Revista Digital: Matemática, Educación e Internet
ISSN: 1659-0643
revistadigitalmatematica@itcr.ac.cr
Instituto Tecnológico de Costa Rica
Costa Rica

Alpizar B., Geisel Y.
Factorización incompleta de Cholesky como técnica de precondicionamiento.
Revista Digital: Matemática, Educación e Internet, vol. 13, núm. 1, agosto-febrero, 2013, pp. 1-13
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica

Artículo
Revista digital Matemática, Educación e Internet (http://www.tec-digital.itcr.ac.cr/revistamatematica/). Vol 13, No 1. Agosto – Febrero 2013.

Factorización incompleta de Cholesky como técnica de precondicionamiento.

Geisel Y. Alpízar B.
galpizar@itcr.ac.cr
Escuela de Matemática
Instituto Tecnológico de Costa Rica

Resumen. Se expone la factorización incompleta de Cholesky como técnica de precondicionamiento. Se presentan experimentos numéricos que muestran la eficencia de este precondicionador, estudiando los tiempos de ejecución al resolver sistemas lineales con el método de gradiente conjugado precondicionado.

Palabras clave: Precondicionamiento, Cholesky, Gradiente Conjugado Precondicionado.

Abstract. There is exposed incomplete Cholesky factorization as preconditioning technique. The article presents numerical experiments that show the efficiency of this preconditioner, studying the times of execution by solving linear systems with the preconditioned conjugate gradient method.

KeyWords. Preconditioning, Cholesky, Preconditioned Conjugate Gradient.

1.1 Introducción

Muchos procesos de la vida real, por ejemplo procesos físicos y de ingeniería, pueden ser modelados por una ecuación diferencial parcial o un sistema de ecuaciones diferenciales parciales cuya discretización en muchas ocasiones genera grandes sistemas de ecuaciones lineales. Razón por la cual ha sido de interés el estudio de los métodos para aproximar la solución de estos sistemas.

Al resolver un sistema de ecuaciones lineales la primera pregunta que nos hacemos es si conviene utilizar un método directo o uno iterativo. La principal desventaja de los métodos directos, como es el método de eliminación Gaussiana, es su complejidad computacional. Para una matriz de tamaño $n \times n$, el método de eliminación Gaussiana es de orden $\mathcal{O}(n^3)$ lo que para matrices muy grandes es bastante costoso. Una solución a este inconveniente la dan los métodos iterativos como por ejemplo: Jacobi, Gauss-Seidel, SOR, gradiente conjugado; explicados en [4]. Estos métodos poseen una complejidad menor, lo que los hace más eficientes. Además, los errores de redondeo, que se pueden producir al trabajar con precisión finita en la aplicación de los métodos directos, hacen más adecuados los métodos iterativos. En la práctica la mayoría de las veces no interesa tener varios de los dígitos de la solución, por lo que los métodos iterativos son más que suficientes para tener una buena aproximación de ella.

En el caso de matrices simétricas definidas positivas, la factorizacion de Cholesky ha mostrado ser uno de los precondicionadores más comunes, ya que solo requiere almacenar una matriz triangular inferior en memoria. En la

práctica muchos de los sistemas que interesa resolver tienen una matriz de coeficientes simétrica definida positiva, como por ejemplo las matrices que surgen de la discretización de problemas de dispersión electromagnética [2].

En la solución de sistemas de ecuaciones lineales, la matriz puede tener una estructura aprovechable para mejorar la eficiencia del método de solución. Las matrices que se van a utilizar en este trabajo son generadas de discretizaciones provenientes de elemento finito dadas por las implementaciones descritas en [3], que son matrices esparcidas de gran tamaño. Si aplicamos factorización de Cholesky para estas matrices, no estaríamos aprovechando la estructura pues dicha factorización puede generar matrices no esparcidas. Razón por la cual se recurre a la factorización incompleta de Cholesky que sí genera matrices esparcidas.

El objetivo de este trabajo es evaluar la factorización incompleta de Cholesky como técnica de precondicionamiento para la solución de sistemas a gran escala con el método del gradiente conjugado. Ya que en el caso particular de sistemas cuya matriz es simétrica definida positiva, el método del gradiente conjugado en combinación con algún precondicionador, es el que presenta los mejores resultados [4].

Se va utilizar la factorización incompleta de Cholesky $IC(p,\tau)$, usando como criterio para la introducción del llenado umbrales numéricos con el fin de aprovechar el patrón de esparcidad de la matriz. En este tipo de factorización los elementos no diagonales son eliminados durante la factorización si su valor absoluto es menor que cierta tolerancia o umbral τ. Existen varias reglas de eliminación de las entradas “pequeñas". Por ejemplo en [4], elimina $a_{ij}^{(k)}$ durante la iteración k-ésima si

$$\left|a_{ij}^{(k)}\right| \leq \tau \|A_i\|_2, \;\; A_i : \text{fila } i$$

Un inconveniente de las estrategias de umbral es no poder predecir la memoria para almacenar el precondicionador [5]. Para evitar este problema [4] propone, además de la regla de eliminación, retener los p elementos mayores en cada fila de la matriz. Esto con el objetivo de mantener un adecuado almacenamiento en memoria para matrices grandes.

