Trabajo de Iniciación a la Investigación

CRITERIOS PARA DECIDIR EL AGREGADO DE DIRECCIONES A LA MATRIZ DE MEMORIA LIMITADA DEL METODO L-BFGS

Alumna: María Ángela de Isasi

Directora: Prof. Nélida Echebest

Iniciación Nº 19

Septiembre de 2006

CRITERIOS PARA DECIDIR EL AGREGADO DE DIRECCIONES

A LA MATRIZ DE MEMORIA LIMITADA DEL METODO L-BFGS

Este trabajo está dirigido a analizar criterios y procedimientos a utilizar para mejorar la aproximación cuadrática del algoritmo de Memoria Limitada BFGS (L-BFGS).En general distintos autores hacen actualizaciones extras de la matriz con ese objetivo.
M. Al-Baalí [1] propone ciertas modificaciones al algoritmo L-BFGS, empleando actualizaciones extras utilizando direcciones de iteraciones previas. Como en algunas iteraciones, estas actualizaciones pueden ser redundantes ó empeorar la calidad de la aproximación del Hessiano, propone un criterio de actualización dependiente del análisis de la calidad. Por lo tanto dichas actualizaciones extras se emplean solo para mejorar la aproximación del Hessiano L-BFGS cuando esa aproximación se considera muy pobre.
También Morales y Nocedal [4] enriquecen el método L-BFGS con el mismo objetivo usando iteraciones del método de Newton Truncado y agregando cierta información obtenida a la matriz de L-BFGS.

1. Introducción:

Consideramos el problema de optimización no lineal:

$$\text{Min } f(x) \quad \text{con} \quad x \in R^n, \text{ siendo } f \text{ una función suave.}$$

Se asume también, que no se dispone del cálculo del Hessiano y que n es grande.

Entre los métodos más usados para este problema, se tienen los métodos Quasi Newton tales como el método BFGS con Memoria Limitada (L-BFGS) de Liu y Nocedal [3]. Este método sólo almacena $2*m$ vectores para generar el modelo cuadrático aproximante, en lugar de la matriz de $n*n$ componentes.

El método, en cada iteración considera los pares (s_i,y_i): $s_i = x_{i+1} - x_i$; $y_i = g_{i+1}-g_i$ **(A)** con $k-\bar{n}+1 \le i \le k$, $m \ge 2$, $\bar{n} = \min(m, k)$, provenientes de las iteraciones anteriores.

A continuación explicaremos las características principales de la metodologia citada.

2. Métodos Quasi –Newton

Los métodos Quasi- Newton en cada iterado, adoptan el modelo cuadrático aproximante previo usando la información aportada por el $\nabla f(x_{k+1})$ y la relación $B_{k+1} * s_k = y_k$ .Tal ecuación se denomina ecuación secante.

2.1 Método BFGS.

El más popular y eficiente de todos los métodos Quasi-Newton es el BFGS, que lleva su nombre debido a sus autores Broyden, Fletcher, Goldfarb y Shanno.

Este algoritmo genera una sucesión de puntos $\{x_k\}$ satisfaciendo: $x_{k+1} = x_k + \alpha_k * p_k$, dado un x_0 inicial, siendo α_k el paso en la dirección $p_k = - H_k * g_k$ y $H_k = B_k^{-1}$.

La matriz H_k se actualiza en cada iteración, mediante: $H_{k+1} =$ BFGS (H_k , s_k, y_k) **(I)** siendo H_k la aproximación inversa de B_k, simétrica y definida positiva.

La fórmula BFGS (H, s, y) = $(I - s * y^t / s^t * y) * H * (I - y * s^t / s^t * y) + (s * s^t / s^t * y)$, define la nueva H_{k+1} agregando la información del par (s_k, y_k):

$s_k = x_{k+1} - x_k$

$y_k = g_{k+1} - g_k$, siendo $g_{k+1} = \nabla f(x_{k+1})$ y $g_k = \nabla f(x_k)$

Tal fórmula satisface la ecuación secante:

$B_{k+1} * s_k = y_k$, o la inversa $H_{k+1} * y_k = s_k$

Se debe destacar que preserva la positividad de los autovalores, si H_k es definida positiva y si α_k se determina de tal forma que el nuevo iterado satisface las condiciones de Wolfe.

1) $f(x_{k+1}) - f(x_k) \geq C_1 * g_k * s_k$ Siendo $C_1 \in (0, ½)$ y $C_2 \in (C_1, 1)$

2) $y_k^t * s_k \geq (1 - C_2) * |g_k^t * s_k|$ (por ejemplo: C_1=10.E-4, C_2=0.9)

La condición 2) garantiza que $s_k^t * y_k > 0$; siempre que $g_k \neq 0$.

Este algoritmo se aplica a la resolución de problemas de pequeña y mediana escala.

2.2 Convergencia global del Método BFGS.

A pesar del excelente comportamiento práctico de éste método, sólo se ha demostrado la convergencia global de BFGS con búsqueda lineal inexacta, al caso de funciones estrictamente convexas. Las hipótesis sobre la función objetivo son:

Suposición N°1:

1) La función objetivo f, es dos veces continuamente diferenciable.

2) El conjunto de nivel $\Omega = \{ x \in R^n : f(x) \leq f(x_0)\}$ es convexo y existen constantes positivas m y M tal que :

$$m * \| z \|^2 \leq z^t * G(x) * z \leq M * \|z\|^2 \quad ; \text{ para todo } z \in R^n \text{ y } x \in \Omega.$$

dónde denotamos a la matriz Hessiana $\nabla^2 f(x) = G(x)$ y $\| . \|$ es la norma Euclideana.

La parte 2) implica que G(x) es definida positiva en Ω y que f tiene un único mínimo x^* en Ω.

Ver detalles: (Nocedal - Wright) [5]

Teorema: Convergencia Global.

Sea B_0 una matriz inicial, simétrica y definida positiva y sea x_0 un punto inicial para el cual la Suposición 1, se satisface, entonces la sucesión $\{x_k\}$ generada por el algoritmo BFGS converge al mínimo x^*.

Demostración: (Nocedal - Wright) [5].

Una extensión del análisis muestra que la velocidad de convergencia de $\{x_k\}$ es superlineal. En particular, se puede mostrar, que la sucesión $\| x_k - x^* \|$ converge a cero rápidamente satisfaciendo que:

$$\sum_{k=1}^{\infty} \| x_k - x^* \| < \infty \qquad \text{(B)}$$

2.3 Convergencia Superlineal. Método BFGS

Aquí se considerará la velocidad de convergencia para funciones no necesariamente convexas. Para obtener ese resultado importante se hace la siguiente suposición:

Suposición N °2

La matriz Hessiana G es Lipschitz continua en un entorno N de x^*:

$\| G(x) - G(x^*) \| \leq L \| x - x^* \|$; para todo $x \varepsilon N$; L es una constante positiva.

Ver detalles:(Nocedal.- Wright) [5]

Teorema: Convergencia Local

Supongamos que f es dos veces continuamente diferenciable y que la sucesión $\{x_k\}$ generada por el algoritmo BFGS converge a un mínimo x^* para el cuál la Suposición 2 se satisface. Supongamos además que se satisface (B), entonces $\{x_k\}$ converge a x^* en forma superlineal.

Demostración: (Nocedal- Wright) [5].

A continuación damos brevemente una idea del procedimiento utilizado para realizar la búsqueda lineal en la dirección $p_k = H_k * (- g_k)$, y determinar el paso que satisfaga las condiciones de Wolfe.

3. Métodos de Búsqueda Lineal

En esta metodología, en cada punto x_k, se considera una dirección de descenso p_k asociada al método y se determina el nuevo iterado x_{k+1} en dicha dirección, mediante:

$x_{k+1} = x_k + \alpha_k * p_k$; dónde $\alpha_k > 0$ es un escalar llamado longitud del paso.

La convergencia del método depende de la elección de α_k y p_k.

Los algoritmos de búsqueda lineal requieren que p_k sea una dirección de descenso –por lo cuál se requiere que $p_k^t * g_k < 0$ - ya que esta propiedad garantiza que la función objetivo desciende en esa dirección.

En el caso de los métodos Quasi-Newton, la dirección de búsqueda tiene la forma:

$$p_k = - B_k^{-1} * g_k = - H_k * (g_k)$$

Siendo B_k una matriz simétrica y no singular que se define de la siguiente manera:

- $B_k = \nabla^2 f(x_k)$ (Hessiano exacto) método de Newton.

- B_k es una aproximación del Hessiano, para los métodos Quasi-Newton propiamente dichos, la que es actualizada en cada iteración.

Además, si p_k es tal que: $p_k = -B_k^{-1} * g_k$ con B_k definida positiva, se tiene:

$$p_k * g_k = -g_k^t * B_k^{-1} * g_k < 0$$

Por lo tanto para los métodos Quasi- Newton con búsqueda lineal se requiere que la matriz B_k tenga la propiedad de ser definida positiva.

