Universidad Mayor de San Andrés
Facultad de Ciencias Puras y Naturales
Carrera de Matemática
Instituto de Investigación Matemática

Revista Boliviana de Matemática – UMSA 04 (2020) 2–7

IIMAT

Método de Gradientes Conjugados Precondicionado☆

Porfirio Suñagua S.[a,1,*]

[a] *Universidad Mayor de San Andrés*
Institito de Investigación Matemática
Primer Piso del Edificio FCPN, Calle 27 Cota–Cota
La Paz – Bolivia

Resumen

En este trabajo presentamos el algoritmo de gradientes conjugados precondicionado que mejora el clásico método de gradientes conjugados cuando la matriz de coeficientes del sistema lineal simétrico a resolver está mal condicionada. El método de gradientes conjugados se aplica al problemas de minimización de una función cuadrática donde la matriz del término cuadrático es definida positiva cuya condición optimal necesaria de primer orden resulta un sistema lineal. Este método funciona bien cuando la matriz de coeficientes del sistema lineal está bien condicionada.

Un precondicionador es una matriz no singular tal que pre y post multiplicando a la matriz dada podemos mejorar su número de condición, lo cual mejora la precisión y la convergencia de los métodos iterativos, especialmente cuando la dimensión del sistema lineal es grande.

Palabras clave: Gradientes Conjugados, Precondicionador, Métodos Iterativos

1. Introducción

En muchas aplicaciones de métodos matemáticos se requiere resolver un sistema lineal de la forma $Ax = b$, donde A es una matriz cuadrada $n \times n$, $x, b \in \mathbb{R}^n$. Cuando la dimensión n es grande, los métodos computacionales más utilizados son los métodos iterativos que a partir de alguna solución aproximada generan una secuencia de soluciones aproximadas que se espera que converja a la solución exacta. La convergencia, como la precisión dependen del número de condición de la matriz de coeficientes A.

Uno de los métodos para resolver sistemas de ecuaciones lineales simétricos de rango completo es la factorización de Cholesky, una descripción de este método se encuentra en Wright [15, Cap. 11]. Pese a que la factorización de Cholesky es numéricamente estable, Higham [8], el error de acarreo por redondeo puede ser grande cuando la matriz es muy mal-condicionada.

Una forma de mejorar el número de condición de la matriz de coeficientes y resolver el sistemas lineal por métodos iterativos es precondicionar la matriz de coeficientes. Ciertamente un precondicionador debe ser adecuado a las características de cada sistema. Para construir un precondicionador, se debe tomar en cuenta si la matriz es simétrica o no. El método de gradientes conjugados se aplica a los sistemas lineales simétricos y definidos positivos, de modo que en adelante daremos énfasis a precondicionadores para matrices simétricas.

Dada una matriz simétrica A y una matriz no singular C las ecuaciones lineales

$$Ax = b \qquad \text{y} \qquad (CAC^T)C^{-T}x = Cb,$$

son equivalentes en el sentido de que ambas tienen la misma solución. La matriz CAC^T también es simétrica y C se conoce como el precondicionador de A. Para que la nueva matriz de coeficientes CAC^T sea bien condicionada, sería bueno que ese producto esté "próximo" a la matriz identidad o tenga sus autovalores "próximos" a la unidad, así su *número de condición sería pequeño*. Esto permitirá resolver el sistema lineal con mejor estabilidad numérica.

Para aclarar las ideas, sobre todo del número de condición,

☆ Artículo aceptado por el Comité Editorial. En contenido está orientado para desarrolladores de algoritmos iterativos computacionales

* ✉ Av. Villazón 1995, PB Edificio Viejo, Carrera de Matemática
✉ psunagua@gmail.com (Porfirio Suñagua S.)
[1] Profesor del Área de Análisis y Optimización Matemática.

consideramos la siguiente matriz simétrica mal-condicionada

$$A = \begin{pmatrix} 10^{20} & 1 \\ 1 & 10^{-10} \end{pmatrix}.$$

los autovalores de ésta matriz son $\lambda_1 = 1,000000000 \times 10^{20}$ y $\lambda_2 = 9,999999999 \times 10^{-11}$. Por tanto, el número de condición es $\kappa(A) = \lambda_1/\lambda_2 = 1,0000000001 \times 10^{30}$.

