Segones Jornades sobre Mètodes Numèrics en
Ciències Aplicades i Enginyería
VALL DE NÚRIA 2003
Laboratori de Càlcul Numèric (LaCàN)

RESOLUCIÓN DE SISTEMAS DE ECUACIONES LINEALES TIPO *SPARSE*: LA ESTRATEGIA RPK

Gustavo Montero[*], Rafael Montenegro[*], José María Escobar[*] y Eduardo Rodríguez[*]

[*] Instituto de Sistemas Inteligentes y Aplicaciones Numéricas en Ingeniería,
Universitad de Las Palmas de Gran Canaria,
Edificio Instituto Polivalente, Campus Universitario de Tafira, 35017 - Las Palmas de Gran Canaria
e-mail: gustavo@dma.ulpgc.es

Palabras clave: Métodos iterativos, Reordenación, Precondicionamiento, Subespacios de Krylov.

Resumen. *Se presenta una visión general de técnicas avanzadas para la resolución de grandes sistemas de ecuaciones lineales con matriz hueca (sparse). En primer lugar se introducen diferentes algoritmos de reordenación orientados a mejorar el efecto del precondicionamiento de un sistema. Seguidamente, se define el concepto de precondicionamiento y se formulan algunos de los precondicionadores más usados en la actualidad, destacando los desarrollados recientemente basándose en la inversa aproximada de una matriz. Por otro lado, se consideran algunos métodos iterativos basados en los subespacios de Krylov para la resolución de sistemas de ecuaciones lineales. Para el caso simétrico se propone el Gradiente Conjugado, mientras que para el no simétrico, existen varias alternativas que se pueden clasificar en tres grandes familias de métodos, los de ortogonalización, los de biortogonalización y los basados en la Ecuación Normal. Esta estrategia de resolución (RPK), que combina las tres técnicas anteriores, parece la más eficiente desde el punto de vista computacional como muestran los experimentos numéricos aquí presentados.*

1. INTRODUCCIÓN

Se plantea la resolución de un sistema de ecuaciones de la forma,

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{1}$$

siendo $\mathbf{A}$ *sparse*, de orden elevado y no singular. La primera cuestión a plantear es si conviene una resolución mediante un método directo o uno iterativo. La principal desventaja de los directos frente a los iterativos es que los errores de redondeo se acumulan durante el proceso, además de requerir mayor espacio de memoria debido al efecto *fill-in*. Por otro lado, en procesos evolutivos donde se tienen que resolver diversos sistemas del mismo tipo, los métodos iterativos pueden aprovechar, como aproximación inicial, la solución obtenida en el paso de tiempo anterior. Todo ello hace que actualmente, en general, se opte por la resolución iterativa.

Las técnicas de reordenación, que se aplicaban fundamentalmente en la resolución de sistemas por métodos directos y que están basadas en la teoría de grafos, proporcionan matrices con ancho de banda o perfil menor, lo que reduce, en la resolución por métodos directos, el coste de almacenamiento en la factorización y, en cambio, son utilizadas en los métodos iterativos para mejorar el efecto del precondicionamiento (distintas formas de reordenación para diversos métodos de Krylov pueden verse en [2, 5, 9, 10, 23, 24, 25, 26])

Por otro lado, las técnicas de precondicionamiento mejoran sensiblemente la convergencia de los métodos iterativos. Además de presentar las distintas formas de precondicionamiento, por la izquierda, por la derecha y por ambos lados, se estudian algunos tipos de precondicionadores, Diagonal, el SSOR, el ILUT, un caso particular de éste, el ILU(0), y, por último, el de la Inversa Aproximada *sparse* y el de estructura Diagonal, denominada precondicionador Diagonal Óptimo. Otros tipos de precondicionadores han sido propuestos en los últimos años (ver por ejemplo [7, 29, 39]).

El tercer lugar se estudian las tres grandes familias de métodos de Krylov (ver [28, 30]): métodos de ortogonalización, los de biortogonalización y los basados en la ecuación normal.

Para el caso particular de sistemas simétricos se estudia el método del Gradiente Conjugado [18], basado en el método de ortogonalización de Lanczos [22], que es una simplificación del algoritmo de Arnoldi para sistemas simétricos.

Dentro de los métodos de ortogonalización orientados a sistemas no simétricos, que utilizan para su implementación el algoritmo de ortogonalización de Arnoldi [1], se estudia el GMRES (*Generalized Minimum Residual Method*) [33]. (Distintas versiones de este algoritmo pueden verse en [16, 20, 21, 31, 37]).

De entre los métodos basados en el algoritmo de biortogonalización de Lanczos se estudian los métodos Bi-CGSTAB (*Biconjugate Gradient Stabilized*) [36] y QMRCGSTAB [6].

Por último, de los basados en la Ecuación Normal, presentamos el LSQR (*Least-square QR*) [32].

2. RENUMERACIÓN

Las técnicas de reordenación, que se aplican fundamentalmente en la resolución de sistemas de ecuaciones lineales por métodos directos, están basadas en la teoría de grafos y proporcionan

una nueva matriz con un ancho de banda o pérfil menor, reduciendo el coste de almacenamiento y donde el efecto de relleno que se produce en la factorización queda disminuido. El objetivo que se persigue al aplicar estas técnicas a los algoritmos que hemos estudiado es conseguir que la factorización incompleta sea más cercana a la factorización completa, para que la matriz de precondicionamiento se asemeje lo más posible a la matriz del sistema inicial $\mathbf{A}$, lo que supone teóricamente, un mejor precondicionador, como reflejan los resultados obtenidos en los experimentos numéricos. Efectos similares se pueden extender al precondicionador SSOR ya que es otra aproximación de la factorización de $\mathbf{A}$.

La reordenación no afecta al almacenamiento de la matriz, ya que el número de elementos no nulos almacenados de forma compacta, se conserva aunque ocupen posiciones distinstas.

Otro objetivo es investigar si la reordenación reduce la cantidad de entradas en el precondicionador SPAI y si su uso mejora la convergencia del método iterativo.

Sea P la matriz de permutación asociada a un algoritmo de reordenación. $(P^TAP)^{-1} = P^TA^{-1}P$, es decir, la inversa de la matriz reordenada es la reordenada de la matriz inversa. Entonces al reordenar una matriz, su inversa aproximada debe tender a la inversa reordenada.

Si la tolerancia de la inversa aproximada está dada por ε, en el subespacio $\mathcal{S} \subset M_n(\mathbb{R})$,

$$\min_{M\in S} \|MA - I\|_F = \|NA - I\|_F < \varepsilon \tag{2}$$

$$\min_{M'\in P^TSP} \left\|M'P^TAP - I\right\|_F = \min_{M'\in P^TSP} \left\|PM'P^TA - I\right\|_F = \min_{M\in S} \|MA - I\| < \varepsilon \tag{3}$$

Sea $\mathcal{S}'$ un subespacio de $M_n(\mathbb{R})$ correspondiente al mismo número de entradas no nulas que $\mathcal{S}$, donde la inversa aproximada óptima es obtienida para dicho número de entradas no nulas. Cabe destacar, también, que el número de entradas no nulas en $\mathcal{S}$ es el mismo que el de $P^T\mathcal{S}P$. En este caso obtenemos,

$$\left\|N'P^TAP - I\right\|_F = \min_{M'\in S'} \left\|M'P^TAP - I\right\|_F \leq \min_{M'\in P^TSP} \left\|M'P^TAP - I\right\|_F < \varepsilon \tag{4}$$

Evidentemente, el número de entradas no nulas necesarias en S' será menor o igual que el número de entradas no nulas en P^TSP y en S. Concluimos que la reordenación reduce la cantidad de entradas no nulas en la inversa aproximada para una tolerancia dada ε, o, al menos, no lo aumenta.

Debido a los resultados dados en (4), los precondicionadores tipo inversa aproximada reordenados adquieren mejores propiedades desde el punto de vista de su utilización para mejorar la convergencia de los métodos iterativos. La cercanía del número de condición de $M'P^TAP$ a 1 se caracteriza por,

$$K_2(M'P^TAP) \leq \frac{1 + \left\|M'P^TAP - I\right\|_2}{1 - \|M'P^TAP - I\|_2} \tag{5}$$

La desviación de la normalidad de $(M'P^TAP)$ está acotada por,

$$\frac{1}{n}\sum_{k=1}^{n}(|\lambda_k| - \sigma_k)^2 \leq \frac{2}{n}\left\|M'P^TAP\right\|_F^2 (1 - \sigma_n) \tag{6}$$

siendo $\{\lambda_k\}_{k=1}^n, \{\sigma_k\}_{k=1}^n$ los autovalores y valores singulares de $P^T APM'$ (en secuencia de módulos decreciente). Finalmente, la agrupación de los autovalores y valores singulares estará dada por,

$$\sum_{k=1}^{n}(1-\sigma_k)^2 \leq \left\|M'P^TAP - I\right\|_F^2 \tag{7}$$

$$\sum_{k=1}^{n}(1-\lambda_k)^2 \leq \left\|M'P^TAP - I\right\|_F^2 \tag{8}$$

En los experimentos númericos, las técnicas que se han adaptado al esquema de almacenamiento descrito, son el algoritmo de grado mínimo (MDG, *Mínimum Degree*) propuesto por George y Liu [14], el algoritmo inverso de Cuthill-Mckee (RCM, *Reverse Cuthill-Mckee*) [13] propuesto por George para mejorar el algoritmo de Cuthill-Mckee [8] y el algoritmo del mínimo vecino (MN, *Minimum Neighboring*) [9], que es una variante del MDG.

2.1. Algoritmo de Grado Mínimo

Este algoritmo se usa para matrices con estructura *sparse* simétrica. Consiste en hacer una renumeración de los nodos en orden creciente de sus grados respectivos (conexiones de dicho nodo en el grafo asociado). Los nodos de mayor grado originarán en teoría, un mayor *fill-in*. La idea es renumerarlos al final para evitar el incremento del *fill-in* durante el proceso.

ALGORITMO MDG
1 - Construir el grafo asociado a la matriz **A**, $g(x) = \langle V, E\rangle$, donde V es el conjunto de nodos y $E = \{\{a,b\} : a \neq b \ / \ a,b \in V\}$.
2 - Mientras $V \neq \emptyset$:
  2.1- Elegir un nodo v de grado mínimo en $g(x) = \langle V, E\rangle$ y reordenar como nodo siguiente.
  2.2 - Definir:
    $V_v = V - \{v\}$,
    $E_v = \{\{a,b\} \in E \ / \ a,b \in V_v\} \cup \{\{a,b\}\, a \neq b \ / \ a,b \in Adj_g(v)\}$.
    Siendo, $Adj_g(v)$ el conjunto de nodos conectados a v en el grafo $g(x)$.
    y hacer
    $V = V_v, \ \ E = E_v$ y $g(x) = \langle V, E\rangle$.
3 - Fin

2.2. Algoritmo de Cuthill-McKee Inverso

El algoritmo de Cuthill-Mckee proporciona un método sencillo para reordenar una matriz *sparse* con objeto de reducir el efecto *fill-in*) transformando la matriz en una matriz banda. La ordenación resultante al invertir el algoritmo de Cuthill-Mckee resulta frecuentemente mejor que el original, en términos de reducción de perfil, manteniendo la anchura de banda.

ALGORITMO RCM
1 - Construir el grafo asociado a la matriz **A**, $g(x) = \langle V, E\rangle$, siendo V el conjunto de nodos y $E = \{\{a, b\} : a \neq b \ / \ a, b \in V\}$.
2 - Determinar un nodo inicial (pseudo-periférico) y renumerarlo como x_1.
3 - Renumerar los nodos conectados a x_i en orden ascendente de grado.
4 - Efectuar el ordenamiento inverso.
5 - Fin

2.3. Algoritmo del Mínimo Vecino

Este algoritmo [14] es una variante del algoritmo de grado mínimo que opera eliminando los nodos seleccionados en la estructura del grafo asociado a la matriz y de forma que no se define ni se inserta en el grafo ninguna nueva conexión. Se selecciona consecutivamente el nodo que tiene el menor número de ”vecinos”. Es especialmente útil cuando se hace una factorización incompleta con el mismo patrón de *sparsidad* que la matriz del sistema, por ejemplo cuando se utilizan precondicionadores como el ILU(0) o el SSOR.

ALGORITMO MN
1 - Construir el grafo asociado a la matriz **A**, $g(x) = \langle V, E\rangle$, donde V es el conjunto de nodos y $E = \{\{a, b\} : a \neq b \ / \ a, b \in V\}$.
2 - Mientras $V \neq \emptyset$:
  2.1- Elegir un nodo v de grado mínimo en $g(x) = \langle V, E\rangle$ y reordenar como nodo siguiente.
  2.2 - Definir:
    $V_v = V - \{v\}\,, \ \ E_v = \{\{a, b\} \in E \ / \ a, b \in V_v\}$.
    y hacer
    $V = V_v, \ \ E = E_v$ y $g(x) = \langle V, E\rangle$.
3 - Fin

2.4. Algoritmo de George

La elección del nodo inicial en el segundo paso de los algoritmos anteriores se ha determinado usando el algoritmo de George [14], que nos determina un nodo pseudo-periférico.

Si definimos la distancia $d(x, y)$ entre dos nodos x e y en un grafo $g(x) = \langle V, E\rangle$, como la longitud de la trayectoria más corta que une ambos nodos, y la excentricidad de un nodo x por $\varepsilon(x) = Max\,\{d(x, y)/x, y \in V\}$, el algoritmo se escribe de la siguiente forma,

ALGORITMO DE GEORGE PARA LA BÚSQUEDA
DE NODOS PSEUDO-PERIFÉRICOS
1- Elegir un nodo arbitrario r de V.

2 . Generar una estructura con niveles enraizada en r,

$$\left\{L_0(r), L_1(r), \ldots, L_{\varepsilon(r)}(r)\right\}.$$

siendo $L_i(r) = \{x/d(x,r) = i\}$.
3 - Elegir un nodo x de grado mínimo en $L_{\varepsilon(r)}(r)$.
4 . Generar una estructura con niveles enraizada en x,

$$\left\{L_0(x), L_1(x), \ldots, L_{\varepsilon(x)}(x)\right\}$$

5 - Si $\varepsilon(x) > \varepsilon(r)$, establecer $x \to r$ y volver al paso 3.
6 - Caso contrario tomamos x como nodo inicial.
7 - Fin

3. PRECONDICIONAMIENTO

La convergencia de los métodos basados en los subespacios de Krylov mejora con el uso de las técnicas de precondicionamiento. Éstas consisten generalmente en cambiar el sistema original $\mathbf{Ax} = \mathbf{b}$ por otro de idéntica solución, de forma que el número de condicionamiento de la matriz del nuevo sistema sea menor que el de $\mathbf{A}$, o bien tenga una mejor distribución de autovalores.