Consideremos ahora la definición de α_k:
Se quiere que la elección de α_k dé una reducción sustancial de la función objetivo f, pero a la vez se quiere que se realice en un tiempo razonable. La idea es encontrar el valor de α para el cuál la función $\varphi(\alpha) = f(x_k + \alpha * p_k)$; ($\alpha > 0$) tenga un mínimo global, pero como eso es costoso, salvo cuando las funciones son cuadráticas, se utilizan los métodos de búsqueda inexacta.

3.1 Condiciones de Wolfe

Una condición muy conocida y usada de búsqueda lineal inexacta estipula en primer lugar que α_k debe permitir una disminución suficiente en la función objetivo f, la cuál es medida por la siguiente desigualdad que usualmente se la conoce como **Condición de Armijo**.

$$f(x_k + \alpha * p_k) \leq f(x_k) + C_1 * \alpha * g_k^t * p_k \quad ; \text{ para alguna constante } 0 < C_1 < 1/2$$

En la práctica C_1 se toma del orden de: $C_1 = 10.E\text{-}4$

La condición de suficiente decrecimiento no basta, para asegurar que el algoritmo haga un progreso razonable; por lo tanto se introduce un segundo requerimiento que se conoce como: **Condición de Curvatura**.

$$[\nabla f(x_k + \alpha_k * p_k)]^t * p_k \geq C_2 * \nabla f(x_k)^t * p_k \; ; \text{ para alguna constante } C_2 \in (C_1, 1).$$

Usualmente $C_2 = 0.9$ (en los métodos de: Newton y Quasi- Newton).

El lado izquierdo de la desigualdad anterior es $\varphi'(\alpha_k)$, por lo tanto la condición de curvatura asegura que la pendiente de $\varphi(\alpha_k)$ es mayor que C_2 veces $\varphi'(0)$.

Estas dos condiciones, (condición de Armijo y la condición de Curvatura) son usualmente llamadas **condiciones de Wolfe**.

3.2 Condición fuerte de Wolfe.

La longitud del paso α puede satisfacer las condiciones de Wolfe sin que ello signifique que estemos cerca del mínimo de $\varphi(\alpha)$.
Sin embargo se puede modificar la condición de Curvatura para forzar a α_k a ubicarse, al menos, en un amplio entorno del mínimo local o un punto estacionario de φ, de la manera siguiente:

$$f(x_k+\alpha_k * p_k) \leq f(x_k)+ C_1\, \alpha_k * \nabla f(x_k) * p_k$$

$$|\nabla f(x_k+\alpha_k * p_k)^t * p_k| \leq C_2 . |\nabla f(x_k)^t * p_k| \quad ; \quad \text{dónde } 0<C_1<C_2<1$$

La última condición se denomina la **condición fuerte de Wolfe.**

Ahora nos ocuparemos de los problemas a gran escala, para ello presentamos aquí, el método de Memoria Limitada BFGS (L-BFGS). [5]

4. Método Memoria Limitada BFGS (L-BFGS)

El método L-BFGS se utiliza para resolver problemas de optimización a gran escala en las que las matrices Hessianas no pueden calcularse con bajo costo computacional.
A fin de definir la aproximación H_k se usa la información de curvatura de las m, más recientes iteraciones.
Este método se define como el BFGS, excepto que la aproximación inversa del Hessiano L-BFGS, contiene menos información que la aproximación inversa del Hessiano BFGS, y la dirección de búsqueda p_k, dada por: $p_k = - H_k * g_k$, se obtiene sin calcular explícitamente la matriz H_k.
Esto se hace almacenando m pares de vectores (s_i, y_i), con $3 \leq m \leq 20$; y los escalares positivos $s_i^t * y_i$ para $i = k-\tilde{n}+1, \ldots\ldots, k$; con $\tilde{n} = \min(m, k)$. Entonces cuando $k>m$, el par más reciente reemplaza al par más antiguo.

Daremos ahora el proceso de actualización de la matriz H_k.
En la iteración k-ésima, se tiene el iterado x_k y el conjunto de pares de vectores $\{s_i, y_i\}$; con $i = k-m, \ldots\ldots\ldots, k-1$.
Se elige una aproximación Hessiana H_k^0 inicial, y de (I) se ve que la aproximación H_k, puede ser escrita de la manera siguiente:

$$H_k = (V_{k-1}^t \ldots V_{k-m}^t) * H_k^0 * (V_{k-m} \ldots V_{k-1}) + \rho_{k-m} * (V_{k-1}^t \ldots V_{k-m+1}^t) * s_{k-m} * s_{k-m}^t * (V_{k-m+1} \ldots V_{k-1}) +$$

$$+ \rho_{k-m+1} * (V_{k-1}^t \ldots V_{k-m+2}^t) * s_{k-m+1} * s_{k-m+1}^t * (V_{k-m+2} \ldots V_{k-1}) + \ldots + \rho_{k-1} * s_{k-1} * s_{k-1}^t \qquad (C)$$

Siendo $V_k = I - \rho_k . y_k s_k^t$ y $\rho_k = \frac{1}{y_k^T s_k}$

Desde la expresión (C) se puede deducir un procedimiento recursivo para calcular el producto $H_k . * g_k$ de forma eficiente, que damos a continuación:

Algoritmo A: (Procedimiento recursivo para realizar el producto matriz por vector)

Dado g_k

$q \leftarrow g_k$

for $i = k-1, k-2, ...k-m$

$\alpha_i \leftarrow \rho_i * s_i^t * q$

$q \leftarrow q - \alpha_i * y_i$

end for.

$r \leftarrow H_k^0 * q$

for $i = k-m, k-m+1, ..., k-1.$

$\beta \leftarrow \rho_i * y_i^t * r$

$r \leftarrow r + s_i * (\alpha_i - \beta)$

end for.

Obteniendo: $r = H_k * g_k$

Sin considerar la multiplicación $H_k^0 * q$, el algoritmo anterior requiere $4 * m * n$ multiplicaciones. Si H_k^0 es diagonal, entonces se necesitan n multiplicaciones adicionales. Además de ser económico, tiene la ventaja que la multiplicación por la matriz inicial H_k^0 es aislada del resto de los cálculos, permitiendo a esta matriz ser elegida libremente y variar entre las iteraciones. Nosotros podemos hacer una elección implícita de H_k^0, incluso definiendo alguna aproximación inicial B_k^0 del Hessiano (no, de su inverso), y obtener r resolviendo el sistema $B_k^0 * r = q$.

Un método para escoger H_k^0, que ha demostrado ser eficaz en la práctica es definir:

$$H_k^0 = \gamma_k * I \quad \text{(Liu-Nocedal).}$$

$$\text{siendo : } \gamma_k = s_{k-1}^t * y_{k-1} / y_{k-1}^t * y_{k-1} \qquad \text{(D)}$$

γ_k es el factor escala que intenta estimar la curvatura de la verdadera matriz Hessiana a lo largo de la más reciente dirección de búsqueda. Esta opción ayuda a que en la dirección p_k la longitud del paso $\alpha_k = 1$ se acepte en la mayoría de las iteraciones.
Para que la actualización de la matriz BFGS sea estable es importante considerar que la búsqueda lineal esté basada en las condiciones de Wolfe.

A continuación presentamos el algoritmo de Memoria Limitada BFGS (L-BFGS).[5]

Algoritmo L-BFGS:

Elegir el punto inicial x_0; sea $m > 0$ entero.

$k \leftarrow 0$

Repetir

Elegir H_k^0 (por ejemplo, usando (D))

Calcular $p_k \leftarrow -H_k * g_k$; mediante el Algoritmo *A*
Calcular $x_{k+1} \leftarrow x_k + \alpha_k * p_k$; dónde α_k cumple las condiciones de Wolfe.

Si $k > m$

Descartar el vector $\{s_{k-m}, y_{k-m}\}$ del almacenamiento

Calcular y guardar $s_k \leftarrow x_{k+1} - x_k$; $y_k \leftarrow g_{k+1} - g_k$

$k = k + 1$

Hasta convergencia.

Durante las primeras m-1 iteraciones, el algoritmo L-BFGS es equivalente al BFGS, si la matriz inicial H_0 es la misma en ambos métodos, y si $H_k^0 = H_0 = \gamma_k * I$ en cada iteración.
La estrategia de guardar el par más reciente {si, yi} trabaja bien en la práctica, y ninguna otra estrategia ha demostrado ser mas eficiente.
Notar que si $m = \infty$, el Hessiano L-BFGS es el mismo que el BFGS, lo cual cuando n es grande es imposible de guardar.

