I_n$	7	3.9978e-14	3.2310e-03	7	4.0018e-14	0.010739
$10^5 I_n$	32	2.9252e-13	8.6079e-03	32	2.9253e-13	0.038707

Tabla A.12: Resultados para el **Problema 4**.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	46	5.7610e-08	0.015625	8	2.9856e-10	0.015625
I_n	44	6.4142e-08	0.011553	7	2.3580e-10	0.010891
$10^{-1}I_n$	46	5.5524e-08	0.012149	8	2.3971e-14	0.013172
$10^{-5}I_n$	46	5.7609e-08	0.013083	8	2.9836e-10	0.011757

Tabla A.13: Resultados para el **Problema 5**.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	46	5.7610e-08	0.014774	8	2.9856e-10	0.013154
I_n	44	4.3894e-16	0.014728	7	2.3580e-10	0.011709
$-10^{1}I_n$	50	6.3018e-08	0.017261	27	5.2347e-14	0.033219
$10^{-5}I_n$	46	35.7609e-08	0.015935	8	2.9836e-10	0.017111

Tabla A.14: Resultados para el **Problema 6**.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
I_n	49	1.2404e-09	0.03333	–	–	–
$10^{2}I_n$	35	2.4682e-09	0.030120	32	7.0100e-07	0.078125
$-10^{2}I_n$	66	7.7272e-10	0.04425	21	4.3323e-10	0.05770

Tabla A.15: Resultados para el **Problema 7**.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
$10^{-2}I_n$	5	4.1674e-10	0.03002	8	2.3656e-07	0.190856
$10^{-1}I_n$	10	2.4682e-09	0.03320	10	4.4444e-06	0.2812
I_n	21	5.1216e-06	0.05824	8	4.2206e-06	0.19870

Tabla A.16: Resultados para el **Problema 10**.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	3	9.9726e-10	0.0049	3	9.9726e-10	0.2699
I_n	2	1.7052e-09	0.0026	2	1.7052e-09	0.1381
$10^1 I_n$	3	7.5003e-06	0.0070	6	1.8311e-09	0.6016

Tabla A.17: Resultados para el **Problema 11**.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	9	4.9281e-11	6.8300e-03	9	4.9281e-11	0.018155
I_n	5	1.4400e-09	4.2129e-03	5	1.4400e-09	0.011597
$10^1 I_n$	9	7.5030e-10	4.9219e-03	9	7.5030e-10	0.016490
$10^2 I_n$	15	7.6371e-13	6.8879e-03	15	7.6371e-13	0.022561
$10^{-2} I_n$	9	2.1126e-11	4.7750e-03	9	2.1126e-11	0.015794

Tabla A.18: Resultados para el **Problema 12**.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
0	36	9.8891e-07	0.014002	11	4.0481e-09	0.016340
I_n	34	8.9759e-07	0.014324	7	4.1810e-12	0.014720
$10^1 I_n$	32	8.3167e-07	0.010851	9	2.1110e-10	0.018172
$10^{-2} I_n$	35	8.3610e-07	0.012498	11	8.6998e-10	0.017093
$-10^2 I_n$	56	1.0935e-06	0.016381	47	1.2232e-11	0.058973

Tabla A.19: Resultados para el **Problema 13**.

	cuasi-Newton			*Newton*		
X_0	N	*Res*	*tiempo*	N	*Res*	*tiempo*
$10^{-2}I_n$	33	1.4131e-07	0.0187	MS	–	–
$-I_n$	28	3.0225e-06	0.0156	8	5.3423e-06	0.1273
I_n	18	5.0382e-06	0.0078	10	4.1356e-06	0.1766
$10^{1}I_n$	12	5.2694e-06	0.0587	12	1.0587e-06	0.2278

Tabla A.20: Resultados para el **Problema 14**.

A.3. Dos algoritmos cuasi-Newton globales

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
0	46	27	8
$-I_n$	46	27	8
$10^{2}I_n$	38	9	16
$10^{1}I_n$	28	15	7

Tabla A.21: Número de iteraciones para la convergencia del **Problema 5**.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
0	27	0.0546	1	9	0.1251	6	0.0732	2
$-I_n$	27	0.0558	1	9	0.1309	6	0.0745	1
$10^{2}I_n$	9	0.0347	1	16	0.2905	6	0.0702	1
$10^{1}I_n$	15	0.0359	1	7	0.109	5	0.0650	1