El documento está organizado de la siguiente forma: en la sección 1.2 se describe el método de gradiente conjugado, en la sección 1.3 se da la idea de precondicionamiento, explicando con algún detalle la factorización incompleta de Cholesky y el método del gradiente conjugado precondicionado. En la sección 1.4 se presenta la implementación de los algoritmos. En la sección 1.5, se presentan los experimentos numéricos y se finaliza con las conclusiones en la sección 1.6.

1.2 Gradiente conjugado

Una de las técnicas iterativas más utilizadas a la hora de trabajar con matrices simétricas definidas positivas (s.d.p) es el método de gradiente conjugado (CG, por sus siglas en inglés), desarrollado por Hestenes y Stiefel en 1952, [**?**]. El objetivo es aproximar la solución del sistema lineal

$$Ax = b \quad x, b \in \mathbb{R}^n \text{ y } A \in \mathbb{R}^{n\times n} \tag{1.1}$$

El interés particular es trabajar con A esparcida, el número de entradas no nulas de A es de orden $\mathcal{O}(n)$, y de gran tamaño. Razón por la cual los métodos directos como eliminación Gaussiana son descartados debido a su complejidad computacional.

Dado un valor inicial x_0 que aproxima a la solución del sistema (1.1), el objetivo del CG es mejorar dicha aproximación realizando una iteración de la forma $x_{k+1} = x_k + \rho_k d_k$, x_k representa la iteración k. Los vectores $\{d_k\}$ son llamados

vectores directores y el escalar ρ_k es escogido para minimizar el funcional

$$\phi(x) = \frac{1}{2}x^T Ax - x^T b$$

en la dirección de d_k. Esto es, ρ_k es escogido para minimizar el funcional $\phi_\rho(x_k + \rho d_k)$ [7]. La iteración x_{k+1} es aceptada como una solución, si el residuo $r_k = Ax_k - b$ satisface alguna condición de parada.

El esquema del método es como sigue:

$$\begin{cases} x_0 \in \mathbb{R}^n \text{ arbitrario} \\ r_0 = b - Ax_0 \\ d_0 = r_0 \\ \rho_0 = \dfrac{r_0^T r_0}{d_0^T A d_0} \\ \\ x_1 = x_0 + \rho_0 d_0 \end{cases} \qquad \begin{cases} \text{Para } k = 1,...,n-1 \\ r_k = r_{k-1} - \rho_{k-1} A d_{k-1} \\ \beta_k = \dfrac{r_k^T r_k}{r_{k-1}^T r_{k-1}} \\ \\ d_k = r_k + \beta_k d_{k-1} \\ \rho_k = \dfrac{r_k^T r_k}{d_k^T A d_k} \\ \\ x_{k+1} = x_k + \rho_k d_k \end{cases}$$

1.3 Precondicionamiento

La convergencia de la mayoría de los métodos iterativos depende de las propiedades espectrales de la matriz de coeficientes del sistema de ecuaciones [8]. Un método basado en una matriz mal condicionada va a requerir de muchas iteraciones, lo que no mejoraría el inconveniente de los métodos directos, porque incluso el método iterativo puede llegar a ser más lento que un método directo. En el caso particular del CG tenemos que cuando $\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} \to 1$, con $\kappa = Cond_2(A) = \frac{\lambda_n}{\lambda_1}$, la convergencia es lenta y eso sucede cuando $\kappa \to \infty$, [7]. Lo que significa que cuando el condicionamiento de la matriz es alto, el número de iteraciones requeridas es también alto y no necesariamente el algoritmo converge, debido a los problemas al trabajar con precisión finita. Por tal razón se usan técnicas de precondicionamiento que tienen por objetivo reducir el condicionamiento de la matriz del sistema, logrando una reducción en el número de iteraciones necesarias para alcanzar la convergencia.

Un precondicionador M es una matriz no singular que transforma el sistema $Ax = b$ en un nuevo sistema $M^{-1}Ax = M^{-1}b$, donde el radio espectral del nuevo sistema es menor que el del sistema original. En estas condiciones, la solución del sistema transformado es equivalente a la solución del original, pero sus propiedades espectrales permiten alcanzar más rápido la convergencia. Así que el objetivo del precondicionamiento es la reducción del número de iteraciones requerido para la convergencia, sin incrementar significativamente la cantidad de cálculos por iteración.

Nótese que si $M = A$, el algoritmo converge en exactamente una iteración. Lo que lleva a concluir que mientras M aproxime mejor a la matriz A, más rápida es la convergencia. En este trabajo M es definida por medio de la factorización incompleta de Cholesky, la cual se explica a continuación.

1.3.1 Factorización incompleta de Cholesky

La factorización de Cholesky, definida para matrices simétricas definidas positivas, (s.d.p) consiste en descomponer A como el producto de una matriz triangular inferior y su transpuesta. Es decir,

$$A = LL^T$$

Por su parte, la factorización la factorización incompleta de Cholesky presenta ligeras diferencias, lo que requiere algunos conceptos previos, los cuales se detallan a continuación.

