# Un algoritmo cuasi Newton global para problemas de complementariedad no lineal

Carlos Andrés Arias Torres

Universidad del Cauca
Facultad de Ciencias Naturales, Exactas y de la Educación
Departamento de Matemáticas
Maestría en Ciencias Matemáticas
Popayán
2014

# Un algoritmo cuasi Newton global para problemas de complementariedad no lineal

Carlos Andrés Arias Torres

Trabajo de investigación presentado como requisito parcial para optar al título de Magíster en Ciencias Matemáticas

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

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

Universidad del Cauca
Facultad de Ciencias Naturales, Exactas y de la Educación
Departamento de Matemáticas
Maestría en Ciencias Matemáticas
Popayán
2014

Nota de aceptación

_____

_____

_____

_____

Director: **Dra. Rosana Pérez Mera**

_____

Jurado: **Dr. Jhon Jairo Duque Robles**

_____

Jurado: **Mg. Favián Enrique Arenas Aparicio**

Popayán, 24 de febrero de 2014

# AGRADECIMIENTOS

A mi familia por motivar cada uno de mis esfuerzos. A mi padre y a mi madre porque han hecho posible cada uno de mis sueños. A Isabel por brindarme su amor y su apoyo incondicional.

A mi directora de tesis, *Dra. Rosana Pérez Mera*, por el conocimiento que me compartió y la motivación que me brindó en este arduo camino. Porque con su visión y su experiencia me guió hacia este logro.

A mi codirector *Dr. Héctor Jairo Martínez*, por sus aportes e invaluables consejos que consolidaron esta investigación.

Al *Dr. Jhon Jairo Duque Robles* y al *Mg. Favián Enrique Arenas Aparicio* por sus acertadas sugerencias, fruto de sus conocimientos y su experiencia investigativa.

A mis profesores por enseñarme a ser un ser humano íntegro.

*Carlos Andrés Arias Torres*

# TABLA DE CONTENIDOS

# ÍNDICE DE TABLAS

# CAPÍTULO 1

# INTRODUCCIÓN

Sea $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n,\ F(\boldsymbol{x}) = (F_1(\boldsymbol{x}),\ F_2(\boldsymbol{x}), ...,\ F_n(\boldsymbol{x}))$ , una función no lineal y continuamente diferenciable. El *Problema de Complementariedad No Lineal* (PCNL), consiste en hallar un vector $\boldsymbol{x} \in \mathbb{R}^n$ tal que

$$\boldsymbol{x} \geq 0\ ;\ F(\boldsymbol{x}) \geq 0\ ;\ \boldsymbol{x}^T F(\boldsymbol{x}) = 0, \tag{1.1}$$

donde la expresión $\boldsymbol{y} \geq 0$ para un vector en $\mathbb{R}^n$ significa que todas sus componentes son no negativas.

La tercera condición de (1.1) implica que $x_i F_i(\boldsymbol{x}) = 0$ para $i = 1, 2, ..., n$ y por lo tanto, $x_i = 0$ o $F_i(\boldsymbol{x}) = 0$ , siendo esta la razón por la cual el problema recibe el calificativo de *complementariedad*. Esta condición trae consigo implícitamente la búsqueda de un equilibrio entre la *variable* del problema y el valor que toma la función que define el problema en dicha variable. Así, el concepto de *complementariedad* es sinónimo del concepto de *sistema en equilibrio* [17].

Por lo mencionado anteriormente, *problemas de complementariedad no lineal* surgen de forma natural en campos como la Ingeniería y la Física, por ejemplo, cuando se presentan situaciones de *contacto mecánico* [17], *equilibrio del tráfico* [35] y *problemas de lubricación elasto-hidrodinámicos* [23], así como en el campo de la Economía, particularmente cuando

se busca un equilibrio en los sistemas económicos [17].

Es por ello que ha sido de gran interés para la comunidad matemática el estudio de diversos métodos que permitan solucionar el PCNL, entre ellos, *métodos de homotopía*, que son derivados de métodos de punto fijo [37] [14]; *métodos basados en redes neuronales* [39][24][25] y *métodos de ecuaciones no diferenciables* [12][11]. En particular, estos últimos, consisten en reformular el PCNL como un sistema de ecuaciones no lineales, no diferenciable, usando para ello una función $\varphi : \mathbb{R}^2 \longrightarrow \mathbb{R}$ tal que

$$\varphi(a, b) = 0 \iff a \geq 0, \;\; b \geq 0, \;\; ab = 0\,, \tag{1.2}$$

conocida como *función de complementariedad* [21][38].

Geométricamente, a partir de (1.2), se infiere que la traza de la función $\varphi$ obtenida por la intersección con el plano $xy$, es la curva formada por los semiejes positivos $x$ y $y$, la cual no es diferenciable en $(0, 0)$, razón por la que $\varphi$ no es diferenciable [1].

Dos de las *funciones de complementariedad* más utilizadas en la solución de PCNL son la *función mínimo*[34][29] y la *función de Fischer*[19][18], definidas respectivamente por:

$$\varphi(a, b) = \text{mín}\{a, b\} \qquad\qquad \varphi(a, b) = \sqrt{a^2 + b^2} - a - b\,.$$

En 1998, *Kanzow y Kleinmichel* [21] presentaron una familia de funciones de complementariedad $\varphi_\lambda$ dependientes de un parámetro $\lambda \in (0, 4)$, definida por

$$\varphi_\lambda(a, b) = \sqrt{(a - b)^2 + \lambda ab} - a - b\,. \tag{1.3}$$

Observemos que esta familia de funciones incluye la *función de Fischer* como un caso particular, cuando $\lambda = 2$, y converge a un múltiplo de la *función mínimo* cuando $\lambda$ tiende a cero.

Así, si definimos la función $\Phi_\lambda : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ por

$$\Phi_\lambda(\boldsymbol{x}) = \begin{pmatrix} \varphi_\lambda(x_1, F_1(\boldsymbol{x})) \\ \vdots \\ \varphi_\lambda(x_n, F_n(\boldsymbol{x})) \end{pmatrix} \tag{1.4}$$

entonces un vector $\boldsymbol{x}^*$ es solución del PCNL, si y solo si, $\boldsymbol{x}^*$ es solución del sistema de ecuaciones

$$\Phi_\lambda(\boldsymbol{x}) = 0. \tag{1.5}$$

Como consecuencia de la no diferenciabilidad de $\varphi_\lambda$, el sistema de ecuaciones no lineales

(1.5) no es diferenciable. De esta forma, el PCNL es reformulado como un sistema de ecuaciones no lineales, no diferenciable.

Entre los métodos más populares para resolver sistemas de ecuaciones no lineales **diferenciables** de la forma $G(\boldsymbol{x}) = 0$, se encuentran el método de *Newton* [27] y métodos *Cuasi Newton* [13]. La principal diferencia entre ellos radica en que en el primero es necesario calcular, en cada iteración, la matriz jacobiana de $G$, lo que computacionalmente resulta costoso, mientras que en el segundo tipo de métodos basta con tomar una aproximación a dicha matriz, pagando a cambio un cierto precio que se ve reflejado en una disminución de la velocidad de convergencia del algoritmo respectivo [13].

Sin embargo, cuando el sistema de ecuaciones $G(\boldsymbol{x}) = 0$ **no es diferenciable**, como es el caso del sistema de ecuaciones (1.5), no tiene sentido hablar de la *"matriz jacobiana de G"*. En 1973, *Frank H. Clarke* [7][8][9] presentó el concepto de *jacobiano generalizado*, definido para $G\colon \mathbb{R}^n \longrightarrow \mathbb{R}^n$ *Lipschitz continua*[1], como el conjunto de matrices

$$\partial G(\boldsymbol{x}) \;=\; conv\Big\{ \lim_{k\to\infty} G'(\boldsymbol{x}_k) \in \mathbb{R}^{n\times n} : \lim_{k\to\infty} \boldsymbol{x}_k \;=\; \boldsymbol{x},\; \boldsymbol{x}_k \in D_G \Big\}, \tag{1.6}$$

donde $D_G$ es el conjunto de todos los vectores de $\mathbb{R}^n$ en los que $G$ es diferenciable y $conv\{\mathrm{A}\}$ representa la envolvente convexa del conjunto $A$. En particular, el conjunto

$$\partial_B G(\boldsymbol{x}) \;=\; \Big\{ \lim_{k\to\infty} G'(\boldsymbol{x}_k) \in \mathbb{R}^{n\times n} : \lim_{k\to\infty} \boldsymbol{x}_k \;=\; \boldsymbol{x},\; \boldsymbol{x}_k \in D_G \Big\} \tag{1.7}$$

se conoce como *B-jacobiano Generalizado* de $G$ en $\boldsymbol{x}$ [34]. Es decir, de (1.6) y (1.7), el *Jacobiano generalizado* de $G$ en $\boldsymbol{x}$ puede verse como el conjunto

$$\partial G(\boldsymbol{x}) = \; conv\,\{\partial_B G(\boldsymbol{x})\}. \tag{1.8}$$

Dado que cada *función de complementariedad* $\varphi_\lambda$ es *Lipschitz continua*, entonces la función $\Phi_\lambda$ también lo es y por tanto, su *jacobiano generalizado* existe.

Usando matrices del conjunto (1.8), Qi [33] propone el llamado método de *Newton generalizado* para resolver sistemas de ecuaciones *no lineales no diferenciables*, el cual, en cada iteración, resuelve un sistema lineal cuya matriz de coeficientes es una de las matrices del conjunto $\partial_B G(\boldsymbol{x})$. Por otra parte, para *problemas de complementariedad no lineal* también se han propuesto métodos *cuasi Newton* [31][1][6], los cuales han mostrado un

[1]Una función $G : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ es *Lipschitz continua*, si existe una constante positiva $K$ tal que

$$\|G(\boldsymbol{x}) - G(\boldsymbol{y})\|_2 \;\leq\; K\|\boldsymbol{x} - \boldsymbol{y}\|_2,$$

para todo $\boldsymbol{x}$ y $\boldsymbol{y}$ en $\mathbb{R}^n$

buen desempeño numérico.

En general, el método de *Newton* para resolver sistemas de ecuaciones no lineales, bajo ciertas hipótesis, presenta una *tasa de convergencia q-cuadrática* [13] siempre y cuando el punto inicial del algoritmo esté suficientemente cerca de la solución del sistema (es decir, es un *método local*). Preguntas que surgen al implementar algoritmos basados en el método de *Newton* y que pueden llevar a problemas en su convergencia son ¿qué tan cerca debe estar el punto inicial de la solución del sistema? ¿cómo hacer una buena elección del punto inicial para implementar el algoritmo?

Con el fin de evitar inconvenientes en la elección del punto inicial, los algoritmos locales son *globalizados*, es decir, son dotados de un criterio que permite ejecutarlos cuando su punto inicial está lejos de la solución del problema.

Algunos de esos criterios se basan en ciertas funciones conocidas como *funciones de mérito* [27]. La *función de mérito* más utilizada en el caso de sistemas de ecuaciones no lineales de la forma $G(\boldsymbol{x}) = 0$, es la función $f\colon \mathbb{R}^n \longrightarrow \mathbb{R}$ definida por $f(\boldsymbol{x}) = \|G(\boldsymbol{x})\|_2^2$. Teniendo como base esta función, una nueva iteración en un algoritmo global será aceptada siempre y cuando el valor de la función $f$ decrezca con respecto a la iteración actual del mismo; este requerimiento conduce a resolver el problema de minimización

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{Minimizar} \quad \frac{1}{2}\|G(\boldsymbol{x})\|_2^2 \tag{1.9}$$

donde el factor $\frac{1}{2}$ es adicionado por conveniencia algebraica.

Cabe destacar que toda solución del sistema de ecuaciones $G(\boldsymbol{x}) = 0$ es un *minimizador* de (1.9), sin embargo, pueden existir soluciones locales de (1.9) que no son soluciones del sistema de ecuaciones no lineales asociado.

Teniendo en cuenta lo anterior y con el fin de *globalizar* algún método tipo *Newton* que busque hallar una solución del sistema de ecuaciones no lineales (1.5), recurriendo al *jacobiano generalizado* de $\Phi_\lambda$, se puede hacer uso de una *función de mérito* asociada a este sistema. En efecto, para tal fin basta considerar $\Psi_\lambda : \mathbb{R}^n \longrightarrow \mathbb{R}$, definida por

$$\Psi_\lambda(\boldsymbol{x}) = \frac{1}{2}\Phi_\lambda(\boldsymbol{x})^T\Phi_\lambda(\boldsymbol{x}) = \frac{1}{2}\|\Phi_\lambda(\boldsymbol{x})\|_2^2. \tag{1.10}$$

Observemos que $\Psi_\lambda$ es *continuamente diferenciable* (Teorema 3.1 en [21] y Proposición 3.4 en [15]). Con esta *función de mérito*, podemos reformular nuevamente el sistema de ecuaciones no lineales, no diferenciable (1.5) como el siguiente *problema de minimización*

*diferenciable*

$$\underset{\boldsymbol{x} \in \mathbb{R}^n}{Minimizar} \quad \Psi_\lambda(\boldsymbol{x})\,. \tag{1.11}$$

En [21], los autores proponen un algoritmo global tipo *Newton generalizado* para resolver el problema de minimización (1.11). Recientemente, en [1], el autor propone un algoritmo *cuasi Newton local* que permite resolver el PCNL recurriendo al sistema de ecuaciones (1.5) y deja abierta la posibilidad de introducir estrategias de globalización para mejorar el desempeño de su algoritmo.