Generalmente, se considera una matriz de precondicionamiento $\mathbf{M}^{-1}$, siendo $\mathbf{M}$ una aproximación de $\mathbf{A}$, esto es,

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

tal que, $K\left(\mathbf{M^{-1}A}\right) < K\left(\mathbf{A}\right)$. El menor valor corresponde a $\mathbf{M} = \mathbf{A}$, $K\left(\mathbf{A}^{-1}\mathbf{A}\right) = 1$, que es el caso ideal y el sistema convergería en una sola iteración, pero el coste computacional del cálculo de $\mathbf{A}^{-1}$ equivaldría a resolver el sistema por un método directo. Se sugiere que $\mathbf{M}$ que sea una matriz lo más próxima a $\mathbf{A}$ sin que su determinación suponga un coste elevado.

Por otro lado, la matriz $\mathbf{M}$ debe ser fácilmente invertible para poder efectuar los productos $\mathbf{M}^{-1}$ por vector que aparecen en los algoritmos precondicionados sin excesivo coste adicional.

Dependiendo de la forma de plantear el producto de $\mathbf{M}^{-1}$ por la matriz del sistema obtendremos distintas formas de precondicionamiento. Éstas son,

$$\begin{array}{ll} \mathbf{M}^{-1}\mathbf{Ax} = \mathbf{M}^{-1}\mathbf{b} & \text{(Precondicionamiento por la izquierda)} \\ \mathbf{AM}^{-1}\mathbf{Mx} = \mathbf{b} & \text{(Precondicionamiento por la derecha)} \\ \mathbf{M}_1^{-1}\mathbf{AM}_2^{-1}\mathbf{M}_2\mathbf{x} = \mathbf{M}_1^{-1}\mathbf{b} & \text{(Precondicionamiento por ambos lados)} \end{array} \tag{9}$$

si $\mathbf{M}$ puede ser factorizada como $\mathbf{M} = \mathbf{M}_1\mathbf{M}_2$.

El campo de posibles precondicionadores es muy amplio. Algunos de los más usados son el de Jacobi, SSOR, ILU, la Inversa Aproximada y el Diagonal Óptimo.

3.1. Precondicionador de Jacobi

Surge comparando la fórmula de recurrencia para la solución que resulta de aplicar el método de Richardson, cuya relación de recurrencia viene dada por $\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha\left(\mathbf{b} - \mathbf{Ax}_i\right)$, con $\alpha > 0$, al sistema precondicionado con la fórmula correspondiente que se obtiene aplicando el

método de Jacobi al sistema sin precondicionar. De la aplicación del método de Richardson al sistema precondicionado $\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}$, se obtiene para el cálculo de los sucesivos valores de la solución,

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha \left(\mathbf{M}^{-1}\mathbf{b} - \mathbf{M}^{-1}\mathbf{A}\mathbf{x}_i\right)$$

Multiplicando por la matriz de precondicionamiento $\mathbf{M}$, queda,

$$\mathbf{M}\mathbf{x}_{i+1} = \mathbf{M}\mathbf{x}_i + \alpha \left(\mathbf{b} - \mathbf{A}\mathbf{x}_i\right) \tag{10}$$

Por otro lado, descomponiendo la matriz del sistema en $\mathbf{A} = \mathbf{D} - \mathbf{E} - \mathbf{F}$, ($\mathbf{D}$ matriz diagonal formada por los elementos de la diagonal de $\mathbf{A}$ y $\mathbf{E}$ y $\mathbf{F}$ matrices triangulares inferior y superior respectivamente), y utilizando el método de Jacobi para la resolución del sistema, se obtiene,

$$\mathbf{x}_{i+1} = \mathbf{D}^{-1}\left(\mathbf{E} + \mathbf{F}\right)\mathbf{x}_i + \mathbf{D}^{-1}\mathbf{b}$$

que multiplicando por $\mathbf{D}$ y operando resulta,

$$\mathbf{D}\mathbf{x}_{i+1} = \mathbf{D}\mathbf{x}_i + \left(\mathbf{b} - \mathbf{A}\mathbf{x}_i\right) \tag{11}$$

Comparando las expresiones de recurrencia finales de ambos métodos, se observa que el método de Jacobi aplicado al sistema sin precondicionar, equivale al de Richardson, con $\alpha = 1$, menos robusto y más simple, cuando este se aplica al sistema precondicionado con la matriz diagonal $\mathbf{D} = diag(\mathbf{A})$.

3.2. Precondicionador SSOR

Si aplicamos el método SSOR al sistema sin precondicionar, considerando la descomposición de la matriz $\mathbf{A}$ en $\mathbf{A} = \mathbf{D} - \mathbf{E} - \mathbf{F}$, como en el caso anterior, y siendo ω el parámetro de relajación, se obtiene,

$$\frac{1}{\omega\left(2-\omega\right)}\left(\mathbf{D}-\omega\mathbf{E}\right)\mathbf{D}^{-1}\left(\mathbf{D}-\omega\mathbf{F}\right)\mathbf{x}_{i+1} = \frac{1}{\omega\left(2-\omega\right)}\left(\mathbf{D}-\omega\mathbf{E}\right)\mathbf{D}^{-1}\left(\mathbf{D}-\omega\mathbf{F}\right)\mathbf{x}_i + \left(\mathbf{b} - \mathbf{A}\mathbf{x}_i\right)$$

con lo que resulta como matriz de precondicionamiento,

$$\mathbf{M} = \left(\mathbf{I}-\omega\mathbf{E}\mathbf{D}^{-1}\right)\left(\frac{\mathbf{D}-\omega\mathbf{F}}{\omega(2-\omega)}\right) \tag{12}$$

que en el caso de sistemas simétricos, podemos expresarla como,

$$\mathbf{M} = \left[\frac{\left(\mathbf{D}-\omega\mathbf{E}\right)\mathbf{D}^{-1/2}}{\sqrt{\omega\left(2-\omega\right)}}\right]\left[\frac{\left(\mathbf{D}-\omega\mathbf{E}\right)\mathbf{D}^{-1/2}}{\sqrt{\omega\left(2-\omega\right)}}\right]^T \tag{13}$$

3.3. Precondicionador ILU(0)

Resulta de la aproximación de $\mathbf{A}$ por una factorización incompleta LU, guardando las mismas entradas nulas en las matrices triangulares $\mathbf{L}$ y $\mathbf{U}$,

$$\mathbf{A} = \mathbf{L}\mathbf{U} \approx \mathbf{ILU}(0) = \mathbf{M} \tag{14}$$

donde m_{ij} son las entradas de $\mathbf{M}$ tal que,

$$m_{ij} = 0 \qquad \text{if} \qquad a_{ij} = 0 \tag{15}$$

$$\{\mathbf{A} - \mathbf{LU}\}_{ij} = 0 \qquad \text{if} \qquad a_{ij} \neq 0 \tag{16}$$

Es decir que lo elementos nulos de la matriz del sistema siguen siendo nulos en las posiciones respectivas de las matrices triangulares para no incrementar el coste computacional. Para sistemas simétricos se define, de forma similar, la factorización incompleta de Cholesky ILL^t.

3.4. Precondicionador Inversa Approximada

Recientemente, el uso de inversas aproximadas se ha convertido en una buena alternativa para los precondicionadores implícitos debido a su naturaleza paralelizable; para más detalles ver, [3, 17], y también un estudio comparativo completo en [4]. Aquí se construye una matriz inversa aproximada usando el producto escalar de Frobenius. Aunque el estudio se ha desarrollado precondicionando por la derecha, los resultados por la izquierda son directos.

Sea $\mathcal{S} \subset \mathcal{M}_n$, el subespacio de las matrices M donde se busca una inversa aproximada explícita con un patrón de *sparsidad* desconocido. La formulación del problema es:
encontrar $M_0 \in \mathcal{S}$ tal que

$$M_0 = \underset{M \in \mathcal{S}}{\arg\min} \|AM - I\|_F \tag{17}$$

Además, esta matriz inicial M_0 pudiera ser posiblemente una aproximada inversa de A en un sentido estricto, es decir,

$$\|AM_0 - I\|_F = \varepsilon < 1 \tag{18}$$

Existen dos razones para esto. Primero, la ecuación (18) permite asegurar que M_0 no es singular (lema de Banach), y segundo, esta será la base para construir un algoritmo explícito para mejorar M_0 y resolver la ecuación (1).

Un trabajo reciente de Grote y otros [17] propone un algoritmo eficiente para obtener una inversa aproximada tan cercana a una matriz no singular A como se requiera. Hemos seguido dicha técnica pero variando el método de selección de las entradas en M_0 y el algoritmo usado para resolver el problema (17). La construcción de M_0 se realiza en paralelo, independizando el cálculo de cada columna. Aunque nuestro algoritmo nos permite comenzar desde cualquier entrada de la columna k, se acepta comúnmente el uso de la diagonal como primera aproximación. Además, la expresión del precondicionador diagonal óptimo es bien conocida. La siguiente entrada a considerar se selecciona dentro del conjunto de entradas candidatas, el cual se define siguiendo el criterio propuesto en [17]. Sea r_k el residuo correspondiente a la columna k-ésima,

$$r_k = Am_k - e_k \tag{19}$$

y sea $\mathcal{I}_k$ el conjunto de índices de las entradas no nulas en r_k, es decir, $\mathcal{I}_k = \{i \in \{1, 2, ..., n\} \ / \ r_{ik} \neq 0\}$. Si $\mathcal{L}_k = \{l \in \{1, 2, ..., n\} \ / \ m_{lk} \neq 0\}$, entonces la nueva entrada se busca en el conjunto $\mathcal{J}_k = \{j \in \mathcal{L}_k^c \ / \ a_{ij} \neq 0, \forall i \in \mathcal{I}_k\}$. En realidad, las únicas entradas consideradas en m_k son aquellas que afectan las entradas no nulas de r_k. En lo que sigue, asumimos que $\mathcal{L}_k \cup \{j\} =$

$\left\{i_1^k, i_2^k, ..., i_{p_k}^k\right\}$ es no vacío, siendo p_k el número actual de entradas no nulas de m_k, y que $i_{p_k}^k = j$, para todo $j \in \mathcal{J}_k$. Para cada j, calculamos,

$$||Am_k - e_k||_2^2 = 1 - \sum_{l=1}^{p_k} \frac{\left[\det\left(D_l^k\right)\right]^2}{\det\left(G_{l-1}^k\right)\det\left(G_l^k\right)} \tag{20}$$

donde, para todo k, $\det\left(G_0^k\right) = 1$ y G_l^k es la matriz de Gram de las columnas $i_1^k, i_2^k, ..., i_l^k$ de la matriz A con respecto al producto escalar euclídeo, D_l^k es la matriz que resulta de reemplazar la última fila de la matriz G_l^k por $a_{k\,i_1^k}, a_{k\,i_2^k}, ..., a_{k\,i_l^k}$, con $1 \leq l \leq p_k$. Se selecciona el índice j_k que minimiza el valor de $\|Am_k - e_k\|_2$. Esta estrategia (ver [27]) define el nuevo índice seleccionado j_k atendiendo solamente al conjunto $\mathcal{L}_k$, lo que nos lleva a un nuevo óptimo donde se actualizan todas las entradas correspondientes a los índices de $\mathcal{L}_k$. Esto mejora el criterio de [17] donde el nuevo índice se selecciona manteniendo las entradas correspondientes a los índices de $\mathcal{L}_k$. Así m_k se busca en el conjunto $\mathcal{S}_k = \{m_k \in \mathbb{R}^n / m_{ik} = 0; \forall i \notin \mathcal{L}_k \cup \{j_k\}\}$,

$$m_k = \sum_{l=1}^{p_k} \frac{\det\left(D_l^k\right)}{\det\left(G_{l-1}^k\right)\det\left(G_l^k\right)} \widetilde{m}_l \tag{21}$$

donde $\widetilde{m}_l$ es el vector con entradas no nulas i_h^k $(1 \leq h \leq l)$. Cada una de ellas se obtiene evaluando el determinante correspondiente que resulta de reemplazar la última fila de $\det\left(G_l^k\right)$ por e_h^t, con $1 \leq l \leq p_k$.

Evidentemente, los cálculos de $||Am_k - e_k||_2^2$ y m_k pueden actualizarse añadiendo la contribución de la última entrada $j \in \mathcal{J}_k$ a la suma previa de 1 a $p_k - 1$. En la práctica, $\det\left(G_l^k\right)$ se calcula usando la descomposición de Cholesky puesto que G_l^k es una matriz simétrica y definida positiva. Esto sólo involucra la factorización de la última fila y columna si aprovechamos la descomposición de G_{l-1}^k. Por otra parte, $\det\left(D_l^k\right) / \det\left(G_l^k\right)$ es el valor de la última incógnita del sistema $G_l^k\, d_l = \left(a_{k\,i_1^k}, a_{k\,i_2^k}, ..., a_{k\,i_l^k}\right)^t$ necesitándose solamente una sustitución por descenso . Finalmente, para obtener $\widetilde{m}_l$ debe resolverse el sistema $G_l^k\, v_l = e_l$, con $\widetilde{m}_{i_h^k l} = v_{hl}$, $(1 \leq h \leq l)$.

3.5. Precondicionador Diagonal Óptimo

En el caso particular en que $\mathcal{S}$ es el subespacio de las matrices diagonales de orden n, el mejor precondicionador diagonal por la izquierda del sistema es,

$$\mathbf{M} = diag\left(\frac{a_{11}}{\|\mathbf{e}_1^T\mathbf{A}\|_2^2}, \frac{a_{22}}{\|\mathbf{e}_2^T\mathbf{A}\|_2^2}, \ldots, \frac{a_{nn}}{\|\mathbf{e}_n^T\mathbf{A}\|_2^2}\right) \tag{22}$$

$$\|\mathbf{MA} - \mathbf{I}\|_F^2 = n - \sum_{i=1}^{n} \frac{a_{ii}}{\|\mathbf{e}_i^T\mathbf{A}\|_2^2} \tag{23}$$

De forma similar se obtiene el precondicionador por la derecha.