Bajo condiciones de suavidad en funciones convexas, el método BFGS converge superlinealmente (Powell [6]), mientras que el algoritmo L-BFGS lo hace linealmente (Liu y Nocedal [3]).

Este método es atractivo debido a su bajo costo de almacenamiento ($3 \leq m \leq 20$) y conveniente para problemas a gran escala.
Sin embargo puede ser muy lento para ciertos problemas, porque la aproximación del Hessiano L-BFGS es pobre comparado con la aproximación del Hessiano BFGS.

Debido a este tipo de inconvenientes varios autores han combinado las mejores propiedades del método de Newton Truncado Discreto y el método L-BFGS, a fin de obtener un algoritmo que sea económico y a la vez capaz de manipular problemas mal condicionados.
La idea es usar la información de curvatura generada durante el cálculo de un paso de Newton Truncado Discreto, con el objeto de mejorar las aproximaciones de L-BFGS.

Para la clase de problemas, con derivadas primeras y con derivadas segundas no disponibles o de costo computacional alto, una alternativa es usar Newton Truncado Discreto.

Se conoce que:

1) El método de Newton Truncado Discreto puede requerir muchas evaluaciones del gradiente por iteración, ya que reemplaza el producto $\nabla^2 f(x_k) * d_k$, por diferencias de gradientes.

2) El método de Memoria Limitada (L-BFGS) puede ser muy lento en problemas mal condicionados, y puede fallar su convergencia en problemas no convexos.

Para sortear estas dificultades, se han hecho modificaciones del algoritmo L-BFGS; las que para muchos problemas han mejorado su eficiencia.

4.1 Algoritmos Modificados L-BFGS

Método: Byrd, Nocedal y Zhu (1996).

Byrd, Nocedal y Zhu [2], diseñaron un nuevo algoritmo que intercala iteraciones de L-BFGS y de Newton Truncado.
En el algoritmo se emplean *mcg* actualizaciones extras en términos de *mcg* pares de vectores; luego el almacenamiento se incrementa de $2*m$ a $2*(m+mcg)$ vectores.
Los *mcg* pares no tienen forma diferente de los pares (A), ya que son generados por las iteraciones internas de Gradientes Conjugados de un paso de Newton Truncado Discreto.

La idea es la siguiente:
Sea $\{x_i\}$ la sucesión de iterados generada por el algoritmo de optimización. Supongamos que en el iterado x_k se decide hacer un paso de Newton Truncado Discreto, es decir se resuelve aproximadamente la ecuación $\nabla^2 f(x_k) * p_k = -g(x_k)$ (*), por el método de Gradientes Conjugados (Min $\frac{1}{2} * p_k^t * \nabla^2 f(x_k) * p_k + g(x_k)^t * p_k$); haciendo uso de la diferencia finita : $\nabla^2 f(x_k) * p_k \approx [g(x_k + \varepsilon * p_k) - g(x_k)] / \varepsilon$ (**) ; dónde ε es pequeño.

La iteración de Gradientes Conjugados en x_k se puede escribir como:

$$z^{i+1} = z^i + \sigma^i * v^i$$

dónde σ^i es la longitud del paso y las direcciones $\{v^i\}$ son conjugadas respecto a $\nabla^2 f(x_k)$.
Cuando se han hecho *mcg* iteraciones de Gradientes Conjugados, la dirección de búsqueda del algoritmo de optimización esta dada por: $p_k = z^{mcg}$
Para calcular: $\nabla^2 f(x_k) * v^i$, la cual es necesaria en toda iteración de G.C usa la fórmula (**).

Una vez que la iteración de Gradientes Conjugados ha finalizado se guardan los pares de vectores (s_k^i, y_k^i) tales que:

$$s_k^i = \varepsilon * v^i \quad ; \quad y_k^i = g(x_k + \varepsilon * v^i) - g(x_k) \quad i = 1, \ldots, mcg$$

Como la iteración de Gradientes Conjugados termina si se detectan direcciones de curvatura negativa, los pares de vectores (s_k^i, y_k^i) satisfacen la condición de Curvatura: $(s_k^i)^t * y_k^i > 0$ y pueden usarse para definir la matriz de Memoria Limitada BFGS H_k.

Ver detalles: Byrd, Nocedal y Zhu [2].

Por lo tanto se puede considerar que el almacenamiento se divide en dos partes: una parte guarda los pares (s_k^i, y_k^i) generados por los pasos internos de Grad. Conjugados, y la otra parte contiene los pares (s_k, y_k) generados por la iteraciones del algoritmo principal.

Con esta técnica los autores [2] muestran para cierto tipo de problemas resultados mejores que del método L-BFGS. Las iteraciones extras mejoran la calidad de la aproximación del Hessiano. Sin embargo al examinar los resultados numéricos, se observa que en algunos casos no sería necesario emplear dichas actualizaciones extras, pues las matrices L-BFGS dan buenas aproximaciones del Hessiano.

Morales y Nocedal [4] toman en cuenta esas consideraciones y proponen una forma para intercalar ambos métodos. Este método se denomina método Enriquecido.

Método Enriquecido: Morales y Nocedal. (2000).

Durante las "*l* "iteraciones del ciclo L-BFGS, la matriz H (m) de Memoria Limitada es actualizada. La matriz que se obtiene al finalizar el ciclo se usa para precondicionar la primera de las "*t* "iteraciones de Newton Truncado. Durante las *t*-1 iteraciones restantes, la matriz de memoria limitada H (m) es actualizada usando la información generada por las iteraciones de Gradientes Conjugados precondicionado y es usada en la próxima iteración de Newton Truncado. Una vez que *t* pasos de Newton Truncado fueron ejecutados, la última actualización de H (m) se utiliza como la matriz inicial de memoria limitada en el nuevo ciclo de L-BFGS.
Luego el proceso continúa de esta manera, alternando ciclos de L-BFGS con ciclos de Newton Truncado y transmitiendo información de curvatura de un ciclo a otro.
La longitud de los ciclos se elige dinámicamente, teniendo en cuenta durante el proceso, la conveniencia o no de realizar tales cambios. Estos cambios en la longitud de los ciclos se basa en el comportamiento del algoritmo de Newton Truncado, y en especial en las

características de la función a tratar. Notar que L-BFGS y Newton Truncado son casos particulares del método Enriquecido:

Si $l = 0$ Método Newton Truncado

Si $t = 0$ Método L-BFGS

Morales y Nocedal [4] utilizan varios criterios para cambiar la longitud del ciclo de Newton Truncado y como consecuencia el de L-BFGS.

- *Hessiano Indefinido*: Si una iteración de Gradientes Conjugados genera una dirección de curvatura negativa, los autores recomiendan aumentar en uno la longitud del ciclo de L-BFGS y disminuir la de Newton Truncado.

- *Si en una iteración de Newton Truncado, la longitud del paso $\alpha < 0.8$*: Los autores considerando que es indicativo que no se está cerca de la región donde converge rápidamente, proponen disminuir la longitud del ciclo t en uno y retornar a L-BFGS.

- *Si un ciclo de t pasos sucesivos de Newton Truncado se ha completado*: Los autores consideran que el proceso ha llegado a la región donde el método de Newton Truncado es rápidamente convergente, entonces proponen aumentar la longitud del ciclo de Newton Truncado.

Ver Detalles: Morales y Nocedal [4]

Si bien estas estrategias en muchos problemas se muestran eficientes, los cambios propuestos no están directamente relacionados con la calidad de las matrices L-BFGS.

Criterio propuesto por M.Al-Baalí. (2000)

Al Baalí [1] propone un criterio de actualización dependiente de la calidad de las aproximaciones, y usa esto para agregar a lo más *mcg* actualizaciones extras.
Tanto el algoritmo de Byrd, Nocedal y Zhu [2], como el de Morales y Nocedal [4] tienen la desventaja de incrementar el espacio de almacenamiento y el costo de evaluar *mcg* pares extras. Sin embargo tienen la ventaja de preservar la forma de cálculo de la matriz L-BFGS.

Para evitar el cálculo extra de direcciones, Al Baalí [1] introduce ciertas modificaciones a la matriz L-BFGS, agregando *mcg* pares extras, elegidos desde los pares (A), en distintas formas.
Los resultados obtenidos de esta manera, muestran que el algoritmo modificado con algunos valores fijos de m y *mcg* trabaja bien en la práctica y mejora el resultado del método L-BFGS en términos de evaluaciones de la función y gradientes, pero puede empeorar en términos del número de actualizaciones.
Al Baalí [1] estudió el comportamiento de las matrices L-BFGS, midiendo su calidad en base a la comparación con las propiedades satisfechas por las matrices del método BFGS. Las matrices de ambos métodos mantienen la propiedad de ser definidas positivas, y

satisfacen la ecuación secante, ya que la última actualización en ambas matrices depende de los pares (s_k, y_k) que satisfacen $s_k^t * y_k > 0$. A consecuencia de esto, hace el estudio de la convergencia de las matrices L-BFGS con búsqueda lineal inexacta.