Teniendo en cuenta lo anterior y pensando en las limitaciones que puede tener un algoritmo *local*, en este trabajo de investigación, proponemos un algoritmo *cuasi Newton global* para resolver el PCNL a través de la minimización de la función de mérito (1.10), es decir, proponemos una globalización del algoritmo presentado en [1], usando ideas del algoritmo expuesto en [21]. Desarrollamos su teoría de convergencia global y realizamos pruebas numéricas preliminares que muestran un buen desempeño del mismo. Adicionalmente, analizamos numéricamente el comportamiento del algoritmo propuesto cuando cambiamos la *búsqueda lineal estándar*, por una *búsqueda lineal no monótona* [20]. También incluimos una exploración preliminar de la variación del parámetro $\lambda$ y su impacto en la convergencia global de nuestro algoritmo

El documento está organizado de la siguiente forma: En el **Capítulo 2**, presentamos algunos resultados teóricos que serán útiles en el desarrollo de la teoría de convergencia del algoritmo que propondremos. En el **Capítulo 3**, proponemos un *Algoritmo cuasi Newton global* para resolver el PCNL mediante la minimización de la *función de mérito* definida en (1.10) y bajo ciertas hipótesis, desarrollamos su teoría de convergencia global. En la primera parte del **Capítulo 4**, presentamos los resultados numéricos obtenidos al resolver con el algoritmo *cuasi Newton global* propuesto, algunos *problemas de complementariedad no lineal* y comparamos estos resultados con los obtenidos al resolver los mismos problemas con el algoritmo tipo *Newton generalizado* presentado en [21]. En la segunda parte, analizamos numéricamente el desempeño de nuestro algoritmo cuando cambiamos la *búsqueda lineal monótona* por una *búsqueda lineal no monótona* [20]. En la última parte de este capítulo, presentamos los resultados obtenidos al explorar el comportamiento del algoritmo cuando variamos el parámetro $\lambda$. Finalmente, en el **Capítulo 5**, hacemos algunos comentarios finales y una propuesta para futuros trabajos en el tema.

# CAPÍTULO 2

# PRELIMINARES

En este capítulo, presentamos algunas definiciones y resultados teóricos que han sido demostrados previamente por algunos autores y que serán útiles en el desarrollo de nuestra investigación.

De (1.1), una solución $\boldsymbol{x}^*$ del PCNL se caracteriza porque

$$\boldsymbol{x}^* \geq 0 \qquad F(\boldsymbol{x}^*) \geq 0 \qquad F(\boldsymbol{x}^*)^T \boldsymbol{x}^* = 0.$$

Asociados a esta solución, tenemos los siguientes conjuntos de índices:

$$\begin{aligned} \alpha =& \{i \in I \,:\, x_i^* > 0,\, F_i(\boldsymbol{x}^*) = 0\} \\ \beta =& \{i \in I \,:\, x_i^* = 0 = F_i(\boldsymbol{x}^*)\} \\ \gamma =& \{i \in I \,:\, x_i^* = 0,\, F_i(\boldsymbol{x}^*) > 0\}. \end{aligned}$$

Cabe mencionar que si $\beta \neq \emptyset$, la solución $\boldsymbol{x}^*$ se denomina *solución degenerada*.

Dado que la función $\Phi_\lambda$ dada en (1.4) es no diferenciable y *Lipschitz continua*, su *jaco-*

*biano generalizado* existe y está definido por

$$\partial\Phi_\lambda(\boldsymbol{x}) \;=\; conv\Big\{\lim_{k\to\infty}\,\Phi'_\lambda(\boldsymbol{x}_k)\in\mathbb{R}^{n\times n}:\,\lim_{k\to\infty}\,\boldsymbol{x}_k\;=\;\boldsymbol{x},\;\boldsymbol{x}_k\in D_{\Phi_\lambda}\Big\}\;=\;conv\{\partial_B\Phi_\lambda(\boldsymbol{x})\}$$

donde $D_{\Phi_\lambda}$ es el conjunto de todos los vectores de $\mathbb{R}^n$ en los que $\Phi_\lambda$ es diferenciable.

En [1], el autor construye matrices $H\in\partial\Phi_\lambda(\boldsymbol{x})$ de la forma

$$H=\begin{pmatrix}[H]_1\\ \vdots\\ [H]_n\end{pmatrix} \tag{2.1}$$

con

$$[H]_i=\begin{cases}(\chi(x_i,F_i(\boldsymbol{x}))-1)\,\boldsymbol{e}_i^T+(\psi(x_i,F_i(\boldsymbol{x}))-1)\nabla F_i(\boldsymbol{x})^T, & i\notin\beta\\[2ex] (\chi(z_i,\nabla F_i(\boldsymbol{x})^T\boldsymbol{z})-1)\,\boldsymbol{e}_i^T+(\psi(z_i,\nabla F_i(\boldsymbol{x})^T\boldsymbol{z})-1)\nabla F_i(\boldsymbol{x})^T, & i\in\beta\end{cases}$$

donde $\boldsymbol{z}$ es un vector tal que $z_i\neq 0$ cuando $i\in\beta$, $\{\boldsymbol{e}_1,\ldots,\boldsymbol{e}_n\}$ es la base canónica de $\mathbb{R}^n$ y las funciones $\chi$ y $\psi$ son las derivadas parciales de la función

$$G_\lambda(a,b)=\sqrt{(a-b)^2+\lambda ab}\,, \tag{2.2}$$

es decir,

$$\chi(a,b)=\frac{2(a-b)+\lambda b}{2G_\lambda(a,b)}\qquad y\qquad \psi(a,b)=\frac{-2(a-b)+\lambda a}{2G_\lambda(a,b)}\,. \tag{2.3}$$

Además, teniendo en cuenta la forma de las matrices $H\in\partial\Phi_\lambda(\boldsymbol{x})$ dada en (2.1), el autor de [1] propone aproximaciones *cuasi Newton* de dichas matrices de la forma

$$B=\begin{pmatrix}[B]_1\\ \vdots\\ [B]_n\end{pmatrix} \tag{2.4}$$

con

$$[B]_i = \begin{cases} (\chi(x_i, F_i(\boldsymbol{x})) - 1)\boldsymbol{e}_i^T + (\psi(x_i, F_i(\boldsymbol{x})) - 1)[A]_i, & i \notin \beta \\ \\ (\chi(z_i, [A]_i\boldsymbol{z}) - 1)\boldsymbol{e}_i^T + (\psi(z_i, [A]_i\boldsymbol{z}) - 1)[A]_i, & i \in \beta \end{cases}$$

donde la matriz

$$A = \begin{pmatrix} [A]_1 \\ \vdots \\ [A]_n \end{pmatrix} \tag{2.5}$$

es una aproximación a $F'(\boldsymbol{x})$, la *matriz jacobiana* de $F$ en $\boldsymbol{x}$.

Con respecto a las derivadas parciales dadas en (2.3), cabe destacar que, para cualquier vector no nulo $(a,\, b\,)$, estas derivadas están acotadas de la siguiente forma [1][21]:

$$|\chi(a,b)| \leq 1 \qquad y \qquad |\psi(a,b)| \leq \sqrt{2}. \tag{2.6}$$

Con base en estas consideraciones, en [1], el autor propone un algoritmo *cuasi Newton local* para resolver el PCNL a través su reformulación como el sistema de ecuaciones no lineales, no diferenciable (1.5), cuya forma genérica presentamos a continuación ya que nuestro interés es introducir en dicho algoritmo una estrategia de globalización que permita obtener un nuevo algoritmo *cuasi Newton global* para resolver el PCNL.

**Algoritmo 2.1.** ***Tipo cuasi Newton local para*** $\Phi_\lambda(\boldsymbol{x}) = 0$.

*Dados* $\boldsymbol{x}_0$ *y* $\lambda \in (0,\, 4)$, *para* $k = 1,\, 2, \ldots,$ *se genera la iteración siguiente por*

$$x_{k+1} = x_k - B_k^{-1}\Phi_\lambda(x_k);$$

donde $B_k$ es de la forma (2.4).

Por otra parte, en [1], se demuestra que la distancia entre una matriz $H$ y su respectiva aproximación $B$ está acotada por un valor constante y, que bajo ciertas condiciones, la matriz $B$ de la forma (2.4) es no singular. Formalmente, los resultados son los siguientes

**Teorema 2.1.** [1] *Sean* $\boldsymbol{x}^*$ *una solución del* PCNL*;* $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$, *una función de clase* $C^1$ *cuya matriz jacobiana es Lipschitz continua,* $\epsilon$ *y* $\delta$ *constantes positivas dadas;*

*$H$ y $B$ definidas por* (2.1) *y* (2.4)*, respectivamente. Entonces, para cada* $\boldsymbol{x} \in \mathcal{B}(\boldsymbol{x}^*, \epsilon)$ *y para cada* $A \in \mathcal{B}(F'(\boldsymbol{x}^*), \delta)$*, existe una constante* $\theta > 0$ *tal que*

$$\|H - B\|_\infty \leq \theta.$$

**Teorema 2.2.** [1] *Sean* $\boldsymbol{x}^*$ *una solución del* PCNL *y* $B$ *la matriz definida por* (2.4)*. Existen constantes positivas* $\epsilon_0$ *y* $\delta_0$ *tales que si*

$$\|\boldsymbol{x} - \boldsymbol{x}^*\|_\infty \leq \epsilon_0 \qquad y \qquad \|A - F'(\boldsymbol{x}^*)\|_\infty \leq \delta_0,$$

*la función* $\mathcal{Q}$ *definida por*

$$\mathcal{Q}(\boldsymbol{x}, A) = \boldsymbol{x} - B^{-1}\Phi_\lambda(\boldsymbol{x})$$

*está bien definida.*

En particular, este teorema permite garantizar que, si una sucesión $\{\boldsymbol{x}_k\}$ converge a $\boldsymbol{x}^*$ entonces existe un entero positivo $\overline{k}$ tal que, para todo $k > \overline{k}$, la matriz $B_k^{-1}$ existe.

En el desarrollo de la teoría de convergencia para algoritmos que resuelvan el PCNL es importante establecer condiciones suficientes para la no singularidad de las matrices en el jacobiano generalizado en una solución del problema. Por ello, introducimos a continuación el concepto de *regularidad* que nos ayuda para tal fin.

**Definición 2.1.** *Sea* $\boldsymbol{x}^*$ *una solución del* PCNL*.*

1. *Si todas las matrices* $H \in \partial_B \Phi_\lambda(\boldsymbol{x}^*)$ *son no singulares,* $\boldsymbol{x}^*$ *es llamada una solución BD-regular, .*

2. *Si la submatriz* $F'(\boldsymbol{x}^*)_{\alpha\alpha}$[1] *es no singular y el complemento de Schur*

$$F'(\boldsymbol{x}^*)_{\beta\beta} - F'(\boldsymbol{x}^*)_{\beta\alpha} F'(\boldsymbol{x}^*)_{\alpha\alpha}^{-1} F'(\boldsymbol{x}^*)_{\alpha\beta}$$

*es una P-matriz* [2]*,* $\boldsymbol{x}^*$ *es llamada una solución R-regular.*

[1] Dada una matriz $A = (a_{ij})$ de tamaño $m \times n$ y conjuntos de índices $\eta$ y $\tau$, la matriz $A_{\eta\tau}$ es aquella con componentes $a_{ij}$ tal que $i \in \eta$ y $j \in \tau$.

[2] Una matriz $M \in \mathbb{R}^{n\times n}$ es una *P-matriz,* si para cualquier vector no nulo $\boldsymbol{y} \in \mathbb{R}^n$, existe un índice $i_0 = i_0(\boldsymbol{y}) \in \{1, 2, \ldots, n\}$, tal que $y_{i_0}[M\boldsymbol{y}]_{i_0} > 0$.

El siguiente teorema da una condición suficiente para garantizar la no singularidad de los elementos del *jacobiano generalizado* de $\Phi_\lambda$ en una solución del PCNL [21].

**Teorema 2.3.** [21] *Si* $\boldsymbol{x}^* \in \mathbb{R}^n$ *es una solución R-regular del* PCNL*, entonces las matrices en el jacobiano generalizado* $\partial\Phi_\lambda(\boldsymbol{x}^*)$ *son no singulares.*

Una consecuencia inmediata de este resultado es dada en el siguiente corolario.

**Corolario 2.1.** *Si* $\boldsymbol{x}^*$ *es una solución R-regular del* PCNL*, entonces* $\boldsymbol{x}^*$ *es una solución BD-regular.*

*Demostración.* Supongamos que $\boldsymbol{x}^*$ es una solución *R-regular* del PCNL. Por hipótesis y el **Teorema 2.3**, todas la matrices de $\partial\Phi_\lambda(\boldsymbol{x}^*)$ son no singulares. Por otra parte, de (1.8) tenemos que

$$\partial\Phi_\lambda(\boldsymbol{x}^*) = \mathit{conv}\{\partial_B\Phi_\lambda(\boldsymbol{x}^*)\},$$

con lo cual

$$\partial_B\Phi_\lambda(\boldsymbol{x}^*) \subseteq \partial\Phi_\lambda(\boldsymbol{x}^*).$$

Así, si $H \in \partial_B\Phi_\lambda(\boldsymbol{x}^*)$, entonces $H \in \partial\Phi_\lambda(\boldsymbol{x}^*)$ y por lo tanto, es no singular. En consecuencia, $\boldsymbol{x}^*$ es una solución BD-regular del PCNL. ⧫

Las dos definiciones siguientes son útiles para la presentación del **Teorema 2.4**, cuya demostración se puede consultar en [16].

