UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

SISTEMA DE OPTIMIZACIÓN NO LINEAL

T E S I S

QUE PARA OBTENER EL TÍTULO DE

A C T U A R I O

P R E S E N T A :

GERARDO NÚÑEZ MEDINA

MÉXICO, D. F. AGOSTO DE 1995

VNIVERSIDAD NACIONAL
AVPNOMA DE
MEXICO

M. en C. Virginia Abrín Batule
Jefe de la División de Estudios Profesionales de la
Facultad de Ciencias
P r e s e n t e

Comunicamos a usted que hemos revisado el trabajo de Tesis: Sistema de Optimización No Lineal

realizado por Núñez Medina Gerardo

con número de cuenta 8632072 6 , pasante de la carrera de Actuaría

Dicho trabajo cuenta con nuestro voto aprobatorio.

Atentamente

Director de Tesis Propietario	Dr. Pablo Barrera Sánchez
Propietario	M en C. María Elena García Alvares
Propietario	M en C. José Guerrero Grajeda
Suplente	M en C. Mariano Lozano Martínez
Suplente	M en C. José Luis Navarro Urrutia

Consejo Departamental de Matemáticas

Mat. Cesar Cuevara Bravo

Sistema de Optimización No Lineal

Gerardo Núñez Medina

Agosto, 95

A mis padres Rosa Medina M. y Sergio Núñez G.

A la memoria de mi abuela Victoria Muñoz V.

Indice

1 Introducción ... 1

1.1 El concepto de mínimo ... 2
1.2 Métodos de optimización ... 3
1.3 Longitud de paso ... 4
1.4 Modelos cuadráticos ... 7
1.5 Convergencia y estabilidad ... 7

2 Un algoritmo de búsqueda en línea ... 9

2.1 El algoritmo de búsqueda para T(μ) ... 11
2.2 Búsqueda de un mínimo local ... 18
2.3 Selección de valores de prueba ... 21
2.4 Resultados numéricos ... 24

3 Método de Newton ... 31

3.1 El algoritmo local ... 33
3.2 Propiedades de las funciones cuadráticas ... 38
3.3 Métodos de la región de confianza ... 43
3.4 Aproximación al gradiente ... 46
3.5 Aproximación a la matriz Hessiana ... 47

4 Métodos Quasi-Newton ... 51

5 Método de direcciones conjugadas ... 55

5.1 Método de gradientes conjugados ... 57

6 Sistema de optimización no lineal ... 61

6.1 Descripción general del sistema ... 61
6.2 Reconocimiento de funciones ... 62
6.2.1 Análisis léxico ... 63
6.2.2 Análisis sintáctico ... 64
6.2.3 Cambio de notación ... 67
6.3 Evaluación ... 68

Bibliografía ... 69

Capítulo 1
INTRODUCCIÓN

El problema de optimización puede ser en principio dividido en dos grandes campos, optimización sin restricciones y optimización sujeta a restricciones.

Dentro de los problemas de optimización se debe considerar otra importante división, que depende de si la función es lineal y todas las restricciones a las que puede estar sujeta también son lineales, entonces se tiene el llamado problema de optimización lineal, de otro modo el problema es de *optimización no lineal* (ver cuadro abajo). Este trabajo tratará enteramente con problemas de optimización no lineal sin restricciones.

El problema de optimización consiste en encontrar aquellos puntos en el dominio de una función (con cualquier número de variables independientes), que al ser transformados por la función tomen valores extremos, que pueden ser locales o globales, de esta forma, los problemas de optimización se subdividen de acuerdo al siguiente cuadro:

$$\textit{Optimización} \begin{cases} \textit{Sin restricciones} \begin{cases} \textit{Lineal} \begin{cases} \textit{Local} \\ \textit{Global} \end{cases} \\ \textit{No} - \textit{Lineal} \begin{cases} \textit{Local} \\ \textit{Global} \end{cases} \end{cases} \\ \textit{Con restricciones} \begin{cases} \textit{Lineal} \begin{cases} \textit{Local} \\ \textit{Global} \end{cases} \\ \textit{No} - \textit{Lineal} \begin{cases} \textit{Local} \\ \textit{Global} \end{cases} \end{cases} \end{cases}$$

Un valor extremo es un punto en el dominio de la función, el cual al ser transformado por f alcanza un valor máximo o un valor mínimo, por lo que es lógico considerar que se tienen dos problemas distintos: el de encontrar un valor máximo o de maximización y el de encontrar un valor mínimo o de minimización, pero existe la siguiente relación entre ambos problemas:

$$\min_x f(x) = -\max_x -f(x)$$

Lo que permite tratar sólo problemas de minimización, dejando de lado los problemas de maximización. El problema central en este trabajo será:

$$\min_x f(x)$$

para una función f no lineal, sin restricción alguna y con $x \in R^n$.

1.1 El Concepto de Mínimo

Si existe un punto x^* tal, que el valor de la función en este punto sea al menos tan pequeño como cualquier otro valor de la función en un punto x vecino arbitrario, se dice que la función f posee un mínimo en x^*, esto al menos en una vecindad de x^*. Si el valor de la función evaluada en x^* es estrictamente más pequeño que el valor de la función evaluada en x, para todo punto x en una vecindad de x^*, x^* es llamado mínimo fuerte, por lo que se pueden distinguir varios tipos de mínimos.

Definición 1.1 *Se dice que un punto x^* es un mínimo fuerte para una función f, si existe un escalar $\delta > 0$ tal que $f(x^*) < f(x^* + \Delta x)$ para todo Δx tal que $\|\Delta x\| \leq \delta$.*

Un mínimo débil será; a diferencia de un mínimo fuerte aquel punto para el cual el valor de la función decrece en algunas direcciones, pero permanece igual en las demás.

Definición 1.2 *Se dice que un punto x^* es un mínimo débil para una función f, si este no es un mínimo fuerte y existe un escalar $\delta > 0$ tal que $f(x^*) \leq f(x^* + \Delta x)$ para todo Δx tal que $\|\Delta x\| \leq \delta$.*

Si la definición 1.1 se cumple para $\delta = \infty$ entonces se dice que el punto mínimo es global, *lo que además implica que es único*, por lo que se debe considerar al mínimo global como el verdadero valor mínimo de una función, esto es, el valor más pequeño que una función puede tomar en todo su dominio, mientras que un mínimo local es el valor más pequeño que una función puede tomar dentro de una vecindad, de manera que podemos encontrar más de un mínimo local en una función.

En realidad no existe forma de garantizar que una vez encontrado un mínimo éste sea global, salvo con un amplio conocimiento de la función, respecto de su continuidad, suavidad e información de derivadas de grado superior, por lo que en general se tratará de encontrar mínimos locales.

Existen funciones cuyo valor mínimo es ∞, o que alcanzan su valor mínimo en un punto en el dominio con coordenada ∞, este tipo de soluciones rara vez modelan situaciones de interés cuya aplicación es de utilidad práctica. Al intentar resolver el problema de optimización uno supone que la solución x^* existe, es única y puede ser encontrada por el método escogido, sin embargo esta situación ideal no siempre ocurre, primero, porque la solución x^* puede no existir, y si existe, puede no ser única, además de que el método usado puede no converger a la solución, por lo que se debe contar con condiciones que garanticen la existencia de al menos un mínimo, y condiciones que garanticen la convergencia del método para el problema dado.