4. MÉTODOS DE KRYLOV

4.1. Método del Gradiente Conjugado (CG)

El método del Gradiente Conjugado está basado en una técnica de proyección ortogonal sobre el subespacio de Krylov $\mathcal{K}_k(\mathbf{A};\mathbf{r}_0)$ donde $\mathbf{r}_0$ es el residuo inicial, y fundamentado en el algoritmo D-Lanczos de la siguiente forma. Una aproximación $\mathbf{x}_{j+1}$ puede ser expresada,

$$\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{p}_j \tag{24}$$

entonces, los vectores residuos deben satisfacer que,

$$\mathbf{r}_{j+1} = \mathbf{r}_j - \alpha_j \mathbf{A}\mathbf{p}_j \tag{25}$$

estos vectores residuos son ortogonales,

$$\left\langle \mathbf{r}_j - \alpha_j \mathbf{A}\mathbf{p}_j, \mathbf{r}_j \right\rangle = 0$$

por tanto,

$$\alpha_j = \frac{\langle \mathbf{r}_j, \mathbf{r}_j \rangle}{\left\langle \mathbf{A}\mathbf{p}_j, \mathbf{r}_j \right\rangle} \tag{26}$$

como las siguientes direcciones $\mathbf{p}_{j+1}$ son una combinación lineal de $\mathbf{r}_{j+1}$ y $\mathbf{p}_j$, después de reescalar los vectores $\mathbf{p}$ apropiadamente,

$$\mathbf{p}_{j+1} = \mathbf{r}_{j+1} + \beta_j \mathbf{p}_j \tag{27}$$

con lo cual,

$$\left\langle \mathbf{A}\mathbf{p}_j, \mathbf{r}_j \right\rangle = \left\langle \mathbf{A}\mathbf{p}_j, \mathbf{p}_j - \beta_{j-1}\mathbf{p}_{j-1} \right\rangle = \left\langle \mathbf{A}\mathbf{p}_j, \mathbf{p}_j \right\rangle$$

ya que $\mathbf{A}\mathbf{p}_j$ es ortogonal a $\mathbf{p}_{j-1}$. Entonces, teniendo en cuenta (26) y (27) obtenemos que,

$$\beta_j = -\frac{\left\langle \mathbf{r}_{j+1}, \mathbf{A}\mathbf{p}_j \right\rangle}{\left\langle \mathbf{p}_j, \mathbf{A}\mathbf{p}_j \right\rangle}$$

y como de (25), podemos escribir,

$$\mathbf{A}\mathbf{p}_j = -\frac{1}{\alpha_j}\left(\mathbf{r}_{j+1} - \mathbf{r}_j\right) \tag{28}$$

entonces,

$$\beta_j = -\frac{1}{\alpha_j}\frac{\langle \mathbf{r}_{j+1}, (\mathbf{r}_{j+1} - \mathbf{r}_j) \rangle}{\left\langle \mathbf{A}\mathbf{p}_j, \mathbf{p}_j \right\rangle} = \frac{\langle \mathbf{r}_{j+1}, \mathbf{r}_{j+1} \rangle}{\langle \mathbf{r}_j, \mathbf{r}_j \rangle}$$

ALGORITMO DEL GRADIENTE CONJUGADO PRECONDICIONADO (PCG)
Aproximación inicial $\mathbf{x}_0$. $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$;
Resolver $\mathbf{M}\mathbf{z}_0 = \mathbf{r}_0$, $\mathbf{p}_0 = \mathbf{z}_0$;
Mientras $\| \mathbf{r}_j \| / \| \mathbf{r}_0 \| \geq \varepsilon \quad (j = 0, 1, 2, 3, ...)$, hacer

$$\alpha_j = \frac{\langle \mathbf{r}_j, \mathbf{z}_j \rangle}{\left\langle \mathbf{A}\mathbf{p}_j, \mathbf{p}_j \right\rangle};$$
$$\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{p}_j;$$

$\mathbf{r}_{j+1} = \mathbf{r}_j - \alpha_j \mathbf{A}\mathbf{p}_j$;
Resolver $\mathbf{M}\mathbf{z}_{j+1} = \mathbf{r}_{j+1}$;
$\beta_j = \dfrac{\langle \mathbf{r}_{j+1}, \mathbf{z}_{j+1} \rangle}{\langle \mathbf{r}_j, \mathbf{z}_j \rangle}$;
$\mathbf{p}_{j+1} = \mathbf{z}_{j+1} + \beta_j \mathbf{p}_j$;
Fin

4.2. Método de Mínimo Residuo Generalizado (GMRES)

El algoritmo GMRES [33] es un método de proyección sobre $\mathcal{K} = \mathcal{K}_k$ con $\mathcal{L} = \mathbf{A}\mathcal{K}_k$ donde $\mathcal{K}_k$ es un subespacio de Krylov de dimensión k, que minimiza la norma del residuo. Así, el desarrollo del algoritmo consiste en encontrar un vector $\mathbf{x}$ de $\mathbf{x}_0 + \mathcal{K}_k$ tal que,

$$\mathbf{x} = \mathbf{x}_0 + \mathbf{V}_k \mathbf{y}$$

imponiendo la condición de mínimo para,

$$\mathbf{J}(\mathbf{y}) = \|\mathbf{b} - \mathbf{A}\mathbf{x}\| \tag{29}$$

Como,

$$\mathbf{b} - \mathbf{A}\mathbf{x} = \mathbf{b} - \mathbf{A}\left(\mathbf{x}_0 + \mathbf{V}_k \mathbf{y}\right) = \mathbf{r}_0 - \mathbf{A}\mathbf{V}_k \mathbf{y}$$

teniendo en cuenta la ecuación obtenida del algoritmo de Arnoldi

$$\mathbf{A}\mathbf{V}_k = \mathbf{V}_{k+1} \overline{\mathbf{H}}_k \tag{30}$$

y llamando y $\mathbf{v}_1 = \mathbf{r}_0 / \parallel \mathbf{r}_0 \parallel$, siendo $\beta = \parallel \mathbf{r}_0 \parallel$, entonces,

$$\mathbf{b} - \mathbf{A}\mathbf{x} = \beta \mathbf{v}_1 - \mathbf{V}_{k+1} \overline{\mathbf{H}}_k \mathbf{y}$$

pero, $\mathbf{v}_1 = \mathbf{V}_{k+1} \mathbf{e}_1$, con $\mathbf{e}_1 \in \Re^{k+1}$, por tanto:

$$\mathbf{b} - \mathbf{A}\mathbf{x} = \mathbf{V}_{k+1} \left(\beta \mathbf{e}_1 - \overline{\mathbf{H}}_k \mathbf{y}\right) \tag{31}$$

y la ecuación (29) quedará,

$$\mathbf{J}(\mathbf{y}) = \left\| \mathbf{V}_{k+1} \left(\beta \mathbf{e}_1 - \overline{\mathbf{H}}_k \mathbf{y}\right) \right\|$$

Como las columnas de la matriz $\mathbf{V}_{k+1}$ son ortonormales por construcción, podemos simplificar la anterior expresión,

$$\mathbf{J}(\mathbf{y}) = \left\| \left(\beta \mathbf{e}_1 - \overline{\mathbf{H}}_k \mathbf{y}\right) \right\| \tag{32}$$

El algoritmo GMRES busca el único vector de $\mathbf{x}_0 + \mathcal{K}_k$ que minimiza la funcional $\mathbf{J}(\mathbf{y})$.

ALGORITMO GMRES
Aproximación inicial $\mathbf{x}_0$. $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$;
Definir la $(k+1) \times k$ matriz $\overline{\mathbf{H}}_k = \{\mathbf{H}\}_{1 \le i \le k+1, 1 \le j \le k}$. Poner $\overline{\mathbf{H}}_k = 0$.
Desde $j = 1, ..., k$ hacer
$\mathbf{w}_j = \mathbf{A}\mathbf{v}_j$
Desde $i = 1, ..., j$ hacer

$\{\mathbf{H}\}_{ij} = \langle \mathbf{w}_j, \mathbf{v}_i \rangle$;
$\mathbf{w}_j = \mathbf{w}_j - \{\mathbf{H}\}_{ij}\, \mathbf{v}_i$;
Fin
$\{\mathbf{H}\}_{j+1,j} = \| \mathbf{w}_j \|$; Si $\{\mathbf{H}\}_{j+1,j} = 0$ poner $k = j$ y parar
$\mathbf{v}_{j+1} = \dfrac{1}{\{\mathbf{H}\}_{j+1,j}}\, \mathbf{w}_j$;
Fin
Hallar $\mathbf{y}_k$ que minimiza $\left\| \left(\beta \mathbf{e}_1 - \overline{\mathbf{H}}_k \mathbf{y}\right) \right\|$;
Determinar $\mathbf{x}_k = \mathbf{x}_0 + \mathbf{V}_k \mathbf{y}_k$ siendo $\mathbf{V}_k = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k]$;
Calcular $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$

El algoritmo GMRES resulta impracticable cuando k es muy grande por el elevado coste computacional y de almacenamiento, por ello se suele usar la técnica de *restart* y truncamiento.

Una modificación del GMRES consiste en utilizar un método directo para la resolución del problema de mínimos cuadrados que aparece en cada iteración [12] en lugar de la factorización QR [33] o la basada en la transformación de Householder [38]. Considérese la proyección ortogonal sobre el subespacio de soluciones mediante la multiplicación por la matriz $\mathbf{H}_k^t$,

$$\mathbf{H}_k^t\, \mathbf{H}_k\, \mathbf{u} = \mathbf{H}_k^t\, \beta_{i-1} \mathbf{e}_1 \tag{33}$$

La forma de $\mathbf{H}_k$ sugiere descomponer el producto $\mathbf{H}_k^t\, \mathbf{H}_k$ en una suma. En efecto, se advierte fácilmente que $\mathbf{H}_k$ es una matriz $(k+1) \times k$ con la siguiente estructura:

$$\mathbf{H}_k = \begin{pmatrix} \boxed{\mathbf{d}_k^t} \\ \boxed{\begin{array}{c} \mathbf{U}_k \\ \\ \mathbf{0} \end{array}} \end{pmatrix}$$

La primera fila está definida por un vector de dimensión k ($\mathbf{d}_k^t$) y el resto forma una matriz cuadrada triangular superior $\mathbf{U}_k$:

$$\{\mathbf{d}_k\}_i = d_i = \{\mathbf{H}\}_{1i} \qquad i = 1, ..., k \tag{34}$$

$$\{\mathbf{U}_k\}_{ij} = u_{ij} = \begin{cases} \{\mathbf{H}\}_{i+1\, j} & 1 \le i \le j \le k \\ 0 & \text{en el resto} \end{cases} \tag{35}$$

Teorema. Sean $\mathbf{d}_k$ y $\mathbf{U}_k$ definidos por (34) y (35), y $\bar{\mathbf{p}}_k$, $\mathbf{p}_k$, respectivamente, las soluciones de los sistemas triangulares $\mathbf{U}_k^t \bar{\mathbf{p}}_k = \mathbf{d}_k$ y $\mathbf{U}_k \mathbf{p}_k = \bar{\mathbf{p}}_k$. Si se define,

$$\lambda_i = \frac{\beta_{i-1}}{1 + \langle \mathbf{d}_k,\, \mathbf{p}_k \rangle}; \qquad \mathbf{u}_k = \lambda_i\, \mathbf{p}_k$$

entonces $\mathbf{u}_k$ minimiza la forma cuadrática $\mathbf{J}(\mathbf{u}) = \|\beta_{i-1}\mathbf{e}_1 - \mathbf{H}_k\, \mathbf{u}\|_2$ sobre R^k.

Demostración: Obsérvese en primer lugar que $1 + \langle \mathbf{d}_k,\, \mathbf{p}_k \rangle \neq 0$, ya que

$$\langle \mathbf{d}_k,\, \mathbf{p}_k \rangle = \left\langle \mathbf{U}_k^t \mathbf{U}_k \mathbf{p}_k,\, \mathbf{p}_k \right\rangle = \|\mathbf{U}_k \mathbf{p}_k\|_2^2 \ge 0$$

y, por tanto, λ_i nunca puede degenerar.

Consideremos ahora el sistema lineal dado en (33). El (i,j)-ésimo elemento de $\mathbf{H}_k^t\,\mathbf{H}_k$ es:

$$\left\{\mathbf{H}_k^t\,\mathbf{H}_k\right\}_{ij} = d_i d_j + \sum_{m=1}^{k} u_{mi}u_{mj} \tag{36}$$

En definitiva, se puede expresar,

$$\left(\mathbf{d}_k\mathbf{d}_k^t + \mathbf{U}_k^t\mathbf{U}_k\right)\mathbf{u} = \mathbf{H}_k^t\,\beta_{i-1}\mathbf{e}_1 \tag{37}$$

Teniendo en cuenta que $\mathbf{H}_k^t\mathbf{e}_1 = \mathbf{d}_k$, se obtiene la siguiente formulación:

$$\left(\mathbf{d}_k\mathbf{d}_k^t + \mathbf{U}_k^t\mathbf{U}_k\right)\mathbf{u} = \beta_{i-1}\mathbf{d}_k \tag{38}$$

Si pasamos al segundo miembro el producto externo y aplicamos la propiedad asociativa de la multiplicación de matrices, se obtiene,

$$\mathbf{U}_k^t\mathbf{U}_k\mathbf{u} = \mathbf{d}_k\left(\beta_{i-1} - \langle\mathbf{d}_k,\mathbf{u}\rangle\right) \tag{39}$$

Si definimos,

$$\lambda_i = \beta_{i-1} - \langle\mathbf{d}_k,\mathbf{u}\rangle \tag{40}$$

$$\mathbf{u} = \lambda_i\mathbf{p}_k \tag{41}$$

entonces se tiene que,

$$\mathbf{U}_k^t\mathbf{U}_k\mathbf{p}_k = \mathbf{d}_k \tag{42}$$

Esto supone resolver sólo dos sistemas triangulares, ya que $\mathbf{U}_k^t$ y $\mathbf{U}_k$ son matrices triangulares inferiores y superiores, respectivamente. Una vez resuelto este sistema, se necesita calcular λ_i para obtener $\mathbf{u}$ en la ecuación (41). Sustituyendo (41) en (40) se llega a,

$$\lambda_i = \beta_{i-1} - \langle\mathbf{d}_k,\mathbf{u}\rangle = \beta_{i-1} - \lambda_i\langle\mathbf{d}_k,\mathbf{p}_k\rangle$$

y por tanto, se demuestra que

$$\lambda_i = \frac{\beta_{i-1}}{1 + \langle\mathbf{d}_k,\ \mathbf{p}_k\rangle} \tag{43}$$

El cálculo del nuevo residuo se basa en el siguiente resultado,

$$\mathbf{r}_i = \mathbf{V}_{k+1}\left(\beta_{i-1}\mathbf{e}_1 - \mathbf{H}_k\,\mathbf{u}\right) \tag{44}$$

donde se sabe que $\mathbf{V}_{k+1} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_{k+1}]$ y es unitaria. Así, se puede escribir,

$$\mathbf{r}_i = \mathbf{V}_{k+1}\hat{\mathbf{r}}_i \tag{45}$$

representando $\hat{\mathbf{r}}_i$ el siguiente vector de dimensión $(k+1)$,

$$\hat{\mathbf{r}}_i = \beta_{i-1}\mathbf{e}_1 - \mathbf{H}_k\,\mathbf{u} \tag{46}$$

Además, es evidente que,

$$\|\mathbf{r}_i\|_2 = \|\hat{\mathbf{r}}_i\|_2 \tag{47}$$

A partir de la descomposición de $\mathbf{H}_k$, la primera componente del vector $(\mathbf{H}_k\,\mathbf{u})$ de dimensión $(k+1)$ es el producto escalar $\langle \mathbf{d}_k, \mathbf{u}\rangle$, y el resto de las componentes vienen dadas por el vector $(\mathbf{U}_k\mathbf{u})$ de dimensión k. Por tanto, la primera componente de $\hat{\mathbf{r}}_i$ es λ_i, y las otras,

$$-\mathbf{U}_k\mathbf{u} = -\lambda_i\mathbf{U}_k\mathbf{p}_k = -\lambda_i\bar{\mathbf{p}}_k \tag{48}$$

siendo posible almacenar el vector $\bar{\mathbf{p}}_k$ después del primer remonte al resolver (42).