**Definición 2.2.** [21] *Una función* $G\colon \mathbb{R}^n \longrightarrow \mathbb{R}^m$ *localmente Lipschitz continua y direccionalmente diferenciable es llamada semisuave en un punto* $\boldsymbol{x}$*, si para cada* $\boldsymbol{d}$ *que converge al vector cero y para cada matriz* $H \in \partial G(\boldsymbol{x}+\boldsymbol{d})$ *se tiene que*

$$H\,\boldsymbol{d} - G'(\boldsymbol{x};\,\boldsymbol{d}) = o(\|\boldsymbol{d}\|).$$

**Definición 2.3.** [12] *Una función* $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ *es de clase* $SC^1$*, si es continuamente diferenciable y su gradiente es una función semisuave.*

Para el enunciado del siguiente teorema, supongamos que $\boldsymbol{x}^*$ es un punto de acumulación de la sucesión $\{\boldsymbol{x}_k\}$.

**Teorema 2.4.** [16] *Sean* $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ *una función de clase* $SC^1$ *y* $\delta > 0$ *. Si* $\overline{Df}(\boldsymbol{x};\, \boldsymbol{d})$ *es una aproximación de la derivada direccional de* $f$ *en* $\boldsymbol{x}$ *en la dirección* $\boldsymbol{d}$*, tal que para todo* $\boldsymbol{x}_k \in \mathcal{B}(\boldsymbol{x}^*; \delta)$

$$\nabla f(\boldsymbol{x}_k)^T \boldsymbol{d}_k = \overline{Df}(\boldsymbol{x}_k; \boldsymbol{d}_k) + o(\|\boldsymbol{d}_k\|^2),$$

*entonces para cualquier* $\sigma \in \left(0, \frac{1}{2}\right)$*, existe un* $\overline{k}$ *tal que para todo* $k > \overline{k}$

$$f(\boldsymbol{x}_k + \boldsymbol{d}) \leq f(\boldsymbol{x}_k) + \sigma \overline{Df}(\boldsymbol{x}_k; \boldsymbol{d}_k).$$

El siguiente resultado presenta una característica particular de la *función de mérito* que queremos minimizar y cuya demostración se puede consultar en [12].

**Teorema 2.5.** [12] *Sean* $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$*,* $F(\boldsymbol{x}) = (F_1(\boldsymbol{x}), F_2(\boldsymbol{x}), \ldots, F_n(\boldsymbol{x}))$ *una función no lineal y continuamente diferenciable,* $\Phi_\lambda$ *la función dada en* (1.4) *asociada a* $F$ *y* $\Psi_\lambda$ *la función de mérito* (1.10) *asociada a* $\Phi_\lambda$ *. Si para todo* $i = 1, 2, \ldots, n$*, la función* $F_i$ *es de clase* $SC^1$*, entonces* $\Psi_\lambda$ *es también de clase* $SC^1$*.*

Para finalizar, recordemos que la función de mérito dada en (1.10) es *diferenciable*. En [21] prueban que el gradiente de esta función se puede calcular usando las matrices $H \in \partial\Phi_\lambda(\boldsymbol{x})$ de la siguiente manera

**Teorema 2.6.** [21] *Para cualquier matriz* $H \in \partial\Phi_\lambda(\boldsymbol{x})$*, la función de mérito* $\Psi_\lambda$ *dada en* (1.10) *es continuamente diferenciable y*

$$\nabla\Psi_\lambda(\boldsymbol{x}) = H^T\Phi_\lambda(\boldsymbol{x}). \tag{2.7}$$

# CAPÍTULO 3

# ALGORITMO Y TEORÍA DE CONVERGENCIA

En este capítulo, proponemos un nuevo algoritmo *cuasi Newton global* para resolver el PCNL indirectamente, a través de la solución del problema de minimización (1.11). Además, para este algoritmo, desarrollamos su teoría de convergencia global.

El algoritmo que proponemos es básicamente un algoritmo de búsqueda direccional, cuya idea geométrica es tomar pasos que conduzcan a un descenso de la función $\Psi_\lambda$. Para cumplir este objetivo, el paso o dirección utilizada en el algoritmo, siempre que sea posible, será un paso *cuasi Newton* como el sugerido en [1] y donde, para seleccionar el tamaño del paso, usamos una estrategia de *búsqueda lineal (monótona)* [13] como la usada en el algoritmo tipo *Newton generalizado* propuesto en [21].

De acuerdo con lo mencionado en el párrafo anterior, el algoritmo que proponemos a continuación es una globalización del algoritmo *cuasi Newton local* propuesto en [1].

**Algoritmo 3.1.** ***Tipo cuasi Newton global para*** $\Phi_\lambda(\boldsymbol{x}) = 0$

***P.0*** *Inicialización*

*Escoja* $\lambda \in (0,4)$, $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\rho > 0$, $\mu \in (0,1)$, $\sigma \in (0, \frac{1}{2})$, $p > 2$, $\varepsilon \geq 0$, $N > 0$ *y haga* $k := 0$ *y* $A_0 = F'(x_0)$.

***P.1*** *Criterio de parada*

*Si* $\|B_k^T \Phi_\lambda(\boldsymbol{x}_k)\| \leq \varepsilon$ *o* $k > N$ *pare.*

***P.2*** *Búsqueda direccional*

*Calcule* $B_k$ *como en* (2.4) *y halle* $\boldsymbol{d}_k \in \mathbb{R}^n$ *tal que*

$$B_k \boldsymbol{d}_k = -\Phi_\lambda(\boldsymbol{x}_k). \tag{3.1}$$

*Si el sistema no tiene solución o si* $\Phi_\lambda(\boldsymbol{x}_k)^T B_k \boldsymbol{d}_k > -\rho \|\boldsymbol{d}_k\|^p$ *haga* $\boldsymbol{d}_k = -B_k^T \Phi_\lambda(\boldsymbol{x}_k)$.

***P.3*** *Búsqueda lineal*

*Haga* $t_k := \max\{\mu^l \,|\, l = 0, 1, 2, ...\}$ *tal que*

$$\Psi_\lambda(\boldsymbol{x}_k + t_k \boldsymbol{d}_k) \leq \Psi_\lambda(\boldsymbol{x}_k) + \sigma t_k B_k^T \Phi_\lambda(\boldsymbol{x}_k) \boldsymbol{d}_k \,. \tag{3.2}$$

***P.4*** *Actualización*

*Haga* $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + t_k \boldsymbol{d}_k$, $k = k + 1$ *, actualize* $A_k$ *como en* (3.4), (3.5) *o* (3.6) *y vuelva a* ***P.1***.

Observemos que la matriz aproximación $B_k$ dada por (2.4) depende a su vez de la aproximación $A_k$ de la matriz jacobiana de $F$ en $\boldsymbol{x}_k$. Así, una pregunta natural que surge es ¿cómo actualizar la matriz $A_k$? Es decir, ¿quién es $A_{k+1}$?

La forma como se actualice $A_{k+1}$ da lugar a un variado abanico de métodos *cuasi Newton*, entre los que se encuentran los llamados *métodos secantes de cambio mínimo*, en los que la actualización de $A_k$, la matriz $A_{k+1}$, debe satisfacer la llamada *ecuación secante* [27]

$$A_{k+1}(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k) = F(\boldsymbol{x}_{k+1}) - F(\boldsymbol{x}_k) \tag{3.3}$$

y su cambio, medido en alguna norma, con respecto a la matriz $A_k$, debe ser mínimo. Ejemplos de este tipo de actualizaciones son las siguientes:

1. Actualización "buena" de *Broyden* [5]

$$A_{k+1} = A_k + \frac{(\boldsymbol{y}_k - B_k \boldsymbol{s}_k)\boldsymbol{s}_k^T}{\boldsymbol{s}_k^T \boldsymbol{s}_k}\,. \tag{3.4}$$

2. Actualización "mala" de *Broyden* [4]

$$A_{k+1} = A_k + \frac{(\boldsymbol{y}_k - B_k \boldsymbol{s}_k)\boldsymbol{e}_{j_k}^T B_k}{\boldsymbol{e}_{j_k}^T B_k \boldsymbol{s}_K} \tag{3.5}$$

   donde $j_k$ es definido por $\|y_k\|_\infty = |\boldsymbol{e}_{j_k}^T \boldsymbol{y}_k|\,.$

3. Actualización de *Schubert* [10]

$$A_{k+1} = A_k + \frac{F(\boldsymbol{x}_k)\boldsymbol{s}_k^T}{\boldsymbol{s}_k^T \boldsymbol{s}_k} \tag{3.6}$$

donde $\boldsymbol{y}_k = F(\boldsymbol{x}_{k+1}) - F(\boldsymbol{x}_k) \quad y \quad \boldsymbol{s}_k = \boldsymbol{x}_{k+1} - \boldsymbol{x}_k.$

Las tres actualizaciones anteriores las usaremos en el **Capítulo 4**, cuando analicemos el desempeño numérico de nuestro algoritmo.

Teniendo en cuenta que nuestro deseo es no calcular las matrices $H_k$ en el *jacobiano generalizado* de la función $\Phi_\lambda$, proponemos algunas variaciones en las condiciones clásicas de búsqueda direccional y en la búsqueda lineal (condición de *Armijo* [27]).

Por el **Teorema 2.6**, para calcular el gradiente de la función $\Psi_\lambda$ en cada iteración necesitaríamos las matrices $H_k$ en el *jacobiano generalizado* de $\Phi_\lambda$ en $\boldsymbol{x}_k$, como alternativa a dicho cálculo proponemos realizar una aproximación a dicho gradiente, esta aproximación, que denotaremos $\overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)$, requiere de las matrices $B_k$ definidas en (2.4) y de la evaluación de la función $\Phi_\lambda$ en $\boldsymbol{x}_k$, explícitamente, proponemos la siguiente aproximación a $\nabla\Psi_\lambda(\boldsymbol{x}_k)$,

$$\overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k) = B_k^T \Phi_\lambda(\boldsymbol{x}_k). \tag{3.7}$$

A continuación presentamos las hipótesis bajo las cuales desarrollamos la teoría de convergencia global del **Algoritmo 3.1**.

## 3.1. Hipótesis

**H1.** Existe $\boldsymbol{x}^* \in \mathbb{R}^n$ tal que $\Phi_\lambda(\boldsymbol{x}^*) = 0$.

**H2.** Las funciones $F_i$ son de clase $SC^1$, es decir, son continuamente diferenciables y su gradiente es una función semisuave.

**H3.** La matriz jacobiana de $F$ es *Lipschitz continua* en una vecindad de $\boldsymbol{x}^*$.

## 3.2. Teoría de convergencia

A continuación, presentamos dos lemas que serán útiles en la demostración de los teoremas de convergencia del **Algoritmo 3.1**. En el primero de ellos, probamos que bajo ciertas condiciones, la norma de las matrices que aproximan a las respectivas matrices del jacobiano generalizado de $\Phi_\lambda$, es acotada. Cabe mencionar que, en nuestra demostración, hacemos uso de la norma infinito para vectores y su respectiva norma inducida para matrices [32]; sin embargo, teniendo en cuenta que las normas en $\mathbb{R}^n$ son equivalentes, el resultado probado es independiente de la norma usada. En el segundo lema, probamos una equivalencia entre el gradiente de la *función de mérito* $\Psi_\lambda$ y la aproximación a este que proponemos en (3.7).

**Lema 3.1.** *Sean $B$ como en (2.4) y $\delta > 0$. Para toda matriz $A \in \mathcal{B}(F'(\boldsymbol{x}^*); \delta)$, existe una constante $\eta$ tal que*

$$\|B\| \leq \eta. \tag{3.8}$$

*Demostración.* Consideremos la matriz $B$ definida en (2.4). De la definición de *norma matricial infinito* [32],

$$\|B\|_\infty = \max_{1 \leq i \leq n} \{\|[B]_i\|_1\}. \tag{3.9}$$

Supongamos que el máximo en (3.9) se alcanza en la fila $j$. Si $j \notin \beta$ entonces $\boldsymbol{x}_j \neq 0$ o $F_j(\boldsymbol{x}) \neq 0$, luego por (2.6)

$$|\chi(x_j,\, F_j(\boldsymbol{x}))| \leq 1 \qquad y \qquad |\psi(x_j,\, F_j(\boldsymbol{x}))| \leq \sqrt{2},$$

además,

$$\begin{aligned}\|B\|_\infty &= \|(\chi(x_j,\, F_j(\boldsymbol{x})) - 1)e_j^T + (\psi(x_j,\, F_j(\boldsymbol{x})) - 1)[A]_j\|_\infty \\ &\leq |\chi(x_j,\, F_j(\boldsymbol{x})) - 1| + |\psi(x_j,\, F_j(\boldsymbol{x})) - 1|\|[A]_j\|_\infty \\ &\leq 2 + \left(1 + \sqrt{2}\right) \|[A]_j\|_\infty \\ &\leq 2 + \left(1 + \sqrt{2}\right) (\|F'(\boldsymbol{x}^*)\|_\infty + \delta).\end{aligned}$$

Si $j \in \beta$, siguiendo un procedimiento idéntico, obtenemos la misma cota anterior. Por lo tanto, existe

$$\eta = 2 + \left(1 + \sqrt{2}\right) (\|F'(\boldsymbol{x}^*)\|_\infty + \delta)$$

tal que $\|B\|_\infty \leq \eta$. ♦

**Lema 3.2.** *Si* $\overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)$ *es la aproximación a* $\nabla\Psi_\lambda(\boldsymbol{x}_k)$ *dada en* (3.7)*, entonces*

$$\nabla\Psi_\lambda(\boldsymbol{x}_k)^T \boldsymbol{d}_k = \overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)^T \boldsymbol{d}_k + o(\|\boldsymbol{d}_k\|_\infty^2). \tag{3.10}$$

*Demostración.* De (2.7) y (3.7),

$$\begin{aligned}|\nabla\Psi_\lambda(\boldsymbol{x}_k)^T \boldsymbol{d}_k - \overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)^T \boldsymbol{d}_k| &= |\Phi_\lambda(\boldsymbol{x}_k)^T H_k \boldsymbol{d}_k - \Phi_\lambda(\boldsymbol{x}_k)^T B_k \boldsymbol{d}_k| \\ &= |\Phi_\lambda(\boldsymbol{x}_k)^T (H_k - B_k)\boldsymbol{d}_k|.\end{aligned} \tag{3.11}$$