1.2 Métodos de Optimización

Dada una función f dos veces continuamente diferenciable (i.e. $f \in C^2$), las siguientes condiciones garantizan que x^* es un punto mínimo.

$$\nabla f(x^*) = 0 \tag{1.1}$$

$$\nabla^2 f(x^*) > 0 \tag{1.2}$$

Otros tipos de puntos además de los mínimos satisfacen la condición (1.1) (máximos y puntos silla por ejemplo) y estos puntos son conocidos como puntos estacionarios o críticos. En suma la condición de que $\nabla f(x) = 0$ es una condición necesaria y suficiente para la existencia de puntos críticos y necesaria pero no suficiente para la existencia de puntos mínimos.

El problema a resolver es el siguiente:

$$\min_{x \in R^n} f(x) \tag{1.3}$$

donde f es una función de $R^n \to R$, acotada inferiormente y que tiene al menos derivadas de segundo orden continuas y supóngase que (1.3) tiene una solución local en x^* en la que $\nabla f(x) = 0$ y $\nabla^2 f(x) \geq 0$.

Un vector $p \in R^n$ es una dirección de descenso para una función $f : R^n \to R$ en un punto $x \in R^n$ si existe una constante $\bar{\alpha} > 0$ tal que

$$f(x + \alpha p) < f(x),\ \alpha \in (0, \bar{\alpha}].$$

Para funciones diferenciables, una manera más fácil es pedir que

$$\nabla f(x)^t p < 0. \tag{1.4}$$

Un *método de descenso*, es tal que a partir de un punto inicial x_0 genera una sucesión de iteraciones de la forma $x_{k+1} = x_k + \alpha_k p_k$, donde p_k es una dirección de descenso y $\alpha_k > 0$. La sucesión debe satisfacer:

1. $f(x_{k+1}) < f(x_k)$.
2. $\lim_{k \to \infty} x_k = x^*$.

La diferencia entre los distintos métodos de descenso está en la forma como se elige la dirección p_k, la que depende de la información que se tenga en el punto x_k. Algunas de las posibilidades de elección son:

1. $p_k = -\nabla f(x_k)$.
2. $p_k = -H_k \nabla f(x_k)$, con H_k es una matriz positiva definida de nxn.
3. $p_k = [\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$.

El método que corresponde a la elección de (1) es conocido como el *método de descenso más rápido* debido a que la dirección opuesta al gradiente es la mejor dirección de descenso de la función en una vecindad del punto. La elección (3) es el conocido *método de Newton* y es la más usada siempre que es posible, ya que tiene una rápida convergencia local. Por último, la elección de (2) es la que da origen a los *métodos de tipo Newton*, pues la matriz H_k aproxima a la inversa de la Hessiana.

1.3 Longitud de Paso

Dado un punto x_k, y una dirección de descenso p_k, la idea básica del método de búsqueda en línea es calcular un parámetro $\alpha_k > 0$ tal que en la siguiente iteración $x_{k+1} = x_k + \alpha_k p_k$ el valor de la función sea mínimo. El parámetro α_k determina el tamaño del paso en la dirección de descenso p_k a partir del punto x_k.

Un algoritmo de búsqueda en línea examina los puntos a lo largo de $\{x + \alpha p : \alpha \geq 0\}$ en busca de un α tal que $f(x + \alpha p) < f(x)$. Si p es una dirección de descenso entonces tal α existe. De hecho, el mínimo α^* de la función univariada

$$\varphi(\alpha) \equiv f(x + \alpha p), \ \alpha \geq 0. \tag{1.5}$$

es un α sobre la línea. El proceso de búsqueda termina hasta después de que un mínimo exacto es encontrado. Encontrar un mínimo exacto a lo largo de la línea generalmente no lleva a grandes reducciones de la función, ya que mayores progresos pueden ser alcanzados tomando reducciones más pequeñas en sucesivas búsquedas, explorando otras direcciones. Para lograr esto se deben desarrollar reglas que determinen cuando detener la búsqueda.

Dados los parámetros $\mu \in (0, \frac{1}{2})$ y $\eta \in (\mu, 1)$, y una dirección de descenso $p \in R^n$, se define el conjunto $T(\mu)$ de longitudes de $\alpha > 0$ como el conjunto que cumple con

$$f(x + \alpha p) \leq f(x) + \alpha \mu \nabla f(x)^t p,$$

y

$$\left|\nabla f(x + \alpha p)^t p\right| \leq \eta \left|\nabla f(x)^t p\right|.$$

En otras palabras, el conjunto $T(\mu)$ especifica las reglas de parada. En términos de la función φ definida por (1.5), un α pertenece a $T(\mu)$ si y sólo si

$$\varphi(\alpha) \leq \varphi(0) + \alpha \mu \varphi'(0), \tag{1.6}$$

y

$$\left|\varphi'(\alpha)\right| \leq \eta \left|\varphi'(0)\right|. \tag{1.7}$$

Si la magnitud de α no es *suficientemente* pequeña, entonces la primera condición de $T(\mu)$ fuerza un decrecimiento suficiente de la función. Sin embargo, (1.6) permite elecciones arbitrariamente pequeñas de $\alpha > 0$, por lo que esta condición no es suficiente para garantizar convergencia. La segunda condición evita elecciones arbitrariamente pequeñas de $\alpha > 0$.

Teorema 1.1 *Sea $f : R^n \to R$ continuamente diferenciable y acotada inferiormente en R^n, suponga que el punto inicial x_0 es tal que ∇f es uniformemente continuo en el conjunto de nivel*

$$\Omega \equiv \{x \in R^n : f(x) \leq f(x_0)\}.$$

Si la sucesión $\{x_k\}$ es definida por $x_{k+1} = x_k + \alpha_k p_k$ donde $\nabla f(x_k)^t p_k < 0$ y α_k define un paso en $T(\mu)$ entonces

$$\lim_{k \to +\infty} \left(\frac{\nabla f(x_k)^t p_k}{\|p_k\|} \right) = 0. \tag{1.8}$$

Prueba 1.1 *Ya que $\nabla f(x_k)^t p_k < 0$ y f es acotada inferiormente, la sucesión $\{x_k\}$ está bien definida en Ω. Además, $\{f(x_k)\}$ es decreciente y por lo tanto converge. La demostración es por contradicción. Si (1.8) no ocurre, entonces existe un $\varepsilon > 0$ y una sucesión indexada por un conjunto K tal que*

$$-\frac{\nabla f(x_k)^t p_k}{\|p_k\|} \geq \varepsilon, \; k \in K.$$

La primera condición de las reglas de parada $T(\mu)$ muestran que

$$f(x_k) - f(x_{k+1}) \geq \mu \alpha_k \|p_k\| \left(-\frac{\nabla f(x_k)^t p_k}{\|p_k\|} \right) \geq \mu \alpha_k \|p_k\| \varepsilon, \; k \in K,$$

y ya que $\{f(x_k)\}$ es una sucesión convergente, $\{\alpha_k p_k : k \in K\}$ converge a cero. La segunda condición de las reglas de parada $T(\mu)$ nos da la siguiente desigualdad

$$(1 - \eta)(-\nabla f(x_k)^t p_k) \leq (\nabla f(x_k + \alpha_k p_k) - \nabla f(x_k))^t p_k, \; k \geq 0,$$

y por lo tanto

$$\varepsilon \leq -\frac{\nabla f(x_k)^t p_k}{\|p_k\|} \leq \left(\frac{1}{1 - \eta} \right) \|\nabla f(x_k + \alpha_k p_k) - \nabla f(x_k)\|, \; k \in K.$$

Sin embargo, ya que hemos mostrado que $\{\alpha_k p_k : k \in K\}$ converge a cero, esto contradice la continuidad uniforme de ∇f en Ω.□

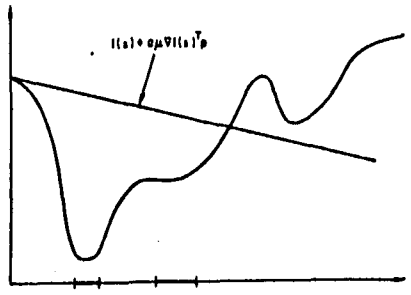


Figura 1

La función φ, que se muestra en la figura 1, describe al conjunto $T(\mu)$, que consta de los intervalos I_1 y I_2.