Ahora como precondicionador, proponemos la siguiente matriz diagonal

$$C = \begin{pmatrix} 10^{-10} & 0 \\ 0 & 10^{5} \end{pmatrix}.$$

En efecto, pre y post multiplicando con ésta matriz, obtenemos

$$\begin{aligned} M = CAC^T &= \begin{pmatrix} 10^{-10} & 0 \\ 0 & 10^{5} \end{pmatrix}\begin{pmatrix} 10^{20} & 1 \\ 1 & 10^{-10} \end{pmatrix}\begin{pmatrix} 10^{-10} & 0 \\ 0 & 10^{5} \end{pmatrix} \\ &= \begin{pmatrix} 1 & 10^{-5} \\ 10^{-5} & 1 \end{pmatrix}. \end{aligned}$$

Los autovalores de la nueva matriz son $\lambda_1 = 1,00001$ y $\lambda_2 = 9,999899 \times 10^{-1}$. Luego, el número de condición de la matriz precondicionada es $\kappa(M) = 1,000020000200002$. En las Figuras 1 y 2, se ilustran gráficamente estos hechos, por medio de curvas de nivel de las formas cuadráticas $\varphi(x) = x^T Ax$ y $\varphi(x) = x^T Mx$ asociadas respectivamente a las matrices A y M, donde $x \in \mathbb{R}^2$.

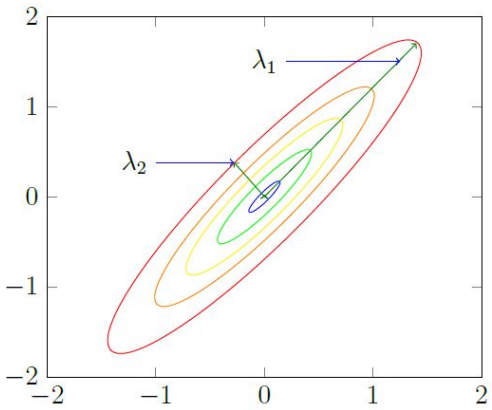

Figura 1. Matriz mal condicionada

Si A fuera definida positiva, teóricamente el mejor precondicionador seria la inversa de la factorización de Cholesky, o sea,

$$A = LL^T \ \wedge \ C = L^{-1} \quad \Rightarrow \quad M = CAC^T = L^{-1}LL^TL^{-T} = I.$$

Está claro que obtener el factor de Cholesky L es ya otro problema, entonces una aproximación de L podría ser aún un buen precondicionador. Una buena opción es usar la factorización incompleta de Cholesky. En algunas aplicaciones este precondicionador funciona bien, pero en los métodos de puntos interiores de Programación Lineal funciona bien solamente en las primeras iteraciones, Oliveira e Sorensen [12]. El sistema precondicionado

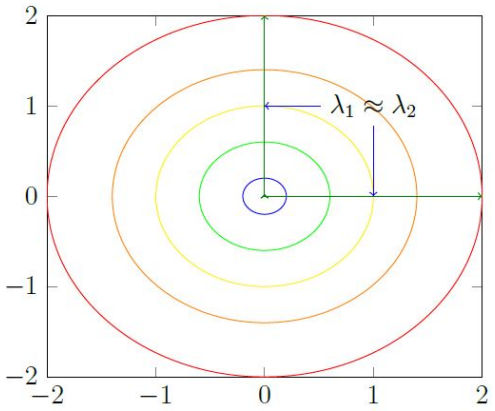

Figura 2. Matriz bien condicionada

tiene la forma $M\tilde{x} = \tilde{b}$, donde $\tilde{b} = Cb$, $x = C^T\tilde{x}$ y $M = CAC^T$. Normalmente, M no se calcula explícitamente, en métodos iterativos se aplica la multiplicación de matriz por un vector, o sea, $Mv = (CAC^T)v = C(A(C^Tv))$.