Definición 1.1 (Esparcidad y llenado).

- Patrón de esparcidad o de no nulos de la matriz A es el conjunto K de todas las posiciones (i,j) para las cuales $a_{ij} \neq 0$.
- Entenderemos por llenado (*fill-in*) cuando por medio de una transformación sobre la matriz una entrada nula se convierte en no nula, es decir, el patrón de esparcidad de la matriz aumenta.

El problema de la factorización de Cholesky al aplicarla a una matriz esparcida es que no necesariamente el resultado, L, es una matriz esparcida. Es decir, se puede producir llenado (*fill-in*) sobre las filas de L. Por tal razón, se recurren a técnicas de factorización incompletas con el objetivo de conservar el patrón de esparcidad de la matriz original, obteniendo una factorización de la forma

$$A = \tilde{L}\tilde{L}^T - R$$

donde $\tilde{L}$ es una matriz esparcida y R una matriz densa conocida como el error en la factorización. En la práctica se desestima R y se aproxima A por $\tilde{L}\tilde{L}^T$. Si bien es cierto, esta factorización no es igual a A pero la aproxima, por lo cual puede ser usada para precondicionar el sistema, tomando $M = \tilde{L}\tilde{L}^T$. La factorización incompleta no siempre existe, el hecho de que sea una heurística hace que no siempre se tenga una factorización posible ya que se puede tener raíces negativas en el procedimiento de obtener la factorización.

Existen cuatro formas fundamentales de factorizaciones de Cholesky incompletas:

1. Factorizaciones sin llenado: $IC(0)$. Presentado en [4], resulta de la aproximación de A por una factorización incompleta IC, guardando las mismas entradas nulas de la matriz A en la matriz triangular L. Es decir, la factorización incompleta tiene la misma cantidad de elementos no nulos y en las mismas posiciones que en la matriz A.

2. Factorizaciones con llenado, usando como criterio para la introducción del llenado la posición dentro de la matriz: $IC(k)$. Consiste en rellenar alguna entrada de la matriz L de la descomposición, que en $IC(0)$ sería nula.

3. Factorizaciones con llenado, usando como criterio para la introducción del llenado umbrales numéricos: $IC(\tau)$. En este tipo de factorización los elementos no diagonales son eliminados durante la descomposición si ellos están por debajo de una cierta tolerancia o umbral. Es decir, se eliminan los valores pequeños o poco significativos en cada fila.

4. Factorizaciones con llenado modificadas: $IC(p,\tau)$. Además de la regla de eliminación descrita en la factorización anterior, retiene los p elementos más grandes en magnitud en cada fila de L [2]. En otras palabras, no se permite más de cierta cantidad p de entradas distintas de cero por fila, esto con el objetivo de mantener un adecuado almacenamiento en memoria para matrices grandes.

1.3.2 CG precondicionado

Para acelerar la convergencia del método de CG en la resolución de $Ax = b$ se pueden utilizar técnicas de precondicionamiento. Para ello, tomaremos como precondicionador la matriz obtenida mediante la técnica de factorización incompleta de Cholesky, es decir, $M = \tilde{L}\tilde{L}^T$. Así como A es s.d.p entonces $\widetilde{A} = \tilde{L}^{-1}A\tilde{L}^{-T}$ también lo es, por lo que puede aplicarse el método del CG al sistema

$$(\tilde{L}^{-1}A\tilde{L}^{-T})(\tilde{L}^T x) = \tilde{L}^{-1}b$$

cuya solución es equivalente a la solución del sistema original $Ax = b$.

En la práctica $\tilde{L}^{-1}A\tilde{L}^{-T}$ no se calcula explícitamente, como CG lo que requiere es multiplicar la matriz por un vector lo que se hace es multiplicar dicho vector de manera individual resolviendo un sistema triangular inferior y un sistema triangular superior.

1.4 Aspectos de la implementación

Se utilizó la factorización incompleta de Cholesky con llenado modificada $IC(p,\tau)$, la cuarta mencionada en la sección 3.1. Para obtener dicha factorización la implemetación se basó en el algoritmo presentado en [2], dicho algoritmo se muestra a continuación.