Si suponemos $\mu < \frac{1}{2}$ y φ cuadrática con $\varphi'(0) < 0$ y $\varphi''(0) > 0$, entonces el mínimo global α^* de φ satisface

$$\varphi(\alpha^*) = \varphi(0) + \frac{1}{2}\alpha^*\varphi'(0)$$

y por lo tanto α^* satisface (1.6) si y sólo si $\mu \leq \frac{1}{2}$. La restricción de $\mu < \frac{1}{2}$ obliga a α a tomar el valor de $\alpha = 1$ en las últimas iteraciones del método de Newton.

La restricción $\mu < \eta$ garantiza que bajo condiciones razonables $T(\mu)$ contiene un intervalo no trivial.

Los algoritmos para seleccionar la longitud de paso α están basados generalmente en la estrategia de minimización de modelos univariados cuadráticos o cúbicos para φ, definidos por la interpolación de la función y la primera derivada para valores de prueba de α.

Un algoritmo de búsqueda en línea seguro, requiere que se determine y actualice un intervalo de incertidumbre I el cual contenga puntos en $T(\mu)$. El proceso de actualización, debe garantizar que la longitud de I tienda a cero y que I esté contenido en $T(\mu)$.

En general los métodos de búsqueda en línea tiene el siguiente esquema:

ALGORITMO

Dada una dirección de búsqueda p_k.

1. Encontrar un α_k que minimice $f(x_k + \alpha p_k)$ con respecto a α.

2. Hacer $x_{k+1} = x_k + \alpha_k p_k$.

1.4 Modelos Cuadráticos

Algunas características de muchos métodos de optimización se obtienen del hecho de que trabajan bien o son incluso exactos si son aplicados a una función cuadrática, estos métodos pueden ser aplicados recursivamente a funciones más generales. No hay duda de que esta aproximación es muy útil en la práctica y de que existe una relación entre ésta y la eficiencia y rapidez de convergencia del método.

A continuación se listan algunas características de los modelos cuadráticos:

- Una función cuadrática es la función no lineal más simple con segundas derivadas continuas, que tiene puntos estacionarios bien determinados.
- Toda función con segundas derivadas continuas se comporta como una función cuadrática sobre una región suficientemente pequeña. Este hecho es central para el desarrollo de algoritmos de minimización, puesto que un algoritmo que alcanza la vecindad del mínimo de una función general no lineal y minimiza una función cuadrática eficientemente, convergerá a la solución rápidamente.
- Aún en puntos lejanos al mínimo parece preferible usar la información cuadrática, ya que ésta es más efectiva que la información lineal para pronosticar direcciones a lo largo de las cuales se puede progresar de manera significativa. Esta se tiene a partir de que la aproximación de la serie de Taylor alrededor de un punto arbitrario x_k, tomando términos cuadráticos, dará una más exacta aproximación en vecindades más grandes de x_k, que los que tomaría la serie en términos de una aproximación lineal.
- Los métodos basados en aproximaciones cuadráticas pueden permanecer invariantes bajo transformaciones lineales.

1.5 Convergencia y Estabilidad

Hay dos factores de fundamental importancia dentro del diseño de algoritmos de minimización, éstos son: la estabilidad y la velocidad de convergencia.

1.5.1 Estabilidad

El diseño de los mejores algoritmos de minimización está asociado a la idea de minimización eficaz de una función cuadrática, con matriz Hessiana positiva definida.

En una vecindad cercana al mínimo muchas funciones pueden ser consideradas como cuadráticas. Sin embargo, un algoritmo debe garantizar ser capaz de acercarse a un mínimo si las propiedades teóricas son cumplidas. Un algoritmo que es capaz de probar convergencia a un punto mínimo (o en algunos casos a un punto estacionario), sin importar el punto inicial ni la presencia de puntos extraños, se dice que es estable o que exhibe convergencia global. En optimización no lineal, la estabilidad es siempre asociada con esquemas iterativos que garanticen una reducción suficiente del valor de la función en cada iteración a no ser que el mínimo halla sido encontrado para el grado de exactitud prescrito.

1.5.2 Convergencia

Dado un método estable, es importante tener un indicador teórico de la velocidad de convergencia, ya que ésta puede ser tan lenta como para no proporcionar al método valor práctico, mientras que por otra parte, es importante que un algoritmo llegue a la vecindad del mínimo en tan pocas iteraciones como sea posible, de ahí el interés por medir teoricamente la velocidad de convergencia. Esto es debido a que las funciones pueden ser muy distintas una de la otra y es sólo cerca del mínimo que tienen un comportamiento (cuadrático) común.

Cuando la vecindad de un mínimo es alcanzada, es posible medir el error definiendo una nueva variable como $h_k = x_k - x^*$, y tener una idea sobre la convergencia del método a partir de la convergencia de h.

Si definen $\|h_{k+1}\| / \|h_k\| \to a$, entonces se dice que si $a \geq 0$ la velocidad de convergencia es lineal o de primer orden, y si $a = 0$ se dice que la velocidad es superlineal.

La convergencia es una característica importante de un algoritmo, ya que es requerida para la terminación de las iteraciones. Lo más útil para el usuario sería garantizar que $f(x_k) - f(x_*) \leq \epsilon$ o $\|x_k - x_*\| \leq \epsilon_i$ donde los parámetros ϵ y ϵ_i son proporcionados por el usuario. Desafortunadamente esto no es aplicable, ya que ello requiere un conocimiento de la solución. Una mejor prueba es $\|g(x_k)\| \leq \epsilon$ la cual ha sido usada en ocasiones, sin embargo tiene la desventaja que no es fácil de usar debido a las dificultades para fijar la magnitud de ϵ y además no trabaja bien en problemas mal condicionados.

Existe una prueba basada en $\|x_k - x_{k+1}\| \leq \epsilon$ o $f(x_k) - f(x_{k+1}) \leq \epsilon$ y normalmente trabaja bien. Una consideración adicional la cual se aplica cuando se dispone de segundas derivadas, es que la prueba puede prevenir la terminación en un punto silla. También es útil en general terminar la prueba cuando un cierto número de iteraciones máximo ha sido ya ejecutado.

Capítulo 2
UN ALGORITMO DE BÚSQUEDA EN LÍNEA

Sea $\varphi : R \to R$ una función continua y diferenciable definida de $[0, \infty)$ con $\varphi'(0) < 0$, y constantes μ y η en $(0, 1)$, estamos interesados en encontrar un $\alpha > 0$ tal que

$$\varphi(\alpha) \leq \varphi(0) + \mu\varphi'(0)\alpha \tag{2.1}$$

y

$$\left|\varphi'(\alpha)\right| \leq \eta \left|\varphi'(0)\right|. \tag{2.2}$$

El desarrollo de un procedimiento de búsqueda que satisfaga estas condiciones es un ingrediente indispensable para el método de búsqueda en línea. Este capítulo describe este procedimiento de búsqueda y la teoría de convergencia asociada a éste.