En primer lugar menciona un resultado de Ge y Powell que dice lo siguiente:

Las matrices BFGS $\{H_k\}$ convergen a una matriz H^* $(k \to \infty)$, si f es cuadrática, estrictamente convexa y los vectores $\{s_k\}$ satisfacen ciertas condiciones.
A pesar que los autores establecen que este resultado no se mantiene para funciones generales, dos veces diferenciables, esto implica que:
Si $H_{k+1} - H_k$ no es suficientemente cercana a la matriz nula, entonces H_{k+1} se aproxima pobremente a H^*. Como consecuencia de ese resultado, Al- Baalí actualiza la aproximación del Hessiano BFGS cuando esto ocurre, agregando unos pocos pares de la forma (A).

Bogg y Tolle analizaron el comportamiento de las matrices BFGS, independientemente de las condiciones de optimización, como sigue:

Sea $\{H_l\}$ generada por la fórmula BFGS, entonces:

$H_{l+1} = BFGS\ \{H_l, s_l, y_l\}$, siendo H_l definida positiva, s_l vectores de R^n que no provienen de un proceso de optimización e y_l dependen de s_l, como por ejemplo:

$y_l = G * s_l$, siendo G el Hessiano de una función cuadrática y estrictamente convexa.

Bajo esas condiciones los autores establecen la convergencia de $\{H_l\}$. Una revisión de este resultado muestra que esto también se mantiene si la secuencia $\{s_l\}$ es un subconjunto de la secuencia de vectores $\{s_i\}_{i=k-m+1}^{k}$ definidos en (A).
En este caso la secuencia de matrices $\{H_l\}$ depende de reutilizar varias veces estos vectores como así también los vectores $\{y_i\}_{i=k-m+1}^{k}$ (Ver: Al Baalí [1]). El autor propone detener las actualizaciones cuando la diferencia $H_{l+1} - H_l$ sea suficientemente cercana a la matriz nula. Como la matriz diferencia no puede calcularse explícitamente, una alternativa es chequear el error relativo:

$$\text{(a)}\quad |d_{l+1} - d_l| / |d_{l+1}| \leq \varepsilon \quad ; \text{ dónde } \varepsilon > 0$$

$$d_l = g^t * H * g \quad ; \text{dónde g es el gradiente } g_{k+1} \text{ más reciente.}$$

El criterio (a) se utiliza para analizar la no convergencia de la sucesión $H_{k+1} - H_k$ ya que cuando $d_{l+1} - d_l \to 0$, no implica la convergencia de $\{H_{l+1} - H_l\}$. (Ver: Al Baalí [1]).

Antes de continuar con el análisis de otros criterios para mejorar las matrices de L-BFGS damos una idea del proceso de precondicionamiento de una matriz en el método de Gradientes Conjugados.
Cuando se precondiciona la matriz del problema A * x = b, se observa que el número de iteraciones realizadas por el método de Grad. Conjugados disminuye considerablemente, y esto significa ahorro de tiempo de cálculo.

I. Precondicionamiento: (en el método de Gradientes Conjugados).
(Nocedal- Wright) [5]

El método de Gradientes Conjugados (Lineal), se usa para resolver sistemas lineales de la forma: $A\,x = b$, dónde A es $n \times n$, simétrica y definida positiva.
La resolución de este problema es equivalente a encontrar el mínimo de la función cuadrática: $\phi(x) = \frac{1}{2} x'.Ax - b'x$, ya que ambos tienen la misma solución única.

Se puede acelerar el método de Gradientes Conjugados, transfor--mando el sistema lineal $A\,x = b$, con el objetivo de mejorar la distribución de los autovalores de A.
Este proceso es conocido como precondicionamiento, consiste en hacer un cambio de variables de x a $\tilde{x}$ mediante una matriz C no singular. Esto es: $\tilde{x} = Cx$

Luego ϕ es transformada:

$$\tilde{\phi}(\tilde{x}) = \frac{1}{2} \tilde{x}^T.(C^{-T}.A.C^{-1})\tilde{x} - (C^{-T}b)^T.\tilde{x} \quad (P)$$

El sistema lineal equivalente queda: $(C^{-T}.A.C^{-1})\tilde{x} = C^{-T}b$

Entonces la tasa de convergencia (Nocedal –Wright.[5]) dependerá de los autovalores de la matriz $C^{-T}.A.C^{-1}$ en lugar de los autovalores A. Por consiguiente, nosotros podemos escoger C tal que los autovalores de $C^{-T}.A.C^{-1}$ sean más favorable para la convergencia. Por ejemplo se puede elegir C tal que el número de condición de $C^{-T}.A.C^{-1}$ sea más pequeño que el número de condición A, ó se puede escoger C tal que los autovalores de $C^{-T}.A.C^{-1}$ estén agrupados, si es posible, en clusters (bloques); ya que se puede asegurar que el número de iteraciones necesario para encontrar una buena aproximación de la solución no es mucho mayor que el número de clusters (Nocedal-Wright)[5].

II. Algoritmo de Gradientes Conjugados Precondicionado.

El algoritmo de Gradientes Conjugados se puede aplicar al problema (P) en términos de la variable $\tilde{x}$ .Una vez que es hallada la solución del problema, se invierten las transformaciones ($x = C^{-1}\tilde{x}$) para expresar las ecuaciones en términos de x.

El procedimiento que se ha descrito anteriormente, se denomina método de Gradientes Conjugados Precondicionado.
Este método no hace uso de la matriz C explícitamente, sinó de la matriz $M = C^T.C$ que es simétrica y definida positiva, por construcción.

A continuación se dá la descripción del algoritmo:

Algoritmo: (Gradientes Conjugados Precondicionado)

Dado x_0 y M (matriz precondicionadora);

$r_0 \leftarrow A.x_0 - b$;

Resolver $M.\ y_0 = r_0$ para y_0 ;

Obtener $p_0 = -r_0$, $k \leftarrow 0$;

While $r_k \neq 0$

$$\alpha_k \leftarrow \frac{r_k^t.y_k}{p_k^t.A.p_k} ;$$

$$x_{k+1} \leftarrow x_k + \alpha_k.p_k ;$$

$$r_{k+1} \leftarrow r_k + \alpha_k.A.p_k ;$$

$$M.y_{k+1} \leftarrow r_{k+1} ;$$

$$\beta_{k+1} \leftarrow \frac{r_{k+1}^t.y_{k+1}}{r_k^t.y_k} ;$$

$$p_{k+1} \leftarrow -y_{k+1} + \beta_{k+1}.p_k ;$$

$$k \leftarrow k+1;$$

End (While).

Si $M = I$ en Grad. Conj. Precondicionado, se obtiene el método de Grad. Conjugados Standard. Una de las propiedades interesantes de este caso es la ortogonalidad de los residuos sucesivos ($r_k^t r_i = 0, i = 0,..,k-1$)

$$r_i^t M r_j = 0 \qquad \forall\, i \neq j .$$

(Más detalles: Nocedal-Wright [5]).

III. Las matrices de precondicionamiento $M_k(m)$ en Gradientes Conjugados.

Estas matrices son construidas usando el procedimiento L-BFGS de memoria limitada, a partir de los vectores guardados $(s_i, A_k.s_i) = (s_i, y_i)$, $i=1,...m$ seleccionados desde las direcciones calculadas durante las iteraciones internas de *PCG*, cuando resuelve el modelo cuadrático aproximante: $q_k(p) = f(x_k) + (g_k)^T.p + 0{,}5.p^T.A_k.p$, dónde A_k representa la matriz Hessiana en x_k.

Se usa la fórmula L-BFGS para obtener la matriz precondicionadora $M_k(m)$, en x_k.

Dados *mcg* pares seleccionados: (s_i, y_i) con $i = 1,....mcg$ (indicando con 1 el primero guardado, y *mcg* el último almacenado, siendo M_k^0 una matriz inicial (por ejemplo: $M_k^0 = \frac{s_{k-1}^t.y_{k-1}}{y_{k-1}^t.y_{k-1}} I_n$) se usa la fórmula (C) (Pag. 6.) y luego usa el Algoritmo A (Procedimiento recursivo para calcular el producto: $M_k(m).(z_j)$) (Nocedal-Wright -1999).

IV. Selección de direcciones en el proceso de Gradientes Conjugados

Hay diferentes maneras para seleccionar los vectores cuando el número de iteraciones internas es mayor al número *mcg* de vectores que se desean almacenar.

En particular, nosotros hemos ensayado varias estrategias diferentes aunque no hemos obtenido diferencias notables en el número de iteraciones del método en sucesivas aplicaciones.