Usando la desigualdad de *Cauchy-Schwarz* [36] y la relación entre las normas vectoriales *dos* e *infinito* en (3.11), obtenemos la siguiente desigualdad

$$|\nabla\Psi_\lambda(\boldsymbol{x}_k)^T \boldsymbol{d}_k - \overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)^T \boldsymbol{d}_k| \leq\ n\, \|\Phi_\lambda(\boldsymbol{x}_k)\|_\infty \|H_k - B_k\|_\infty \|\boldsymbol{d}_k\|_\infty.$$

Por el **Teorema 2.1**, para un $k$ suficientemente grande, existe una constante $\theta$ tal que $\|H_k - B_k\|_\infty \leq \theta$ , en consecuencia

$$|\nabla\Psi_\lambda(\boldsymbol{x}_k)^T \boldsymbol{d}_k - \overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)^T \boldsymbol{d}_k| \ \leq\ n\,\theta\, \|\Phi_\lambda(\boldsymbol{x}_k)\|_\infty \|\boldsymbol{d}_k\|_\infty$$

de donde,

$$0 \leq\ \frac{|\nabla\Psi_\lambda(\boldsymbol{x}_k)^T \boldsymbol{d}_k - \overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)^T \boldsymbol{d}_k|}{\|\boldsymbol{d}_k\|_\infty^2} \leq\ \frac{n\,\theta\, \|\Phi_\lambda(\boldsymbol{x}_k)\|_\infty}{\|\boldsymbol{d}_k\|_\infty},$$

por lo tanto, teniendo en cuenta que con el **Algoritmo 3.1** se busca minimizar la función de mérito $\Psi_\lambda$ y que $\boldsymbol{d}_k$ es una dirección de descenso, podemos inferir que aunque la sucesión $\{\|\boldsymbol{d}_k\|\}$ converja a cero, la sucesión $\{\|\Phi_\lambda(\boldsymbol{x}_k)\|\}$ lo hará más rápido, en consecuencia

$$\lim_{k\to\infty} \frac{|\nabla\Psi_\lambda(\boldsymbol{x}_k)^T\boldsymbol{d}_k - \overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)^T\boldsymbol{d}_k|}{\|\boldsymbol{d}_k\|_\infty^2} = 0$$

lo que demuestra (3.10). ♦

Los siguientes teoremas de convergencia son análogos a los presentados para un método tipo Newton (no suave) que busque resolver PCNL.

**Teorema 3.1.** *Todo punto de acumulación de la sucesión* $\{\boldsymbol{x}_k\}$ *generada por el* **Algoritmo 3.1** *es un punto estacionario de* $\Psi_\lambda$.

*Demostración.* Sea $\boldsymbol{x}^*$ un punto de acumulación de la sucesión $\{\boldsymbol{x}_k\}$ generada por el **Algoritmo 3.1**.

Si para un conjunto infinito de índices $J$, la dirección dada en el paso 2 del **Algoritmo 3.1** es siempre $\boldsymbol{d}_k = -\nabla\Psi_\lambda(\boldsymbol{x}_k)$, entonces cualquier punto límite de la subsucesión $\{\boldsymbol{x}_k\}_J$ es un punto estacionario de $\Psi_\lambda$ [2]. Luego, sin pérdida de generalidad, podemos suponer que la dirección siempre está dada por

$$B_k\boldsymbol{d}_k = -\Phi_\lambda(\boldsymbol{x}_k). \tag{3.12}$$

Supongamos que $\boldsymbol{x}^*$ no es un punto estacionario de $\Psi_\lambda$, es decir, $\nabla\Psi_\lambda(\boldsymbol{x}^*)$ no es el vector cero. Demostraremos que existen constantes positivas $m$ y $M$ tales que

$$m \le \|\boldsymbol{d}_k\| \le M. \tag{3.13}$$

En efecto, de (3.12)

$$\|\Phi_\lambda(\boldsymbol{x}_k)\| \le \|B_k\|\|\boldsymbol{d}_k\|$$

de donde

$$\frac{\|\Phi_\lambda(\boldsymbol{x}_k)\|}{\|B_k\|} \le \|\boldsymbol{d}_k\|$$

por tanto, si la sucesión $\{\boldsymbol{d}_k\}$ converge al vector cero, entonces la sucesión $\{\|\Phi_\lambda(\boldsymbol{x}_k)\|\}$ converge a cero ya que, por el **Lema 3.1**, $\|B_k\|$ es acotada para todo $k$, en consecuencia, la sucesión $\{\Phi_\lambda(\boldsymbol{x}_k)\}$ converge al vector cero, luego $\nabla\Psi_\lambda(\boldsymbol{x}_k)$ también converge al vector

cero y dado que la función $\Phi_\lambda$ es continua, entonces

$$\nabla\Psi_\lambda(\boldsymbol{x}^*) = \mathbf{0},$$

lo que contradice nuestro supuesto. Es decir, existe una constante positiva $m$ tal que $m \leq \|\boldsymbol{d}_k\|$.

Por otra parte, si $\|\boldsymbol{d}_k\|$ no es acotada superiormente, teniendo en cuenta que $\overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)$ es acotado y que $p \geq 2$, entonces

$$\lim_{k\to\infty} \frac{\|\overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)\| \cos\theta}{\|\boldsymbol{d}_k\|^{p-1}} = 0$$

donde $\theta$ es el ángulo formado por los vectores $\overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)$ y $\boldsymbol{d}_k$. Equivalentemente,

$$\lim_{k\to\infty} \frac{\overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)^T \boldsymbol{d}_k}{\|\boldsymbol{d}_k\|^{p}} = 0$$

lo que contradice el hecho que

$$\overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)^T \boldsymbol{d}_k \leq -\rho\|\boldsymbol{d}_k\|^p. \tag{3.14}$$

Esto significa que existe una constante positiva $M$ tal que $\|\boldsymbol{d}_k\| \leq M$.

Ahora, dado que la sucesión $\{\boldsymbol{x}_k\}$ converge a $\boldsymbol{x}^*$ y que la iteración siguiente en el **Algoritmo 3.1** es generada por $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + t_k\boldsymbol{d}_k$ entonces

$$t_k\boldsymbol{d}_k \to \mathbf{0}. \tag{3.15}$$

Por (3.13), podemos asumir que si $k \to \infty$, $\boldsymbol{d}_k \to \overline{\boldsymbol{d}} \neq \mathbf{0}$, luego $t_k = \mu^{l_k} \to 0$. Además, de (3.2)

$$\frac{\Psi_\lambda(\boldsymbol{x}_k + \mu^{l_k-1}\boldsymbol{d}_k) - \Psi_\lambda(\boldsymbol{x}_k)}{\mu^{l_k-1}} > \sigma\overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)^T\boldsymbol{d}_k \tag{3.16}$$

Así, de (3.16), si $k \to \infty$

$$\nabla\Psi_\lambda(\boldsymbol{x}^*)^T\,\overline{\boldsymbol{d}} \geq \sigma\,\overline{\nabla\Psi_\lambda}(\boldsymbol{x}^*)^T\,\overline{\boldsymbol{d}}$$

luego, por el **Lema (3.2)** podemos inferir que

$$\sigma \geq 1,$$

lo que contradice el hecho que $\sigma \in \left(0, \frac{1}{2}\right)$. Así, de (3.15) tenemos que, necesariamente

la sucesión $\{\boldsymbol{d}_k\}$ converge al vector cero y por lo tanto, la sucesión $\{\|\boldsymbol{d}_k\|\}$ converge a cero, lo cual contradice el hecho que $\nabla\Psi_\lambda(\boldsymbol{x}^*) \neq \mathbf{0}$. En conclusión, $\nabla\Psi_\lambda(\boldsymbol{x}^*) = \mathbf{0}$. ♦

En el **Teorema 3.2**, hacemos referencia a un término aparentemente contradictorio, por ello y para evitar ambigüedades realizamos la siguiente definición.

**Definición 3.1.** *Sea $\Omega$ el conjunto de puntos de acumulación de la sucesión $\{\boldsymbol{x}_k\}$. Decimos que $\boldsymbol{x}^* \in \Omega$ es un punto de acumulación aislado, si $\boldsymbol{x}^*$ es un punto aislado en $\Omega$, es decir, si existe $\delta > 0$ tal que*

$$\mathcal{B}(\boldsymbol{x}^*; \delta) \cap \Omega = \{\boldsymbol{x}^*\}.$$

**Teorema 3.2.** *Si $\boldsymbol{x}^*$ es un punto de acumulación aislado de la sucesión generada por el* **Algoritmo 3.1***, entonces la sucesión converge a $\boldsymbol{x}^*$.*

*Demostración.* Sean $\{\boldsymbol{x}_k\}$ la sucesión generada por el **Algoritmo 3.1** y $\boldsymbol{x}^*$ un punto de acumulación aislado de esta sucesión. Como $\Psi(\boldsymbol{x}^*) = 0$ y $\Psi$ es una función convexa, entonces $\boldsymbol{x}^*$ es un minimizador global de $\Psi$.

Sea $\Omega$ el conjunto de puntos de acumulación de $\{\boldsymbol{x}_k\}$, entonces $\Omega \neq \emptyset$, pues $\boldsymbol{x}^* \in \Omega$. Definamos

$$\delta = \begin{cases} dist(\boldsymbol{x}^*;\, \Omega\backslash\{\boldsymbol{x}^*\}) & \text{si} \quad \Omega\backslash\{\boldsymbol{x}^*\} \neq \emptyset \\ 1 & \text{si} \quad \Omega = \{\boldsymbol{x}^*\} \end{cases}$$

donde

$$dist(a,\, A) = \inf_{x \in A} \|a - \boldsymbol{x}\|$$

indica la distancia del punto $a$ al conjunto $A$.

Teniendo en cuenta que $\boldsymbol{x}^*$ es único, localmente, entonces $\delta > 0$. Si

$$\Omega_1 = \left\{ \boldsymbol{y} \in \mathbb{R}^n : \; dist(\boldsymbol{y};\, \Omega) \leq \frac{\delta}{4} \right\},$$

entonces existe $\overline{k}$ tal que $\boldsymbol{x}_k \in \Omega_1$ para todo $k \geq \overline{k}$.

Ahora, si

$$\Lambda = \left\{ k \in \mathbb{N} : \; \|\boldsymbol{x}_k - \boldsymbol{x}^*\| \leq \frac{\delta}{4} \right\},$$

entonces $\{\boldsymbol{x}_k\}_\Lambda \subset \overline{\mathcal{B}}\left(x^*;\, \frac{\delta}{4}\right)$. Como $\boldsymbol{x}^*$ es el único punto de acumulación de $\{\boldsymbol{x}_k\}$ en $\overline{\mathcal{B}}\left(\boldsymbol{x}^*;\, \frac{\delta}{4}\right)$, podemos concluir que $\{\boldsymbol{x}_k\}_\Lambda$ converge a $\boldsymbol{x}^*$ y como $\Psi(\boldsymbol{x}^*) = 0$, entonces $\{\|\Psi(\boldsymbol{x}_k)\|\}_\Lambda$ converge a cero, lo que a su vez, por (3.14) y (3.10) implica que la sucesión $\{\boldsymbol{d}_k\}$ converge al vector cero. Así, existe $\widetilde{k}$ tal que si $k \in \Lambda$ y $k \geq \widetilde{k} \geq \overline{k}$ entonces

$$\|\boldsymbol{d}_k\| \leq \frac{\delta}{4}.$$

Sea $\widehat{k} \in \Lambda$ tal que $\widehat{k} \geq \widetilde{k}$, observemos que

$$\boldsymbol{x}_{\widehat{k}+1} = \boldsymbol{x}_{\widehat{k}} + t_{\widehat{k}}\boldsymbol{d}_{\widehat{k}} \tag{3.17}$$

con $t_{\widehat{k}} \in (0,\, 1]$. Luego

$$\|\boldsymbol{x}_{\widehat{k}+1} - \boldsymbol{x}_{\widehat{k}}\| \leq \|\boldsymbol{d}_{\widehat{k}}\| \leq \frac{\delta}{4},$$

por lo tanto,

$$\begin{aligned} dist(\boldsymbol{x}^*;\, \Omega\backslash\{\boldsymbol{x}^*\}) &\leq dist(\boldsymbol{x}_{\widehat{k}+1};\, \Omega\backslash\{\boldsymbol{x}^*\}) + \|\boldsymbol{x}^* - \boldsymbol{x}_{\widehat{k}+1}\| \\ &\leq dist(\boldsymbol{x}_{\widehat{k}+1};\, \Omega\backslash\{\boldsymbol{x}^*\}) + \|\boldsymbol{x}^* - \boldsymbol{x}_{\widehat{k}}\| + \|\boldsymbol{x}_{\widehat{k}} - \boldsymbol{x}_{\widehat{k}+1}\| \\ &\leq dist(\boldsymbol{x}_{\widehat{k}+1};\, \Omega\backslash\{\boldsymbol{x}^*\}) + \frac{\delta}{4} + \frac{\delta}{4} \end{aligned}$$

es decir,

$$dist(\boldsymbol{x}_{\widehat{k}+1};\, \Omega\backslash\{\boldsymbol{x}^*\}) \geq dist(\boldsymbol{x}^*;\, \Omega\backslash\{\boldsymbol{x}^*\}) - \frac{\delta}{2} \geq \delta - \frac{\delta}{2} = \frac{\delta}{2}$$

en consecuencia, $\boldsymbol{x}_{\widehat{k}+1} \notin \Omega_1\backslash\overline{\mathcal{B}}\left(\boldsymbol{x}^*;\, \frac{\delta}{4}\right)$ y como $\boldsymbol{x}_{\widehat{k}+1} \in \Omega_1$ entonces $\boldsymbol{x}_{\widehat{k}+1} \in \overline{\mathcal{B}}\left(x^*;\, \frac{\delta}{4}\right)$, es decir, $\widehat{k}+1 \in \Lambda$ y dado que $\widehat{k}+1 > \widetilde{k}$, entonces por inducción, $\widehat{k} \in \Lambda$ para todo $\widehat{k} > \widetilde{k}$, en consecuencia, $\boldsymbol{x}_k \in \overline{\mathcal{B}}\left(\boldsymbol{x}^*;\, \frac{\delta}{4}\right)$ para todo $k \geq \widetilde{k}$, así, $\{\boldsymbol{x}_k\}$ converge a $\boldsymbol{x}^*$. ♦

El siguiente resultado muestra que existe un entero positivo $\overline{k}$ tal que para todo $k > \overline{k}$, el **Algoritmo 3.1** tiene un comportamiento *local* idéntico al del **Algoritmo 4.1** propuesto en [1] lo que nos permitirá garantizar una tasa de convergencia similar a la obtenida para dicho algoritmo. Cabe mencionar que si $\boldsymbol{x}^*$ es *R-regular*, la hipótesis **H3** en [1] es inmediatamente satisfecha (teorema 2.3).