Un buen precondicionador no debe ser muy caro de ser calculado computacionalmente, pero de alguna forma, debería obtener una matriz razonablemente bien condicionada, un equilibrio entre éstas dos propiedades determina un buen precondicionador. Una de las variantes de la factorización incompleta de Cholesky es la factorización controlada de Cholesky introducida por Campos y Rollet [3], y Kershaw [9], también obtiene buenos resultados para la mayoría de los problemas. Inicialmente éstas aproximaciones fueron desarrollados para resolver ecuaciones diferenciales por métodos numéricos donde funcionan mejor, pero hoy en día se aplican en las iteraciones de los métodos de puntos interiores de programación lineal.

En la implementación computacional de PCx (Predictor-Corrector de Mehrotra *et al* [5]) modificada por Oliveira e Sorensen [11], Bocanegra *et al* [1] e Velazco *et al* [14], fue incorporado el precondicionador obtenido desde la factorización controlada de Cholesky, el cual se utiliza con éxito en las primeras iteraciones del método de puntos interiores para resolver el sistema de ecuaciones normales por el método de gradientes conjugados.

Antes de abordar el método de gradientes conjugados, hacemos una breve descripción del método del gradiente que forma parte del grupo de métodos llamados *búsqueda lineal exacta*.

2. Método del gradiente o de máximo descenso

Dada una función real f diferenciable de varias variables, en la búsqueda lineal exacta (BLE), se minimiza $\phi(t) = f(\bar{x} + td)$ a partir del punto $\bar{x}$ en la dirección del vector d de acuerdo al criterio de primer orden $\phi'(t) = 0$. En general, ésta ecuación es no lineal, por lo que podría ser necesario resolver la misma por métodos numéricos. Si fuera el caso, el proceso en cada iteración sería computacionalmente caro.

En el método del gradiente se toma $d = -\nabla f(\bar{x})$, luego $\nabla f(\bar{x})^T d = -\nabla f(\bar{x})^T \nabla f(\bar{x}) = -\|\nabla f(\bar{x})\|^2 < 0$, así d siempre será una dirección de descenso. Para encontrar el siguiente punto buscamos un $\bar{t}$ tal que minimice globalmente $\phi(t) = f(\bar{x} + td)$, de este modo, éste método se conoce también como de *máximo descenso*. El algoritmo computacional se expresa en las siguientes líneas.

Dado x^0, $k = 0$.
Mientras $\nabla f(x^k) \neq 0$, haga

1. Calcule $d^k = -\nabla f(x^k)$
2. t_k minimizador global de $\min_{t>0} \phi(t) = f(x^k + td^k)$
3. $x^{k+1} = x^k + t_k d^k$,
4. $k \leftarrow k + 1$.

2.1. *Método del Gradiente: Caso cuadrático*

Para A simétrica y definida positiva ($A > 0$) el minimizador global de la función cuadrática $f(x) = \frac{1}{2}x^T Ax + b^T x + c$, puede ser encontrado explícitamente resolviendo el sistema lineal $\nabla f(x) = Ax + b = 0$. Desde que $\nabla^2 f(x) = A > 0$, entonces la solución del sistema lineal es único y es el minimizador global estricto. En efecto, $\phi'(t) = 0 \Leftrightarrow \nabla f(x^k + td^k)^T d^k = 0 \Leftrightarrow \nabla f(x^k)^T d^k + t(d^k)^T A d^k = 0$, así

$$t_k = -\frac{\nabla f(x^k)^T d^k}{(d^k)^T A d^k} = \frac{\nabla f(x^k)^T \nabla f(x^k)}{\nabla f(x^k)^T A \nabla f(x^k)}$$

Luego en el algoritmo de máximo descenso, el cálculo del tamaño de paso t_k queda explícitamente determinado. En la Figura 3 se ilustra el recorrido de los puntos estimados en cada iteración. En la figura tridimencional se puede observar que el punto estimado corresponde efectivamente al minimizador de $\phi(t)$, donde $\phi(t)$ puede ser visto como la intersección de $f(x)$ con el plano vertical formado por $x^k + td^k$.