Tabla A.22: Resultados para **Problema 5**.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
$10^{-2}I_n$	7	3	3
$-10^{1}I_n$	9	3	3
$-10^{2}I_n$	14	2	2
$-10^{4}I_n$	25	2	2

Tabla A.23: Número de iteraciones para la convergencia del **Problema 6**.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
$10^{-2}I_n$	3	0.01398	1	3	0.0340	4	0.0158	2
$-10^{1}I_n$	3	0.0149	1	3	0.0399	4	0.0156	1
$-10^{2}I$	2	0.0132	0	2	0.0336	3	0.0140	1
$-10^{4}I$	2	0.0136	2	2	0.0330	3	0.0150	1

Tabla A.24: Resultados para **Problema 6**.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
$10^{1}I_n$	4	2	2
$10^{2}I_n$	6	3	3
$10^{4}I_n$	10	5	5
$10^{5}I_n$	13	5	5

Tabla A.25: Número de iteraciones para la convergencia del **Problema 11**.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
$10^{1}I_n$	2	0.0115	0	2	0.0231	3	0.2672	0
$10^{2}I$	3	0.0162	0	3	0.0454	5	0.4844	0
$10^{4}I_n$	5	0.0189	0	5	0.0678	6	0.5938	0
$10^{5}I_n$	5	0.0192	0	5	0.0683	6	0.5670	0

Tabla A.26: Resultados para **Problema 11**.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
0	9	4	4
$-I_n$	6	4	4
$10^1 I_n$	9	3	3
$10^2 I_n$	15	2	2

Tabla A.27: Número de iteraciones para la convergencia del **Problema 12**.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
0	4	0.0147	0	4	0.0533	5	0.0264	0
$-I_n$	4	0.0141	0	4	0.0443	5	0.0251	0
$10^1 I_n$	3	0.0137	1	3	0.0405	4	0.0221	1
$10^2 I_n$	2	0.0126	1	2	0.0354	3	0.0198	1

Tabla A.28: Resultados para **Problema 12**.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
0	36	26	13
I_n	34	27	7
$-10^{-2} I_n$	30	21	13

Tabla A.29: Número de iteraciones para la convergencia del **Problema 13**.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
0	26	0.0468	0	13	0.1449	9	0.0564	0
I_n	27	0.0487	0		0.1092	4	0.0487	0
$-10^{-2} I_n$	21	0.0445	0	13	0.1328	9	0.0552	1

Tabla A.30: Resultados para **Problema 13**.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
$10^{-2}I_n$	33	2	2
I_n	18	14	10
$10^2 I_n$	17	6	6

Tabla A.31: Número de iteraciones para la convergencia del **Problema 14**.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
$10^{-2}I_n$	2	0.0134	0	2	0.0367	4	0.0716	1
I_n	14	0.0327	0	10	0.1180	7	0.1719	0
$10^2 I$	6	0.0232	0	6	0.0826	8	0.1406	1

Tabla A.32: Resultados para **Problema 14**.

X_0	*Sin búsqueda lineal exacta*	*Con búsqueda lineal exacta*	
	Algoritmo QN	Algoritmo EQN	Algoritmo IQN
$-10^{-2}I_n$	16	13	13
$-10^{-1}I_n$	25	18	11
$-10^2 I_n$	30	18	12

Tabla A.33: Número de iteraciones para la convergencia del **Problema 15**.

	Algoritmo EQN			Algoritmo IQN		Algoritmo EN		
X_0	N	*time*	*con*	N	*time*	N	*time*	con
$10^{-2}I_n$	12	0.0547	0	4	0.0687	4	0.0680	1
$-10^{-1}I$	18	0.06676	0	11	0.1143	6	0.1098	0
$-10^2 I_n$	18	0.0690	12		0.1210	7	0.1305	1

Tabla A.34: Resultados para **Problema 15**.

Bibliografía

[1] A. S. Alfa, *Combined elapsed time and matrix-analytic method for the discrete time $gi/g/1$ and $gi^X/g/1$ systems*, Queueing Syst. **45** (2003), 5– 25.

[2] N. G. Bean, L. Bright, G. Latouche, C. E. M. Pearce, P. K. Pollet, and P. G. Taylor, *The quasi-stationary behavior of quasi-birth-anddeath processes*, Anal. Appl. Probab. **7** (1997), 134– 155.

[3] M. Berhanu, *The polynomial eigenvalue problem*, Ph.D. thesis, Manchester University, 2005.