Por ejemplo, tres diferentes estrategias son:

Dado x_k, el modelo local $q_k(p)$:

S_0. Guardar las direcciones (s_k^i, y_k^i) para las cuales $\frac{s_k^t A s_k^i}{\|s_k^i\|^2}$ tienen los menores valores, es decir las direcciones con curvaturas menores.

S_1. Guardar las direcciones (s_k^i, y_k^i) obtenidas en las iteraciones internas que resulten uniformemente espaciadas o casi uniformes de acuerdo a la propuesta de Nocedal-Morales.

S_2: Guardar pares espaciados en intervalos regulares o casi regulares, como en la estrategia previa, pero eligiendo de cada subintervalo el representante de menor curvatura.

Ilustración: Comportamiento del precondicionamiento con matrices de Memoria Limitada.

Ilustramos el comportamiento del precondicionamiento realizado por las matrices de L-BFGS, usando los pares salvados en las iteraciones internas del PCG, que satisfagan alguna ``condición relacionada a la curvatura del problema" .

Para ilustrar este tema, hemos usado dos matrices que tienen diferentes distribuciones de autovalores que aparecen en el trabajo de Morales y Nocedal (2000): Automatic Preconditioning by Quasi-Newton Updating, SIAM J.On Optimization, 10, 4, 1079-1096.

A_1	N=50	λ=1	$\lambda=2*10^9$
A_2	N=170	λ=1	$\lambda=1.3*10^8$

A_1: dimensión $n = 50$.

Esta matriz tiene un autovalor λ=1, otro $\lambda=2*10^6$, y los restantes están distribuidos en una amplia franja y también agrupados cerca del mayor $\lambda=2*10^9$.

A_2: dimensión $n = 170$.

Esta matriz tiene 10 autovalores iguales a 1, otro $\lambda=0.54*10^4$ y los restantes forman un cluster entre $\lambda=10^7$ y $\lambda=1.3*10^8$

V. Experiencias:

Se resuelve el sistema: $A\_ x = b_0$

En el caso de la matriz A_1, el término b_0 es:

$$b_0^1=b_0^n=0,\quad b_0^i=\frac{i}{n-1}*10^2,\ i=2,..n-1$$

dónde los superíndices indican las componentes del vector.
En el caso de la matriz A_2, b_0 tiene sus componentes ceros, salvo en las posiciones:

$$b_0^{34}=b_0^{68}=b_0^{102}=b_0^{136}=b_0^{170}=-8000$$

Aplicamos el algoritmo CG para resolver el $A\_ x = b_0$, y construimos las matrices precondicionadoras M (m), para distintos valores de m.
Hemos considerado m = 4, 8, 16, 20.

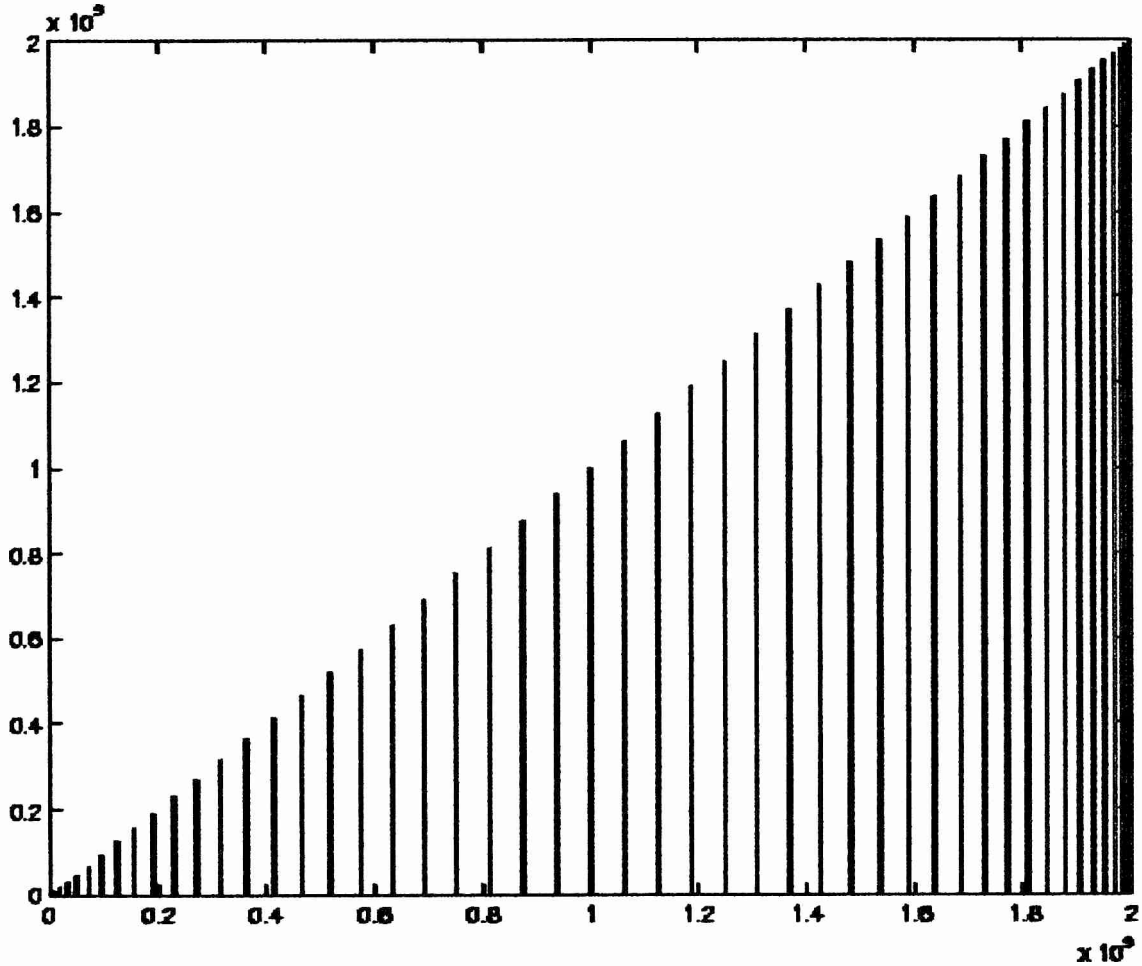
Para cada M (m), resolvemos $A\_ x = b_k$, k =1...., 20 donde los vectores del término de la derecha se obtienen usando b_k para definir b_{k+1} introduciendo perturbaciones del ± 5% con signos al azar para cada componente no nula de b_k.
En la tabla siguiente se reporta el número promedio de iteraciones necesitadas por el algoritmo PCG para cada problema, usando la matriz precondicionada M (m)*A, para aceptar el criterio de terminación:

$$\|r_i\|_\infty < 10^{-7}(\|A\|_\infty \|p_i\|_\infty + \|b\|_\infty)$$

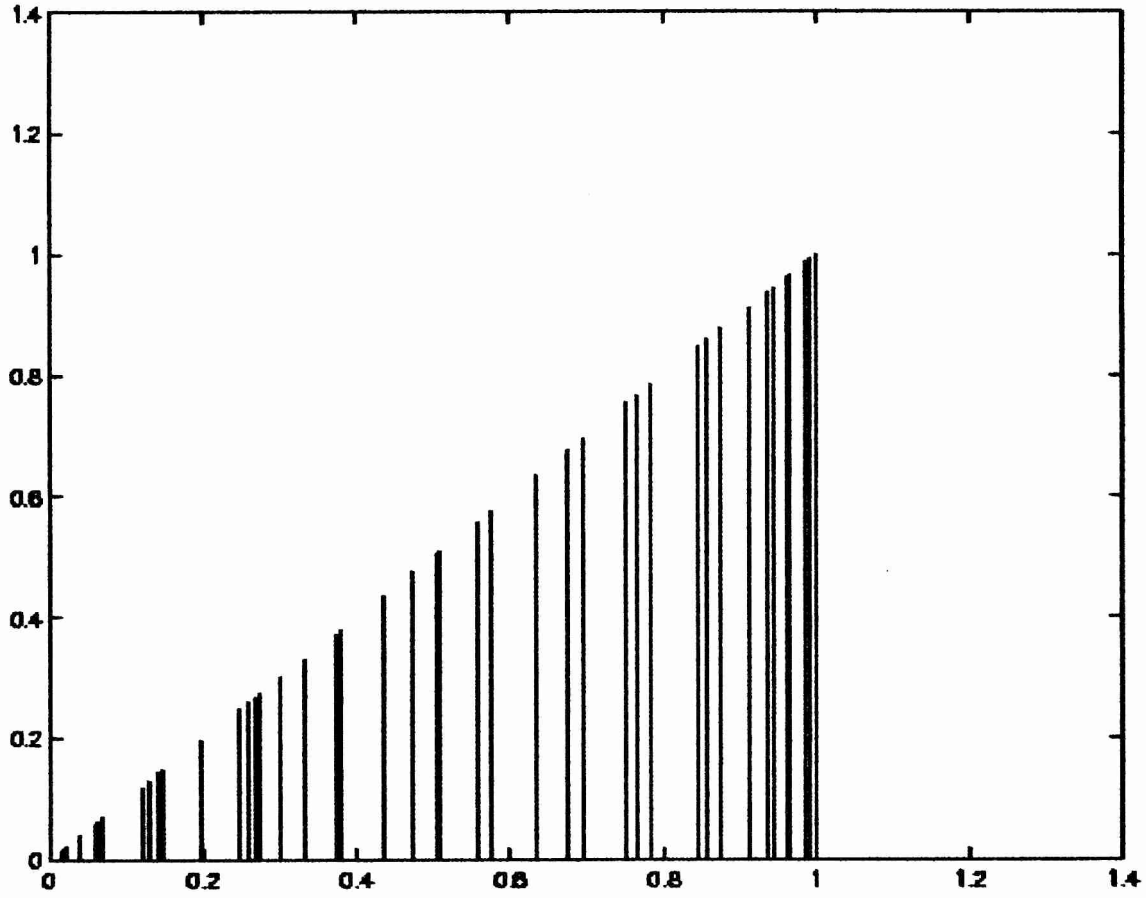
M (m) con m =	A_1		A_2	
	S_1	S_2	S_1	S_2
0	49	49	57	57
4	43	36	48	38
8	24	24	26	28
16	14	16	18	17
20	12	12	17	16