**Teorema 3.3.** *Suponga que* $\boldsymbol{x}^*$ *es un punto de acumulación de la sucesión* $\{\boldsymbol{x}_k\}$ *generada por el* **Algoritmo 3.1** *tal que* $\boldsymbol{x}^*$ *es una solución R-regular del* PCNL*, entonces la sucesión* $\{\boldsymbol{x}_k\}$ *converge a* $\boldsymbol{x}^*$*, la dirección de descenso es eventualmente dada por la solución del sistema de ecuaciones* $B_k\boldsymbol{d}_k = -\Phi_\lambda(\boldsymbol{x}_k)$ *y el tamaño del paso* $t_k = 1$ *es aceptado para todo* $k$ *suficientemente grande.*

*Demostración.* Si $\boldsymbol{x}^*$ es una solución *R-regular* del PCNL, $x^*$ es también una solución *BD-regular* (**Corolario 2.1**), luego por la **Proposición 3** en [28], $\boldsymbol{x}^*$ es un punto de acumulación aislado de la sucesión $\{\boldsymbol{x}_k\}$ generada por el **Algoritmo 3.1** y en consecuencia, del **Teorema 3.2**, se sigue que la sucesión $\{\boldsymbol{x}_k\}$ converge a $\boldsymbol{x}^*$.

Por otra parte, por el **Teorema 2.2**, para algún $k$ suficientemente grande, la matriz $B_k^{-1}$ existe y por lo tanto, el sistema de ecuaciones

$$B_k\boldsymbol{d}_k = -\Phi_\lambda(\boldsymbol{x}_k)$$

tiene solución. Así, si $\boldsymbol{d}_k$ es una solución de dicho sistema,

$$\boldsymbol{d}_k = -B_k^{-1}\Phi_\lambda(\boldsymbol{x}_k), \tag{3.18}$$

luego, para cualquier norma vectorial y su respectiva norma matricial inducida

$$\|\boldsymbol{d}_k\| \leq \|\Phi_\lambda(\boldsymbol{x}_k)\|\|B_k^{-1}\|. \tag{3.19}$$

De (3.7), (3.18) y (3.19)

$$\begin{aligned}
\overline{\nabla\Psi_\lambda}(\boldsymbol{x}_k)^T\boldsymbol{d}_k &= \Phi_\lambda(\boldsymbol{x}_k)^T B_k\boldsymbol{d}_k \\
&= -\Phi_\lambda(\boldsymbol{x}_k)^T B_k B_k^{-1}\Phi_\lambda(\boldsymbol{x}_k) \\
&= -\|\Phi_\lambda(\boldsymbol{x}_k)\|^2 \\
&\leq -\frac{\|d_k\|^2}{\|B_k^{-1}\|^2}
\end{aligned}$$

luego, podemos tomar la constante $\rho = \eta^2$, donde $\eta$ es la cota superior de $\|B_k\|$ dada en el **Teorema 2.2**. Es decir, teniendo en cuenta que la sucesión $\{\|\boldsymbol{d}_k\|\}$ converge a cero entonces

$$\overline{\nabla\Psi_\lambda(\boldsymbol{x}_k)\,\boldsymbol{d}_k} \leq -\rho\|\boldsymbol{d}_k\|^p$$

para todo $p \geq 2$. Esto prueba que para un $k$ suficientemente grande, la dirección de descenso es siempre dada por (3.13).

Ahora, por el **Lema 3.2**, el **Teorema 2.4** y teniendo en cuenta que $\Psi_\lambda$ es una función

de clase $SC^1$, se tiene que para cualquier $\sigma \in \left(0, \frac{1}{2}\right)$ existe $\overline{k}$ tal que para todo $k > \overline{k}$

$$\Psi_\lambda(\boldsymbol{x}_k + \boldsymbol{d}_k) \leq \Psi_\lambda(\boldsymbol{x}_k) + \sigma B_k^T \Phi_\lambda(\boldsymbol{x}_k)\boldsymbol{d}_k$$

es decir, el tamaño del paso $t_k = 1$ es aceptado para $k$ suficientemente grande. Esto completa la demostración. ⧫

# CAPÍTULO 4

# PRUEBAS NUMÉRICAS

En la primera parte de este capítulo, analizamos numéricamente el algoritmo propuesto en el capítulo anterior (**Algoritmo 3.1**). Para ello, comparamos su desempeño con el algoritmo tipo *Newton generalizado* presentado en [21] que llamaremos **Algoritmo 4.1** el cual, para mayor claridad en la lectura de este documento, incluimos a continuación.

**Algoritmo 4.1.** ***Newton generalizado para*** $\Phi_\lambda(\boldsymbol{x}) = 0$.

**P.0** *Inicialización:*

*Escoja* $\lambda \in (0,4),\ \mathbf{x}_0 \in \mathbb{R}^n,\ \rho > 0,\ \mu \in (0,1),\ \sigma \in (0,\frac{1}{2}),\ p > 2,\ \varepsilon \geq 0,\ N > 0$ *y haga* $k := 0$.

**P.1** *Criterio de parada:*

*Si* $\|\nabla \Psi_\lambda(\mathbf{x}_k)\| \leq \varepsilon$ *o* $k > N$ *pare.*

**P.2** *Búsqueda direccional:*

*Calcule* $H_k \in \partial\Phi_\lambda(\mathbf{x}_k)$ *y halle* $\mathbf{d}_k \in \mathbb{R}^n$ *tal que*

$$H_k\mathbf{d}_k = -\Phi_\lambda(\mathbf{x}_k).$$

*Si este sistema no tiene solución o si* $\nabla\Psi_\lambda(\mathbf{x}_k)^T\mathbf{d}_k > -\rho\|\mathbf{d}_k\|^p$ *haga* $\mathbf{d}_k := -\nabla\Psi_\lambda(\mathbf{x}_k)$.

**P.3** *Búsqueda lineal:*

*Haga* $t_k := \max\{\mu^l \,|\, l = 0, 1, 2, ...\}$ *tal que*

$$\Psi_\lambda(\mathbf{x}_k + t_k\mathbf{d}_k) \leq \Psi_\lambda(\mathbf{x}_k) + \sigma t_k \nabla\Psi_\lambda(\mathbf{x}_k)^T\mathbf{d}_k$$

**P.4** *Actualización:*

*Haga* $\mathbf{x}_{k+1} = \mathbf{x}_k + t_k\mathbf{d}_k$, $k = k + 1$ *y vuelva a* ***P.1***.

Como se puede ver, este algoritmo usa en cada iteración las matrices $H_k \in \partial\Phi_\lambda(\boldsymbol{x})$. Es básicamente en esta etapa donde radica su diferencia con el **Algoritmo 3.1** que proponemos en el capítulo anterior, ya que en lugar de calcular dichas matrices, calculamos aproximaciones a ellas como la dada en (2.4). De (2.7) y por lo mencionado anteriormente, es imposible calcular el gradiente de la *función de mérito* $\Psi_\lambda$ en cada iteración con el **Algoritmo 3.1**, motivo por el cual precisamente, proponemos la aproximación (3.7) a dicho gradiente para ser usada en nuestro algoritmo.

Motivados por el buen desempeño numérico de la *búsqueda lineal no monótona* [20] en algoritmos de complementariedad no lineal y problemas relacionados [21][30], en la segunda parte de este capítulo, presentamos el **Algoritmo 4.2** en el cual, modificamos el paso **P.3** del **Algoritmo 3.1** e introducimos una *búsqueda lineal no monótona*. Analizamos numéricamente el **Algoritmo 4.2** comparando su desempeño con el respectivo **Algoritmo 4.1** tras hacerle la misma modificación y además realizamos pruebas numéricas con el **Algoritmo 4.2** al variar los parámetros propios del tipo de búsqueda utilizada en el mismo.

Finalmente, teniendo en cuenta el buen desempeño *local* que han mostrado los algoritmos de complementariedad que usan la función *mínimo* y el buen desempeño *global* que han mostrado los algoritmos de complementariedad que usan la función de *Fischer* [31] [21], en la tercera parte de este capítulo realizamos una exploración numérica preliminar del comportamiento del **Algoritmo 3.1** al variar el parámetro inicial $\lambda$ que define la función

de complementariedad dada en (1.3).

Para las pruebas numéricas, consideramos cuatro *problemas de complementariedad no lineal* asociados respectivamente a las siguientes funciones:

1. **Kojima Shindo** (Koj-Shi): $F : \mathbb{R}^4 \to \mathbb{R}^4$ definido por

$$F(\boldsymbol{x}) = \begin{pmatrix} 3x_1^2 + 2x_1x_2 + 2x_2^2 + x_3 + 3x_4 - 6 \\ 2x_1^2 + x_2^2 + x_1 + 10x_3 + 2x_4 - 2 \\ 3x_1^2 + x_1x_2 + 2x_2^2 + 2x_3 + 9x_4 - 9 \\ x_1^2 + 3x_2^2 + 2x_3 + 3x_4 - 3 \end{pmatrix}.$$

2. **Kojima Josephy** (Koj-Jo): $F : \mathbb{R}^4 \to \mathbb{R}^4$ definido por

$$F(\boldsymbol{x}) = \begin{pmatrix} 3x_1^2 + 2x_1x_2 + 2x_2^2 + x_3 + 3x_4 - 6 \\ 2x_1^2 + x_2^2 + x_1 + 3x_3 + 2x_4 - 2 \\ 3x_1^2 + x_1x_2 + 2x_2^2 + 2x_3 + 3x_4 - 9 \\ x_1^2 + 3x_2^2 + 2x_3 + 3x_4 - 3 \end{pmatrix}.$$

3. **Mathiesen Modificado** (Mathiesen): $F : \mathbb{R}^4 \to \mathbb{R}^4$ definido por

$$F(\boldsymbol{x}) = \begin{pmatrix} -x_2 + x_3 + x_4 \\ x_1 - \dfrac{4.5x_3 + 2.7x_4}{x_2 + 1} \\ 5 - x_1 - \dfrac{0.5x_3 + 0.3x_4}{x_3 + 1} \\ 3 - x_1 \end{pmatrix}.$$

   Cabe mencionar que este problema tiene infinitas soluciones de la forma

$$\boldsymbol{x}^* = (a\ 0\ 0\ 0\ ),$$

   con $a \in [0,\ 3]$.

4. **Billups** (Billups): $F : \mathbb{R} \to \mathbb{R}$ definido por

$$F(x) = (x-1)^2 - 1.1.$$

Estos problemas, tomados de [28] y [3], son considerados problemas difíciles en cuanto a su convergencia y por ello, de uso obligado para probar algoritmos de complementariedad no lineal. Cabe destacar que los problemas de *Kojima Shindo* (aplicación en Economía al problema de *equilibrio económico* [22]), *Kojima Josephy* y *Mathiesen Modificado* (aplicación al problema de *equilibrio económico walrasiano* [26]) son analizados localmente en [1]. Además, es de subrayar que el *problema de Billups* fue construido por *Billups* [3] con el objetivo de hacer que los métodos usados para resolverlo fallen [21], es decir, es un problema de alta exigencia en cuanto a su convergencia.

Para escribir los códigos de los algoritmos y de las *funciones de prueba*, utilizamos el software MATLAB® y realizamos las pruebas numéricas en un computador con procesador Intel(R) Atom(TM) de 1.67 GHz.

En la implementación de los **Algoritmos 3.1 y 4.1**, para cada problema, usamos como puntos iniciales los utilizados en [21] para sus pruebas. Además, usamos los siguientes valores para los parámetros requeridos

$$\rho = 10^{-8},\ \sigma = 10^{-4},\ p = 2.1,\ \varepsilon = 10^{-12},\ N = 200,\ \mu = 0.5.$$

De igual forma, teniendo en cuenta los resultados presentados en [21], el buen desempeño que ha mostrado la función de *Fischer* cuando se ejecutan algoritmos iterativos con puntos iniciales alejados de la solución y la eficiencia local de la función *mínimo* para este tipo de algoritmos, implementamos una escogencia dinámica para el parámetro $\lambda$, el cual se actualiza en cada iteración de la siguiente manera:

1. Escoja $\lambda = 2$
2. Si $\Psi_\lambda(\boldsymbol{x}_k) \leq \gamma_1$
$$\lambda := \Psi_\lambda(\boldsymbol{x}_k),$$
si no,
$$\lambda := \text{mín}\{c_1\Psi(\boldsymbol{x}_k),\ \lambda\}$$
3. Si $\Psi(\boldsymbol{x}_k) \leq \gamma_2$,
$$\lambda := \text{mín}\{c_2,\ \lambda\}$$

donde

$$\gamma_1 = 10^{-2},\ \gamma_2 = 10^{-4},\ c_1 = 10,\ c_2 = 10^{-8}.$$

Esta elección del parámetro $\lambda$ permite que los algoritmos realicen sus primeras iteraciones usando la *función de complementariedad de Fischer* [19][18], la cual ha mostrado un

mejor comportamiento cuando el punto inicial en los algoritmos está alejado de la solución y culmine su ciclo con funciones de complementariedad que tienden a ser múltiplos de la *función mínimo* [34][29], la cual ha mostrado un buen desempeño local [34].