En el método de búsqueda en línea damos una función $f : R^n \to R$ continua y diferenciable, y una dirección de descenso p para f en un punto $x \in R^n$. Así, si

$$\varphi(\alpha) \equiv f(x + \alpha p),\ \alpha \geq 0 \tag{2.3}$$

entonces (2.1) y (2.2) definen un paso aceptable. Si α no es muy pequeña, la condición (2.1) forza un *decrecimiento suficiente* en el valor de la función. Sin embargo, esta condición no es suficiente para garantizar convergencia, debido a que permite elegir valores de $\alpha > 0$ arbitrariamente pequeños. La condición (2.2) descarta valores de α arbitrariamente pequeños y normalmente garantiza que α está cerca de un mínimo local de φ. La condición (2.2) es conocida como la condición de curvatura ya que implica que

$$\varphi'(\alpha) - \varphi'(0) > (1 - \eta)\left|\varphi'(0)\right|,$$

y así la curvatura promedio de φ en $(0, \alpha)$ es positiva. La condición de curvatura (2.2) es particularmente importante para los métodos Quasi-Newton debido a que garantiza que una actualización positiva definida Quasi-Newton es posible.

Como motivación final para la solución de (2.1) y (2.2), mencionamos que si el paso satisface estas condiciones, entonces el método de búsqueda en línea es convergente para una elección razonable de la dirección.

En muchas situaciones prácticas es importante imponer condiciones adicionales sobre α. En particular, es natural pedir que α cumpla con los límites

$$0 \leq \alpha_{\min} \leq \alpha \leq \alpha_{\max}. \tag{2.4}$$

La principal razón para pedir un límite inferior $\alpha_{\min}$ es para terminar la iteración, mientras que el límite superior $\alpha_{\max}$ es necesario cuando la búsqueda es utilizada en problemas de optimización lineal sujeta a restricciones o cuando la función φ no está acotada inferiormente. Un problema no acotado, puede ser aproximado aceptando cualquier α en $[\alpha_{\min}, \alpha_{\max}]$ tal que $\varphi(\alpha) \leq \varphi_{\min}$, donde $\varphi_{\min} < \varphi(0)$ es un límite inferior especificado por el usuario de la búsqueda. En este caso

$$\alpha_{\max} = \frac{1}{\mu}\left(\frac{\varphi(0) - \varphi_{\min}}{-\varphi'(0)}\right) \tag{2.5}$$

es una elección razonable debido a que si $\alpha_{\max}$ satisface la condición (2.1) de decrecimiento suficiente entonces $\varphi(\alpha_{\max}) \leq \varphi_{\min}$. Por otra parte, si $\alpha_{\max}$ no satisface la condición de decrecimiento suficiente, entonces mostraremos que es posible determinar un α aceptable.

El principal problema que consideraremos es encontrar un α aceptable en el sentido que α pertenezca al conjunto

$$T(\mu) \equiv \{\alpha > 0 : \varphi(\alpha) \leq \varphi(0) + \alpha\mu\varphi'(0),\ \left|\varphi'(\alpha)\right| \leq \mu\left|\varphi'(0)\right|\}. \tag{2.6}$$

Para expresar este resultado en términos de $T(\mu)$ debemos aclarar que el algoritmo de búsqueda es independiente de η; el parámetro η es utilizado sólo para la prueba de terminación del algoritmo. Otra ventaja de expresar el resultado en términos de $T(\mu)$ es que $T(\mu)$ es generalmente no vacío. Por ejemplo, $T(\mu)$ es no vacío cuando φ no está acotada inferiormente.

Fletcher sugiere que es posible calcular una sucesión de intervalos que contengan puntos que satisfagan (2.1) y (2.2), esta sugerencia nos lleva a los algoritmos desarrollados por Al-Baali y Fletcher así como de Moré y Thuente. Aquí proporcionaremos una análisis de convergencia, detalles de implementación y resultados numéricos del algoritmo de Moré y Thuente.

El algoritmo para $T(\mu)$ es definido en la siguiente sección. Mostraremos que el algoritmo de búsqueda produce una sucesión de iteraciones que convergen a un punto en $T(\mu)$ y que, excepto para casos patológicos, el algoritmo produce una sucesión finita $\alpha_0, \alpha_1, \ldots, \alpha_m$ de valores de prueba en $[\alpha_{\min}, \alpha_{\max}]$, donde $\alpha_m \in T(\mu)$ o es uno de los límites. La terminación en uno de los puntos límite puede ser evitada con una selección adecuada de los límites. Por ejemplo, si $\alpha_{\min} = 0$ y $\alpha_{\max}$ es definida por (2.5), entonces α_m pertenece a $T(\mu)$ o $\varphi(\alpha_m) \leq \varphi_{\min}$.

Los resultados de la siguiente sección mostrarán que el algoritmo de búsqueda puede ser usado para encontrar un α que satisfaga (2.1) y (2.2) cuando $\mu \leq \eta$. Un

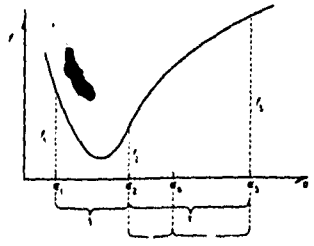


Figura 1

resultado para $\eta \in (0, \mu)$ requiere de suposiciones adicionales debido a que quizá no exista un α que satisfaga (2.1) y (2.2) incluso si φ es acotada inferiormente. Adelante mostraremos que si el algoritmo de búsqueda genera una iteración de α_k que satisfacen la condición de decrecimiento suficiente y $\varphi'(\alpha_k) > 0$, entonces el algoritmo de búsqueda termina en un α_k que satisface (2.1) y (2.2).

Dado α_0 en $[\alpha_{\min}, \alpha_{\max}]$, el algoritmo de búsqueda genera una sucesión de intervalos anidados $\{I_k\}$ y una sucesión de iteraciones $\alpha_k \in I_k \cap [\alpha_{\min}, \alpha_{\max}]$. La sección (2.3) describe la elección especifica para los valores de prueba α_k que son usados en nuestro algoritmo. Los resultados numéricos indican que estas elecciones llevan a una rápida terminación.

La sección (2.4) describe un conjunto de problemas de prueba y resultados numéricos para el procedimiento de búsqueda. Las primeras tres funciones en el conjunto de prueba tienen regiones de concavidad, mientras las últimas tres funciones son convexas. En todos los casos las funciones tienen un único mínimo. El énfasis en los resultados numéricos es para explicar las características cualitativas del algoritmo para un amplio rango de valores de μ y η.

2.1 El Algoritmo de Búsqueda para T(μ)

En esta sección presentaremos el algoritmo de búsqueda para determinar un α en $T(\mu)$. Asumimos que φ es continua y diferenciable en $[0, \alpha_{\max}]$ con $\varphi'(0) < 0$. El algoritmo de búsqueda en línea supone que $\mu < \frac{1}{2}$, debido a que si es cuadrática y $\varphi'(0) < 0$ y $\varphi''(0) > 0$, entonces el mínimo global α^* de φ satisface

$$\varphi(\alpha^*) = \varphi(0) + \frac{1}{2}\alpha^*\varphi'(0),$$

y así α^* satisface (2.1) sólo si $\mu \leq \frac{1}{2}$. La restricción $\mu < \frac{1}{2}$ también nos lleva a $\alpha = 1$ como un valor aceptable a la larga para los métodos de Newton y Quasi-Newton.

Dado α_0 en $[\alpha_{\min}, \alpha_{\max}]$, el algoritmo de búsqueda genera una sucesión de intervalos anidados $\{I_k\}$ y una sucesión de iteraciones $\alpha_k \in I_k \cap [\alpha_{\min}, \alpha_{\max}]$ de acuerdo al

siguiente procedimiento.