El algoritmo Variable GMRES es una variante del GMRES-Modificado. Además de la resolución directa del problema de mínimos cuadrados [12] la idea básica es usar el *full* GMRES al principio del algoritmo hasta que satisface una cierta subtolerancia δ que puede ser función de ε, la tolerancia exigida a la solución, y se incrementa la dimensión del subespacio de Krylov k una unidad en cada paso mientras la norma del vector residuo sea mayor o igual a δ y k sea menor que la dimensión máxima permitida (por ejemplo, por exigencias de memoria) del subespacio de Krylov k_{top}. A partir de este punto, con ese último valor de k, comienza el algoritmo como si se tratase del estándar *restarted* GMRES hasta alcanzar la tolerancia ε dada [11].

ALGORITMO VGMRES
Aproximación inicial $\mathbf{x}_0$. $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$;
Elegir k_{init}, k_{top}, $\delta \in [0,1]$, $k = k_{init}$
Mientras $\| \widehat{\mathbf{r}}_{i-1} \| \,/\, \| \mathbf{r}_0 \| \geq \varepsilon \quad (i = 1, 2, 3, ...)$,
  $\beta_{i-1} = \| \mathbf{r}_{i-1} \|$, $\mathbf{v}_i = \mathbf{r}_{i-1}/\beta_{i-1}$;
  Si $\| \mathbf{r}_{i-1} \| \,/\, \| \mathbf{r}_0 \| \geq \delta$ y $k < k_{top}$ hacer $k = k+1$;
  Para $j = 1, ..., k$ hacer
    Resolver $\mathbf{M}\mathbf{z}_j = \mathbf{v}_j$;
    $\mathbf{w} = \mathbf{A}\mathbf{z}_j$;
    Para $n = 1, ..., j$ hacer
      $\{\mathbf{H}\}_{nj} = \mathbf{w}^t\mathbf{v}_n$;
      $\mathbf{w} = \mathbf{w} - \{\mathbf{H}\}_{nj}\,\mathbf{v}_n$;
    Fin
    $\{\mathbf{H}\}_{j+1,j} = \| \mathbf{w} \|$;
    $\mathbf{v}_{j+1} = \mathbf{w} / \{\mathbf{H}\}_{j+1,j}$:
  Fin
  Resolver $\mathbf{U}_k^t\bar{\mathbf{p}} = \mathbf{d}_k$ y $\mathbf{U}_k\mathbf{p} = \bar{\mathbf{p}}$;
    con $\begin{matrix}\{\mathbf{d}_k\}_m = \{\mathbf{H}\}_{1m} \\ \{\mathbf{U}_k\}_{lm} = \{\mathbf{H}\}_{l+1,m}\end{matrix}$ $l, m = 1, ..., k$;
  $\lambda_i = \dfrac{\beta_{i-1}}{1 + \mathbf{d}_k^t\,\mathbf{p}}$;
  $\mathbf{u}_k = \lambda_i\,\mathbf{p}$;
  $\mathbf{x}_i = \mathbf{x}_{i-1} + \mathbf{Z}_k\mathbf{u}_k$; siendo $\mathbf{Z}_k = [\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_k]$;
  $\mathbf{r}_i = \mathbf{Z}_{k+1}\hat{\mathbf{r}}_i$; con $\begin{matrix}\{\hat{\mathbf{r}}_i\}_1 = \lambda_i \\ \{\hat{\mathbf{r}}_i\}_{l+1} = -\lambda_i\,\{\bar{\mathbf{p}}\}_l\end{matrix}$ $l = 1, ..., k$;
Fin

Este algoritmo, al igual que el FGMRES, permite cambiar el precondicionador en cada iteración.

4.3. Método Bi-CGSTAB (*Biconjugate Gradient Stabilized*)

En el CGS [34], los vectores residuos verifican $\widehat{\mathbf{r}}_j = \phi_j^2(\mathbf{A})\mathbf{r}_0$. Van der Vorst en el Bi-CGSTAB [36], propone obtener un vector residuo por aplicación sucesiva de dos polinomios reductores distintos de la forma,

$$\mathbf{r}_j^{'} = \psi_j(\mathbf{A})\phi_j(\mathbf{A})\mathbf{r}_0 \tag{49}$$

tal que las relaciones de recurrencia del algoritmo donde intervenga este polinomio $\psi_j(\mathbf{A})$ no deben ser excesivamente complicadas y los parámetros que figuren en su definición sean fácilmente optimizables. En este sentido sugiere para $\psi_j(\mathbf{A})$ la expresión,

$$\psi_{j+1}(\mathbf{A}) = (\mathbf{I} - w_j\mathbf{A})\psi_j(\mathbf{A}) \tag{50}$$

determinando w_j, por la condición de mínimo para $\mathbf{r}_j$ en la iteración j-ésima.

Las relaciones de recurrencia se derivan de forma similar que en el algoritmo CGS. Así,

$$\psi_{j+1}(\mathbf{A})\phi_{j+1}(\mathbf{A}) = (\mathbf{I} - w_j\mathbf{A})\psi_j(\mathbf{A})\phi_{j+1}(\mathbf{A}) = (\mathbf{I} - w_j\mathbf{A})\left[\psi_j(\mathbf{A})\phi_j(\mathbf{A}) - \alpha_j\mathbf{A}\psi_j(\mathbf{A})\pi_j(\mathbf{A})\right] \tag{51}$$

Necesitamos una relación de recurrencia para la correspondiente dirección conjugada,

$$\begin{aligned}\psi_j(\mathbf{A})\pi_j(\mathbf{A}) &= \psi_j(\mathbf{A})\left[\phi_j(\mathbf{A}) + \beta_{j-1}\pi_{j-1}(\mathbf{A})\right] \\ &= \psi_j(\mathbf{A})\phi_j(\mathbf{A}) + \beta_{j-1}(\mathbf{I} - w_{j-1}\mathbf{A})\psi_{j-1}(\mathbf{A})\pi_{j-1}(\mathbf{A})\end{aligned}$$

Definimos,

$$\mathbf{r}_j = \psi_j(\mathbf{A})\phi_j(\mathbf{A})\mathbf{r}_0,$$
$$\mathbf{p}_j = \psi_j(\mathbf{A})\pi_j(\mathbf{A})\mathbf{r}_0$$

Teniendo en cuenta las anteriores relaciones, podemos encontrar las correspondientes fórmulas de recurrencia en función de los escalares α_j y β_j,

$$\mathbf{r}_{j+1} = (\mathbf{I} - w_j\mathbf{A})\left(\mathbf{r}_j - \alpha_j\mathbf{A}\mathbf{p}_j\right) \tag{52}$$

$$\mathbf{p}_{j+1} = \mathbf{r}_{j+1} + \beta_j(\mathbf{I} - w_j\mathbf{A})\mathbf{p}_j \tag{53}$$

Para obtener el algoritmo, se toma como referencia al igual que en el CGS el algoritmo BiCG, efectuando las transformaciones necesarias para que las relaciones de recurrencia de la solución sean función del nuevo vector residuo.

Recordando que en el algoritmo BiCG, $\beta_j = \rho_{j+1}/\rho_j$ con,

$$\rho_j = \left\langle \phi_j(\mathbf{A})\mathbf{r}_0, \phi_j(\mathbf{A}^T)\mathbf{r}_0^* \right\rangle = \left\langle \phi_j^2(\mathbf{A})\mathbf{r}_0, \mathbf{r}_0^* \right\rangle$$

Pero como ρ_j no lo hallamos porque ninguno de los vectores $\phi_j(\mathbf{A})\mathbf{r}_0, \phi_j(\mathbf{A}^T)\mathbf{r}_0^*$ ó $\phi_j^2(\mathbf{A})\mathbf{r}_0$ están disponibles, lo podemos relacionar con el escalar,

$$\widetilde{\rho}_j = \left\langle \phi_j(\mathbf{A})\mathbf{r}_0, \psi_j(\mathbf{A}^T)\mathbf{r}_0^* \right\rangle$$

que podemos obtener como,

$$\widetilde{\rho}_j = \langle \psi_j(\mathbf{A})\phi_j(\mathbf{A})\mathbf{r}_0, \mathbf{r}_0^* \rangle = \langle \mathbf{r}_j, \mathbf{r}_0^* \rangle$$

Desarrollando $\phi_j(\mathbf{A}^T)\mathbf{r}_0^*$ explícitamente para relacionar los escalares, ρ_j y $\widetilde{\rho}_j$, incorporando la ortogonalidad de $\phi_j(\mathbf{A})\mathbf{r}_0$ respecto de todos los vectores $(\mathbf{A}^T)^i\mathbf{r}_0^*$, con $i < j$ y teniendo en cuenta que sólo los coeficientes principales de los polinomios son relevantes en el desarrollo del anterior producto,

$$\widetilde{\rho}_j = \left\langle \phi_j(\mathbf{A})\mathbf{r}_0, \eta_1^j(\mathbf{A}^T)^j\mathbf{r}_0^* + \eta_2^j(\mathbf{A}^T)^{j-1}\mathbf{r}_0^* + ... \right\rangle$$

Si γ_1^j es el principal coeficiente para el polinomio $\phi_j(\mathbf{A})$, entonces,

$$\widetilde{\rho}_j = \left\langle \phi_j(\mathbf{A})\mathbf{r}_0, \frac{\eta_1^j}{\gamma_1^j}\phi_j(\mathbf{A}^T)\mathbf{r}_0^* \right\rangle = \frac{\eta_1^j}{\gamma_1^j}\rho_j$$

Examinando las relaciones de recurrencia para $\phi_{j+1}(\mathbf{A})$ y $\psi_{j+1}(\mathbf{A})$, los coeficientes principales para estos polinomios fueron hallados para satisfacer las relaciones,

$$\eta_1^{j+1} = -w_j\eta_1^j, \quad \gamma_1^{j+1} = -\alpha_j\gamma_1^j$$

Por tanto,

$$\frac{\widetilde{\rho}_{j+1}}{\widetilde{\rho}_j} = \frac{w_j}{\alpha_j}\frac{\rho_{j+1}}{\rho_j}$$

que nos permite encontrar la siguiente relación para β_j,

$$\beta_j = \frac{\widetilde{\rho}_{j+1}}{\widetilde{\rho}_j}\frac{\alpha_j}{w_j}$$

De forma similar, y por una simple fórmula de recurrencia podemos encontrar α_j como,

$$\alpha_j = \frac{\left\langle \phi_j(\mathbf{A})\mathbf{r}_0, \phi_j(\mathbf{A}^T)\mathbf{r}_0^* \right\rangle}{\langle \mathbf{A}\pi_j(\mathbf{A})\mathbf{r}_0, \pi_j(\mathbf{A}^T)\mathbf{r}_0^* \rangle}$$

De la misma manera que anteriormente, en los productos de polinomios, tanto en el numerador como en el denominador, sólo se consideran los términos correpondientes a sus respectivos coeficientes principales y como éstos para $\phi_j(\mathbf{A}^T)\mathbf{r}_0^*$ y $\pi_j(\mathbf{A}^T)\mathbf{r}_0^*$ son idénticos, se tiene,

$$\alpha_j = \frac{\left\langle \phi_j(\mathbf{A})\mathbf{r}_0, \phi_j(\mathbf{A}^T)\mathbf{r}_0^* \right\rangle}{\langle \mathbf{A}\pi_j(\mathbf{A})\mathbf{r}_0, \phi_j(\mathbf{A}^T)\mathbf{r}_0^* \rangle} = \frac{\left\langle \phi_j(\mathbf{A})\mathbf{r}_0, \psi_j(\mathbf{A}^T)\mathbf{r}_0^* \right\rangle}{\langle \mathbf{A}\pi_j(\mathbf{A})\mathbf{r}_0, \psi_j(\mathbf{A}^T)\mathbf{r}_0^* \rangle} = \frac{\langle \psi_j(\mathbf{A})\phi_j(\mathbf{A})\mathbf{r}_0, \mathbf{r}_0^* \rangle}{\langle \mathbf{A}\psi_j(\mathbf{A})\pi_j(\mathbf{A})\mathbf{r}_0, \mathbf{r}_0^* \rangle}$$

Y como $\mathbf{p}_j = \psi_j(\mathbf{A})\pi_j(\mathbf{A})\mathbf{r}_0$, entonces,

$$\alpha_j = \frac{\widetilde{\rho}_j}{\left\langle \mathbf{A}\mathbf{p}_j, \mathbf{r}_0^* \right\rangle} \tag{54}$$

A partir de la ecuación (52), si hacemos,

$$\mathbf{s}_j = \mathbf{r}_j - \alpha_j\mathbf{A}\mathbf{p}_j \tag{55}$$

el valor óptimo para el parámetro w_j que interviene en la construcción del polinomio reductor $\psi_j(\mathbf{A})$ y que figura en las relaciones de recurrencia del algoritmo lo obtendremos con la condición de minimizar la norma del vector residuo,

$$\mathbf{r}_{j+1} = \mathbf{s}_j - w_j \mathbf{A}\mathbf{s}_j$$

$$\|\mathbf{r}_{j+1}\|^2 = \langle \mathbf{s}_j - w_j \mathbf{A}\mathbf{s}_j, \mathbf{s}_j - w_j \mathbf{A}\mathbf{s}_j \rangle = \langle \mathbf{s}_j, \mathbf{s}_j \rangle - 2w_j \langle \mathbf{s}_j, \mathbf{A}\mathbf{s}_j \rangle + w_j^2 \langle \mathbf{A}\mathbf{s}_j, \mathbf{A}\mathbf{s}_j \rangle$$

$$\frac{\partial \|\mathbf{r}_{j+1}\|^2}{\partial w_j} = -2 \langle \mathbf{s}_j, \mathbf{A}\mathbf{s}_j \rangle + 2w_j \langle \mathbf{A}\mathbf{s}_j, \mathbf{A}\mathbf{s}_j \rangle = 0$$

con lo que,

$$w_j = \frac{\langle \mathbf{A}\mathbf{s}_j, \mathbf{s}_j \rangle}{\langle \mathbf{A}\mathbf{s}_j, \mathbf{A}\mathbf{s}_j \rangle} \tag{56}$$

La ecuación (52) ahora nos quedará,

$$\mathbf{r}_{j+1} = \mathbf{s}_j - w_j \mathbf{A}\mathbf{s}_j = \mathbf{r}_j - \alpha_j \mathbf{A}\mathbf{p}_j - w_j \mathbf{A}\mathbf{s}_j \tag{57}$$

y la relación de recurrencia para el vector solución viene dada por,

$$\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{p}_j + w_j \mathbf{s}_j \tag{58}$$

Una vez expresados todos los vectores en función del nuevo residuo y determinadas sus relaciones de recurrencia, así como los escalares que intervienen en las mismas, se puede escribir el algoritmo, en el que figuran dos productos matriz por vector y cuatro productos internos, mientras que el CGS exige los mismos productos matriz por vector y sólo dos productos internos. Sin embargo, en la mayoría de los casos la convergencia del Bi-CGSTAB es más rápida y uniforme e incluso necesita menor carga computacional para alcanzar una determinada tolerancia, pues la reducción del número de iteraciones compensa su mayor coste.