Algoritmo 1.1: Factorización incompleta de Cholesky $IC(A,\tau,p)$

Datos: $A = (a_{ij})_{n\times n}, \tau, p$
Salida: L

for $i = 1,...,n$ **do**
  $w = a_{i,1:i}$
  for $j = 1,...,i-1$ **do**
    for $k = 1,...,j-1$ **do**
      $w_{ij} = w_{ij} - l_{ik}l_{jk}$
    $w_{ij} = w_{ij}/l_{jj}$;
  $\tau_i = \tau \cdot \|w\|_2$;
  for $k = 1,...,i-1$ **do**
    $w_{ik} = 0$ cuando $|w_{ik}| < \tau_i$
  Se mantienen los índices $I = (i_k)_{k=1...p}$ de los p elementos más grandes $|w_{ij}|$ para $j = 1$ hasta $i-1$
  for $k = 1,...,p$ **do**
    $l_{i_k j} = w_{i_k j}$
  for $k = 1,...,i-1$ **do**
    $a_{ii} = a_{ii} - l_{ik}^2$
  $l_{ii} = \sqrt{a_{ii}}$
return $L = (l_{ij})_{n\times n}$

Todos los algoritmos se utilizaron para matrices en formato *compressed sparse row* (CSR), el cual se utiliza comúnmente para el almacenamiento de matrices esparcidas, para detalles de este formato se puede consultar [4]. Se implementaron dos algoritmos para resolver los sistemas: el gradiente conjugado y el gradiente conjugado precondicionado, en

ambos casos para matrices en formato CSR. Para lo cual también se implementó la multiplicación matriz por vector para matrices en formato CSR.

Algoritmo 1.2: Gradiente conjugado $GCCSR(A, b, x_0, tol, Max)$

Datos: $A, b, x_0, tol, \text{Max}$
Salida: x

$r = b - Ax$;
$d = -r$;
$z = Ad$;
$a = (r' \cdot d)/(d' \cdot z)$;
$x = x_0 + a \cdot d$;
$k = 1$;
while $\|r_k\|_2 \leq tol \cdot \|r_0\|_2$ and $k \leq$Max **do**
    $r = r - a \cdot z$;
    $B = (r' \cdot z)/(d' \cdot z)$;
    $d = -r + B \cdot d$;
    $a = (r' \cdot d)/(d' \cdot z)$;
    $x = x + a \cdot d$;
    $k = k + 1$;
return x

El algoritmo del CG precondicionado tiene un parámetro más, la matriz L de la factorización de Choslesky.

Algoritmo 1.3: Gradiente conjugado Precondicionado $GCPrec(A, L, b, x_0, tol, \text{Max})$

Datos: $A, L, b, x_0, TOL, \text{Max}$
Salida: x

$r = b - Ax$;
Resolver el sistema $LL^T q = r$;
$d = -q$;
$z = Ad$;
$a = (r' \cdot d)/(d' \cdot z)$;
$x = x0 + a \cdot d$;
$k = 1$;
while $\|r_k\|_2 \leq tol \cdot \|r_0\|_2$ and $k \leq$Max **do**
    $r = r - a \cdot z$;
    $B = (r' \cdot z)/(d' \cdot z)$;
    Resolver el sistema $LL^T q = r$;
    $d = -q + B \cdot d$;
    $a = (r' \cdot d)/(d' \cdot z)$;
    $x = x + a \cdot d$;
    $k = k + 1$;
return x

Para resolver $LL^T q = r$, primero se resuelve un sistema triangular inferior $Ly = r$ y luego se resuelve el sistema triangular superior $L^T q = y$. Para resolver el segundo sistema se implementó de forma tal que no sea necesario calcular L^T.

1.5 Experimentos numéricos

Con este estudio se pretende determinar de manera experimental, la eficiencia del precondicionador Cholesky Incompleto. Todos los cálculos presentados en esta sección se realizaron en una computadora portátil con procesador Intel Core Duo de 64 bits, 4GB de memoria RAM y 320GB de disco duro.

Se utilizó el método CG precondicionado con la factorización incompleta de Cholesky en cuatro sistemas tomados de [3], todos con matriz simétrica definida positiva. Estas matrices provienen de la discretización de un problema de valor frontera utilizando el método de elemento finito. De esta manera, se trabajará con las matrices presentadas en el Cuadro 1.1.

Matriz	n	Entradas no nulas (nnz)
M1	5380	583564
M2	5500	369946
M3	5260	200040
M4	8380	328292

Cuadro 1.1 Detalle de las matrices

Los experimentos realizados se presentan en las siguientes figuras y cuadros de resumen. Todas las pruebas numéricas se realizaron con un número máximo de 500 iteraciones y con una tolerancia de 10^{-8}. El vector inicial en todos los casos fue de ceros ya que se obtuvieron mejores resultados que con el vector de unos.

El primer resultado se muestra en el Cuadro 1.2 que corresponde al número de iteraciones, tiempo de ejecución y residuo del CG sin precondicionar para los 4 sistemas. Se muestran los datos para el algoritmo de CG para la matriz en formato CSR (GCCSR) y por el algoritmo del gradiente conjugado para matrices densas de MATLAB (pcg).

Matriz	# Iteraciones		Último Residuo		Tiempo	
	GCCSR	pcg	GCCSR	pcg	GCCSR	pcg
M1	277	276	2.6259e-9	9.2180e-9	5.5539 s	16.4285 s
M2	221	221	2.1671e-9	9.4933e-9	2.9645 s	14.6156 s
M3	126	124	1.1948e-9	9.7693e-9	1.0161 s	7.2327 s
M4	138	137	1.0241e-9	9.1588e-9	1.8018 s	21.1818 s

Cuadro 1.2 Gradiente conjugado sin precondicionar

En el Cuadro 1.2 se puede observar cómo el número de iteraciones y el tiempo para resolver aumentan, según crece el número de entradas diferente de cero en la matriz. También se observa una reducción en el tiempo entre el algoritmo que trabaja con matrices densas y el algoritmo para las matrices en formato CSR. En la Figura 1.1 se muestra la curva característica de convergencia para las pruebas mencionadas.