Presentamos los resultados de las pruebas numéricas en tablas cuyas columnas contienen la siguiente información:

*Problema*: Nombre del problema.
$t$: Tiempo (en segundos) de ejecución de los **Algoritmos**.
$\boldsymbol{x}_0$: Punto inicial.
$k$: Número de iteraciones.
$\boldsymbol{x}^*$: Solución.

De igual manera, incluimos una columna en la que hacemos alusión al algoritmo y la actualización usada:

NG: **Algoritmo 4.1**
BB: **Algoritmo 3.1** y actualización (3.4) de Broyden "bueno".
BM: **Algoritmo 3.1** y actualización (3.5) de Broyden "malo".
SN: **Algoritmo 3.1** y actualización (3.6) de Shubert.
-: indica que hubo divergencia.

## 4.1. Pruebas con los Algoritmos 3.1 y 4.1

En la siguiente tabla, encontramos los resultados obtenidos al resolver los problemas de complementariedad anteriormente expuestos con los **Algoritmos 4.1** y **3.1**, este último con las actualizaciones dadas en (3.4), (3.5) y (3.6), respectivamente.

| *Problema* | *Algoritmo* | $\boldsymbol{x}_0$ | $t\,(seg)$ | $k$ | $\boldsymbol{x}^*$ |
|---|---|---|---|---|---|
| Koj-Jo | NG | $(0\ 0\ 0\ 0)^T$ | 0.152 | 21 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | $(0\ 0\ 0\ 0)^T$ | 0.141 | 21 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BM | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-Jo | SN | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-Jo | NG | $(1\ 1\ 1\ 1)^T$ | 0.060 | 8 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | $(1\ 1\ 1\ 1)^T$ | 0.086 | 13 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BM | $(1\ 1\ 1\ 1)^T$ | 0.074 | 12 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | SN | $(1\ 1\ 1\ 1)^T$ | - | - | - |

| *Problema* | *Algoritmo* | $\boldsymbol{x}_0$ | $t\,(seg)$ | $k$ | $\boldsymbol{x}^*$ |
|---|---|---|---|---|---|
| Koj-Jo | NG | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Jo | BB | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Jo | BM | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Jo | SN | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Jo | NG | $(1\ 0\ 1\ 0)^T$ | 0.038 | 6 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | $(1\ 0\ 1\ 0)^T$ | 0.051 | 8 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BM | $(1\ 0\ 1\ 0)^T$ | 0.056 | 8 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | SN | $(1\ 0\ 1\ 0)^T$ | - | - | - |
| Koj-Jo | NG | $(1\ 0\ 0\ 0)^T$ | 0.082 | 8 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | $(1\ 0\ 0\ 0)^T$ | 0.067 | 10 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BM | $(1\ 0\ 0\ 0)^T$ | 0.087 | 12 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | SN | $(1\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-Jo | NG | $(0\ 1\ 1\ 0)^T$ | 0.066 | 9 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | $(0\ 1\ 1\ 0)^T$ | 0.089 | 14 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BM | $(0\ 1\ 1\ 0)^T$ | 0.229 | 17 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | SN | $(0\ 1\ 1\ 0)^T$ | - | - | - |
| Koj-Shi | NG | $(0\ 0\ 0\ 0)^T$ | 0.212 | 15 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BB | $(0\ 0\ 0\ 0)^T$ | 0.131 | 21 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BM | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-Shi | SN | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-Shi | GN | $(1\ 1\ 1\ 1)^T$ | 0.067 | 10 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BB | $(1\ 1\ 1\ 1)^T$ | 0.141 | 16 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BM | $(1\ 1\ 1\ 1)^T$ | 0.102 | 16 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | SN | $(1\ 1\ 1\ 1)^T$ | - | - | - |
| Koj-Shi | GN | $(100\ 100\ 100\ 100)^T$ | 0.076 | 12 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | BB | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Shi | BM | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Shi | SN | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Shi | GN | $(1\ 0\ 1\ 0)^T$ | 0.039 | 6 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BB | $(1\ 0\ 1\ 0)^T$ | 0.046 | 7 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BM | $(1\ 0\ 1\ 0)^T$ | 0.044 | 7 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | GN | $(1\ 0\ 0\ 0)^T$ | 0.051 | 8 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | BB | $(1\ 0\ 0\ 0)^T$ | 0.060 | 9 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | BM | $(1\ 0\ 0\ 0)^T$ | 0.057 | 9 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | SN | $(1\ 0\ 0\ 0)^T$ | - | - | - |

| *Problema* | *Algoritmo* | $\boldsymbol{x}_0$ | $t\,(seg)$ | $k$ | $\boldsymbol{x}^*$ |
|---|---|---|---|---|---|
| Koj-Shi | GN | $(0\ 1\ 1\ 0)^T$ | 0.070 | 10 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BB | $(0\ 1\ 1\ 0)^T$ | 0.093 | 15 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BM | $(0\ 1\ 1\ 0)^T$ | 0.125 | 15 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | SN | $(0\ 1\ 1\ 0)^T$ | - | - | - |
| Mathiesen | GN | $(1\ 1\ 1\ 1)^T$ | 0.186 | 11 | $(0.0302\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | $(1\ 1\ 1\ 1)^T$ | 0.178 | 11 | $(0.0302\ 0\ 0\ 0)^T$ |
| Mathiesen | BM | $(1\ 1\ 1\ 1)^T$ | 0.091 | 11 | $(0.0302\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | $(1\ 1\ 1\ 1)^T$ | 0.070 | 5 | $(1.497\ 0\ 0\ 0)^T$ |
| Mathiesen | GN | $(100\ 100\ 100\ 100)^T$ | 0.102 | 13 | $(2.999\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | $(100\ 100\ 100\ 100)^T$ | 0.132 | 7 | $(3\ 0\ 0\ 0)^T$ |
| Mathiesen | BM | $(100\ 100\ 100\ 100)^T$ | 0.054 | 7 | $(3\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | $(100\ 100\ 100\ 100)^T$ | 0.094 | 9 | $(2.320\ 0\ 0\ 0)^T$ |
| Mathiesen | GN | $(1\ 0\ 1\ 0)^T$ | 0.091 | 4 | $(0\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | $(1\ 0\ 1\ 0)^T$ | 0.046 | 6 | $(0\ 0\ 0\ 0)^T$ |
| Mathiesen | BM | $(1\ 0\ 1\ 0)^T$ | 0.043 | 6 | $(0\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | $(1\ 0\ 1\ 0)^T$ | 0.071 | 6 | $(0.018\ 0\ 0\ 0)^T$ |
| Mathiesen | GN | $(0\ 1\ 1\ 0)^T$ | 0.037 | 5 | $(0.752\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | $(0\ 1\ 1\ 0)^T$ | 0.036 | 5 | $(0.608\ 0\ 0\ 0)^T$ |
| Mathiesen | BM | $(0\ 1\ 1\ 0)^T$ | 0.036 | 5 | $(0.647\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | $(0\ 1\ 1\ 0)^T$ | 0.037 | 5 | $(0.778\ 0\ 0\ 0)^T$ |
| Billups | NG | 0 | - | - | - |
| Billups | BB | 0 | 2.843 | 16 | 2.0488 |
| Billups | BM | 0 | 2.827 | 16 | 2.0488 |
| Billups | SN | 0 | - | - | - |

**Tabla** 4.1: Resultados numéricos para los **Algoritmos 3.1 y 4.1**.

Como se puede ver, estos resultados muestran la importancia de tener en cuenta el tiempo de ejecución de los algoritmos ya que, evaluar solo el número de iteraciones realizadas por cada uno de ellos no es un buen comparativo pues, si bien en general el número de iteraciones realizadas por el **Algoritmo 3.1** es mayor que las realizadas por el Algoritmo 4.1, el número de operaciones que realiza en cada iteración el **Algoritmo 3.1** es en general menor que el número de operaciones que realiza en cada iteración el **Algoritmo 4.1**.

Prueba de lo mencionado anteriormente son los resultados obtenidos para el problema de *Kojima-Shindo* con punto inicial $(0\ 0\ 0\ 0)$, ya que el Algoritmo **4.1** resuelve el problema

en 15 iteraciones, mientras que el **Algoritmo 3.1** lo resuelve en 21 iteraciones, sin embargo, si comparamos los tiempos de ejecución vemos que el algoritmo propuesto en esta investigación fue un poco más rápido que el **Algoritmo 4.1**. Esto fue algo que sucedió en varias ocasiones en las pruebas pero que sobresalió con el problema de *Mathiesen* pues en todas las ocasiones el **Algoritmo 3.1** fue más rápido que el **Algoritmo 4.1**.

Por otra parte, los resultados muestran el buen desempeño que tiene el **Algoritmo 3.1** cuando se utiliza la actualización de *Broyden "buena"* pues convergió para un 93 % de los problemas y puntos iniciales, sin contar los resultados obtenidos al resolver el problema de *Billups*, pues en este caso, el **Algoritmo 4.1** divergió, entre tanto, el **Algoritmo 3.1** convergió.

## 4.2. Búsqueda lineal no monótona

Como lo mencionamos en la primera parte de este capítulo, motivados en el buen comportamiento de los algoritmos de complementariedad que usan búsqueda lineal no monótona, en esta parte, analizamos el efecto, en cuanto a convergencia y velocidad de convergencia, de introducir una búsqueda lineal no monótona en los **Algoritmos 3.1 y 4.1**. Para mayor claridad en la lectura, mostramos a continuación la estructura del nuevo algoritmo.

**Algoritmo 4.2.** ***Tipo cuasi Newton globalizado con búsqueda no monótona***

***P.0*** *Inicialización*

*Escoja* $\lambda \in (0,4)$, $\boldsymbol{x}_0 \in \mathbb{R}^n$, $\rho > 0$, $\mu \in (0,1)$, $\sigma \in (0, \frac{1}{2})$, $p > 2$, $\varepsilon \geq 0$, $N > 0$ $s > 0$, $M > 0$ *y haga* $k = 0$ $m_k = 0$ *y* $A_0 = F'(x_0)$

***P.1*** *Criterio de parada*

*Si* $\|B_k^T \Phi_\lambda(\boldsymbol{x}_k)\| \leq \varepsilon$ *o* $k > N$ *pare.*

***P.2*** *Búsqueda direccional*

*Calcule* $B_k$ *como en* (2.4) *y halle* $\boldsymbol{d}_k \in \mathbb{R}^n$ *tal que*

$$B_k \boldsymbol{d}_k = -\Phi_\lambda(\boldsymbol{x}_k).$$

*Si el sistema no tiene solución o si* $\Phi_\lambda(\boldsymbol{x}_k)^T B_k d_k > -\rho \|\boldsymbol{d}_k\|^p$, *haga* $\boldsymbol{d}_k = -B_k^T \Phi_\lambda(\boldsymbol{x}_k)$.

***P.3*** *Búsqueda lineal no monótona*

*Si* $k \leq s$ *o* $\boldsymbol{d}_k = -B_k^T \Phi_\lambda(\boldsymbol{x}_k)$,

*haga* $m_k = 0$

*si no*

*haga* $m_k = \min\{m_{k-1} + 1,\ M\}$

*Calcule*

$$R_k = \max_{k-m_k \leq j \leq k} \Psi_\lambda(\boldsymbol{x}_j)$$

*Haga* $t_k = \max\{\mu^l \mid l = 0, 1, 2, ...\}$ *tal que*

$$\Psi_\lambda(\boldsymbol{x}_k + t_k \boldsymbol{d}_k) \leq R_k + \sigma t_k B_k^T \Phi_\lambda(\boldsymbol{x}_k)^T \boldsymbol{d}_k.$$

***P.4*** *Actualización*

*Haga* $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + t_k \boldsymbol{d}_k$, $k = k + 1$ *, actualize* $A_k$ *como en* (3.4), (3.5) *o* (3.6) *y vuelva a* ***P.1***.

Cabe destacar que con la *búsqueda lineal no monótona* usada en el **Algoritmo 4.2**, es posible comparar el valor que toma la función a minimizar en la iteración actual con el máximo valor que toma la función en alguna iteraciones anteriores con el fin de seleccionar el tamaño del paso, es decir, es posible que la función no decrezca entre dos iteraciones consecutivas. El número de iteraciones anteriores con la cuales se puede comparar el valor de la función en la iteración actual depende del parámetro $M$, por lo tanto, si $M = 0$ tendremos una *búsqueda lineal monótona*. De igual manera, el parámetro $s$ indica el número de iteraciones iniciales con las que el **Algoritmo 4.2** realizará una *búsqueda lineal monótona*.