Dado que A es una matriz simétrica sus autovalores son reales, luego considere $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ los autovalores de A, donde λ_1 y λ_n son el menor y el mayor autovalor de A respectivamente. El cociente $\dfrac{\lambda_n}{\lambda_1} = \kappa(A)$, donde $\kappa(A)$ es el número de condición de A respecto a la norma euclidiana, está fuertemente relacionado a la razón de convergencia del método de gradiente. En efecto, si $\{x^k\}$ es la sucesión generada por el método del gradiente, entonces $\|x^{k+1}\|_2 \leq \gamma \|x^k\|_2$, donde $\gamma = \sqrt{1 - \dfrac{\lambda_1}{\lambda_n}}$. Así, cuando la matriz es mal condicionada, el autovalor λ_n es grande, entonces el cociente λ_1/λ_n será pequeño, de modo que γ estará cerca de uno. Lo cual hace que la convergencia sea lenta.

Para problemas generales, según Luenberger [10], para $f \in C^2$, si $\bar{x}$ es un minimizador local de f, $\nabla^2 f(\bar{x})$ es definida positiva y $\{x^k\}$ es la sucesión generada por el método de gradiente que converge a $\bar{x}$, entonces la sucesión $\{f(x^k)\}$ converge linealmente a $f(\bar{x})$ con tasa de convergencia no superior a $\left(\dfrac{\bar{\lambda}_n - \bar{\lambda}_1}{\bar{\lambda}_n + \bar{\lambda}_1}\right)^2$, donde $\bar{\lambda}_1$ y $\bar{\lambda}_n$ son el menor y el mayor autovalor de $\nabla^2 f(\bar{x})$ respectivamente.

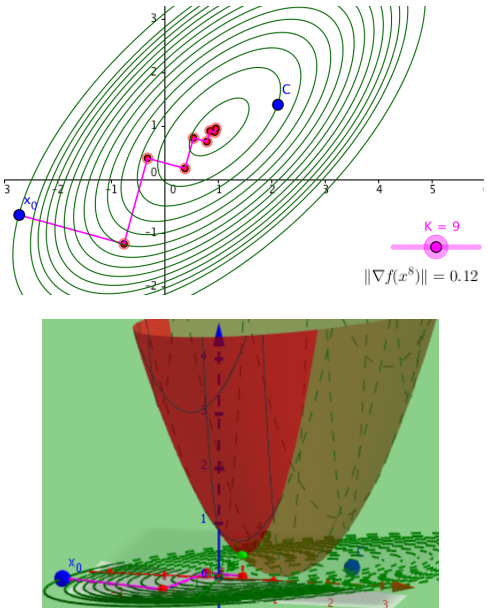

Figura 3. Método de Gradiente

3. Método de Gradientes Conjugados

Definición 3.1. Dada una matriz A simétrica y definida positiva, los vectores u y v son direcciones $A-$conjugadas si $u^T Av = 0$.

Para una función cuadrática convexa $f(x) = \frac{1}{2}x^T Ax + b^T x + c$, donde A es una matriz simétrica y definida positiva ($A = A^T > 0$), el método de gradientes conjugados, introducida por Hestenes y Stiefel [7], consiste en encontrar n direcciones conjugadas, basados en las gradientes de f, que conducen a encontrar el minimizador global de f. El tamaño de paso se encuentra con el mismo criterio de máximo descenso. Es decir, para $\phi(t) = f(x^k + td^k)$, entonces

$$\begin{aligned} \phi'(t) &= \nabla f(x^k + td^k)^T d^k = (A(x^k + td^k) + b)^T d^k \\ &= \nabla f(x^k)^T d^k + t(d^k)^T A d^k = 0 \end{aligned}$$

Así, la solución óptima para el tamaño del paso a partir del punto x^k resulta $t_k = -\dfrac{\nabla f(x^k)^T d^k}{(d^k)^T A d^k}$, luego $x^{k+1} = x^k + t_k d^k$. Para encontrar la siguiente dirección conjugada se plantea $d^{k+1} = -\nabla f(x^{k+1}) + \beta d^k$, donde β se calcula de manera que d^k y d^{k+1} sean $A-$conjugados. En efecto,

$$\begin{aligned} 0 &= (d^k)^T A d^k = -(d^k)^T A \nabla f(x^{k+1}) + \beta (d^k)^T A d^k \\ \beta_k &= \frac{(d^k)^T A \nabla f(x^{k+1})}{(d^k)^T A d^k} \end{aligned}$$