ALGORITMO DE BÚSQUEDA

Hacer $I_0 = [0, \infty]$.

Para $k = 0, 1, \ldots$

1. Elegir $\alpha_k \in I_k \cap [\alpha_{\min}, \alpha_{\max}]$.
2. Prueba de convergencia.
3. Actualizar el intervalo I_k.

En este algoritmo elegir α_k protejida se refiere a las reglas que forzan la convergencia del algoritmo. Por el momento suponemos que la elección ha sido hecha y discutiremos la actualización de I_k.

El objetivo del proceso de actualización para los intervalos I_k es identificar y generar un intervalo I_k tal que $T(\mu) \cap I_k$ sea no vacío, y entonces refinar el intervalo de tal forma que $T(\mu) \cap I_k$ permanezca no vacío. Ahora especificaremos las condiciones de los puntos finales de un intervalo I que garantizan que I tiene una intersección no vacía con $T(\mu)$. Las condiciones sobre los puntos finales α_l y α_u están especificadas en términos de la función auxiliar ψ definida por

$$\psi(\alpha) \equiv \varphi(\alpha) - \varphi(0) - \mu\varphi'(0)\alpha.$$

Suponemos que $\alpha_l \neq \alpha_u$ pero no supondremos que α_l y α_u están ordenados.

Teorema 2.1 *Sea I un intervalo cerrado con puntos finales α_l y α_u. Si los puntos finales satisfacen*

$$\psi(\alpha_l) \leq \psi(\alpha_u),\ \psi(\alpha_l) \leq 0,\ \psi'(\alpha_l)(\alpha_u - \alpha_l) < 0, \tag{2.7}$$

entonces hay un α^* en I con $\psi(\alpha^*) \leq \psi(\alpha_l)$ y $\psi'(\alpha^*) = 0$. En particular, $\alpha^* \in (T(\mu) \cap I)$.

Prueba 2.1 *Suponga que* $\alpha_u > \alpha_l$; *la prueba en el otro caso es similar. Defina*

$$\alpha_m = \sup\{\alpha \in [\alpha_l, \alpha_u] : \psi(\beta) \leq 0,\ \beta \in [\alpha_l, \alpha]\}.$$

Entonces $\alpha_m > \alpha_l$, *debido a* $\psi'(\alpha_l) < 0$. *Primero pedimos que* $\psi(\alpha_m) \geq \psi(\alpha_l)$. *El supuesto sobre* α_u *muestra que es cierto el caso cuando* $\alpha_m = \alpha_u$. *Esto también se sostiene si* $\alpha_m < \alpha_u$, *debido en este caso a que la definición de* α_m *implica que* $\psi(\alpha_m) = 0$, *y así* $0 = \psi(\alpha_m) \geq \psi(\alpha_l)$.
Definimos α^* *como el mínimo global de* ψ *en* $[\alpha_l, \alpha_m]$. *Afirmamos que* $\alpha^* \in T(\mu)$. *El mínimo global no puede ser alcanzado en* α_m. *Ya que,* α^* *está en el interior de* $[\alpha_l, \alpha_m]$. *En particular,* $\psi'(\alpha^*) = 0$, *y así* $|\varphi'(\alpha^*)| = \mu\,|\varphi'(0)|$. *También sabemos que* α^* *satisface (2.1), debido a que* $\psi(\alpha) \leq 0$ *para toda* α *en* $[\alpha_l, \alpha_m]$. *Ya que,* $\alpha^* \in T(\mu)$, *como deseábamos.*□

El teorema anterior proporciona la motivación para el algoritmo de búsqueda mostrando que si los puntos finales de I satisfacen (2.7), entonces ψ tiene un mínimo α^* en el interior de I además, que α^* pertenece a $T(\mu)$. Así el algoritmo puede ser visto como un procedimiento para localizar un mínimo de ψ.

La suposición (2.7) puede ser expresada diciendo que α_l es el punto final con el valor más pequeño de ψ, que α_l satisface la condición de decrecimiento suficiente (2.1), y que $\alpha_u - \alpha_l$ es una dirección de descenso para ψ en α_l y $\psi(\alpha) < \psi(\alpha_l)$ para todo α en I suficientemente cercana α_l. En particular, esta última suposición garantiza que ψ puede decrecer buscando cerca de α_l.

Ahora describiremos un algoritmo para actualizar el intervalo I, y entonces mostraremos cómo usaremos este algoritmo para obtener un intervalo que satisfaga las condiciones del teorema anterior.

ALGORITMO DE ACTUALIZACIÓN

Dado un valor de prueba α_t en I, los puntos finales α_l^+ y α_u^+ del intervalo actualizado I_+ son determinados como sigue:

Caso U1 : Si $\psi(\alpha_t) > \psi(\alpha_l)$, entonces $\alpha_l^+ = \alpha_l$ y $\alpha_u^+ = \alpha_t$.

Caso U2 : Si $\psi(\alpha_t) \leq \psi(\alpha_l)$ y $\psi'(\alpha_t)(\alpha_l - \alpha_t) > 0$, entonces $\alpha_l^+ = \alpha_t$ y $\alpha_u^+ = \alpha_u$.

Caso U3 : Si $\psi(\alpha_t) \leq \psi(\alpha_l)$ y $\psi'(\alpha_t)(\alpha_l - \alpha_t) < 0$, entonces $\alpha_l^+ = \alpha_t$ y $\alpha_u^+ = \alpha_l$.

Es directo mostrar que si los puntos finales α_l y α_u satisfacen (2.7), entonces los puntos finales actualizados α_l^+ y α_u^+ también satisfacen (2.7), a menos que $\psi'(\alpha_t) = 0$ y $\psi(\alpha_t) \leq \psi(\alpha_l)$. Por supuesto, en este caso no es necesario actualizar I debido a que α_t pertenece a $T(\mu)$.

Al-Baali y Fletcher presentan dos formas de actualización. El objetivo del esquema $S1$ consiste en identificar un punto que satisfaga (2.1) y $\varphi'(\alpha) \geq \eta\varphi'(0)$, mientras el esquema $S2$ busca un punto que satisfaga (2.1) y (2.2). En el esquema $S2$ los puntos finales α_l^+ y α_u^+ son determinados como sigue:

Si $\psi(\alpha_t) > 0$ O Si $\varphi(\alpha_t) > \varphi(\alpha_l)$, entonces

$\alpha_l^+ = \alpha_l$ y $\alpha_u^+ = \alpha_t$,

Si No, Si $\varphi'(\alpha_t)(\alpha_l - \alpha_u) > 0$, entonces

$\alpha_l^+ = \alpha_t$ y $\alpha_u^+ = \alpha_u$,

Si No $\alpha_l^+ = \alpha_t$ y $\alpha_u^+ = \alpha_l$.

Los dos algoritmos de actualización producen las mismas iteraciones mientras

$$\psi(\alpha_t) > 0, \ O \ \psi(\alpha_t) \leq 0, \ \psi(\alpha_t) \leq \psi(\alpha_l),$$

pero difieren en el tratamiento de la situación cuando

$$\psi(\alpha_t) \leq 0, \ \psi(\alpha_t) > \psi(\alpha_l), \ \psi(\alpha_t) < \psi(\alpha_l), \ \alpha_l < \alpha_t < \alpha_u.$$

En este caso, el algoritmo de actualización elige $I_+ = [\alpha_t, \alpha_u]$, mientras que el esquema $S2$ elige $[\alpha_t, \alpha_u]$ si $\varphi'(\alpha_t) < 0$. Nuestro algoritmo parece ser preferible en esta situación debido a que el intervalo I_+ contiene un punto aceptable, mientras el intervalo generado por la forma $S2$ no garantiza contener un punto aceptable.