Para cada forma de precondicionamiento [28], el valor del parámetro $\widetilde{\omega}$ resulta:

$$\left.\begin{aligned}
\widetilde{\omega} &= \frac{(\mathbf{A}\mathbf{s})^T \mathbf{s}}{(\mathbf{A}\mathbf{s})^T \mathbf{A}\mathbf{s}} && \text{precondicionamiento por la derecha} \\
\widetilde{\omega} &= \frac{(\mathbf{M}^{-1}\mathbf{A}\mathbf{s})^T (\mathbf{M}^{-1}\mathbf{s})}{(\mathbf{M}^{-1}\mathbf{A}\mathbf{s})^T (\mathbf{M}^{-1}\mathbf{A}\mathbf{s})} && \text{precondicionamiento por la izquierda} \\
\widetilde{\omega} &= \frac{(\mathbf{L}^{-1}\mathbf{A}\mathbf{s})^T (\mathbf{L}^{-1}\mathbf{s})}{(\mathbf{L}^{-1}\mathbf{A}\mathbf{s})^T (\mathbf{L}^{-1}\mathbf{A}\mathbf{s})} && \text{precondicionamiento por ambos lados}
\end{aligned}\right\} \tag{59}$$

De este modo, el cálculo de $\widetilde{\omega}$ incluye un proceso de sustitución por iteración para el precondicionamiento por la izquierda y dos para el precondicionamiento por ambos lados. No obstante, si obtenemos $\widetilde{\omega}$ a partir de la minimización del residuo del sistema original sin precondicionar, todos los valores dados en la ecuación (59) coinciden con el del precondicionamiento por la derecha. En este caso se obtiene también un único algoritmo,

ALGORITMO BICGSTAB PRECONDICIONADO
Aproximación inicial $\mathbf{x}_0$. $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$;

Elegir $\mathbf{r}_0^*$ arbitrario tal $\langle \mathbf{r}_0, \mathbf{r}_0^* \rangle \neq 0$;
Resolver $\mathbf{M}\mathbf{z}_0 = \mathbf{r}_0$;
$\mathbf{p}_0 = \mathbf{z}_0$;
Mientras $\| \mathbf{r}_{j-1} \| / \| \mathbf{r}_0 \| \geq \varepsilon \;\; (j = 1, 2, 3, ...)$, hacer
Resolver $\mathbf{M}\mathbf{z}_j = \mathbf{r}_j$;
$\mathbf{y}_j = \mathbf{A}\mathbf{p}_j$;
Resolver $\mathbf{M}\mathbf{v}_j = \mathbf{y}_j$;
$\alpha_j = \dfrac{\langle \mathbf{z}_j, \mathbf{r}_0^* \rangle}{\langle \mathbf{v}_j, \mathbf{r}_0^* \rangle}$;
$\mathbf{s}_j = \mathbf{r}_j - \alpha_j \mathbf{y}_j$;
$\mathbf{u}_j = \mathbf{A}\mathbf{s}_j$;
Resolver $\mathbf{M}\mathbf{t}_j = \mathbf{u}_j$;
$\widetilde{\omega}_j = \dfrac{\langle \mathbf{t}_j, \mathbf{s}_j \rangle}{\langle \mathbf{t}_j, \mathbf{t}_j \rangle}$;
$\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{p}_j + \widetilde{\omega}_j \mathbf{u}_j$;
$\mathbf{z}_{j+1} = \mathbf{s}_j - \widetilde{\omega}_j \mathbf{t}_j$;
$\beta_j = \dfrac{\langle \mathbf{z}_{j+1}, \mathbf{r}_0^* \rangle}{\langle \mathbf{z}_j, \mathbf{r}_0^* \rangle} \times \dfrac{\alpha_j}{\widetilde{\omega}_j}$;
$\mathbf{p}_{j+1} = \mathbf{z}_{j+1} + \beta_j \left(\mathbf{p}_j - \widetilde{\omega}_j \mathbf{v}_j \right)$;
Fin

Cualquier otra elección del residuo inicial conducirá a una nueva forma de precondicionamiento (ver [35]).

4.4. Método QMRCGSTAB

Desarrollado por Chan y otros [6] está basado en la aplicación del principio de minimización usado en el algoritmo Bi-CGSTAB al método QMR. Sea,

$$\mathbf{Y}_k = [\mathbf{y}_1, \mathbf{y}_2, \ldots \mathbf{y}_k] \;\; ,$$
$$\mathbf{W}_{k+1} = [\mathbf{w}_0, \mathbf{w}_1, \ldots \mathbf{w}_k]$$

tal que,

$$\begin{cases} \mathbf{y}_{2l-1} = \mathbf{p}_l \;\; \text{para} \;\; l = 1, \ldots, [(k+1)/2] \\ \mathbf{y}_{2l} = \mathbf{s}_l \;\; \text{para} \;\; l = 1, \ldots, [k/2] \end{cases}$$

y,

$$\begin{cases} \mathbf{w}_{2l-1} = \mathbf{s}_l \;\; \text{para} \;\; l = 1, \ldots, [(k+1)/2] \\ \mathbf{w}_{2l} = \mathbf{r}_l \;\; \text{para} \;\; l = 0, 1, \ldots, [k/2] \end{cases}$$

donde $[k/2]$ y $[(k+1)/2]$ son la parte entera de $k/2$ y $(k+1)/2$ respectivamente.

Definiendo,

$$[\delta_1, \delta_2, \ldots \delta_k] \, / \begin{cases} \delta_{2l} = \omega_l \;\; \text{para} \;\; l = 1, \ldots, [(k+1)/2] \\ \delta_{2l-1} = \alpha_l \;\; \text{para} \;\; l = 1, \ldots, [(k+1)/2] \end{cases}$$

Entonces, para cada columna de $\mathbf{W}_{k+1}$ y $\mathbf{Y}_k$ las expresiones (55) y (57) resultan,

$$\mathbf{A}\mathbf{y}_m = (\mathbf{w}_{m-1} - \mathbf{w}_m)\,\delta_m^{-1}, \qquad m = 1, \ldots, k \tag{60}$$

o usando notación matricial,

$$\mathbf{A}\mathbf{Y}_k = \mathbf{W}_{k+1}\mathbf{E}_{k+1}$$

donde $\mathbf{E}_{k+1}$ es una matriz bidiagonal $(k+1)\times k$ con elementos diagonales δ_m^{-1} y en la diagonal inferior $-\delta_m^{-1}$.

Esto puede ser facilmente comprobado hasta que el grado de los polinomios correspondientes a los vectores $\mathbf{r}_j$, $\mathbf{s}_j$ y $\mathbf{p}_j$ sean $2j$, $2j-1$, y $2j-2$ respectivamente. Entonces, $\mathbf{Y}_k$ y $\mathbf{W}_k$ generarán el mismo subespacio de Krylov generado por $\mathbf{r}_0$ pero de grado $k-1$.

La idea principal del QMRCGSTAB es encontrar una aproximación a la solución del sistema (1) usando el subespacio de Krylov $\mathcal{K}_{k-1}$ en la forma,

$$\mathbf{x}_k = \mathbf{x}_0 + \mathbf{Y}_k\mathbf{g} \qquad \text{con} \qquad \mathbf{g} \in \Re^n$$

la expresión para el vector residuo queda,

$$\mathbf{r}_k = \mathbf{r}_0 - \mathbf{A}\mathbf{Y}_k\mathbf{g} = \mathbf{r}_0 - \mathbf{W}_{k+1}\mathbf{E}_{k+1}\mathbf{g}$$

Teniendo en cuenta el hecho de que el primer vector de $\mathbf{W}_{k+1}$ es justamente $\mathbf{r}_0$, entonces,

$$\mathbf{r}_k = \mathbf{W}_{k+1}\left(\mathbf{e}_1 - \mathbf{E}_{k+1}\mathbf{g}\right)$$

Como las columnas de $\mathbf{W}_{k+1}$ no están normalizadas, se usará una matriz de escala $\mathbf{\Sigma}_{k+1} = diag\left(\sigma_1, \ldots, \sigma_{k+1}\right)$ con $\sigma_j = \|\mathbf{w}_j\|$ para hacer unitarias las columnas de $\mathbf{W}_{k+1}$. Entonces,

$$\mathbf{r}_k = \mathbf{W}_{k+1}\mathbf{\Sigma}_{k+1}^{-1}\mathbf{\Sigma}_{k+1}\left(\mathbf{e}_1 - \mathbf{E}_{k+1}\mathbf{g}\right) = \mathbf{W}_{k+1}\mathbf{\Sigma}_{k+1}^{-1}\left(\sigma_1\mathbf{e}_1 - \mathbf{H}_{k+1}\mathbf{g}\right)$$

con $\mathbf{H}_{k+1} = \mathbf{\Sigma}_{k+1}\mathbf{E}_{k+1}$.

La aproximación QMR consiste en la minimización de $\|\sigma_1\mathbf{e}_1 - \mathbf{H}_{k+1}\mathbf{g}\|$ para obtener el óptimo $\mathbf{g_k} \in \Re^k$, donde este problema de mínimos cuadrados es resuelto usando la descomposición QR de la matriz $\mathbf{H}_{k+1}$ de forma incremental utilizando las rotaciones de Givens y como $\mathbf{H}_{k+1}$ es bidiagonal inferior, solamente es necesaria la rotación en los pasos previos.

ALGORITMO QMRCGSTAB PRECONDICIONADO
Aproximación inicial $\mathbf{x}_0$. $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$;
Resolver $\mathbf{M}\mathbf{z}_0 = \mathbf{r}_0$;
Elegir $\widetilde{\mathbf{r}}_0$ tal que $\langle \mathbf{z}_0, \widetilde{\mathbf{r}}_0 \rangle \neq 0$
$\mathbf{p}_0 = \mathbf{v}_0 = \mathbf{d}_0 = \mathbf{0}$;
$\rho_0 = \alpha_0 = \widetilde{\omega}_0 = 1$, $\tau_0 = \|\mathbf{z}_0\|$, $\theta_0 = 0$, $\eta_0 = 0$;
Mientras $\sqrt{j+1}\,|\widetilde{\tau}| \,/\, \|\mathbf{r}_0\| \geq \varepsilon (j = 1, 2, ...)$, hacer:
$\rho_j = \langle \mathbf{z}_{j-1}, \widetilde{\mathbf{r}}_0 \rangle$, $\beta_j = (\rho_j/\rho_{j-1})(\alpha_{j-1}/\widetilde{\omega}_{j-1})$;
$\mathbf{p}_j = \mathbf{z}_{j-1} + \beta_j(\mathbf{p}_{j-1} - \widetilde{\omega}_{j-1}\mathbf{v}_{j-1})$;
$\mathbf{y}_j = \mathbf{A}\mathbf{p}_j$;
Resolver $\mathbf{M}\mathbf{v}_j = \mathbf{y}_j$;
$\alpha_j = \rho_j / \langle \mathbf{v}_j, \widetilde{\mathbf{r}}_0 \rangle$;

$\mathbf{s}_j = \mathbf{z}_{j-1} - \alpha_j \mathbf{v}_j$;
Primera cuasi-minimización
$\widetilde{\theta}_j = \|\mathbf{s}_j\| / \tau$, $c = \dfrac{1}{\sqrt{1+\widetilde{\theta}_j^2}}$; $\widetilde{\tau} = \tau\, \widetilde{\theta}_j c$;
$\widetilde{\eta}_j = c_j^2 \alpha_j$;
$\widetilde{\mathbf{d}}_j = \mathbf{p}_j + \dfrac{\theta_{j-1}^2 \eta_{j-1}}{\alpha_j} \mathbf{d}_{j-1}$;
$\widetilde{\mathbf{x}}_j = \mathbf{x}_{j-1} + \widetilde{\eta}_j \widetilde{\mathbf{d}}_j$;
$\mathbf{u}_j = \mathbf{A}\mathbf{s}_j$;
Resolver $\mathbf{M}\mathbf{t}_j = \mathbf{u}_j$;
$\widetilde{\omega}_j = \dfrac{\langle \mathbf{s}_j, \mathbf{t}_j \rangle}{\langle \mathbf{t}_j, \mathbf{t}_j \rangle}$;
$\mathbf{z}_j = \mathbf{s}_j - \widetilde{\omega}_j \mathbf{t}_j$;
Segunda cuasi-minimización
$\theta_j = \|\mathbf{z}_j\| / \widetilde{\tau}$, $c = \dfrac{1}{\sqrt{1+\theta_j^2}}$; $\tau = \widetilde{\tau}\, \theta_j c$;
$\eta_j = c^2 \widetilde{\omega}_j$;
$\mathbf{d}_j = \mathbf{s}_j + \dfrac{\widetilde{\theta}_j^2 \widetilde{\eta}_j}{\widetilde{\omega}_j} \widetilde{\mathbf{d}}_j$;
$\mathbf{x}_j = \widetilde{\mathbf{x}}_j + \eta_j \mathbf{d}_j$;
Fin

4.5. Método LSQR (*Least-Square QR*)

La resolución del sistema de ecuaciones (1), donde la matriz $\mathbf{A}$ es no simétrica, es equivalente a resolver el sistema $\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^\mathbf{T}\mathbf{b}$, donde $\mathbf{A}^T\mathbf{A}$ es simétrica definida positiva. La ecuación $\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^\mathbf{T}\mathbf{b}$ recibe el nombre de Ecuación Normal.

Al igual que el CG, los métodos desarrollados a partir de la Ecuación Normal cumplen las dos condiciones fundamentales de minimización de la norma residual y optimización del coste computacional, sin embargo presentan el inconveniente de que el condicionamiento del nuevo sistema es el cuadrado del inicial ($K_2\left(\mathbf{A}^T\mathbf{A}\right) = K_2\left(\mathbf{A}\right)^2$), lo cual, para sistemas mal condicionados puede resultar desastroso y además en cada iteración aparecen dos productos matriz por vector correspondientes a las matrices $\mathbf{A}$ y $\mathbf{A}^T$ aumentando el coste computacional.