La primera dirección conjugada se elige simplemente como la dirección de máximo descenso, $d^0 = -\nabla f(x^0)$, luego el algoritmo de gradientes conjugados se puede escribir como sigue
Dado $x^0 \in \mathbb{R}^n$, $d^0 = -\nabla f(x^0)$, $k = 0$
Mientras $\nabla f(x^k) \neq 0$, haga

1. $t_k = -\dfrac{(d^k)^T \nabla f(x^k)}{(d^k)^T A d^k}$, $x^{k+1} = x^k + t_k d^k$
2. $\beta_k = \dfrac{\nabla f(x^{k+1})^T A d^k}{(d^k)^T A d^k}$, $d^{k+1} = -\nabla f(x^{k+1}) + \beta_k d^k$
   $k \leftarrow k + 1$.

En la Figura 4 se tiene las dos iteraciones del método de gradientes conjugados, pues en general, este método converge exactamente en n iteraciones, aunque en la práctica, dependiendo de la condición de la matriz, el método puede converger en menor número de iteraciones. En la figura tridimensional también se observa que el método minimiza f en la dirección d^k a partir del punto x^k.

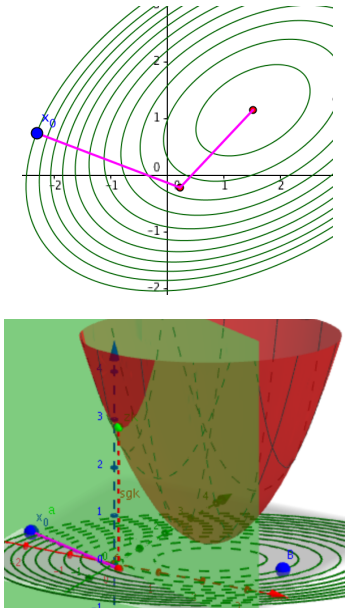

Figura 4. Método de gradientes conjugados

Un algoritmo de gradientes conjugados, conocido como la versión de Hestenes-Stiefel, basado en Golub y Van Loan [6], implementable en cualquier lenguaje de programación se muestra en el Algoritmo 1.

Este algoritmo converge a lo sumo en n iteraciones si no hubiera errores de redondeo, porque las direcciones conjugadas forman una base para $\mathbb{R}^n$. En Golub y Van Loan [6], se establece que las iteraciones $\{x_k\}$ generadas por el algoritmo satisfacen:

$$\|x - x_k\|_A \leq 2\left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^k \|x - x_0\|_A, \tag{1}$$

Algoritmo 1 Gradientes conjugados

Datos: x^0, tal que $Ax_0 \approx b$
  $k = 0$
  Calcule el residuo inicial $r_0 = b - Ax_0$
  mientras $r_k \neq 0$ **haga**
    $k = k + 1$
    si k=1 **entonces**
      Inicialice $p_1 = r_0$
    sino
      Calcule $\beta_k = r_{k-1}^T r_{k-1} / r_{k-2}^T r_{k-2}$
      Actualice $p_k = r_{k-1} + \beta_k p_{k-1}$
    fin si
    Calcule $\alpha_k = r_{k-1}^T r_{k-1} / p_k A p_k$
    Actualice $x_k = x_{k-1} + \alpha_k p_k$
    Actualice $r_k = r_{k-1} - \alpha_k A p_k$
  fin mientras
  retorne $x = x_k$

donde, x es la solución exacta que satisface $Ax = b$, $\kappa = \text{cond}_2(A)$ e $\|w\|_A = \sqrt{w^T M w}$. Entonces, podemos verificar también que en norma-2

$$\|x_k - A^{-1}b\|_2 \leq 2\sqrt{\kappa}\left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^k \|x_0 - A^{-1}b\|_2. \tag{2}$$

Por tanto, la tasa de convergencia, depende, principalmente del número de condición κ de la matriz A.