Mostraremos ahora como el algoritmo de actualización puede ser usado para determinar un intervalo I en $[0, \alpha_{\max}]$ con puntos finales que satisfacen (2.7). Inicialmente $\alpha_l = 0$ y $\alpha_u = \infty$. Considere cualquier valor de prueba α_t en $[\alpha_{\min}, \alpha_{\max}]$. Si los casos $U1$ o $U3$ ocurren, entonces habremos determinado un intervalo con puntos finales α_l y α_u que satisfacen las condiciones del teorema. De otra forma el caso $U2$ se sostiene, y entonces repetir el proceso para alguna α_l^+ en $[\alpha_t, \alpha_{\max}]$. Continuamos generando valores de prueba en $[\alpha_l, \alpha_{\max}]$ mientras el caso $U2$ se presente, pero requerimos que eventualmente $\alpha_{\max}$ sea usada como un valor de prueba. Esto se logra eligiendo

$$\alpha_t^+ \in [\min\{\delta_{\max}\alpha_t; \alpha_{\max}\}, \alpha_{\max}] \tag{2.8}$$

para algún factor $\delta_{\max} > 1$. En nuestra implementación usamos

$$\alpha_t^+ = \min\{\alpha_t + \delta(\alpha_t - \alpha_l), \alpha_{\max}\},\ \delta \in [1.1, 4],$$

diversos argumentos muestran que (2.8) se sostiene con $\delta_{\max} = 1.1$.

Ya que inicialmente $\alpha_l = 0$ y $\psi(0) = 0$, la sucesión $\alpha_0, \alpha_1, \ldots$ de valores de prueba es creciente con

$$\psi(\alpha_k) \leq 0 \text{ y } \psi'(\alpha_k) < 0,\ k = 0, 1, \ldots \tag{2.9}$$

mientras el caso $U2$ se presente. El algoritmo de búsqueda termina en $\alpha_{\max}$ si $\psi(\alpha_{\max}) \leq 0$ y $\psi'(\alpha_{\max}) < 0$. Este es un criterio razonable de terminación debido a que el teorema (2.1) muestra que cuando estas condiciones no se sostienen hay un $\alpha^* \in T(\mu)$ con $\alpha^* \leq \alpha_{\max}$. Así, después de un número finito de valores prueba, ambos algoritmos de búsqueda terminan en $\alpha_{\max}$, o el algoritmo de búsqueda genera un intervalo con puntos finales que satisfacen las condiciones (2.7).

Dado un intervalo que satisface las condiciones (2.7), el algoritmo de búsqueda usa el algoritmo de actualización para refinar I. Afirmamos que si el algoritmo no genera un intervalo I en $[\alpha_{\min}, \alpha_{\max}]$ que satisfaga las condiciones (2.7), entonces la sucesión $\{\alpha_k\}$ de valores prueba es decreciente con

$$\psi(\alpha_k) > 0 \text{ o } \psi'(\alpha_k) \geq 0,\ k = 0, 1, \ldots \tag{2.10}$$

Esta condición es establecida al considerar los tres casos del algoritmo de actualización. Si usamos un α_t con $\psi(\alpha_t) \leq 0$ y $\psi'(\alpha_t) < 0$, y el caso $U2$ o $U3$ se presente, entonces el algoritmo de actualización muestra que el intervalo I_+ está ubicado a la derecha de α_t. Ya que $\alpha_t \geq \alpha_{\min}$, el intervalo I contiene a $[\alpha_{\min}, \alpha_{\max}]$. Si el caso $U1$ ocurre, entonces $\psi(\alpha_l) < 0$, y así $\alpha_l \geq \alpha_{\min}$. Ya que, el intervalo actualizado contiene a $[\alpha_{\min}, \alpha_{\max}]$.

Forzamos al algoritmo de búsqueda a usar $\alpha_{\min}$ como un valor de prueba cuando (2.10) ocurre y $\alpha_{\min} > 0$. Esto se logra eligiendo

$$\alpha_t^+ \in [\alpha_{\min}, \max\{\delta_{\min}\alpha_t, \alpha_{\min}\}] \tag{2.11}$$

para algún factor $\delta_{\min} < 1$. En nuestra implementación (2.11) se cumple con $\delta_{\min} = \frac{7}{12}$.

El algoritmo de búsqueda termina en $\alpha_{\min}$ si $\psi(\alpha_{\min}) > 0$ o $\psi'(\alpha_{\min}) \geq 0$. Este es un criterio de finalización razonable debido a que el teorema (2.1) muestra que hay un $\alpha^* \in T(\mu)$ con $\alpha^* \leq \alpha_{\min}$ cuando las condiciones se cumplen. Así, después de un número finito de valores de prueba, ambos algoritmos de búsqueda terminan en $\alpha_{\min}$, o generan un intervalo I en $[\alpha_{\min}, \alpha_{\max}]$ con puntos finales que satisfacen (2.7).

Los requisitos (2.8) y (2.11) son dos reglas de protección. Observe que (2.8) es impuesto sólo cuando se cumple (2.9), mientras (2.11) es usado cuando se cumple

(2.10). Si el algoritmo de búsqueda se genera un intervalo I en $[\alpha_{\min}, \alpha_{\max}]$, entonces necesitamos una tercera regla que garantice que el α_t elegido fuerce la longitud de I a cero. En nuestra implementación esto se logra monitoreando la longitud de I; si la longitud de I no decrece por un factor de $\delta < 1$ después de dos pruebas, entonces un paso de bisección es usado en la siguiente prueba de α_t (usamos en la $\delta = 0.66$ implementación).

Teorema 2.2 *El algoritmo de búsqueda produce una sucesión $\{\alpha_k\}$ en $[\alpha_{\min}, \alpha_{\max}]$ tal que después de un número finito de valores de prueba, una de las siguientes condiciones se sostiene :*
El algoritmo termina en $\alpha_{\max}$, la sucesión de valores de prueba es creciente y se cumple (2.9).
El algoritmo termina en $\alpha_{\min}$, la sucesión de valores de prueba es decreciente y se cumple (2.10).
Un intervalo $I_k \subset [\alpha_{\min}, \alpha_{\max}]$ es generado.