En los Cuadros 1.3, 1.4 y 1.5 se muestra el número de iteraciones al resolver los mismos sistemas anteriores, pero usando CG precondicionado. Primeramente usando como precondicionador la factorización de Cholesky.

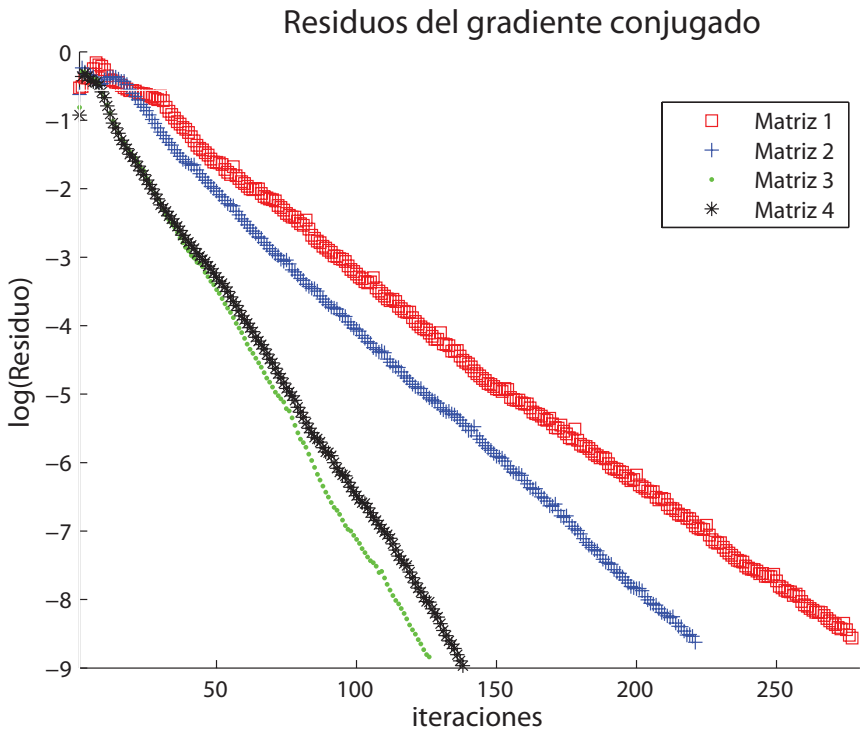

Figura 1.1 Curva característica de convergencia. Residuos por iteración en el método del gradiente conjugado sin precondicionar

Matriz	iteraciones	Residuo	Tiempo solución	Tiempo construcción de L	nnz
M1	1	7.2543e-15	1.0786 s	252.059 s	6832325
M2	1	5.0087e-15	1.2420 s	327.077 s	8154437
M3	1	2.4149e-15	1.1248 s	293.915 s	7477508
M4	1	2.5398e-15	3.42225 s	1352.9 s	20470873

Cuadro 1.3 Gradiente conjugado precondicionado con Cholesky

Los datos del Cuadro 1.3 coinciden con la teoría, ya que este caso el precondicionador $M = LL^T$ es igual a la matriz dada, por lo tanto el algoritmo converge en una interación.

Ahora se mostrarán las pruebas realizadas con las mismas matrices pero precondicionando con Cholesky incompleto. Estas pruebas serán orientadas con el fin de observar lo que sucede cuando se varían los dos parámetros de los que depende dicha factorización τ y p. En los Cuadros 1.4 y 1.5 se presenta una muestra de los resultados obtenidos fijando alguno de los dos parámetros de la factorización. En cada caso se muestra el número de iteraciones necesarias y el residuo obtenido por el método.

Primero se varía el parámetro τ dejando $p = n$, es decir sin aplicar el parámetro p. Para las cuatro matrices podemos obervar, en el Cuadro 1.4, que para $\tau = 10^{-5}$, $\tau = 10^{-4}$ y $\tau = 10^{-3}$ no hay un cambio significativo en el número de iteraciones, pero sí hay un cambio importante en el tiempo para construir la matriz L y en el número de entradas diferentes de cero. Para $\tau = 10^{-2}$ aumenta el número de iteraciones pero es menor que el número de iteraciones para el CG sin precondicionar. Para $\tau = 10^{-1}$ la factorización incompleta de Cholesky no existe para las dos primeras