4. Método de Gradientes Conjugados Precondicionado

Cuando la matriz A es mal condicionada, precondicionamos el sistema $Ax = b$ con una matriz C no singular, para obtener, $M\tilde{x} = \tilde{b}$, donde $\tilde{x} = C^T x$, $M = C^{-1}AC^{-T}$ y $\tilde{b} = C^{-1}b$. Se espera que la nueva M sea bien condicionada, o bien, que M sea una matriz con autovalores concentrados, de manera que el menor y el mayor autovalores no estén muy separados. Para implementar el método podríamos aplicar el Algoritmo 1 y resolver para $\tilde{x}$, luego al final resolver el sistema adicional $C^T x = \tilde{x}$. Sin embargo, definiendo $p_k \leftarrow C^T p_k$ en el algoritmo anterior y haciendo $M = CC^T$, basado en Golub y Van Loan [6] y en Oliveira y Sorensen [12], el método de gradientes conjugados precondicionado, presentado en el Algoritmo 2, llamado también como el algoritmo Mz, obtiene directamente la solución para x.

Una aplicación de este método, para resolver problemas de programación lineal de dimensión grande, se encuentra dentro la aplicación computacional PCx que fue utilizado por Suñagua y Oliveira [13] y otros autores que aplican este algoritmo en el método de puntos interiores, donde el precondicionador se obtiene usando la factorización controlada de Cholesky.

Dada una matriz A simétrica y definida positiva, un precondicionador trivial que puede ser útil es $C = \text{diag}(A)^{-1}$, o bien,

Algoritmo 2 Gradientes conjugados precondicionado

Datos: x^0, tal que $Ax_0 \approx b$
  $k = 0$
  Calcule el residuo inicial $r_0 = b - Ax_0$
  mientras $r_k \neq 0$ **haga**
    Resuelva $Mz_k = r_k$
    $k = k + 1$
    si k=1 **entonces**
      Inicialice $p_1 = z_0$
    sino
      Calcule $\beta_k = r_{k-1}^T z_{k-1} / r_{k-2}^T z_{k-2}$
      Actualice $p_k = z_{k-1} + \beta_k p_{k-1}$
    fin si
    Calcule $\alpha_k = r_{k-1}^T z_{k-1} / p_k^T A p_k$
    Actualice $x_k = x_{k-1} + \alpha_k p_k$
    Actualice $r_k = r_{k-1} - \alpha_k A p_k$
  fin mientras
  retorne $x = x_k$

$C = \text{diag}(A)^{-1/2}$, donde diag$(A)$ es la matriz diagonal cuya diagonal principal es el vector diagonal de A, $(a_{11}, a_{22}, \cdots, a_{nn})$.

Como se ha considerado en las secciones precedentes, el método de gradientes conjugados, precondicionado o no, resuelve sistemas lineales simétricos de rango completo donde la matriz de coeficientes tenga su número de condición razonablemente bueno.

En métodos de puntos interiores de programación lineal, el método de gradientes conjugados se aplica en cada iteración del método de puntos interiores para determinar la dirección de búsqueda. La mayoría de los problemas de programación lineal, como de NETLIB (`www.netlib.org`) se resuelve aproximadamente entre 10 y 100 iteraciones. Aunque el método termina en un número finito de iteraciones, para problemas con dimensiones grandes, el método converge en número menor de iteraciones que la dimensión de la matriz del sistema lineal que está dado por el número de restricciones. Mas detalles, se puede encontrar en Bocanegra *et al* [1], Velazco *et al* [14] y Suñagua y Oliveira [13].

Otra aplicación importante, descrita en Burden y Faires [2], está en la resolución numérica de ecuaciones diferenciales parciales mediante la aplicación de métodos de elementos finitos, donde afortunadamente las matrices involucradas no son tan mal condicionadas como en el método de puntos interiores.

En química computacional, en la optimización de estructura molecular llamado también minimización geométrica se minimiza la energía potencial de la superficie. Una referencia para este tipo de aplicaciones está en Chatzieleftheriou *et al* [4].