El método LSQR, propuesto por Paige y Saunders [32], intenta corregir el posible empeoramiento del número de condición de los sistemas al aplicar el método de Ecuación Normal. La idea básica del LSQR es hallar la solución del sistema simétrico,

$$\begin{pmatrix} \mathbf{I} & \mathbf{A} \\ \mathbf{A}^T & -\lambda^2\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{r} \\ \mathbf{x} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{0} \end{pmatrix} \tag{61}$$

minimizando,

$$\left\| \begin{pmatrix} \mathbf{A} \\ \lambda \mathbf{I} \end{pmatrix} \mathbf{x} - \begin{pmatrix} \mathbf{b} \\ \mathbf{0} \end{pmatrix} \right\|_2 \tag{62}$$

donde λ es un número real arbitrario. De la aplicación del proceso de Lanczos a este sistema simétrico resultan dos formas de bidiagonalización propuestas por Golub y Kahan [15].

Forma bidiagonal inferior

Si comenzamos tomando $\mathbf{r}_0 = \beta_1 \mathbf{u}_1, \quad \alpha_1 \mathbf{v}_1 = \mathbf{A}^T \mathbf{u}_1$

$$\left.\begin{aligned} \beta_{j+1}\mathbf{u}_{j+1} &= \mathbf{A}\mathbf{v}_j - \alpha_j \mathbf{u}_j \\ \alpha_{j+1}\mathbf{v}_{j+1} &= \mathbf{A}^T\mathbf{u}_{j+1} - \beta_{j+1}\mathbf{v}_j \end{aligned}\right\} \quad j = 1, 2, ... \tag{63}$$

los escalares $\alpha_j \geq 0$ y $\beta_j \geq 0$ son elegidos tal que $\|\mathbf{u}_j\| = \|\mathbf{v}_j\| = 1$ y,

$$\mathbf{B}_k = \begin{pmatrix} \alpha_1 & & & & \\ \beta_2 & \alpha_2 & & & \\ & \beta_3 & \alpha_3 & & \\ & & \cdots & \cdots & \\ & & & \beta_k & \alpha_k \\ & & & & \beta_{k+1} \end{pmatrix}$$

Las relaciones de recurrencia dadas en (63) pueden reescribirse,

$$\mathbf{r}_0 = \mathbf{U}_{k+1}\left(\beta_1 \mathbf{e}_1\right), \tag{64}$$

$$\mathbf{A}\mathbf{V}_k = \mathbf{U}_{k+1}\mathbf{B}_k\,, \tag{65}$$

$$\mathbf{A}^T\mathbf{U}_{k+1} = \mathbf{V}_k\mathbf{B}_k^T + \alpha_{k+1}\mathbf{v}_{k+1}\mathbf{e}_{k+1}^T \tag{66}$$

teniendo en cuenta que en aritmética exacta se verifica que $\mathbf{U}_{k+1}^T\mathbf{U}_{k+1} = \mathbf{I}$ y $\mathbf{V}_k^T\mathbf{V}_k = \mathbf{I}$.

Forma bidiagonal superior

En este caso se comienza a partir de $\mathbf{A}^T\mathbf{r}_0 = \theta_1\mathbf{v}_1, \quad \rho_1\mathbf{p}_1 = \mathbf{A}\mathbf{v}_1$

$$\left.\begin{aligned} \theta_{j+1}\mathbf{v}_{j+1} &= \mathbf{A}^T\mathbf{v}_j - \rho_j\mathbf{v}_j \\ \rho_{j+1}\mathbf{p}_{j+1} &= \mathbf{A}\mathbf{v}_{j+1} - \theta_{j+1}\mathbf{p}_j \end{aligned}\right\} \quad j = 1, 2, ... \tag{67}$$

Los escalares $\rho_j \geq 0$ y $\theta_j \geq 0$ son elegidos tal que $\|\mathbf{p}_j\| = \|\mathbf{v}_j\| = 1$ y,

$$\mathbf{R}_k = \begin{pmatrix} \rho_1 & \theta_2 & & & \\ & \rho_2 & \theta_3 & & \\ & & \cdots & \cdots & \\ & & & \rho_{k-1} & \theta_k \\ & & & & \rho_k \end{pmatrix}$$

Las relaciones dadas en (67) pueden reescribirse,

$$\mathbf{A}^T\mathbf{r}_0 = \mathbf{V}_k\left(\theta_1\mathbf{e}_1\right), \tag{68}$$

$$\mathbf{A}\mathbf{V}_k = \mathbf{P}_k\mathbf{R}_k\,, \tag{69}$$

$$\mathbf{A}^T\mathbf{P}_k = \mathbf{V}_k\mathbf{R}_k^T + \theta_{k+1}\mathbf{v}_{k+1}\mathbf{e}_k^T \tag{70}$$

y también en aritmética exacta se verifica que $\mathbf{P}_k^T\mathbf{P}_k = \mathbf{I}$ y $\mathbf{V}_k^T\mathbf{V}_k = \mathbf{I}$.

Otra forma de resolver el problema de Ecuación Normal se puede derivar de forma análoga. Definiendo $\mathbf{s} = -\mathbf{A}\mathbf{x}$, podemos escribir la ecuación (61) como,

$$\begin{pmatrix} \mathbf{I} & \mathbf{A} \\ \mathbf{A}^T & -\lambda^2\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{s} \\ \mathbf{x} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ -\mathbf{A}^T\mathbf{b} \end{pmatrix} \tag{71}$$

Aplicando de nuevo el algoritmo de Lanczos, después de $2k$ pasos se habrá transformado en,

$$\begin{pmatrix} \mathbf{I} & \mathbf{R}_k \\ \mathbf{R}_k^T & -\lambda^2\mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{q}_k \\ \mathbf{y}_k \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ -\theta_1\mathbf{e}_1 \end{pmatrix}, \tag{72}$$

$$\begin{pmatrix} \mathbf{s}_k \\ \mathbf{x}_k \end{pmatrix} = \begin{pmatrix} \mathbf{P}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_k \end{pmatrix} \begin{pmatrix} \mathbf{q}_k \\ \mathbf{y}_k \end{pmatrix} \tag{73}$$

donde $\mathbf{R}_k$ es una matriz bidiagonal superior $k \times k$ e $\mathbf{y}_k$ satisface,

$$\left(\mathbf{R}_k^T\mathbf{R}_k + \lambda^2\mathbf{I}\right)\mathbf{y}_k = \theta_1\mathbf{e}_1 \tag{74}$$

Se puede ver que las matrices generadas en ambos procesos de bidiagonalización, $\mathbf{B}_k$, $\mathbf{U}_{k+1}$, $\mathbf{R}_k$, $\mathbf{P}_k$ y $\mathbf{V}_k$ son independientes de λ, por tanto, son generadas igualmente cuando $\lambda = 0$.

Las matrices, los vectores y parámetros generados en la primera bidiagonalización son usados para resolver el problema de Ecuación Normal,

$$\text{mín}\,\|\mathbf{b} - \mathbf{A}\mathbf{x}\|$$

Los vectores,

$$\mathbf{x}_k = \mathbf{V}_k\mathbf{y}_k \tag{75}$$

$$\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k \tag{76}$$

$$\mathbf{t}_{k+1} = \beta_1\mathbf{e}_1 - \mathbf{B}_k\mathbf{y}_k \tag{77}$$

dependen de $\mathbf{y}_k$, y se tiene que,

$$\mathbf{r}_k = \mathbf{U}_{k+1}\mathbf{t}_{k+1} \tag{78}$$

Debemos minimizar $\|\mathbf{r}_k\|$ y como teóricamente las columnas de la matriz $\mathbf{U}_{k+1}$ son ortonormadas, la elección de $\mathbf{y}_k$ debe ser tal que minimice $\|\mathbf{t}_{k+1}\|$. Es decir, el problema a resolver es,

$$\text{mín}\,\|\beta_1\mathbf{e}_1 - \mathbf{B}_k\mathbf{y}_k\| \tag{79}$$

La factorización QR de $\mathbf{B}_k$ es la misma de los procesos de bidiagonalización y toma la forma,

$$\mathbf{Q}_k \begin{pmatrix} \mathbf{B}_k & \beta_1\mathbf{e}_1 \end{pmatrix} = \begin{pmatrix} \mathbf{R}_k & \mathbf{f}_k \\ & \overline{\phi}_{k+1} \end{pmatrix} \equiv \begin{pmatrix} \rho_1 & \theta_2 & & & & \phi_1 \\ & \rho_2 & \theta_3 & & & \phi_2 \\ & & \cdots & \cdots & & \cdots \\ & & & \rho_{k-1} & \theta_k & \phi_{k-1} \\ & & & & \rho_k & \phi_k \\ & & & & & \overline{\phi}_{k+1} \end{pmatrix}$$

donde $\mathbf{Q}_k \equiv \mathbf{Q}_{k,k+1}\ldots\mathbf{Q}_{2,3}\mathbf{Q}_{1,2}$ es un producto de rotación de planos y los vectores $\mathbf{y}_k$ y $\mathbf{t}_{k+1}$

pueden ser obtenidos de la forma,

$$\mathbf{R}_k\mathbf{y}_k = \mathbf{f}_k \tag{80}$$

$$\mathbf{t}_{k+1} = \mathbf{Q}_k^T \begin{pmatrix} \mathbf{0} \\ \overline{\phi}_{k+1} \end{pmatrix} \tag{81}$$

Como $\begin{pmatrix} \mathbf{R}_k & \mathbf{f}_k \end{pmatrix}$ es la misma matriz que $\begin{pmatrix} \mathbf{R}_{k-1} & \mathbf{f}_{k-1} \end{pmatrix}$ pero con una nueva fila y columna añadidas, se pueden combinar adecuadamente las ecuaciones (75) y (80) para obtener,

$$\mathbf{x}_k = \mathbf{V}_k\mathbf{R}_k^{-1}\mathbf{f}_k = \mathbf{D}_k\mathbf{f}_k \tag{82}$$

donde las columnas de $\mathbf{D}_k = \begin{pmatrix} \mathbf{d}_1 & \mathbf{d}_2 & \cdots & \mathbf{d}_k \end{pmatrix}$se obtienen sucesivamente de la resolución del sistema $\mathbf{R}_k^T\mathbf{D}_k = \mathbf{V}_k^T$. Con $\mathbf{d}_0 = \mathbf{x}_0 = \mathbf{0}$, se obtienen,

$$\mathbf{d}_k = \frac{1}{\rho_k}\left(\mathbf{v}_k - \theta_k\mathbf{d}_{k-1}\right), \tag{83}$$

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \phi_k\mathbf{d}_k \tag{84}$$

La factorización QR anterior es deteminada para construir el k-ésimo plano de rotación $\mathbf{Q}_{k,k+1}$ que opera en las filas k y $k+1$ de $\begin{pmatrix} \mathbf{B}_k & \beta_1\mathbf{e}_1 \end{pmatrix}$ para eliminar β_{k+1}. Las relaciones de recurrencia pueden expresarse,

$$\begin{pmatrix} c_k & s_k \\ s_k & -c_k \end{pmatrix}\begin{pmatrix} \overline{\rho}_k & 0 & \overline{\phi}_k \\ \beta_{k+1} & \alpha_{k+1} & 0 \end{pmatrix} = \begin{pmatrix} \rho_k & \theta_{k+1} & \phi_k \\ 0 & \overline{\rho}_{k+1} & \overline{\phi}_{k+1} \end{pmatrix} \tag{85}$$

donde $\overline{\rho}_1 = \alpha_1$, $\overline{\phi}_1 = \beta_1$ y los escalares c_k y s_k son elementos no triviales de $\mathbf{Q}_{k,k+1}$. Los escalares $\overline{\rho}_k$ y $\overline{\phi}_k$ se van reemplazando secuencialmente por ρ_k y ϕ_k. Por otro lado, en la ecuación (83) se puede usar el vector $\mathbf{w}_k = \rho_k\mathbf{d}_k$ en lugar de $\mathbf{d}_k$.

Para la estimación de $\|\mathbf{r}_k\|$, se tiene en cuenta las expresiones (78) y (81),

$$\mathbf{r}_k = \overline{\phi}_{k+1}\mathbf{Q}_k^T\mathbf{U}_{k+1}\mathbf{e}_{k+1} \tag{86}$$

y, asumiendo que $\mathbf{U}_{k+1}^T\mathbf{U}_{k+1} = I$, se obtiene,

$$\|\mathbf{r}_k\| = \overline{\phi}_{k+1} = \beta_1 s_k s_{k-1}\cdots s_1 \tag{87}$$

ALGORITMO LSQR PRECONDICIONADO
Aproximación inicial $\mathbf{x}_0$. $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$;
Resolver $\mathbf{M}\mathbf{z}_0 = \mathbf{r}_0$;
$\beta_1 = \|\mathbf{z}_0\|$, $\mathbf{u}_1 = \mathbf{z}_0/\beta_1$;
Resolver $\mathbf{M}^T\mathbf{s}_1 = \mathbf{u}_1$;
$\alpha_1 = \left\|\mathbf{A}^T\mathbf{s}_1\right\|$, $\mathbf{v}_1 = \mathbf{A}^T\mathbf{s}_1/\alpha_1$, $\mathbf{w}_1 = \mathbf{v}_1$;
$\overline{\phi}_1 = \beta_1$, $\overline{\rho}_1 = \alpha_1$;
Mientras $\overline{\phi}_j / \parallel \mathbf{r}_0 \parallel \geq \varepsilon$ ($j = 1, ...$), hacer
$\mathbf{p}_j = \mathbf{A}\mathbf{v}_j$;
Resolver $\mathbf{M}\mathbf{q}_j = \mathbf{p}_j$;
$\beta_{j+1} = \|\mathbf{q}_j - \alpha_j\mathbf{u}_j\|$;

$$\mathbf{u}_{j+1} = \frac{\mathbf{q}_j - \alpha_j \mathbf{u}_j}{\beta_{j+1}};$$
Resolver $\mathbf{M}\mathbf{s}_{j+1} = \mathbf{u}_{j+1}$;
$$\alpha_{j+1} = \left\| \mathbf{A}^T \mathbf{s}_{j+1} - \beta_{j+1} \mathbf{v}_j \right\|;$$
$$\mathbf{v}_{j+1} = \frac{\mathbf{A}^T \mathbf{s}_{j+1} - \beta_{j+1} \mathbf{v}_j}{\alpha_{j+1}};$$
$$\rho_j = \left(\overline{\rho}_j^2 + \beta_{j+1}^2 \right)^{\frac{1}{2}};$$
$$c_j = \frac{\overline{\rho}_j}{\rho_j};$$
$$s_j = \frac{\beta_{j+1}}{\rho_j};$$
$$\theta_{j+1} = s_j \alpha_{j+1};$$
$$\overline{\rho}_{j+1} = -c_j \alpha_{j+1};$$
$$\phi_j = c_j \overline{\phi}_j;$$
$$\overline{\phi}_{j+1} = s_j \overline{\phi}_j;$$
$$\mathbf{x}_j = \mathbf{x}_{j-1} + \left(\frac{\phi_j}{\rho_j} \right) \mathbf{w}_j;$$
$$\mathbf{w}_{j+1} = \mathbf{v}_{j+1} - \left(\frac{\theta_{j+1}}{\rho_j} \right) \mathbf{w}_j;$$

Fin

5. EXPERIMENTOS NUMÉRICOS

En esta sección consideramos algunos problemas obtenidos de aplicaciones científicas e industriales. Mostraremos la efectividad tanto de los precondicionadores *sparse* con tres de los métodos iterativos expuestos anteriormente. Todos los cálculos númericos se realizaron en FORTRAN con doble precisión. Se tomó siempre como aproximación inicial $x_0 = 0$, $\tilde{r}_0 = r_0 = b$, y como criterio de parada $\frac{\|b - Ax_i\|_2}{\|b\|_2} < 10^{-9}$. A continuación mostramos una breve descripción de los problemas.