Siguiendo la propuesta hecha en [21], decidimos comparar el valor que toma la función de prueba en la iteración actual hasta con ocho de las iteraciones inmediatamente anteriores, es decir, tomamos $M = 8$, además, pedimos al **Algoritmo 4.2** que realizara una *búsqueda lineal monótona* sólo en la primera iteración, es decir, tomamos $s = 1$. Los resultados obtenidos se muestran en la siguiente tabla.

| *Problema* | *Algoritmo* | $\boldsymbol{x}_0$ | *t (seg)* | *k* | $x^*$ |
|---|---|---|---|---|---|
| Koj-Jo | NG | $(0\ 0\ 0\ 0)^T$ | 1.57 | 134 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | $(0\ 0\ 0\ 0)^T$ | 0.98 | 76 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BM | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-Jo | SN | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-Jo | NG | $(1\ 1\ 1\ 1)^T$ | 0.096 | 10 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | $(1\ 1\ 1\ 1)^T$ | 0.129 | 13 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BM | $(1\ 1\ 1\ 1)^T$ | 0.122 | 12 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | SN | $(1\ 1\ 1\ 1)^T$ | - | - | - |
| Koj-Jo | NG | $(100\ 100\ 100\ 100)^T$ | 1.53 | 131 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Jo | BM | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Jo | SN | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Jo | NG | $(1\ 0\ 1\ 0)^T$ | 0.052 | 6 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | $(1\ 0\ 1\ 0)^T$ | 0.068 | 8 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BM | $(1\ 0\ 1\ 0)^T$ | 0.086 | 8 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | SN | $(1\ 0\ 1\ 0)^T$ | - | - | - |
| Koj-Jo | NG | $(1\ 0\ 0\ 0)^T$ | 0.082 | 8 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | $(1\ 0\ 0\ 0)^T$ | 0.098 | 10 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BM | $(1\ 0\ 0\ 0)^T$ | 0.110 | 11 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | SN | $(1\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-Jo | NG | $(0\ 1\ 1\ 0)^T$ | 0.072 | 8 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | $(0\ 1\ 1\ 0)^T$ | 0.132 | 13 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BM | $(0\ 1\ 1\ 0)^T$ | 0.154 | 15 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | SN | $(0\ 1\ 1\ 0)^T$ | - | - | - |
| Koj-Shi | NG | $(0\ 0\ 0\ 0)^T$ | 0.161 | 15 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | BB | $(0\ 0\ 0\ 0)^T$ | 0.494 | 44 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | BM | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-Shi | SN | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-Shi | GN | $(1\ 1\ 1\ 1)^T$ | 0.080 | 9 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BB | $(1\ 1\ 1\ 1)^T$ | 0.149 | 14 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BM | $(1\ 1\ 1\ 1)^T$ | - | - | - |
| Koj-Shi | SN | $(1\ 1\ 1\ 1)^T$ | - | - | - |
| Koj-Shi | GN | $(100\ 100\ 100\ 100)^T$ | 0.141 | 13 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | BB | $(100\ 100\ 100\ 100)^T$ | 1.23 | 102 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |

| *Problema* | *Algoritmo* | $\boldsymbol{x}_0$ | *t (seg)* | *k* | $x^*$ |
|---|---|---|---|---|---|
| Koj-Shi | BM | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Shi | SN | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-Shi | GN | $(1\ 0\ 1\ 0)^T$ | 0.054 | 6 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BB | $(1\ 0\ 1\ 0)^T$ | 0.061 | 7 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BM | $(1\ 0\ 1\ 0)^T$ | 0.065 | 7 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | GN | $(1\ 0\ 0\ 0)^T$ | 0.054 | 6 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | BB | $(1\ 0\ 0\ 0)^T$ | 0.080 | 9 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | BM | $(1\ 0\ 0\ 0)^T$ | 0.071 | 8 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | GN | $(0\ 1\ 1\ 0)^T$ | 0.091 | 10 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BB | $(0\ 1\ 1\ 0)^T$ | 0.155 | 15 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BM | $(0\ 1\ 1\ 0)^T$ | 0.173 | 17 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | SN | $(0\ 1\ 1\ 0)^T$ | - | - | - |
| Mathiesen | GN | $(1\ 1\ 1\ 1)^T$ | 0.094 | 11 | $(0.0302\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | $(1\ 1\ 1\ 1)^T$ | 0.125 | 11 | $(0.0302\ 0\ 0\ 0)^T$ |
| Mathiesen | BM | $(1\ 1\ 1\ 1)^T$ | 0.105 | 11 | $(0.0302\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | $(1\ 1\ 1\ 1)^T$ | 0.053 | 5 | $(1.497\ 0\ 0\ 0)^T$ |
| Mathiesen | GN | $(100\ 100\ 100\ 100)^T$ | 0.090 | 9 | $(3\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | $(100\ 100\ 100\ 100)^T$ | 0.077 | 8 | $(2.999\ 0\ 0\ 0)^T$ |
| Mathiesen | BM | $(100\ 100\ 100\ 100)^T$ | 0.073 | 8 | $(3\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | $(100\ 100\ 100\ 100)^T$ | 0.639 | 9 | $(2.320\ 0\ 0\ 0)^T$ |
| Mathiesen | GN | $(1\ 0\ 1\ 0)^T$ | 0.089 | 11 | $(0\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | $(1\ 0\ 1\ 0)^T$ | 0.054 | 6 | $(0\ 0\ 0\ 0)^T$ |
| Mathiesen | BM | $(1\ 0\ 1\ 0)^T$ | 0.054 | 6 | $(0\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | $(1\ 0\ 1\ 0)^T$ | - | - | - |
| Mathiesen | GN | $(0\ 1\ 1\ 0)^T$ | 0.041 | 5 | $(0.752\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | $(0\ 1\ 1\ 0)^T$ | 0.029 | 4 | $(2.087\ 0\ 0\ 0)^T$ |
| Mathiesen | BM | $(0\ 1\ 1\ 0)^T$ | 0.065 | 7 | $(0\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | $(0\ 1\ 1\ 0)^T$ | 0.042 | 5 | $(0.778\ 0\ 0\ 0)^T$ |
| Billups | NG | 0 | - | - | - |
| Billups | BB | 0 | - | - | - |
| Billups | BM | 0 | - | - | - |
| Billups | SN | 0 | - | - | - |

**Tabla** 4.2: Resultados del **Algoritmo 4.2** con búsqueda no monótona.

Observemos que el **Algoritmo 4.2** en general tuvo un desempeño similar al **Algoritmo 3.1** que usa *búsqueda lineal monótona*, sin embargo, hay que destacar que aunque la diferencia en tiempo de ejecución entre un algoritmo y otro es relativamente pequeña, esta siempre es a favor del **Algoritmo 3.1**, algo que no ocurre cuando se implementa el **Algoritmo 4.1** haciendo *búsqueda lineal no monótona* en el paso **P.3**, ya que en este caso la diferencia en tiempo de ejecución es a favor del algoritmo con *búsqueda lineal no monótona*.

Un caso que llama la atención es el problema de *Kojima-Shindo* con punto inicial $(100\ 100\ 100\ 100)^T$, pues el **Algoritmo 3.1** diverge cuando se implementa la actualización de *Broyden "buena"* dada en (3.4), sin embargo, el **Algoritmo 4.2** converge bajo estas mismas condiciones. Por el contrario, el problema de *Billups* no fue resuelto por el **Algoritmo 4.2** mientras que el **Algoritmo 3.1** si lo hizo.

Estos comportamientos nos motivaron a explorar el desempeño del **Algoritmo 4.2**, cuando se varía el valor del parámetro $M$.

A continuación, presentamos algunos resultados relevantes, para los cuales cabe mencionar que tomamos $s = 1$.

| *Problema* | *Algoritmo* | $M$ | $\boldsymbol{x}_0$ | *t (seg)* | $k$ | $\boldsymbol{x}^*$ |
|---|---|---|---|---|---|---|
| Mathiesen | SN | 0 | $(500\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 2 | $(500\ -250\ 500\ -250)^T$ | 0.075 | 8 | $(1.100\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | 5 | $(500\ -250\ 500\ -250)^T$ | 0.068 | 8 | $(1.100\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | 10 | $(500\ -250\ 500\ -250)^T$ | 0.070 | 8 | $(1.100\ 0\ 0\ 0)^T$ |
| Koj-shi | BB | 0 | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-shi | BB | 2 | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-shi | BB | 5 | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Koj-shi | BB | 10 | $(100\ 100\ 100\ 100)^T$ | 1.63 | 121 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-jo | BB | 0 | $(0\ 0\ 0\ )^T$ | 0.145 | 21 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 2 | $(0\ 0\ 0\ )^T$ | - | - | - |
| Koj-jo | BB | 5 | $(0\ 0\ 0\ )^T$ | 0.309 | 32 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 10 | $(0\ 0\ 0\ )^T$ | 1.543 | 117 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 0 | $(0\ 1\ 1\ 0)^T$ | 0.093 | 15 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 2 | $(0\ 1\ 1\ 0)^T$ | 0.152 | 15 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 5 | $(0\ 1\ 1\ 0)^T$ | 0.138 | 15 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 10 | $(0\ 1\ 1\ 0)^T$ | 0.158 | 15 | $(1\ 0\ 3\ 0)^T$ |

**Tabla** 4.3: Resultados numéricos al variar el parámetro $M$.

Esto en primera instancia nos permite, hipotéticamente, deducir que usar la búsqueda lineal no monótona con $5 \leq M \leq 10$ es una buena opción cuando el **Algoritmo 3.1** no converja, sin embargo, en caso que este algoritmo converja, en general lo hará un poco más rápido que si se usa la búsqueda lineal no monótona, así el número de iteraciones realizadas por los dos algoritmos sea el mismo.

Similarmente, estudiamos el comportamiento del **Algoritmo 4.2** al variar el número de iteraciones iniciales para las cuales realiza una búsqueda lineal monótona, es decir, el desempeño del algoritmo al variar el parámetro $s$. En la **Tabla 4.4**, reportamos los resultados más relevantes de nuestras pruebas.

| *Problema* | *Algoritmo* | $s$ | $\boldsymbol{x}_0$ | *t (seg)* | $k$ | $\boldsymbol{x}^*$ |
|---|---|---|---|---|---|---|
| Koj-Jo | BB | 1 | $(0\ 0\ 0\ 0)^T$ | 1.525 | 117 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | 2 | $(0\ 0\ 0\ 0)^T$ | 0.713 | 57 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | 3 | $(0\ 0\ 0\ 0)^T$ | 0.248 | 23 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Jo | BB | 5 | $(0\ 0\ 0\ 0)^T$ | 0..264 | 25 | $(1\ 0\ 3\ 0)^T$ |
| Koj-Shi | BB | 1 | $(0\ 0\ 0\ 0)^T$ | 0.531 | 44 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | BB | 2 | $(0\ 0\ 0\ 0)^T$ | 0.255 | 22 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | BB | 3 | $(0\ 0\ 0\ 0)^T$ | 0.198 | 19 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Koj-Shi | BB | 5 | $(0\ 0\ 0\ 0)^T$ | 0.186 | 19 | $\left(\frac{\sqrt{6}}{2}\ 0\ 0\ \frac{1}{2}\right)^T$ |
| Mathiesen | SN | 1 | $(100\ 100\ 100\ 100)^T$ | - | - | - |
| Mathiesen | SN | 2 | $(100\ 100\ 100\ 100)^T$ | 0.087 | 9 | $(2.322\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | 3 | $(100\ 100\ 100\ 100)^T$ | 0.070 | 9 | $(2.322\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | 5 | $(100\ 100\ 100\ 100)^T$ | 0.065 | 9 | $(2.322\ 0\ 0\ 0)^T$ |

**Tabla** 4.4: Desempeño del **Algoritmo 4.2** con $M = 10$.

Estos resultados nos permiten ratificar las conclusiones realizadas por Grippo, Lampariello y Lucidi en [20], pues los mejores resultados se obtienen para $s$ entre 3 y 5, es decir, el **Algoritmo 4.2** presenta un mejor desempeño si se realizan entre tres y cinco iteraciones iniciales usando una *búsqueda lineal monótona*.

## 4.3. Variación del parámetro $\lambda$

Teniendo en cuenta el buen desempeño local que presentan los algoritmos de complementariedad que hacen uso de la *función mínimo* y el buen desempeño global que presentan los algoritmos de complementariedad que hacen uso de la *función de Fischer*, realizamos una exploración numérica al variar el parámetro inicial $\lambda$ en el **Algoritmo 3.1**. Recordemos que la función de complementariedad dada en (1.3) es idéntica a la *función de Fischer* si su parámetro $\lambda$ es igual a dos y converge a un múltiplo de la función mínimo si $\lambda$ converge a cero, por lo tanto, hemos decidido tomar valores aleatorios para el parámetro inicial $\lambda$ iniciando en uno y haciendo incrementos de 0.1 unidades hasta tres.