Prueba 2.2 *En esta prueba resumimos los argumentos presentados anteriormente. Sean $\alpha_l^{(k)}$ y $\alpha_u^{(k)}$ los puntos finales de I_k, y definimos*

$$\beta_l^{(k)} = \min\left\{\alpha_l^{(k)}, \alpha_u^{(k)}\right\}, \ \beta_u^{(k)} = \max\left\{\alpha_l^{(k)}, \alpha_u^{(k)}\right\}.$$

El punto final izquierdo $\beta_l^{(k)}$ de I_k es no decreciente, mientras el punto final derecho es no creciente.
Primero mostraremos que $\beta_u^{(k)} = \infty$ no puede ocurrir para todo $k \geq 0$. Si $\beta_u^{(k)} = \infty$, sólo el caso U2 del algoritmo de actualización sucede, debido a que en los otros dos casos ambos puntos finales toman valores finitos. Ya que, sólo el caso U2 ocurre, es claro que (2.9) sucede, y así la regla (2.8) muestra que el límite $\alpha_{\max}$ es eventualmente utilizado como un valor de prueba. Si el algoritmo no termina en $\alpha_{\max}$, entonces $\beta_u^{(k)} = \alpha_{\max}$.
Un argumento similar muestra que $\beta_l^{(k)} = 0$ no puede ocurrir para todo $k \geq 0$. Si $\beta_l^{(k)} = 0$, entonces sólo el caso U1 o U3 del algoritmo de actualización ocurre, debido a que en el caso U2 ambos puntos finales están en un conjunto de valores positivos. Además, en este caso (2.10) ocurre. La regla (2.11) muestra que (2.10) no puede suceder para todo $k \geq 0$ cuando $\alpha_{\min} = 0$, y si $\alpha_{\min} > 0$, entonces $\alpha_{\min}$ es eventualmente usado como un valor de prueba. Si la búsqueda no termina en $\alpha_{\min}$, entonces $\beta_u^{(k)} = \alpha_{\min}$.
Hemos así mostrado que después de un número finito de valores de prueba, ambas búsquedas terminan en una de las dos cotas $\alpha_{\min}$ o $\alpha_{\max}$, o $\beta_l^{(k)} > 0$ y $\beta_u^{(k)} < \infty$. Por supuesto, en este último caso I_k es un subconjunto de $[\alpha_{\min}, \alpha_{\max}]$.□

El caso más interesante del teorema (2.2) ocurre cuando un intervalo $I_k \subset [\alpha_{\min}, \alpha_{\max}]$ es generado. En este caso las reglas de seguridad garantizan que la lon-

gitud de los intervalos $\{I_k\}$ converge a cero, y así la sucesión $\{\alpha_k\}$ converge a alguna α^* en $T(\mu)$.

Podemos eliminar la terminación finita en uno de los límites excluyendo (2.9) y (2.10). La forma más simple de hacer esto es suponer que $\alpha_{\min}$ satisface

$$\psi(\alpha_{\min}) \leq 0 \; y \; \psi'(\alpha_{\min}) < 0, \tag{2.12}$$

y que $\alpha_{\max}$ satisface que

$$\psi(\alpha_{\max}) > 0 \; y \; \psi'(\alpha_{\max}) \geq 0. \tag{2.13}$$

Bajo estas suposiciones, el teorema (2.2) muestra que un intervalo $I_k \subset [\alpha_{\min}, \alpha_{\max}]$ después de un número finito de valores de prueba.

Las condiciones (2.12) y (2.13) pueden ser fácilmente satisfechas. Por ejemplo, si $\alpha_{\min} = 0$, entonces se cumple (2.12). La condición (2.13) se cumple si $\alpha_{\max}$ es definida por (2.5) y $\varphi_{\min}$ es un límite inferior estricto de φ. La condición (2.13) también se cumple si $\varphi'(\alpha_{\max}) \geq 0$.

Teorema 2.3 *Si los límites $\alpha_{\min}$ y $\alpha_{\max}$ cumplen con (2.12) y (2.13), entonces el algoritmo de búsqueda termina en un número finito de pasos con un $\alpha_k \in T(\mu)$, o las iteraciones $\{\alpha_k\}$ convergen a alguna $\alpha^* \in T(\mu)$ con $\psi'(\alpha^*) = 0$. Si el algoritmo no termina en un número finito de pasos, entonces existe un índice k_0 tal que los puntos finales $\alpha_l^{(k)}, \alpha_u^{(k)}$ de un intervalo I_k satisfacen que $\alpha_l^{(k)} < \alpha^* < \alpha_u^{(k)}$. Además, si $\psi(\alpha^*) = 0$, entonces ψ' cambia de signo en $[\alpha_l^{(k)}, \alpha^*]$ o $[\alpha^*, \alpha_u^{(k)}]$ para todo $k \geq k_0$.*

Prueba 2.3 *Suponemos que $\alpha_k \notin T(\mu)$ para todas las iteraciones generadas por el algoritmo de búsqueda. Ya que los intervalos I_k están uniformemente acotados y sus longitudes tienden a cero, cualquier sucesión $\{\theta_k\}$ con $\theta_k \in I_k$ debe converger a un límite común α^*. El teorema (2.1) garantiza que existe $\theta_k \in (T(\mu) \cap I_k)$ con $\varphi'(\theta_k) = \mu\varphi'(0)$. Esto implica que $\alpha^* \in T(\mu)$ y que $\varphi'(\alpha^*) = \mu\varphi'(0)$. En particular, $\psi'(\alpha^*) = 0$.*

Definimos k_0 observando que la continuidad de φ' muestra que existe un $k_0 > 0$ tal que $\varphi'(0) < 0$ para toda $\alpha \in I_k$ y toda $k \geq k_0$. Ya que $\psi(\alpha_l^{(k)}) \leq 0$ y $\alpha_l^{(k)} \notin T(\mu)$, debemos tener $\left|\varphi'(\alpha_l^{(k)}\right| > \mu \left|\varphi'(0)\right|$. También sabemos que $\varphi'(\alpha_l^{(k)}) < 0$ para $k \geq k_0$, y así $\varphi'(\alpha_l^{(k)}) < \mu\varphi'(0)$. Por lo tanto, $\psi'(\alpha_l^{(k)}) < 0$. La condición (2.7) en los puntos finales implica que $\alpha_l^{(k)} < \alpha_u^{(k)}$, y en particular, $\alpha_l^{(k)} < \alpha^* < \alpha_u^{(k)}$.

Consideremos el caso cuando $\psi(\alpha^*) = 0$. No puede ocurrir $\psi'(\alpha) \leq 0$ en $[\alpha_l^{(k)}, \alpha^*]$ debido a que esto implicaría que $\psi(\alpha_l^{(k)}) > \psi(\alpha^*) = 0$. Así $\psi'(\beta_k) > 0$ para algún $\beta_k \in [\alpha_l^{(k)}, \alpha^*]$. Ya que $\psi'(\alpha_l^{(k)}) < 0$, hemos mostrado que ψ' cambia de signo en $[\alpha_l^{(k)}, \alpha^*]$.

Finalmente, considere el caso cuando $\psi(\alpha^*) < 0$. *Suponga que* k_0 *es tal que* $\psi(\alpha_u^{(k)}) < 0$ *para todo* $k \geq k_0$. *Si* $\psi(\alpha_u^{(k)}) \geq 0$, *entonces* $\varphi'(\alpha_u^{(k)}) \geq \mu\varphi'(0)$, *y puesto que* $\varphi'(\alpha_u^{(k)}) < 0$, *tenemos que* $\alpha_u^{(k)} \in T(\mu)$. *Esta contradicción muestra que* $\psi'(\alpha_u^{(k)}) < 0$. *hemos mostrado ya que* $\psi'(\alpha_l^{(k)}) < 0$, *así que* ψ' *cambia de signo en* $[\alpha_l^{(k)}, \alpha^*]$ *o* $[\alpha^*, \alpha_u^{(k)}]$ *para algún* β_k *en* $[\alpha_l^{(k)}, \alpha_u^{(k)}]$. *Esto es claro debido a que si* $\psi'(\alpha) \leq 0$ *en* $[\alpha_l^{(k)}, \alpha_u^{(k)}]$, *entonces* $\psi(\alpha_l^{(k)}) > \psi(\alpha_u^{(k)})$.□

Si el algoritmo de búsqueda no termina en un número finito de pasos, entonces el teorema (2.3) implica que ψ' cambia de signo un número infinito de veces en el sentido que existe una sucesión monótona $\{\beta_k\}$ que converge a α^* tal que $\psi'(\beta_k)\psi'(\beta_{k+1}) < 0$. Excepto para casos patológicos el algoritmo termina en un número finito de iteraciones. Resultados cercanos han sido establecidos por Al-Baali y Fletcher , por Moré y Sorensen. En estos resultados, sin embargo, el énfasis está en mostrar que el algoritmo de búsqueda genera eventualmente un α_k que satisface (2.1) y (2.2) siempre que $\mu < \eta$.