Comparando Precondicionadores:

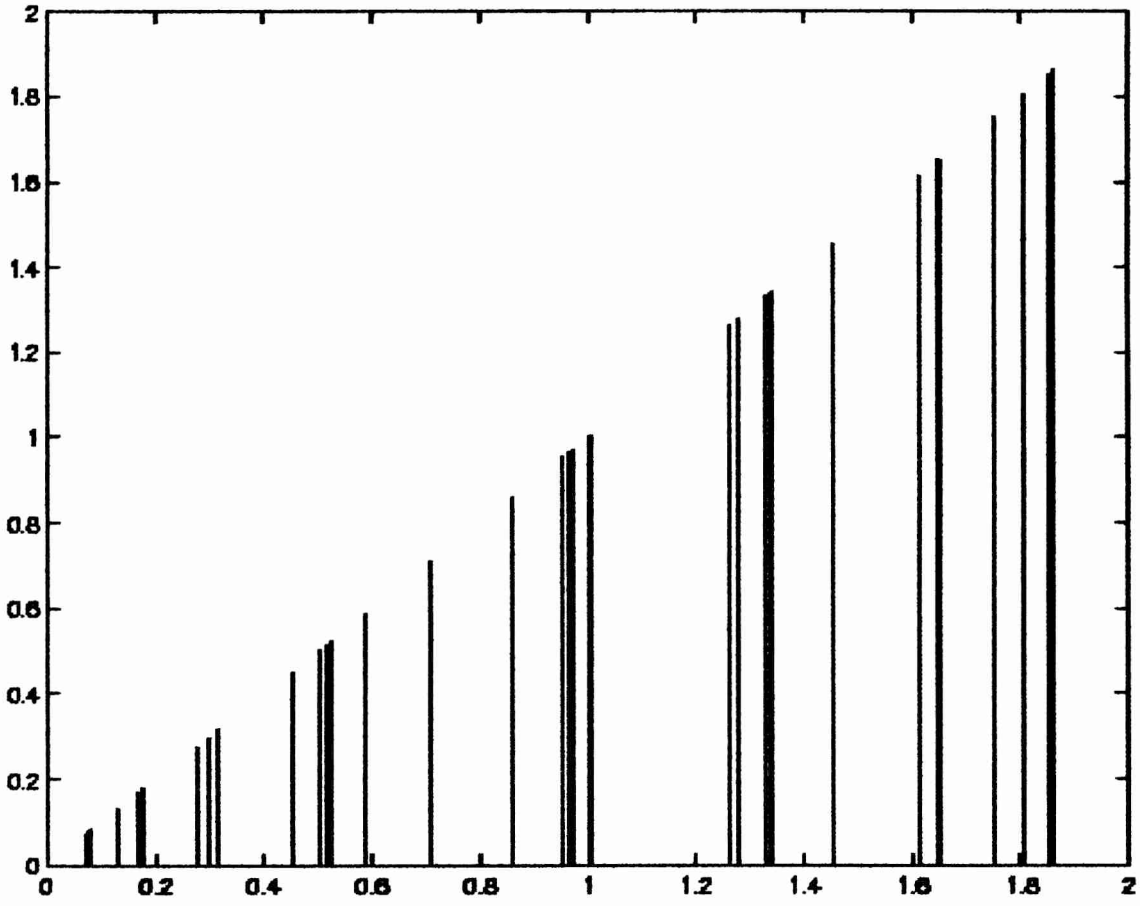
Ahora en imágenes se observa el efecto producido al considerar el precondicionamiento dado por M (m), es decir los autovalores de A se comparan con los de M (m)* A:



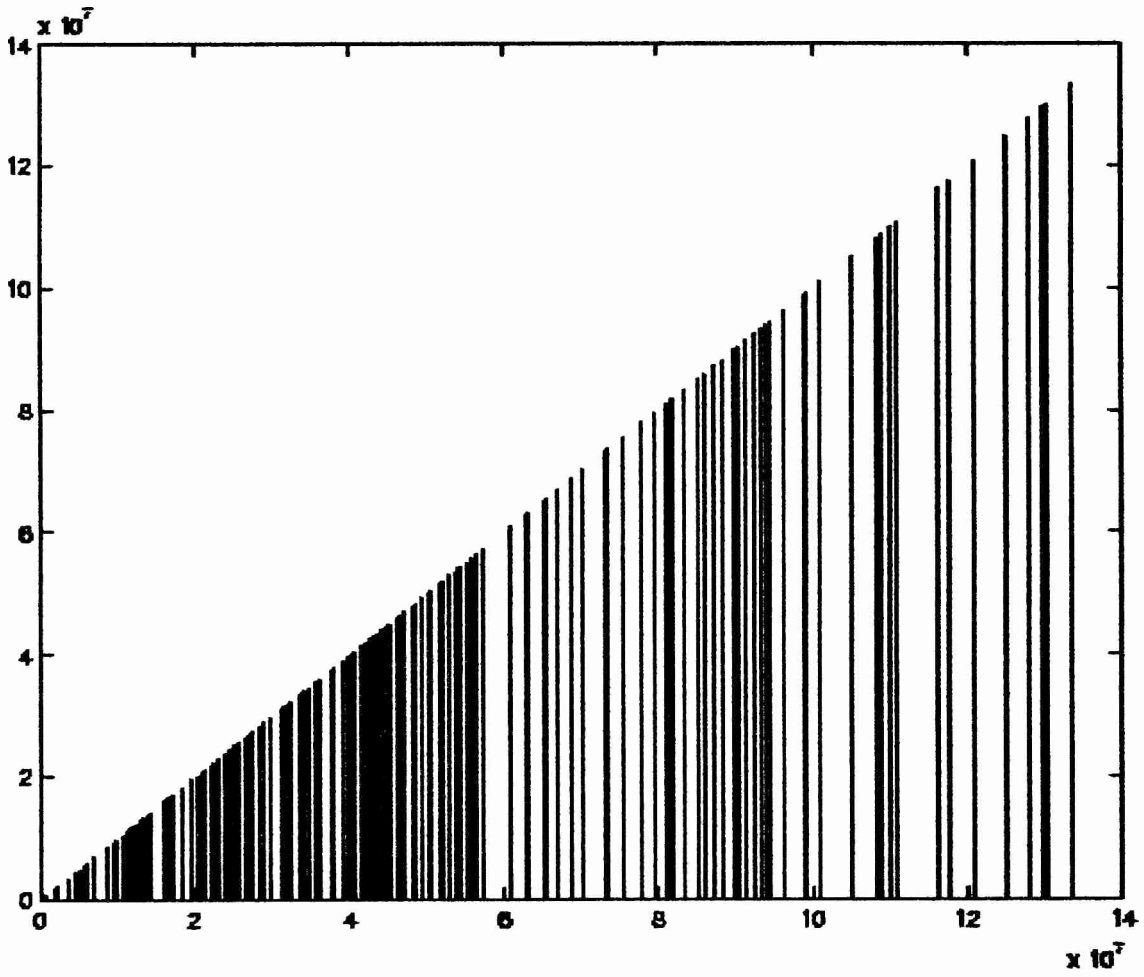
Autovalores de la matriz A_1



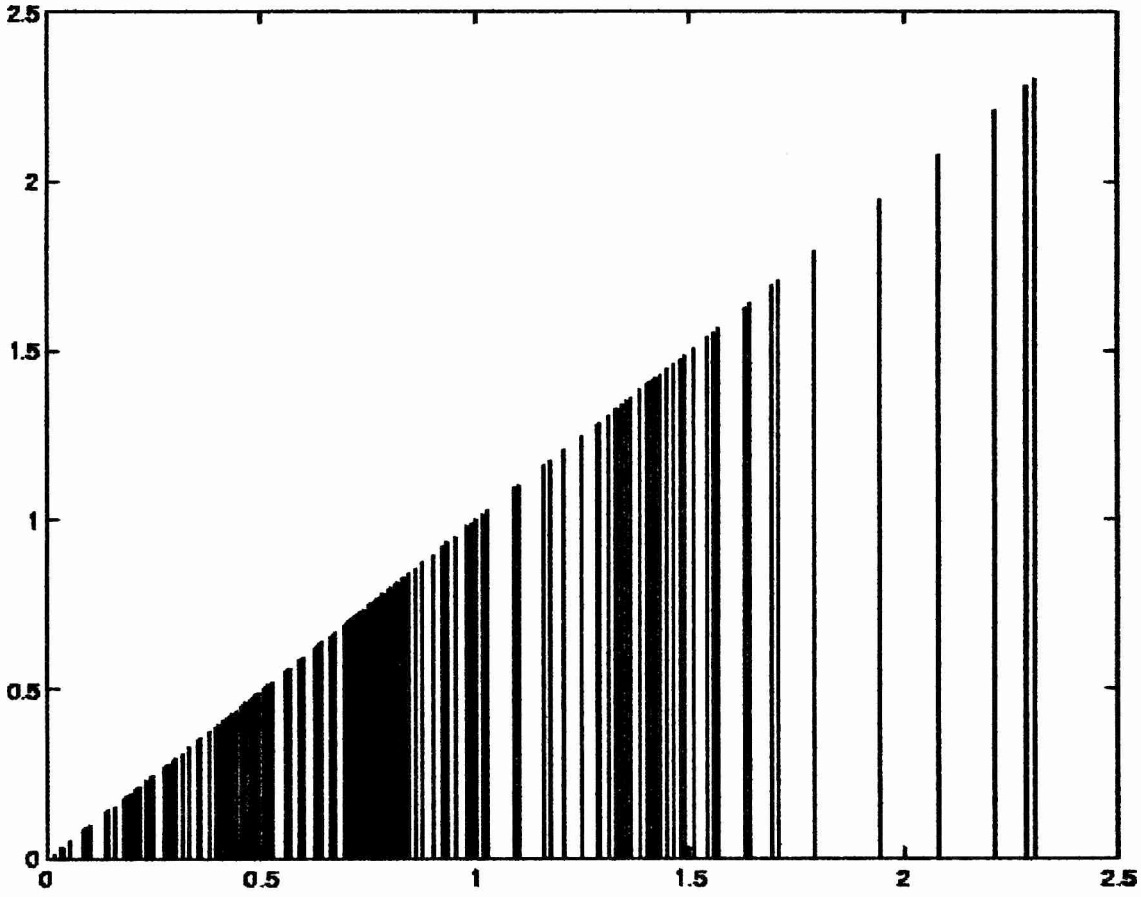
Autovalores de M (4) * A_1.



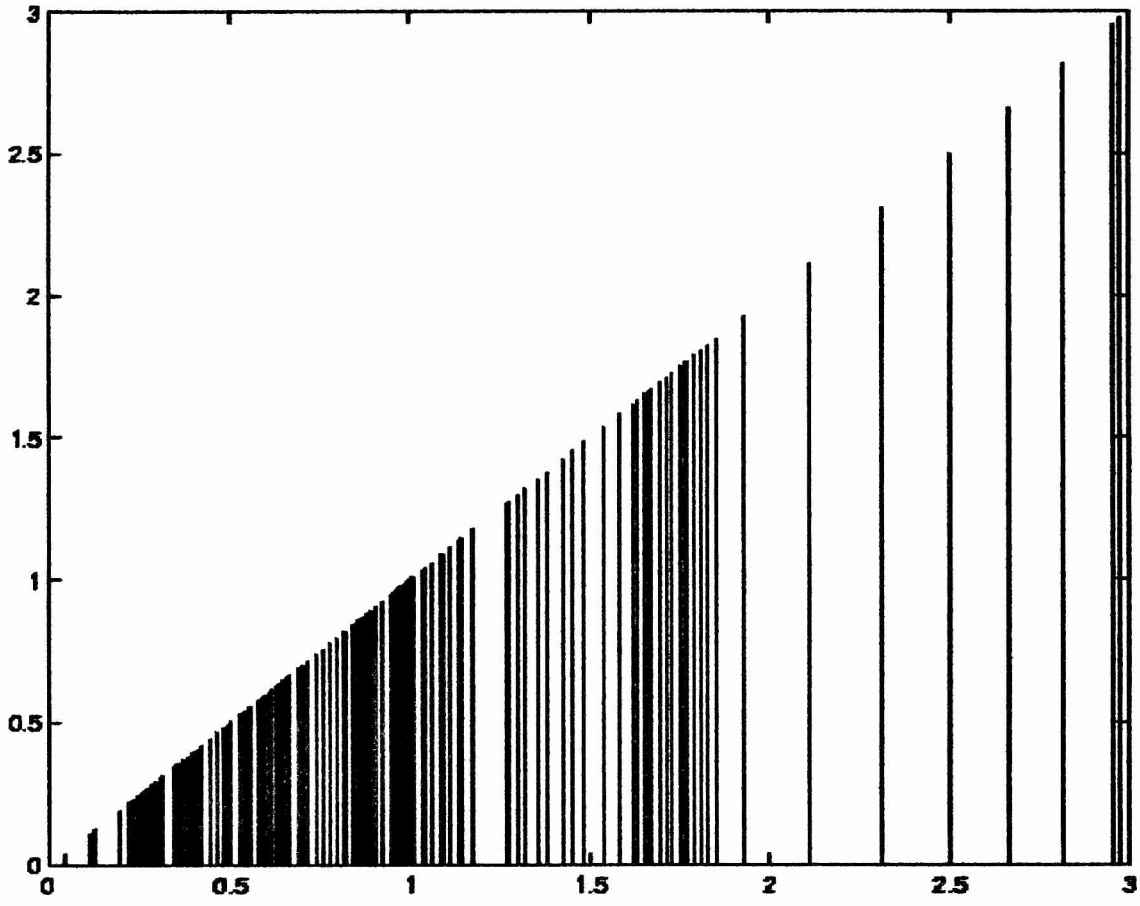
Autovalores de $M(16)^* A_1$



Autovalores de la matriz A_2



Autovalores de M (4)* A_2



Autovalores de M (16)* A_2

Para finalizar, se puede decir, basados en el análisis realizado sobre precondicionamiento que la distribución de los autovalores juega un rol muy importante, ya que esto implica que una buena distribución de los mismos acelera la convergencia del método.

5. Método L-BFGS. (Modificado).

Basados en los resultados analizados previamente, hemos querido analizar algunas modificaciones del método L-BFGS.
En lo que sigue, proponemos aplicar L-BFGS mientras su comportamiento se muestre eficaz de acuerdo a criterios que describiremos. Cuando estos indicadores predigan un comportamiento poco eficaz de L-BFGS, se propone enriquecerlo haciendo dos iteraciones del método de Newton Truncado, almacenando la información de curvatura en los pares (s_i, y_i) para agregar a la próxima matriz L-BFGS.

El algoritmo comienza el proceso de optimización utilizando el método L-BFGS, hasta que éste falle (de acuerdo a ciertos criterios).En caso de ocurrir esto por dos veces consecutivas

define la dirección por el método de de Newton Truncado, usando el método de Gradientes Conjugados. Sale con la dirección de Newton Truncado y usando alguna estrategia guarda las direcciones intermedias de gradientes conjugados para ser adicionadas a la matriz H_{k+1}.
Luego, retorna al procedimiento iterativo de L-BFGS, y en caso nuevamente de fallar repite el procedimiento descrito anteriormente, y así sucesivamente hasta la convergencia.

En ambos casos, una vez que tiene la dirección p_k hace una búsqueda lineal, basada en las condiciones fuertes de Wolfe, usando el algoritmo Gsrch de Powell.

Este algoritmo almacena m vectores de los cuales *mcg* lugares son para guardar los vectores (s_k^i, y_k^i) de Gradientes Conjugados, para formar las matrices H_k y m_r = m-*mcg* lugares para los pares de vectores (s_k, y_k).de L-BFGS.

Resumiendo: El algoritmo combina pasos de Newton Truncado y pasos de L-BFGS, como el algoritmo de Morales y Nocedal [4], aunque con criterios diferentes.
La idea es preservar la información generada en las iteraciones internas de Gradientes Conjugados para mejorar la calidad de la matriz siguiente H_{k+1} de L-BFGS.