En problema *LightTruck* proviene de una simulación numérica de un filtro de carbón activo en $3D$. El fenómeno físico dominante en estos dispositivos es el transporte de hidrocarburos. Huerta y otros [19], han desarrollado un modelo de convección-difusión-reacción del tipo,

$$\frac{\partial u}{\partial t} + \mathbf{v} \cdot \nabla u - \nu \Delta u + \sigma\left(u\right) \, u = f\left(u\right)$$

donde u es la concentración de hidrocarburos en el aire, $\mathbf{v}$ representa el campo de velocidades del aire, calculado previamente resolviendo un problema de flujo potencial y ν es el coeficiente de difusión. El término de reacción $\sigma\left(u\right) \, u$ y el término fuente $f\left(u\right)$ son fuertemente no lineales.

Se han considerado tres sistemas de ecuaciones correspondientes a la discretización por elementos finitos para tres pasos de tiempo diferentes (4030, 10044 y 16802) del proceso evolutivo, si bien sólo se presentan resultados para el último, al ser prácticamente los mismos en los tres

casos. La matriz de coeficientes de orden $n = 17914$, es simétrica y la misma para los tres sistemas, cambiando únicamente el segundo miembro en cada caso. Las figuras 1 y 2 muestran la comparativa de tiempos e iteraciones, respectivamente, del CG con varios precondicionadores, donde se observa que este caso el SSOR es la mejor elección. En la figura 3 se comprueba que el coste computacional del CG es muy inferior al de cualquier otro método correspondiente a sistemas no simétricos.

El ejemplo (*convdifhor*) es un problema de convección difusión definido en $[0,1] \times [0,1]$ por,

$$v_1 \frac{\partial u}{\partial x} - K\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = F$$

donde $v_1 = 10^4\left(y - \frac{1}{2}\right)\left(x - x^2\right)\left(\frac{1}{2} - x\right)$, $K = 10^{-5} - 10^2$, y $F = 10^3 - 1$.

La matriz corresponde a una malla no estructurada de elementos finitos con $n = 441$ y $nz = 2927$, $n = 1960$ y $nz = 13412$, y $n = 13190$ y $nz = 91227$, respectivamente.

La convergencia del BiCGSTAB mejora cuando se usa el precondicionador *sparse* para *convdifhor*. El número de iteraciones disminuye claramente con ε_k. Sin embargo, para obtener una inversa aproximada de A, las entradas por colmunas deben aumentarse a 200. La figura 5 se muestra que las inversas aproximadas tipo *sparse* pueden mejorar el comportamiento de algunos precondicionadores clásicos como el ILU(0) y ser competitivas cuando se realizan los cálculos en paralelos. En las figuras 4a-4b se muestran los gráficos de la estructura sparse de A y M, respectivamente, para este problema. Podemos observar que los patrones de sparsidad de A y A^{-1} son totalmente diferentes y no podemos, por tanto, definir la estructura de M igual a la de A, ni en general a la de una matriz fija.

En la figura 6 se compara la convergencia de BiCGSTAB-SPAI(0.2) para las diferentes reordenaciones en el caso de $n = 1960$. Claramente, las reordenaciones producidas por los algoritmos de Grado Mínimo y Cuthill-Mckee Inverso tiene un efecto beneficioso en la velocidad de convergencia del algoritmo BiCGSTAB-SPAI. La reducción del número de entradas en la SPAI es del 40-50 % para los casos de reordenación con Grado Mínimo y Cuthill-Mckee Inverso. El algoritmo de Mínimo Vecino no afecta a nz. Además, el número de iteraciones del BiCGSTAB se redujo mediante las renumericaciones del 60-70 %. Como estamos interesados en el efecto de la reordenación de A en las características de los precondicionadores SPAI, en las figuras 7a-7b se muestra la estructura *sparse* de la matriz M con $\varepsilon_k = 0{,}3$ para ordenación Original, Grado Mínimo, Cuthill-Mckee Inverso y Mínimo Vecino, respectivamente. La estructuras correspondientes representan matrices llenas, como cabía esperar. Sin embargo, se advierte cierto paralelismo con la estructura de A para las diferentes reordenaciones. La reducción del ancho de banda y del perfil llevada a cabo por el algoritmo de Cuthill-Mckee Inverso en la matriz A se conserva de alguna manera en la matriz M, incluso cuando existe una tendencia a explotar algunas entradas fuera del perfil. Los patrones de las matrices SPAI correspondientes a Grado Mínimo y Mínimo Vecino también conservan en parte las estructuras de la matriz A reordenada, respectivamente, aún cuando nuestro algoritmo SPAI no tiene por qué producir matrices M con estructura simétrica.

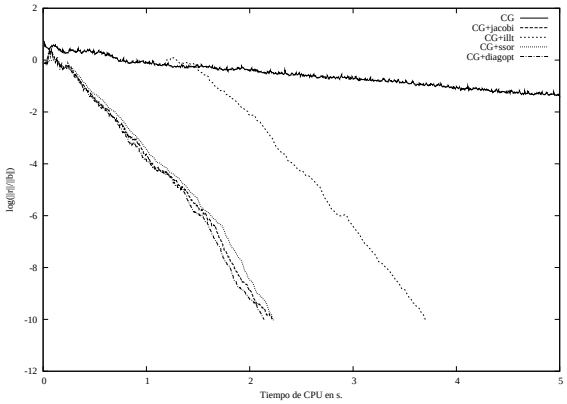


Figura 1. Comportamiento de los prencondicionadores con CG (Tiempos)

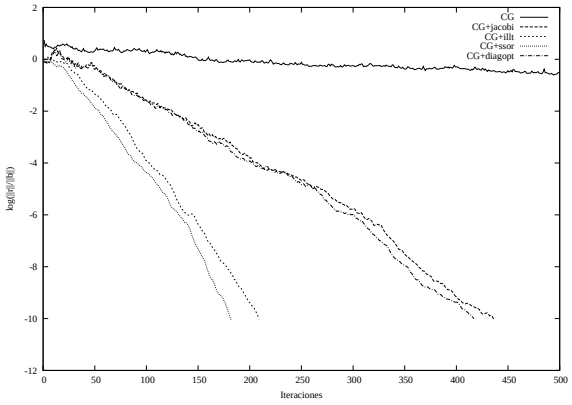


Figura 2. Comportamiento de los prencondicionadores con CG (Iteraciones)

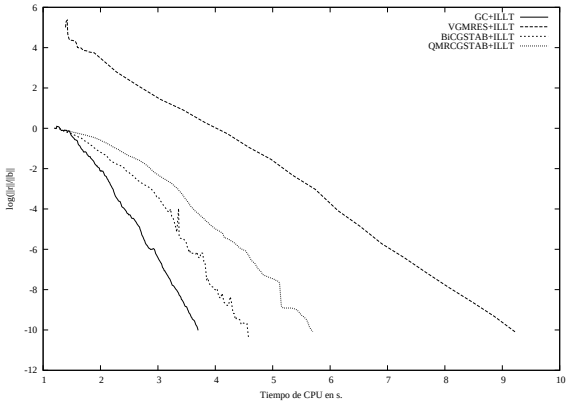
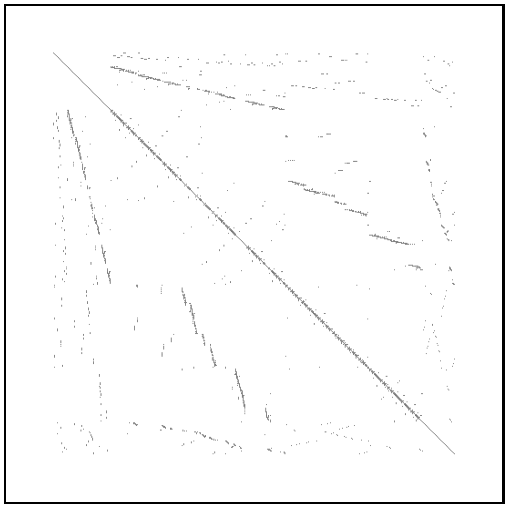
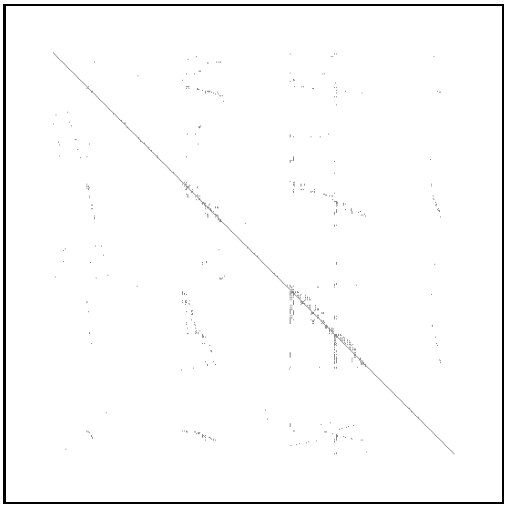


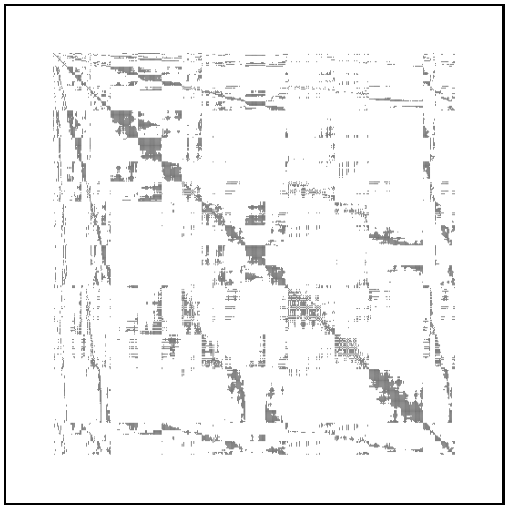
Figura 3. Comportamiento de los métodos de Krylov (Tiempos)



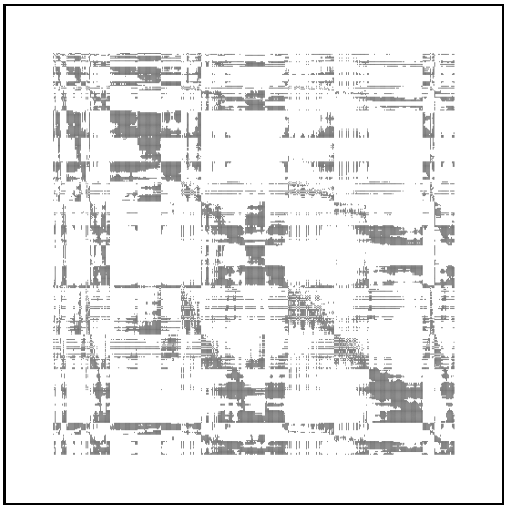
(a) Estructura *sparse* de la matriz **A**



(b) Estructura *sparse* de la inversa aproximada con $\varepsilon_k = 0{,}5$ y $\max nz(m_{0k}) = 50$



(c) Estructura *sparse* de la inversa aproximada con $\varepsilon_k = 0{,}5$ y $\max nz(m_{0k}) = 50$



(d) Estructura *sparse* de la inversa aproximada con $\varepsilon_k = 0{,}05$ y $\max nz(m_{0k}) = 200$

Figura 4. Patrón de *sparsidad* de **A** e inversas aproximadas para *convdifhor*, $n = 441$.

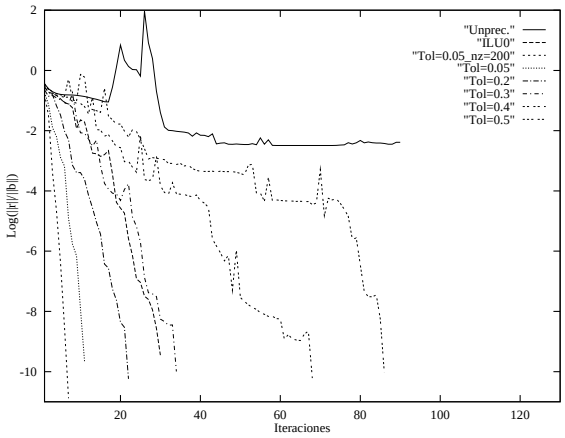


Figura 5. Comportamiento de los prencondicionadores con BiCGSTAB para *convdifhor*, $n = 441$.

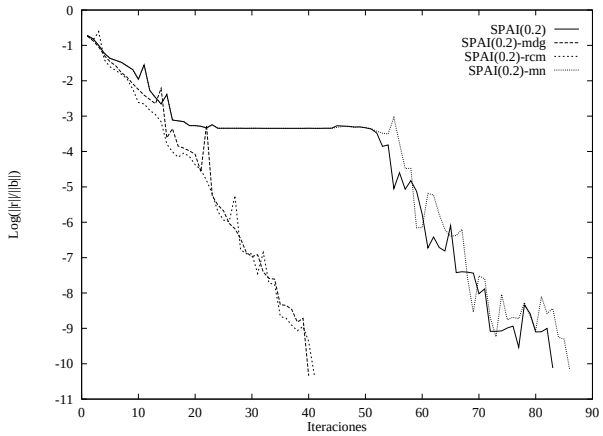
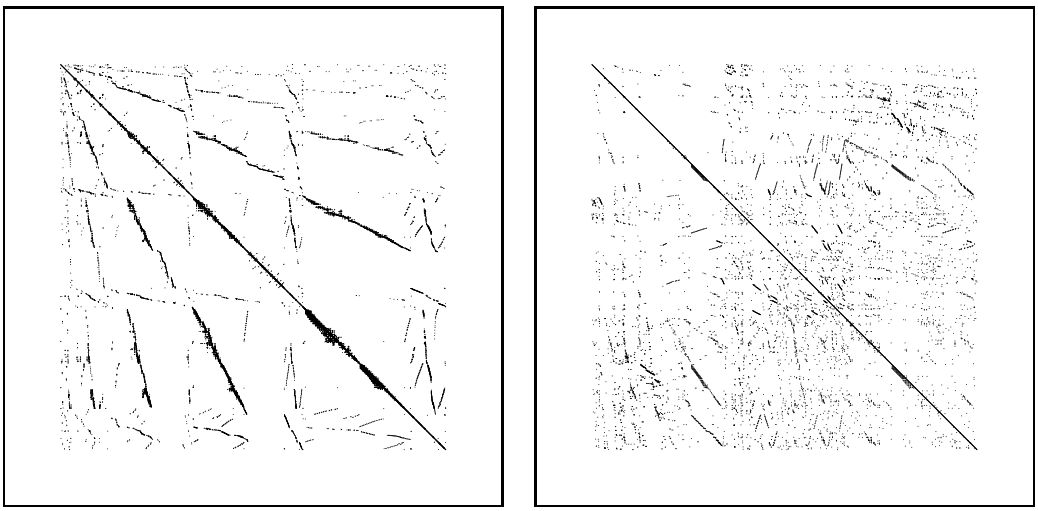


Figura 6. Comparación del comportamiento de BiCGSTAB-SPAI con reordenación para *convdifhor*, $n = 1960$.