Matriz	τ	# Iter.	Residuo	Tiempo solución	Tiempo constr. de L	$nnz\ L$
M1	10^{-5}	4	2.0985e-11	1.1274 s	101.709 s	2627990
	10^{-4}	4	1.2135e-10	0.8795 s	73.1655 s	1670373
	10^{-3}	8	5.6679e-9	0.8426 s	44.6830 s	881962
	10^{-2}	28	1.2421e-7	1.4874 s	25.8376 s	338458
	10^{-1}	-	-	-	n.s.p.d	-
M2	10^{-5}	5	6.8118e-10	1.0852 s	84.7090s	2081187
	10^{-4}	5	1.5425e-9	0.6977 s	56.0129 s	1226495
	10^{-3}	8	3.0949e-9	0.6359 s	34.4903 s	611267
	10^{-2}	30	6.0980e-8	1.2178 s	20.1014 s	239874
	10^{-1}	-	-	-	n.s.p.d	-
M3	10^{-5}	5	2.8876e-10	0.6989 s	52.8868 s	1341951
	10^{-4}	5	5.2354e-10	0.4610 s	34.9170 s	765735
	10^{-3}	7	5.8705e-9	0.3567 s	21.7202 s	377285
	10^{-2}	25	2.9254e-8	0.7598 s	13.6282 s	149571
	10^{-1}	501	8.6395e-4	9.5751 s	8.26631 s	3753
M4	10^{-5}	5	1.7475e-10	1.2684 s	150.119 s	2499299
	10^{-4}	5	3.4011e-10	0.7624 s	98.6794 s	1375555
	10^{-3}	9	6.2449e-9	0.7363 s	57.9547 s	647712
	10^{-2}	34	3.5915e-8	1.3870 s	33.6950 s	245079
	10^{-1}	501	1.2533e-3	12.6499 s	19.5183 s	59712

Cuadro 1.4 Gradiente conjugado precondicionado con $IC(n,\tau)$

matrices y para las otras dos el método alcanza el número máximo de iteraciones.

En la Figura 1.2 podemos obervar cómo la aplicación del parámetro τ reduce considerablemente los tiempos de construcción de la matriz L, sin incrementar el número de iteraciones como se indica en el Cuadro 1.4.

En el Cuadro 1.5 se presentan los resultados cuando se varía el parámetro p. En este caso se fijó $\tau = 10^{-3}$. Se observa que para $p = 2000$ el número de entradas diferente de cero no ha cambiado con respecto al caso en que solo se aplique $\tau = 10^{-3}$, y hasta $p = 500$ no hay gran cambio en el número de entradas distintas de cero ni en el tiempo de construcción de L. El cambio se nota a partir de $p = 200$. Para la matriz 1 el cambio en el número de iteraciones de $p = 100$ a $p = 50$ es grande, en este último valor el número de iteraciones se acerca al número de iteraciones sin precondicionar; y ya para $p = 40$ la factorización incompleta de Cholesky no existe. En la matriz 2 el cambio significativo se da de $p = 50$ a $p = 40$ y para $p = 30$ el número de iteraciones sobrepasa las iteraciones del método del CG sin precondicionar. Para las matrices 2 y 3 de $p = 40$ a $p = 30$ es donde se da el cambio grande en el número de iteraciones y para $p = 20$ el número de iteraciones sobrepasa las iteraciones del método del CG sin precondicionar. Estos resultados se muestran en la Figura 1.3, en la cual es claro que cuando p aumenta el número de iteraciones disminuye.

Los resultados del Cuadro 1.5 muestran una reducción en los tiempos de cálculo de L de hasta un 50% para las cuatro matrices.

En la Figura 1.4 se presenta el tiempo total del cálculo para todo el proceso del gradiente conjugado, que consiste en el tiempo de las iteraciones del gradiente conjugado más el tiempo de la factorización incompleta de Cholesky. Dado