5.1 Criterios para cambiar del método L-BFGS al método de Newton Truncado.

1) Si el descenso es insignificante; si: $| f(x_k + \alpha_k * p_k) - f(x_k) | \leq 10^{-8} * |f(x_k)|$

2) Si $\cos(p_k, -g_k) \approx 0$ (ejemplo: $\cos(p_k, -g_k) < 0.15$). Esto indica que la dirección es casi ortogonal al gradiente y puede ocurrir que el algoritmo falle.
El objetivo es mantener $\cos(p_k, -g_k) \geq \mu \quad ; \quad (\mu > 0)$
Se busca que $\theta_k = \text{argcos}(p_k, -g_k) < \pi/2 - \Delta\theta$, con $\Delta\theta$ fijo; ya que esto garantiza que $\|gk\|_{k \to \infty} \to 0$. (Nocedal - Wright [5]).

3) Cuando en la búsqueda lineal se tiene: $x_{k+1} = x_k + \alpha_k * p_k$, determinando que $\alpha_k < 1$ (ejemplo: $\alpha_k = 0.3$), implica que el paso $\alpha_k = 1$ no es aceptado.
Como el valor deseable de α_k es: $\alpha_k \approx 1$, pues una condición necesaria para la convergencia superlineal es que $\alpha_k = 1$, $(k > k_0)$ (Nocedal –Wright [5]), si $\alpha_k << 1$, se puede analizar como se comporta el det (H_{k+1}) en relación al det (H_k), es decir si cambia, en ese sentido H_{k+1} respecto de H_k.

Se conoce (Nocedal –Wright [5]) que bajo ciertas hipótesis, el algoritmo BFGS cumple:

$$\det(H_{k+1}) = \det(H_k) * [(\alpha_k)^2 * p_k * (-g_k) / s_k' * y_k]$$

De la igualdad anterior se desprende que si $| [(\alpha_k)^2 * p_k * (-g_k) / s_k^t * y_k] | \approx 1$ entonces $\det(H_{k+1}) \approx \det(H_k)$. Esto sirve para analizar si $H_{k+1} \neq H_k$, ó si al menos H_{k+1} tiene algunas propiedades de H_k.
Mientras el $\det(H_{k+1}) \neq \det(Hk)$ entonces con seguridad esas matrices son distintas.

Cuando en L-BFGS el det $(H_{k+1}) \neq$ det (H_k) podemos considerar que el procedimiento no es convergente aún, a un punto x^*, tal que det $H(x^*) \approx$ det (H_{k+1}) lo que indicaría que el modelo m_k es convergente a m^* si $k \geq k_N$, así que menos aún se estaría cerca de la convergencia de la sucesión.
Así, cuando el procedimiento de L-BFGS se ha estacionado en un modelo cuadrático aproximante de la función, sería deseable que el paso $\alpha_k = 1$.
En este caso, si $\alpha_k << 1$, consideramos que el modelo H_k no concuerda con el verdadero modelo aproximante de f(x) en x_k, por lo tanto se recurre al modelo aproximante dado por $\nabla^2 f(x_k)$, pasando en esa iteración al método de Newton Truncado.

En los casos descriptos anteriormente se cambia el método L-BFGS momentáneamente por el método de Newton Truncado, con el objeto de mejorar la matriz H_{k+1}, que corresponde a la del modelo cuadrático próximo a considerar.

A continuación damos el algoritmo L-BFGS Modificado (M – LBFGS).

Algoritmo (M-LBFGS)

Dado x_0, $g_0 = \nabla f(x_0)$ $H_0 = I_n$, $\varepsilon > 0$.
Sea $k \leftarrow 0$; $imeth = \text{L} - \text{BFGS}$; $ind = 0$;

Paso Iterativo : Dado x_k, $g_k = \nabla f(x_k)$, $H_k(m)$, *imeth*, *ind*:

- Si $\|g_k\| \leq \max(1, \|x_k\|)\varepsilon$ **STOP.**
  sino,
- Calcula $p_k^L = H_k(m)(-g_k)$

- Llama al procedimiento " *Selección del Método* " *y calcula la dirección* p_k.

- Obtiene x_{k+1}, usando un procedimiento de búsqueda lineal (Powell-Gsrch), satisfaciendo las condiciones de Wolfe .
- Adapta $H_{k+1}(m)$ adicionando el último par (s_k, y_k),
  siendo $s_k = x_{k+1} - x_k$, e $y_k = g_{k+1} - g_k$, bajo ciertas condiciones de estabilidad.
- END

Procedimiento para seleccionar el método y calcular la dirección p_k.

1. *Si* $imeth = \text{L} - \text{BFGS}$,

   - *Si una de las condiciones: (1), (2)ó (3) ocurre, entonces define* $imeth = PTN$, *e* $ind = 1$, ***va a 2.***

sino

- *Define* $p_k = p_k^L$; ***retorna***

2. *Si imeth* = PTN entonces

- Computa p^{CG}, and *define* $p_k = p^{CG}$.

-Adapta H_k, adicionando los pares (s_l, y_l) almacenados en las iteraciones internas de *PCG*.

- *Si ind* > 1, *redefine* $ind = 0$, $imeth = L\text{-}BFGS$,
  sino, $ind = ind + 1$; ***retorna***.

6. Experiencias Numéricas.

Presentamos experiencias que comparan los resultados del algoritmo M- LBFGS con los obtenidos con NEWTON TRUNCADO y L-BFGS resolviendo algunos problemas no lineales.
Se ha considerado m = 8 pares de vectores para obtener las matrices H_k, M_k (m).

Parámetros: $\varepsilon = 10^{-5}$, $C_1 = 10^{-4}$, $C_2 = 0.7$ (Condiciones de Wolfe)

Condición de terminación en Gradientes Conjugados:

$$\text{Si } \|r_i\| \le 10^{-6}\|g_k\|$$

Problema(n)	M-LBFGS		NTP		L-BFGS	
	iter	grad	iter	grad	iter	Grad
Dixmaani(1500)	673	1010	70	1687	975	1020
Fminsrf2(1500)	132	209	67	563	189	227
Fminsurf (1200)	146	201	19	555	194	246
Genrose (500)	1056	1256	392	2797	1070	1391
Nondquar(1200)	160	174	35	754	407	474
Eigenals (100)	31	104	17	159	259	286
Eigenbls(100)	661	1012	77	1527	960	1027
Vardim(1000)	36	37	26	53	36	37
Watson(31)	37	65	12	67	43	59
Rosenb.Ext(1000)	22	49	23	69	33	46
Big(6)	22	62	18	65	47	63
RosenChained (25)	48	62	20	87	66	72

Conclusiones:

Del análisis que hemos realizado se desprende que es conveniente enriquecer el procedimiento de L-BFGS, agregando información sobre la curvatura de la función en ciertos iterados x_k.
En particular, los resultados indican que el agregado de información local es conveniente cuando la dirección p_k de L-BFGS no es suficientemente buena ($\cos(p_k, -g_k) \approx 0$).
El pasaje a Newton Truncado en la situación previamente citada siempre aportó eficiencia al procedimiento. También en varios problemas, cuando el decrecimiento de la función no concordaba con el decrecimiento del modelo de L-BFGS, el pasaje a Newton Truncado generó una mejoría notable en el proceso iterativo.
Cuando las derivadas segundas no son calculables o son muy costosas, el algoritmo L-BFGS intercalándolo con el algoritmo de Newton Truncado en algunas iteraciones, es una alternativa de resolución mas eficiente que L-BFGS y menos costosa que Newton Truncado, ya que este último requiere muchas evaluaciones de gradientes.

Referencias.

[1] M. Al -Baali. Extra-Updates Criterion for the Limited Memory BFGS Algorithm for Large Scale Nonlinear Optimization. December (2000).Sultan Qaboos University. Department of Mathematics and Statistics.

[2] R.H. Byrd, J. Nocedal y C. Zhu. Towards a Discrete Newton Method with Memory for Large- Scale Optimization. Technical Report OTC 95/01. (February 15, 1996). Optimization Technology Center.

[3] D. C. Liu y J. Nocedal. On the limited memory BFGS method for Large Scale Optimization. Math. Programming (Series B), 45:503-528. (1989).

[4] J. L. Morales- J. Nocedal. Enriched Methods for Large-Scale Unconstrained Optimization. (January 25, 2000) .Tech. Report, Dept. of Electrical Engineering and Computer Science, Northwestern University, Evaston, USA.

[5] J.Nocedal-S.J.Wright. Numerical Optimization – (Springer series in operations Research). 1999. Springer-Verlag. New York, Inc.

[6] M.J.D. Powell. Some global convergence properties of a variable metric algorithm for minimization without exact line searches, SIAM .Publications, 1976