[4] A. Bermúdez, R. G. Durán, Rodríguez R., and J. Solomin, *Finite element analysis of a quadratic eingenvalue problem arising in dissipative acoustics*, SIAM Journal **38** (2000), no. 1, 267– 291.

[5] T. Betcke, N. J. Higham, V. Mehrmann, C. Schröder, and F. Tisseur, *NLEVP: A Collection of Nonlinear Eigenvalue Problems*, ACM Transactions on Mathematical Software **39** (2013), no. 2, 1 – 28.

[6] D. Bini, G. Latouche, and B. Maini, *Solving matrix polynomial equations arising in queueing problems*, Linear Algebra and its Applications **340** (2002), 225 – 244.

[7] G. J. Butler, C. R. Johnson, and H. Wolkowicz, *Nonnegative solutions of a quadratic matrix equation arising from comparison theorems in ordinary differential equations*, SIAM J. Algebr. Discrete Methods **6** (1985), no. 1, 47–53.

[8] R. A. Canfield, *Quadratic Multipoint Exponential Approximation: Surrogate Model for Large-Scale Optimization*, Advances in Structural and Multidisciplinary Optimization, Springer International Publishing (2017), 648–661.

[9] K-W. E. Chu, *The solution of the matrix equations* $AXB - CXD = E$ *and* $(YA - DZ, YC - BZ) = (E, F)$, Linear Algebra Appl. **93** (1987), 93– 105.

[10] C . E. Davila, *A subspace approach to estimation of autoregressive parameters from noisy measurements*, IEEE Trans. Signal Process. **46** (1988), no. 2, 531 – 534.

[11] J. G. Davis, *Numerical solution of a quadratic matrix equation*, SIAM J. Sci. Statict. Comput. **2** (1981), no. 2, 164– 175.

[12] J. E. Dennis and J. J. Moré, *A characterization of superlinear convergence and its application to quasi-newton methods*, Mathematics of computation **28** (1974), no. 126, 549–560.

[13] J. E. Jr. Dennis, J. F. Traub, and R. P. Weber, *On the Matrix Polynomial, Lamda-Matrix and Block Eigenvalue Problems*, Cornell Computing and Information Science (1971), 71–109.

[14] J. E. Dennis Jr and R. B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*, SIAM Philadelphia, 1996.

[15] R. Fletcher, *Practical methods of optimization*, 2 ed., Wiley, Chichester, UK, 1987.

[16] H. G. Golub and C. F. Van Loan, *Matrix Computations*, third ed., Jonhs Hopskins University Press, 1996.

[17] J.-H. Han and H.-M. Kim, *Solving a matrix polynomials by Newton's method*, J. KSIAM. **14** (2010), no. 2, 113– 124.

[18] C. He, B. Meini, N. H. Rhee, and K. Sohraby, *A quadratically convergent Bernoulli-like algorithm for solving matrix polynomial equations in Markov chains*, Electron. Trans. **17** (2004), 151 – 167.

[19] N. J. Higham, *Functions of matrices theory and computations*, SIAM. Philadephia, 2008.

[20] N. J. Higham and H.-M. Kim, *Numerical Analysis of a Quadratic Matrix Equation*, IMA Journal of Numerical Analysis **20** (2000), no. 4, 499– 519.

[21] ———, *Solving a quadratic matrix equation by Newton's methods with exact line searches*, SIAM J. Anal. Appl. **23** (2001), no. 2, 303– 316.

[22] H. M. Kim, *Numerical methods for solving a quadratic matrix equation*, Ph.D. thesis, Manchester University, October 2000.

[23] H. J. Ko and H.-M. Kim, *Solving a Matrix Polynomial by Conjugate Gradient Methods*, J. KSIAM **11** (2007), no. 4, 39– 46.

[24] W. Kratz and E. Stickel, *Numerical solution of matrix polynomial equations by Newton's method*, IMA J. Numer. Anal. **7** (1987), 355– 369.

[25] P. Lancaster, *Lambda-Matrices and Vibrating Systems*, Pergamon Press, Oxford, 1966.

[26] P. Lancaster, L. Rodman, and I. Gohberg, *Matrix Polynomials*, Academic Press, 1982.

[27] P. Lancaster and M. Tismenetsky, *The Theory of Matrices with Applications*, second ed., Academic Press, 1985.

[28] A. J. Laub, *Efficient multivariable frequency response computations*, 1980 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes (1980), 39–41.

[29] E. M. Macías, *Métodos secantes de cambio mínimo para el cálculo de ceros de funciones de matrices*, Master's thesis, Universidad del Cauca, Diciembre 2013.