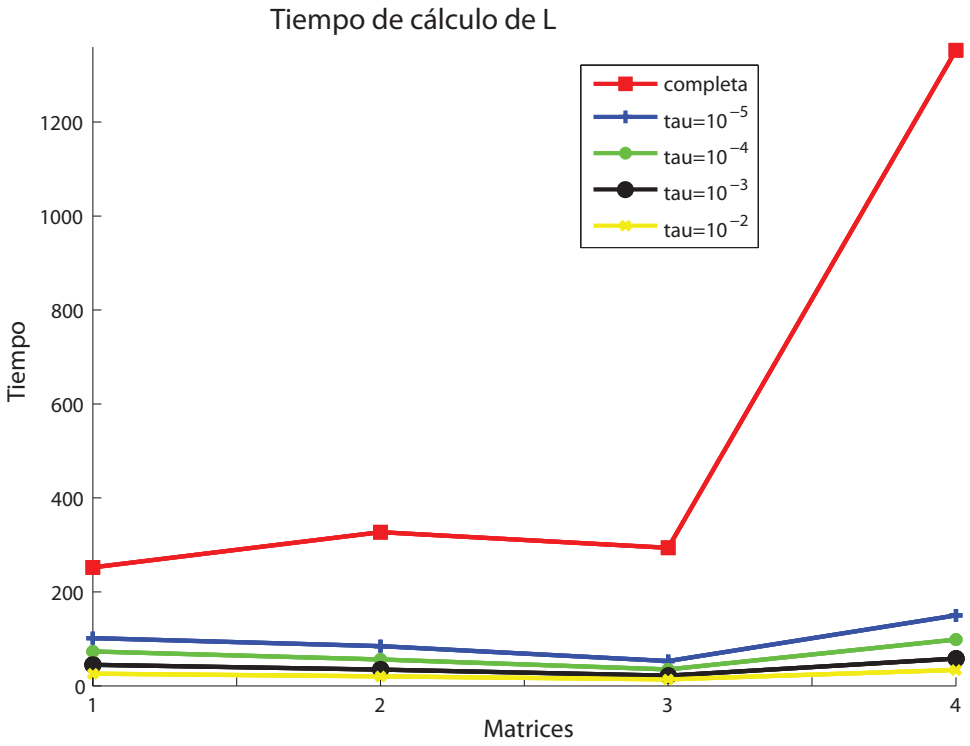

Figura 1.2 Tiempos de Construcción de L variando τ.

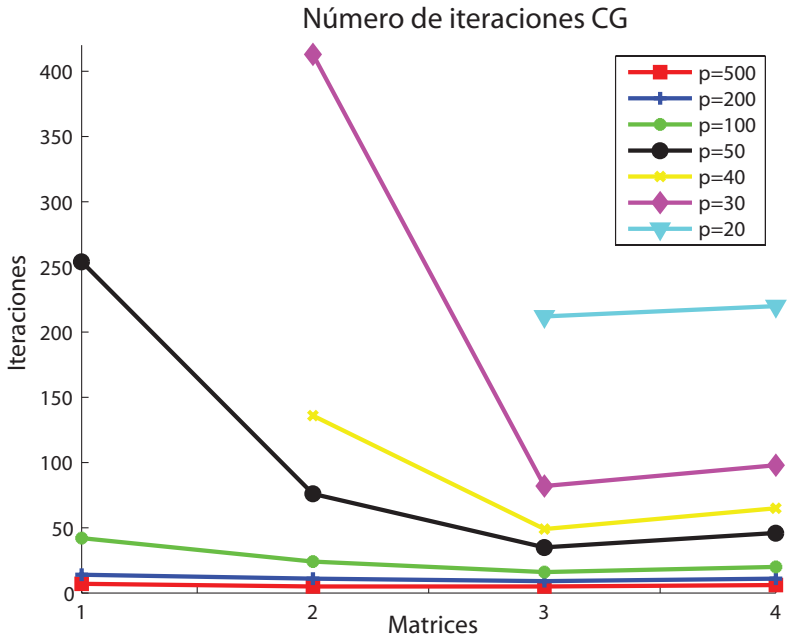

Figura 1.3 Número de iteraciones en el CG precondicionado con IC, variando p.

Matriz	p	# Iter.	Residuo	Tiempo solución	Tiempo constr. de L	$nnz\ L$
M1	2000	4	1.2135e-10	0.8552 s	78.3425 s	1670373
	1000	5	2.6069e-11	1.0901 s	75.7115 s	1626152
	500	7	1.1150e-8	1.0074 s	74.4029 s	1296721
	200	14	7.5943e-8	1.2250 s	56.0341 s	725797
	100	42	1.4548e-7	2.5066 s	39.6188 s	427113
	50	254	1.9719e-7	11.1612 s	26.1658 s	238423
	40	-	-	-	n.s.p.d	-
M2	2000	5	1.5425e-9	0.7188 s	62.8421 s	1226495
	1000	5	1.6369e-9	0.7339 s	60.1756 s	1224708
	500	5	3.0949e-9	0.6336 s	60.1464 s	1099216
	200	11	1.4726e-8	0.8772 s	49.6251 s	676822
	100	24	4.4555e-8	1.2629 s	35.2203 s	405813
	50	76	1.1912e-7	2.7671 s	25.1436 s	229755
	40	136	1.26152e-7	4.6100 s	21.7773 s	189819
	30	413	1.4592e-7	12.1913 s	18.7125 s	147867
M3	2000	5	5.2354e-10	0.4612 s	42.0342 s	765735
	1000	5	5.2924e-10	0.5129 s	41.3433 s	764665
	500	5	7.9411e-10	0.4328 s	37.1162 s	733882
	200	9	3.1893e-9	0.5769 s	32.4403 s	529033
	100	16	2.0672e-8	0.7099 s	26.2134 s	339876
	50	35	4.4857e-8	1.0317 s	19.5985 s	19.8997
	40	49	5.0969e-8	1.2839 s	17.6534 s	165747
	30	82	6.0919e-8	1.8859 s	15.4083 s	130453
	20	212	7.3821e-8	4.3771 s	13.6554 s	92743
M4	2000	5	3.4011e-10	0.8521 s	109.582 s	1375555
	1000	5	3.6512e-10	0.7991 s	104.1970 s	1371588
	500	6	3.9514e-10	0.8828 s	101.760 s	1291102
	200	11	7.3648e-9	1.1057 s	87.7153 s	893005
	100	20	2.3708e-8	1.4030 s	69.3135 s	561357
	50	46	4.6405e-8	2.1851 s	50.6483 s	325961
	40	65	6.0633e-8	2.7884 s	45.3207 s	271397
	30	98	6.7736e-8	3.7966 s	39.6381 s	213190
	20	220	7.1489e-8	7.5313 s	12.3811 s	151049

Cuadro 1.5 Gradiente conjugado precondicionado con $IC(p, 10^{-3})$

que el número de entradas diferentes de cero de L aumenta con p, el costo en el cálculo de L y el de cada iteración del CG conjugado también aumenta. Sin embargo, como se muestra en la Figura 1.4 el aumento es moderado.