La figura 8 muestra la convergencia de los diferentes métodos de Krylov con ILU(0) para el caso $n = 13190$. Se observa una convergencia rápida pero irregular en el BiCGSTAB. En cambio, el QMRCGSTAB elimina estas irregularidades aunque con un ligero aumento del coste. En este caso, el ILU(0) se muestra más eficiente que el resto de precondicionadores para el QMRCGSTAB, como se observa en la figura 9. Finalmente, la reordenación reduce el coste a menos del 50 % para obtener convergencia con QMRCGSTAB-ILU(0).

6. CONCLUSIONES

Los métodos basados en subespacios de Krylov ofrecen un amplio abanico de posibilidades para la resolución de sistemas de ecuaciones lineales. Si bien en el caso simétrico la elección está clara (CG), en el caso no simétrico ésta depende de varios factores: proceso evolutivo, capacidad de memoria disponible, posibilidad de grandes oscilaciones en la convergencia, ...



(a) Ordenación Original

(b) Grado Mínimo

(c) Cuthill-Mckee Inverso

(d) Mínimo Vecino

Figura 7. Patrón de *sparsidad* de la matriz SPAI(0.3) con diferentes reordenaciones para *convdifhor*, $n = 1960$.

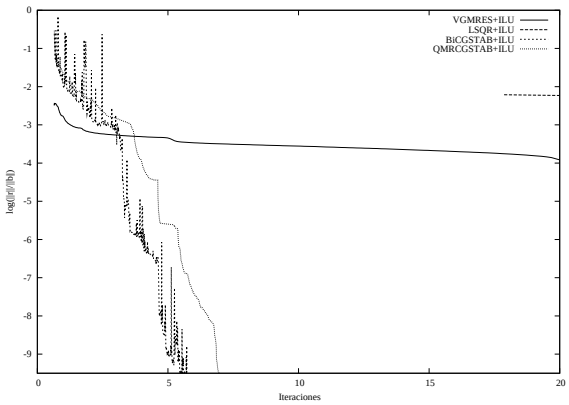


Figura 8. Comportamiento de los métodos de Krylov (Iteraciones)

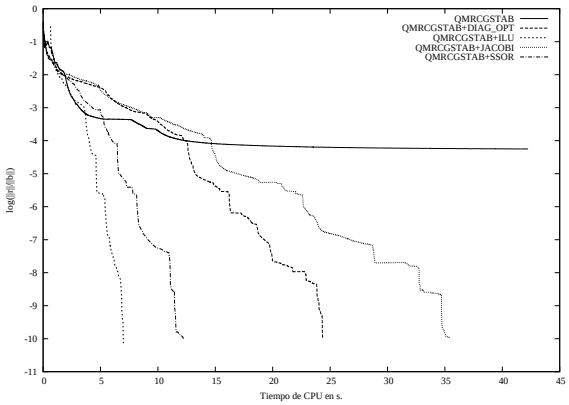


Figura 9. Comportamiento de los precondicionadores con el QMRCGSTAB (Tiempos)

El algoritmo propuesto para calcular un precondicionador *sparse* M_0 de una matriz *sparse* no simétrica A, se puede construir en paralelo ya que las columnas (o filas) de M_0 se obtienen independientemente unas de otras. El patrón de *sparsidad* de estos precondicionadores se construye dinámicamente partiendo del diagonal aumentando el número de entradas no nulas. Evidentemente, este precondicionador puede ser competitivo, si se trabaja en paralelo, con los tradicionales precondicionadores implícitos. Esto dependerá en gran medida del problema y de la arquitectura del ordenador. No obstante, se ha demostrado de forma teórica y práctica la eficacia de este precondicionador en la mejora la convergencia de los métodos iterativos.

Hemos probado experimentalmente que las técnicas de reordenación tienen efectos beneficiosos sobre los precondicionadores para la convergencia de los métodos de Krylov, y en concreto, en la ejecución de inversas aproximadas *sparse*. La reducción del número de entradas no nulas debido a la reordenación permite obtener inversas aproximadas *sparse* con una exactitud

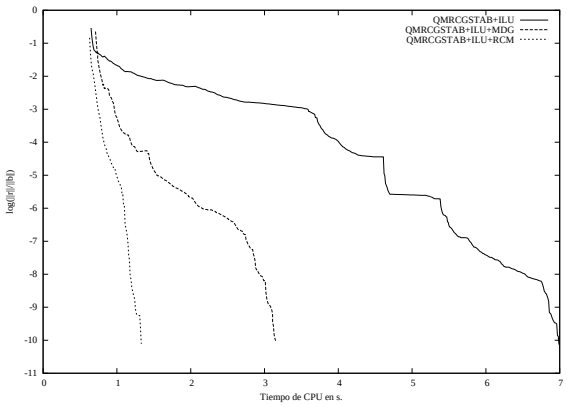


Figura 10. Efecto de la reordenación en el QMRCGSTAB precondicionado con ILU (Tiempos)

similar a las obtenidas sin reordenación, pero con menores requerimientos de almacenamiento y coste computacional. Además, la reordenación produce precondicionadores con mejores cualidades puesto que generalmente se reduce el número de pasos para alcanzar la convergencia.

Sería interesante estudiar el efecto de otras técnicas de reordenación que tengan en cuenta las entradas numéricas de A (ver [23], [26]). Aún cuando estas técnicas son caras, en el caso en que haya que resolver muchos sistemas de ecuaciones lineales con la misma matriz, dichas técnicas pueden ser competitivas en máquinas en paralelo.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente subvencionado por el Ministerio de Ciencia y Tecnología y fondos FEDER a través del proyecto REN2001-0925-C03-02/CLI.

REFERENCIAS

[1] W.E. Arnoldi, The Principle of Minimized Iteration in the Solution of the Matrix Eingenvalue Problem, *Quart. Appl. Math.,* **9**, 17-29 (1951)

[2] M. Benzi, D.B. Szyld y A. van Duin, Orderings for incomplete factorization preconditioning of nonsymmetric problems, *SIAM J. Sci. Comput.*, **20**, 5, 1652-1670, 1999

[3] M. Benzi y M. Tůma, A sparse approximate inverse preconditioner for nonsymmetric linear systems, *SIAM J. Sci. Comput.,* **19**, 3, 968-994, 1998

[4] M. Benzi y M. Tůma, A comparative study of sparse approximate inverse preconditioners, *Appl. Num. Math.*, **30**, 305-340, 1999

[5] M. Benzi y M. Tůma, Orderings for factorized sparse approximate inverse preconditioners, *SIAM J. Sci. Comput.*, to appear

[6] T.F. Chan, E. Gallopoulos, V. Simoncini, T. Szeto y C.H. Tong, A Quasi-Minimal Residual Variant of the Bi-CGSTAB Algorithm for Nonsymmetric Systems, *SIAM J. Sci. Statist. Comput.,* **15**, 338-247 (1994)

[7] T.F. Chan y T. Szeto, Composite Step Product Methods for Solving Nonsymmetric Linear Systems, *SIAM J. Sci. Comput.*, **17**, 6, 1491-1508 (1996)

[8] E.H. Cuthill y J.M. Mckee, Reducing the Bandwidth of Sparse Symmetric Matrices, *Proc. 24th National Conference of the Association for Computing Machinery,* Brondon Press, New Jersey, 157-172 (1969)

[9] L.C. Dutto, The Effect of Ordering on Preconditioned GMRES Algorithm, *Int. Jour. Num, Meth. Eng.*, **36**, 457-497 (1993)

[10] E. Flórez, M.D. García, L. González y G. Montero, The effect of orderings on sparse approximate inverse preconditioners for non-symmetric problems, *Adv. Engng. Software*, **33**, 7-10, 611-619, 2002

[11] M. Galán, G. Montero y G. Winter, Variable GMRES: an Optimizing Self-Configuring Implementation of GMRES(k) with Dynamic Memory Allocation, *Tech. Rep. of CEANI*, (1994)

[12] M. Galán, G. Montero y G. Winter, A Direct Solver for the Least Square Problem Arising From GMRES(k), *Com. Num. Meth. Eng.*, **10**, 743-749 (1994)

[13] A. George, Computer Implementation of the Finite Element Method, *Report Stan CS-71-208,* (1971)

[14] A. George y J.W. Liu, The Evolution of the Minimum Degree Ordering Algorithms, *SIAM Rev.* **31**, 1-19 (1989)

[15] G.H. Golub y Kahan W., Calculating the Singular Values and Pseudoinverse of a Matrix. *SIAM J. Numer. Anal.*, **2**, 205-224 (1965)

[16] A. Greenbaum y L.N. Trefethen, GMRES/CR and Arnoldi/Lanczos as Matrix Approximation Problems, *SIAM J. Sci. Comput.*, **15**, 2, 359-368 (1994)

[17] M. Grote y T. Huckle, Parallel preconditioning with sparse approximate inverses, *SIAM J. Sci. Comput.*, **18**, 3, 838-853, 1997

[18] M.R. Hestenes y E. Stiefel, Methods of Conjugate Gradients for Solving Linear Systems, *Jour. Res. Nat. Bur. Sta.*, **49**, 6, 409-436, (1952)

[19] A. Huerta, A. Rodríguez-Ferrán, J. Sarrate, P. Díez. y S. Fernández-Méndez, Numerical Modelling, a Tool to Improve the Design of Active Carbon Canisters, in *Abstracts of the Sixth U.S. National Congress on Computational Mechanics,* Dearborn, Michigan, (2001)

[20] W. Joubert, A Robust GMRES-based Adaptive Polynomial Preconditioning Algorithm for Nonsymmetric Linear Systems, *SIAM J. Sci. Comput.*, **15**, 2, 427-439 (1994)

[21] E.M. Kasenally, GMBACK: A Generalised Minimum Backward Error Algorithm for Nonsymmetric Linear Systems, *SIAM J. Sci. Comput.*, **16**, 3, 698-719 (1995)

[22] C. Lanczos, Solution of Systems of Linear Equations by Minimized Iterations, *Jour. Res. Nat. Bur. Sta.*, **49**, 1, 33-53, (1952)

[23] R.R. Lewis, Simulated Annealing for Profile and Fill Reduction of Sparce Matrices, *Int. Jour. Num, Meth. Eng.*, **37**, 905-925 (1994)

[24] I.L. Lim, I.W. Johnston y S.K. Choi, A Comparison of Algorithms for Profile Reduction of Sparse Matrices, *Computers & Structures*, **57**, 2, 297-302 (1995)

[25] M. Monga-Made, Ordering Strategies for Modified Block Incomplete Factorizations,

SIAM J. Sci. Comput., **16**, 2, 378-399 (1995)

[26] G. Montero, M. Galán, P. Cuesta y G. Winter, Effects of stochastic ordering on preconditioned GMRES algorithm, in B.H.V. Topping & M. Papadrakakis editors,.*Advances in Structural Optimization*, 241-246, Civil-Comp Press, Edinburgh, 1994

[27] G. Montero, L. González, E. Flórez, M.D. García y A. Suárez, Short Communication: Approximate inverse computation using Frobenius inner product, *Num. Lin. Alg. Appl.*, **9**, 239-247, (2002)

[28] G. Montero, G. Winter, A. Suárez, M. Galán y D. García, Contribution to Iterative Methods for Nonsymmetric Linear Systems: GMRES, BCG and QMR Type Methods, in *Computational Methods and Neural Networks. Part Two: Computational Methods*, M. Sambandhamy M.P. Bekakos, Eds., Dynamic Publishers, Inc., 97-128 (1999)

[29] G. Montero, R. Montenegro, G. Winter y L. Ferragut, Aplicacion de Esquemas EBE en Procesos Adaptativos, *Rev. Int. Met. Num. Cal. Dis. Ing.*, **6**, 311-332 (1990)

[30] N.M. Nachttigal, S.C. Reddy y L.N. Trefethen, How Fast Are Nonsymmetric Matrix Iterations?, *SIAM J. Matr. Anal. Appl.*, **13**, 3, 796-825 (1992)

[31] N.M. Nachttigal, L. Reichel y L.N. Trefethen, A Hybrid GMRES Algorithm for Nonsymmetric Linear Systems, *SIAM J. Matr. Anal. Appl.*, **13**, 3, 778-795 (1992)

[32] C.C. Paige y M.A. Saunders, LSQR: An Algorithm for Sparse Linear Equations and Sparse Least Squares, *ACM Trans. Math. Sof.*, **8**, 1, 43-71 (1982)

[33] Y. Saad y M. Schultz, GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems, *SIAM J. Sci. Statist. Comput.*, **7**, 856-869 (1986)

[34] P. Sonneveld, CGS: a Fast Lanczos-Type Solver for Nonsymmetric Linear Systems, *SIAM J.Sci. Statist. Comput.*, **10**, 36-52 (1989)

[35] A. Suárez, G. Montero, D. García y E. Flórez, Efecto de la Elección del Vector Inicial en la Convergencia de los Algoritmos CGS y BICGSTAB, en *Encuentro de Análisis Matricial y Aplicaciones (EAMA 97)*, Sevilla (1997)

[36] H.A. van der Vorst, Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems, *SIAM J. Sci. Comput.*, **13**, 631-644 (1992)

[37] H.A. van der Vorst y C. Vuik, GMRESR: A Family of Nested GMRES Methods, *Tech, Rep. 91-80, Delft University of Technology, Mathematics and Informatics*, Delft, The Netherlands, (1991)

[38] H.F. Walker, Implementation of the GMRES Method Using Householder Transformations, *SIAM J. Sci. Comput.*, **9**, 152-163 (1988)

[39] P.M. de Zeeuw, Incomplete Line LU as Smoother and as Preconditioner, *en Eighth GAMM-Seminar, Kiel*, 215-224 (1992)