[30] E. M. Macías, F. E. Arenas, and R. Pérez, *Un algoritmo tipo Newton globalizado para resolver la ecuación cuadrática matricial*, Revista Integración Temas de Matemáticas **36** (2018), no. 2, 117 – 132.

[31] E. M. Macías, H. J. Martínez, and R. Pérez, *Un algoritmo cuasi-Newton para resolver la ecuación cuadrática matricial*, Revista Integración Temas de Matemáticas **34** (2016), no. 2, 187– 206.

[32] E. M. Macías, R. Pérez, and H. J. Martínez, *Two global quasi-Newton algorithms for solving matrix polynomial equations*, [Manuscript submitted for publication].

[33] ———, *An explicit polynomial to globalize algorithms for solving matrix polynomial equations*, Journal of Computational and Applied Mathematics **420** (2023), 114806.

[34] ———, *On the local convergence of a quasi-Newton method for solving matrix polynomial equations*, Applied Mathematics and Computation **441** (2023), 127678.

[35] MATLAB, *9.7.0.1190202 (r2019b)*, The MathWorks Inc., Natick, Massachusetts, 2018.

[36] V. Mehrmann and D. Watkins, *Polynomial eigenvalue problems with hamiltonian structure*, Electronic Transactions on Numerical Analysis **13** (2002), 106– 118.

[37] B. Meini, *Solving QBD problems: the cyclic reduction algorithm versus the invariant subspace method*, Adv. Perf. Anal. **1** (1998), 215 – 225.

[38] J. Meng, S.-H. Seo, and H.-M. Kim, *Condition Numbers and Backward Error of a Matrix Polynomial Equation Arising in Stochastic Models*, J Sci Comput **76** (2018), 759 – 776.

[39] V. Niendorf and H. Voss, *Detecting hyperbolic and definite matrix polynomials*, Linear Algebra and its Applications **432** (2010), 1017 – 1035.

[40] J. Nocedal and G. Wright, *Numerical Optimization*, Springer-Verlag, New York, 1999.

[41] E. Pereira, R. Serôdio, and J. Vitória, *Newton's method for matrix polynomials*, International Journal of Mathematics, Game Theory and Algebra **4** (2008), 183–188.

[42] E. Pereira and J. Vitória, *Deflation for block eigenvalues of block partitioned matrices with an application to matrix polynomials of commuting matrices*, Computers & Mathematics with Applications **42** (2001), 1177–1188.

[43] S. C. Reddy, P. J. Schmid, and D. S. Henningson, *Pseudospectra of the Orr-Sommerfeld Operator*, SIAM J. App. Math. **53** (1993), no. 1, 15– 47.

[44] W. Schiehlen, *Multibody Systems Handbook*, Springer-Verlag, 1990.

[45] J.-H. Seo and H.-M. Kim, *Solving matrix polynomials by Newton's method with exact line searches*, J. KSIAM **12** (2008), no. 2, 55– 68.

[46] _____, *Convergence of pure and relaxed Newton methods for solving a matrix polynomial equation arising in stochastic models*, Linear Algebra and its Applications **440** (2014), 34 – 49.

[47] S.-H. Seo, Seo J.-H., and H.-M. Kim, *Convergence of a modified Newton method for a matrix polynomial equation arising in stochastic problem*, ELA Electronic Journal of Linear Algebra **34** (2018), 113 –124.

[48] H. A. Smith, R. K. Singh, and D. C. Sorensen, *Formulation and solution of the nonlinear, damped eigenvalue problem for skeletal systems*, International Journal for Numerical Methods in Engineering **38** (1995), no. 18, 3071– 3085.

[49] D. Stowell, *Computing eigensolutions for singular Sturm-Liouville problems in photonics*, Ph.D. thesis, Southern Methodist University, Dallas, TX, USA, 2010.

[50] D. Stowell and J. Taush, *Variational formulation for guided and leaky modes in multilayer dielectric waveguides*, Commun. Comput. Phys. **7** (2010), no. 3, 564 – 579.

[51] F. Tisseur, *Backward error and condition of polynomial eigenvalue problems*, Linear Algebra Appl. **309** (2000), 339– 361.

[52] F. Tisseur and K. Meerbergen, *The quadratic eigenvalue problem*, SIAM **43** (2001), no. 2, 235 – 286.

[53] D. S. Watkins, *Fundamentals of Matrix Computations*, second ed., John Wiley & Sons Inc., 2002.