5. Implementaciones Computacionales

Muchos métodos de optimización en alguna etapa de sus iteraciones utilizan el método de gradientes conjugados o gradientes conjugados precondicionados si la matriz involucrada queda mal condicionada. Los programas computacionales que resuelven problemas reales con dimensiones grandes son muy complejos como para listar en cualquier publicación, pues ellos están implementados en lenguajes como `C`, `C++`, `Fortran`, etc. Quizá las implementaciones más didácticas están escritas en MATLAB y Python. Algunos enlaces de los métodos de optimización o de implementaciones públicas son

- Implementaciones de gradientes conjugados para diferentes lenguajes de alto y bajo nivel.
- Rutina de gradientes conjugados implementada en la librería de Álgebra Lineal de Python
- Implementación conceptual en Python para problemas pequeños
- Lista de algunos programas computacionales de PL libres y pagos
- CUTEr: Problemas de optimización para pruebas de algoritmos en formato SIF
- Problemas de Programación Lineal para pruebas en formato MPS

6. Conclusiones

Como se ha mostrado en el desarrollo de las secciones, el método de gradientes conjugados es uno de los métodos iterativos eficientes para minimizar funciones cuadráticas cuya matriz principal es simétrica y definida positiva. Este método no funciona bien si la matriz de la forma cuadrática está mal condicionada. Una forma de mejorar tal número de condición es aplicar un precondicionador y aplicar el método de gradientes conjugados precondicionado según el Algoritmo 2. Las heurísticas para encontrar el precondicionador sigue siendo un tema de investigación. Para matrices muy mal condicionadas, uno de los preconcionadores utilizados en la práctica se obtiene por la factorización controlada de Cholesky.

Referencias

[1] S. Bocanegra, F. Campos, A. R. Oliveira, Using a hybrid preconditioner for solving large-scale linear systems arising from interior point methods, Computational Optimization and Applications 36 (2-3) (2007) 149–164.

[2] R. L. Burden, J. D. Faires, Numerical analysis, Cengage Learning 9.

[3] F. F. Campos, J. S. Rollet, Controlled Cholesky factorization for preconditioning the conjugate gradient method, Oxford University Computing Laboratory, Numerical Analysis Group, 1995.

[4] S. Chatzieleftheriou, M. R. Adendorff, N. D. Lagaros, Generalized Potential Energy Finite Elements for Modeling Molecular Nanostructures, Journal of chemical information and modeling 56 (10) (2016) 1963–1978.

[5] J. Czyzyk, S. Mehrotra, M. Wagner, S. J. Wright, PCx user guide (Version 1.1), Optimization Technology Center, Northwestern University .

[6] G. H. Golub, C. F. Van Loan, Matrix computations, JHU Press, 4th. edn., ISBN 978-1-4214-0794-4, 2013.

[7] M. R. Hestenes, E. Stiefel, Methods of conjugate gradients for solving linear systems, vol. 49, NBS Washington, DC, 1952.
[8] N. J. Higham, Analysis of the Cholesky decomposition of a semi-definite matrix, M. G. Cox and S. J. Hammarling, (eds). Reliable Numerical Computation (1990) 161–185.
[9] D. S. Kershaw, The incomplete Cholesky—conjugate gradient method for the iterative solution of systems of linear equations, Journal of Computational Physics 26 (1) (1978) 43–65.
[10] D. G. Luenberger, Linear and nonlinear programming, Springer, 2003.
[11] A. Oliveira, D. Sorensen, CRPC-TR97772 November 1997, Tech. Rep., 1997.
[12] A. R. Oliveira, D. C. Sorensen, A new class of preconditioners for large-scale linear systems from interior point methods for linear programming, Linear Algebra and its applications 394 (2005) 1–24, England.
[13] P. Suñagua, A. R. Oliveira, A new approach for finding a basis for the splitting preconditioner for linear systems from interior point methods, Computational Optimization and Applications 67 (1) (2017) 111–127.
[14] M. Velazco, A. R. Oliveira, F. Campos, A note on hybrid preconditioners for large-scale normal equations arising from interior-point methods, Optimization Methods & Software 25 (2) (2010) 321–332.
[15] S. J. Wright, Primal-dual interior-point methods, vol. 54, SIAM, 1997.