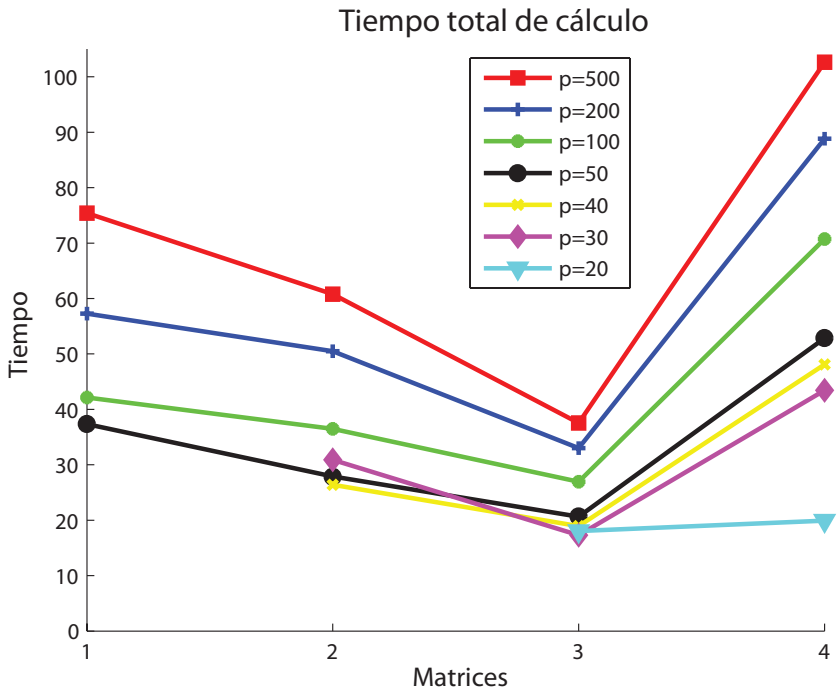

Figura 1.4 Tiempo total de cálculo en el CG precondicionado con IC, variando p.

1.6 Conclusiones

En este artículo se ha presentado la Factorización incompleta de Cholesky como técnica de precondicionamiento, con el objetivo de reducir el tiempo y el número de iteraciones al aproximar la solución de un sistema lineal generado de la discretización. Se describe brevemente la técnica y se presentan resultados para estudiar su funcionamiento.

Los resultados obtenidos en este estudio, muestran que el método del gradiente conjugado precondicionado es efectivo y eficiente. Se puede concluir que el precondicionador Cholesky incompleto funcionó mejor que el algoritmo de gradiente conjugado sin precondicionamiento, pues redujo el número de iteraciones. Una adecuada elección de los parámetros p y τ puede reducir el tiempo total de cálculo y los requisitos de memoria.

Merece la pena destacar que el parámetro p involucrado, que define el máximo llenado en cada fila del precondicionador, ayuda a reducir el tiempo de cálculo total, como se muestra en los experimentos numéricos.

Bibliografía

[1] Y. Saad (2003). *Iterative Methods for Sparse Linear Systems.* Society for Industrial and Applied Mathematics, USA, second edition.

[2] L. Li W. Shao S. Lai T. Huang, Y. Zhang (2009). Modified incomplete cholesky factorization for solving electromagnetic scattering problems (2009). *Progress In Electromagnetics Research B*, 13:41?58

[3] F. Sequeira (2010). *Aspectos computacionales del método "Local Discontinuous Galerkin" para mallas no estructuradas en 3D.* Tesis de Maestría, Universidad de Puerto Rico, Mayagüez.

[4] Y. Saad. (1994). ILUT: A dual threshold incomplete LU factorization. *Numerical Linear Algebra with Applications*, pages 387?402.

[5] C. Lin and J. Moré (1997). Incomplete Cholesky Factorization with limited memory. *SIAM Journal Scientific Computing*, 21:24?45

[6] M. Hestenes y E. Stiefel (1952). Methods of Conjugate Gradients for Solving Linear Systems. *Journal of Research of the National Bureau of Standards*, 49:409?436

[7] B. Datta (2010). *Numerical Linear Algebra and Applications.* Society for Industrial and Applied Mathematics, USA, second edition edition.

[8] T. Chan J. Demmel J. Donato J. Dongarra V. Eijkhout R. Pozo C. Romine R. Barrett, M. Berry and H. Vorst (1994). *Templates for the solution of linear systems: Building Blocks for Iterative Methods.* SIAM.