En la siguiente tabla mostramos los resultados obtenidos para algunos de los problemas analizados anteriormente.

| *Problema* | *Algoritmo* | $\lambda$ | $\boldsymbol{x}_0$ | *t (seg)* | $k$ | $\boldsymbol{x}^*$ |
|---|---|---|---|---|---|---|
| Koj-jo | BB | 1 | $(0\ 0\ 0\ 0)^T$ | 0.232 | 23 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 1.1 | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-jo | BB | 1.2 | $(0\ 0\ 0\ 0)^T$ | 0.314 | 29 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 1.3 | $(0\ 0\ 0\ 0)^T$ | 0.192 | 26 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 1.4 | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-jo | BB | 1.5 | $(0\ 0\ 0\ 0)^T$ | 0.265 | 32 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 1.6 | $(0\ 0\ 0\ 0)^T$ | 0.150 | 23 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 1.7 | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-jo | BB | 1.8 | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-jo | BB | 1.9 | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Koj-jo | BB | 2 | $(0\ 0\ 0\ 0)^T$ | 0.141 | 21 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 2.1 | $(0\ 0\ 0\ 0)^T$ | 0.140 | 21 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 2.2 | $(0\ 0\ 0\ 0)^T$ | 0.156 | 23 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 2.3 | $(0\ 0\ 0\ 0)^T$ | 0.185 | 22 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 2.4 | $(0\ 0\ 0\ 0)^T$ | 0.129 | 18 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 2.5 | $(0\ 0\ 0\ 0)^T$ | 0.139 | 21 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 2.6 | $(0\ 0\ 0\ 0)^T$ | 0.152 | 21 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 2.7 | $(0\ 0\ 0\ 0)^T$ | 0.137 | 19 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 2.8 | $(0\ 0\ 0\ 0)^T$ | 0.198 | 29 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 2.9 | $(0\ 0\ 0\ 0)^T$ | 0.162 | 24 | $(1\ 0\ 3\ 0)^T$ |
| Koj-jo | BB | 3 | $(0\ 0\ 0\ 0)^T$ | - | - | - |
| Mathiesen | SN | 1 | $(550\ -250\ 500\ -250)^T$ | - | - | - |

| *Problema* | *Algoritmo* | $\lambda$ | $\boldsymbol{x}_0$ | *t (seg)* | $k$ | $\boldsymbol{x}^*$ |
|---|---|---|---|---|---|---|
| Mathiesen | SN | 1.1 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 1.2 | $(550\ -250\ 500\ -250)^T$ | 0.088 | 8 | $(2.6727\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | 1.3 | $(550\ -250\ 500\ -250)^T$ | 0.081 | 8 | $(2.2882\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | 1.4 | $(550\ -250\ 500\ -250)^T$ | 0.087 | 8 | $(1.9697\ 0\ 0\ 0)^T$ |
| Mathiesen | SN | 1.5 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 1.6 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 1.7 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 1.8 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 1.9 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 2 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 2.1 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 2.2 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 2.3 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 2.4 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 2.5 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 2.6 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 2.7 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 2.8 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 2.9 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | SN | 3 | $(550\ -250\ 500\ -250)^T$ | - | - | - |
| Mathiesen | BB | 1 | $(1\ 1\ 1\ 1)^T$ | 0.187 | 11 | $(0.0089\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 1.1 | $(1\ 1\ 1\ 1)^T$ | 0.087 | 11 | $(0.0107\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 1.2 | $(1\ 1\ 1\ 1)^T$ | 0.081 | 11 | $(0.0128\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 1.3 | $(1\ 1\ 1\ 1)^T$ | 0.109 | 11 | $(0.0149\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 1.4 | $(1\ 1\ 1\ 1)^T$ | 0.079 | 11 | $(0.0170\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 1.5 | $(1\ 1\ 1\ 1)^T$ | 0.084 | 11 | $(0.0191\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 1.6 | $(1\ 1\ 1\ 1)^T$ | 0.078 | 11 | $(0.0213\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 1.7 | $(1\ 1\ 1\ 1)^T$ | 0.091 | 11 | $(0.0235\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 1.8 | $(1\ 1\ 1\ 1)^T$ | 0.095 | 11 | $(0.0258\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 1.9 | $(1\ 1\ 1\ 1)^T$ | 0.094 | 11 | $(0.0280\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 2 | $(1\ 1\ 1\ 1)^T$ | 0.178 | 11 | $(0.0302\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 2.1 | $(1\ 1\ 1\ 1)^T$ | 0.085 | 11 | $(0.0346\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 2.2 | $(1\ 1\ 1\ 1)^T$ | 0.167 | 11 | $(0.0346\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 2.3 | $(1\ 1\ 1\ 1)^T$ | 0.064 | 7 | $(0.0421\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 2.4 | $(1\ 1\ 1\ 1)^T$ | 0.139 | 11 | $(0.0389\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 2.5 | $(1\ 1\ 1\ 1)^T$ | 0.079 | 11 | $(0.0410\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 2.6 | $(1\ 1\ 1\ 1)^T$ | 0.169 | 11 | $(0.0430\ 0\ 0\ 0)^T$ |

| *Problema* | *Algoritmo* | $\lambda$ | $\boldsymbol{x}_0$ | *t (seg)* | $k$ | $\boldsymbol{x}^*$ |
|---|---|---|---|---|---|---|
| Mathiesen | BB | 2.7 | $(1\ 1\ 1\ 1)^T$ | 0.086 | 11 | $(0.0449\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 2.8 | $(1\ 1\ 1\ 1)^T$ | 0.099 | 11 | $(0.0468\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 2.9 | $(1\ 1\ 1\ 1)^T$ | 0.144 | 11 | $(0.0486\ 0\ 0\ 0)^T$ |
| Mathiesen | BB | 3 | $(1\ 1\ 1\ 1)^T$ | 0.089 | 11 | $(0.0502\ 0\ 0\ 0)^T$ |

**Tabla** 4.5: Desempeño del **Algoritmo 3.1** al variar $\lambda$.

Estos resultados sugieren una elección del parámetro $\lambda$ entre uno y dos, pues en este rango convergió el algoritmo con mayor frecuencia, aunque en general no lo hizo más rápido que para valores de $\lambda$ mayores que dos. Un dato que ratifica nuestras observaciones y que no está condensado en la Tabla 4.5 es el comportamiento del **Algoritmo 3.1** al resolver problema de *Kojima Shindo* cuando se tomó como punto inicial el punto $(100\ 100\ 100\ 100)$, ya que bajo estas condiciones, convergió solo para $\lambda = 1.1$, para los demás valores el algoritmo divergió.

Cabe destacar que la variación del parámetro $\lambda$ es sólo una experiencia numérica y por tanto, deja las puertas abiertas para futuras investigaciones en algoritmos que usen funciones de complementariedad.

# CAPÍTULO 5

# COMENTARIOS FINALES

Resolver el PCNL a través de la minimización de la *función de mérito* (1.10) es importante en el sentido que da robustés a los algoritmos cuyo objetivo es resolver el sistema de ecuaciones no lineales (1.5), los cuales por si solos ya son bastante significativos en la solución de problemas de este tipo.

En este trabajo, propusimos un algoritmo *cuasi Newton global* que permite resolver el PCNL y que puede ser útil, en primer lugar, cuando la matriz jacobiana de la función que define el problema es costosa de calcular y en segundo lugar, cuando el usuario no tiene información de la posible solución del problema. Para dicho algoritmo desarrollamos la teoría de convergencia global y demostramos que después de algunas iteraciones, este tiene una *tasa de convergencia* idéntica a la del *algoritmo local* propuesto en [1].

En un algoritmo de minimización, el gradiente de la función a minimizar (la función de mérito, en nuestro caso) desempeña un papel fundamental. En nuestro problema, dicho gradiente depende de las matrices del *jacobiano generalizado* de $\Phi_\lambda$, las cuales, siguiendo la filosofía *cuasi Newton*, no estábamos interesados en calcular, por ello, propusimos una aproximación al gradiente y analizamos, qué tan buena es esta aproximación.

Algunos experimentos numéricos preliminares realizados en esta investigación nos permi-

tieron ver la efectividad y competitividad del algoritmo propuesto, cuando comparamos su rendimiento con el de un algoritmo tipo *Newton generalizado*.

Además, las variaciones realizadas a nuestro algoritmo nos permitieron formular conclusiones *a priori* sobre el desempeño del mismo, por lo tanto, creemos conveniente realizar más pruebas numéricas que consideren dichas variaciones. De igual forma, creemos necesario incorporar estrategias que permitan elegir el parámetro inicial $\lambda$ con el objetivo de mejorar la efectividad en cuanto a convergencia del algoritmo.

# BIBLIOGRAFÍA


[1] Favian E. Arenas, *Métodos secantes de cambio mínimo para el problema de complementariedad no lineal*, Tesis de Maestría, Universidad del Cauca, 2013.

[2] Dimitri Bertzekas, *Constrained optimization and lagrange multiplier methods*, Athena Scientific, 1996.

[3] Stephen C. Billups, *Algorithms for complementarity problems and Generalized Equations*, Ph.D. thesis, University of Wisconsin, 1995.

[4] Peter N. Brown and Youcef Saad, *Convergence theory of nonlinear newton-krylov algorithms*, SIAM Journal on Optimization **4** (1994), no. 2, 297–330.

[5] Charles G. Broyden, *A Class of Methods for Solving Nonlinear Simultaneous Equations*, Mathematics of Computation **19** (1965), no. 92, 577–593.

[6] Sandra Buhmiler and Nataža Krejić, *A new smoothing quasi-newton method for nonlinear complementarity problems*, Journal of Computational and Applied Mathematics **211** (2008), no. 2, 141 – 155.

[7] Frank H. Clarke, *Necessary Conditions for Nonsmooth Problems in Optimal Control and the Calculus of Variations*, Ph.D. thesis, University of Washington, 1973.

[8] Frank H. Clarke, *Generalized gradients and applications*, Transactions of the American Society **205** (1975), 247–262.

[9] Frank H. Clarke, *Optimization and nonsmooth analisis*, SIAM, 1990.

[10] Thomas F. Coleman, Burton S. Garbow, and Jorge J. More, *Software for estimating sparse jacobian matrices*, ACM Trans. Math. Softw. **10** (1984), no. 3, 329–345.

[11] Tecla De luca, Francisco Facchinei, and Kanzow Christian, *A theorical and numerical comparison of some semismooth algorithms for complementarity problems*, Mathematical Programming (1997), 1 – 34.

[12] Tecla De Luca, Francisco Facchinei, and Christian Kanzow, *A semismooth equation approach to the solution of nonlinear complementarity problems*, Mathematical Programming **75** (1996), 407–439.

[13] John E. Dennis and Robert B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*, Society for Industrial and Applied Mathematics, 1996.

[14] Jundi Ding and Hongyou Yin, *A new homotopy method for solving non-linear complementarity problems*, Numerical Mathematics **16** (2007), 155–163.

[15] F. Facchinei and J. Soarez, *A new merit function for nonlinear complementarity problems and a related algorithm*, SIAM **7** (1997), 225–247.

[16] Francisco Facchinei, *Minimization of $SC^1$ functions and the maratos effect*, Operations research letters **23** (1994), no. 17, 131–137.

[17] Michael C. Ferris and Jong-Shi Pang, *Engineering and economic applications of complementarity problems*, SIAM Review **39** (1997), 669–713.

[18] Andreas Fischer, *A special newton-type optimization method*, Optimization (1992), 269 – 284.

[19] Andreas Fischer and Christian Kanzow, *On finite termination of an iterative method for linear complementarity problems*, Math. Program. **74** (1996), no. 3, 279–292.

[20] L. Grippo, F. Lampariello, and S Lucidi, *A nonmonotone line search technique for newton's method*, SIAM **23**.

[21] Christian Kanzow and Helmut Kleinmichel, *A new class of semismooth newton-type methods for nonlinear complementarity problems*, Comput. Optim. Appl. **11** (1998), no. 3, 227–251.

[22] Masakazu Kojima and Susumu Shindoh, *Extensions of newton and quasi-newton methods to systems of $PC\,1$ equations*, Research reports on information sciences, Inst. of Technology, Department of Information Sciences, 1986.

[23] Michael Kostreva, *Elasto-hydrodynamic lubrication: A non-linear complementarity problem*, International Journal for Numerical Methods in Fluids **4** (1984), no. 4, 377–397.

[24] Li-Zhi Liao and Houduo Qi, *A neural network for the linear complementarity problem*, Math. Comput. Model. **29** (1999), 9 – 18.

[25] Li-Zhi Liao, Houduo Qi, and Liqun Qi, *Solving nonlinear complementarity problems with neural networks*, Comput. Appl. Math. **131** (2001), 343 – 359.

[26] Lars Mathiesen, *An algorithm based on a sequence of linear complementarity problems applied to a walrasian equilibrium model: An example*, Mathematical Programming **37** (1987), 1–18.

[27] Jorge Nocedal and Wright Stephen J., *Numerical optimization*, Springer Series in Operations Research, 2000.

[28] Jong-Shi Pang and Steven A. Gabriel, *A robust algorithm for the nonlinear complementarity problem*, Mathematical Programming **60** (1993), 295–337.

[29] Jong-Shi Pang and Liqun Qi, *Nonsmooth equations: Motivation and algorithms*, SIAM Journal on Optimization **3** (1993), no. 3, 443–465.

[30] Rosana Pérez and Vera Lucía Rocha, *Recent applications and numerical implementation of quasi-newton methods for solving nonlinear systems of equations*, Numerical Algorithms **35** (2004), 261–285.

[31] Rosana Pérez M., *Algoritmos para complementariedade não lineal e problemas relacionados*, Ph.D. thesis, IMECC, UNICAMP, Campinas, Brazil, 1997.

[32] Rosana Pérez M. and Diaz V. Tomás H., *Minimización sin restricciones*, Universidad del Cauca, 2010.

[33] Liqun Qi, *Convergence analysis of some algorithms for solving nonsmooth equations*, Math. Oper. Res. **18** (1993), no. 1, 227–244.

[34] Vera Lucia Rocha, Jose M. Martínez, and Rosana Pérez, *On the local convergence of quasi-newton methods for nonlinear complementary problems*, Applied Numerical Mathematics **30** (1999), 3–22.

[35] Jhon Glen Wardrop, *Some theoretical aspects of road traffic research*, Proceeding of the Institute of Civil Engineers (1952), 325 – 378.

[36] David Watkins, *Fundamentals of matrix computations*, John Wiley and Sons, 2002.

[37] Qing Xu and Chuangyin Dang, *A new homotopy method for solving non-linear complementarity problems*, Optimization **57** (2008), no. 5, 681–689.

[38] Longquan Yong, *Nonlinear complementarity problem and solution methods*, Proceedings of the 2010 International Conference on Artificial Intelligence and Computational Intelligence: Part I, Springer-Verlag, 2010, pp. 461–469.

[39] Xia Youshen and Feng Gang, *A neural network for solving nonlinear projection equations*, Neural Networks **5** (2007), 577 – 589.