2.2 Búsqueda de un Mínimo Local

El teorema (2.3) garantiza terminar en un α_k que satisfaga (2.1) y (2.2) siempre que $\eta > \mu$. En esta sección modificaremos el algoritmo de búsqueda y mostraremos, que bajo condiciones razonables, podemos garantizar que el algoritmo de búsqueda modificado genera un α_k que satisface (2.1) y (2.2) para toda $\eta > 0$.

La dificultad de poner $\eta < \mu$ es que, incluso si $T(\mu)$ es no vacío, quizá no exista un $\alpha > 0$ que satisfaga (2.1) y (2.2). Ilustraremos este ejemplo con una pequeña modificación de un ejemplo de Al-Baali y Fletcher. Definimos

$$\varphi(\alpha) \begin{cases} \frac{1}{2}(1-\sigma)\alpha^2 - \alpha, & 0 \leq \alpha \leq 1 \\ \frac{1}{2}(\sigma - 1) - \sigma\alpha, & 1 \leq \alpha, \end{cases}$$

donde $\eta < \sigma < \mu$. La curva sólida en la figura 2 es la función φ con $\sigma = 0.1$; la curva punteada es la gráfica de la función $l(\alpha) = \varphi(0) + \mu\varphi'(0)\alpha$ con $\mu = 0.25$. Un cálculo muestra que φ es continuamente diferenciable y que

$$\left|\varphi'(\alpha)\right| \geq \sigma > \eta$$

para toda $\alpha \geq 0$. Además, si $\mu < \frac{1}{2}$, entonces

$$T(\mu) = \left[\frac{1-\mu}{1-\sigma}, \frac{1-\sigma}{2(\mu-\sigma)}\right].$$

Así $T(\mu)$ es un intervalo no vacío con $\alpha = 1$ en el interior. En la figura 2 tenemos $\sigma = 0.1$ y $\mu = 0.25$ y así $T(\mu) = \left[\frac{5}{6}, 3\right]$.

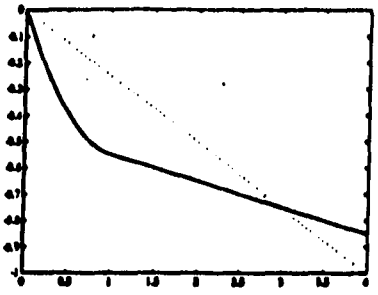


Figura 2

Mostraremos que si durante la búsqueda de $T(\mu)$ calculamos valores de prueba α_k tales que $\psi(\alpha_k) \leq 0$ y $\psi'(\alpha_k) \leq 0$, entonces α_k pertenece a $T(\mu)$ o hemos identificado un intervalo que contiene puntos que satisfacen la condición de decrecimiento suficiente (2.1) y la condición de curvatura (2.2).

Teorema 2.4 *Suponga que los límites $\alpha_{\min}$ y $\alpha_{\max}$ satisfacen (2.12) y (2.13). Sea $\{\alpha_k\}$ la sucesión generada por el algoritmo de búsqueda, y sean $\alpha_l^{(k)}$ y $\alpha_u^{(k)}$ los puntos finales del intervalo I_k generado por el algoritmo de búsqueda. Si α_k es la primera iteración que satisface*

$$\psi(\alpha_k) \leq 0, \ \psi'(\alpha_k) \geq 0, \tag{2.14}$$

entonces $\alpha_l^{(k)} < \alpha_u^{(k)}$. Además, $\alpha_k \in T(\mu)$ o $\varphi'(\alpha_k) > 0$. Si $\varphi'(\alpha_k) > 0$, entonces el intervalo

$$I^* \equiv \left[\alpha_l^{(k)}, \alpha_k\right],$$

contiene un α^* que satisface (2.1) y $\varphi'(\alpha^*) = 0$. Además, cualquier $\alpha \in I^*$ con $\varphi(\alpha) \leq \varphi(\alpha_k)$ también satisface (2.1).

Prueba 2.4 *La primera condición es $\psi'(\alpha_l^{(k)}) < 0$. Si este no es el caso, entonces $\psi'(\alpha_j) \geq 0$ para algún índice $j < k$ debido a que $\alpha_l^{(k)}$ es una iteración anterior. Sin embargo, esto contradice la suposición de que α_k es la primera iteración que satisface (2.14). Esto prueba que $\psi'(\alpha_l^{(k)}) < 0$.*

Ya que $\psi'(\alpha_l^{(k)}) < 0$, suponemos que (2.7) implica que $\alpha_l^{(k)} < \alpha_u^{(k)}$. Esto implica, en parti-cular, que $\alpha_k > \alpha_l^{(k)}$ así I^* está bien definido.

Si $\psi'(\alpha_k) \leq 0$ y $\psi'(\alpha_k) \geq 0$, entonces es claro que $\left|\varphi'(\alpha_k)\right| \leq \mu\left|\varphi'(0)\right|$. Ya que $\psi'(\alpha_k) > 0$ y α^* es un mínimo, $\alpha^* \neq \alpha_k$. Similarmente, ya que probamos antes que $\psi'(\alpha_l^{(k)}) < 0$, y que α^* es un mínimo, $\alpha^* \neq \alpha_l^{(k)}$.

Hemos mostrado que α^* está en el interior de I^*. De aquí, $\varphi'(\alpha^*) = 0$ como deseábamos. Completamos la prueba observando que si $\varphi(\alpha) \leq \varphi(\alpha_k)$ para algún

$\alpha \in I^*$, entonces

$$\varphi(\alpha) \leq \varphi(\alpha_k) \leq \varphi(0) + \mu\varphi'(0)\alpha_k \leq \varphi(0) + \mu\varphi'(0)\alpha.$$

La segunda desigualdad se sostiene a causa de que α_k satisface (2.1), mientras la tercera desigualdad se sostiene debido a que $\alpha \leq \alpha_k$. De aquí, cualquier $\alpha \in I^*$ con $\varphi(\alpha) \leq \varphi(\alpha_k)$ también satisface (2.1).□

No hay garantía de que el algoritmo de búsqueda genere una iteración α_k tal que $\psi(\alpha_k) \leq 0$ y $\varphi'(\alpha_k) > 0$. Por ejemplo, si φ es la función mostrada en la figura 2, entonces $\varphi'(\alpha) < 0$ para todo α. Incluso si φ tuviera un mínimo en α^* que satisficiera la condición de decrecimiento suficiente, el algoritmo de búsqueda quizá tomaría la región que contiene puntos en $T(\mu)$, pero donde (2.1) y (2.2) no sean satisfechas.

El teorema (2.4) es uno de los ingredientes necesarios para desarrollar un algoritmo de búsqueda para minimización que satisfaga las condiciones de decrecimiento suficiente (2.1) y de curvatura (2.2). También necesitamos mostrar que el intervalo I^* especificado por el teorema (2.4) satisface las hipótesis del siguiente resultado.

Teorema 2.5 *Sea I un intervalo cerrado con puntos finales α_l y α_u. Si los puntos finales satisfacen*

$$\varphi(\alpha_l) \leq \varphi(\alpha_u),\ \varphi'(\alpha_l)(\alpha_u - \alpha_l) < 0,$$

entonces existe un α^* en I con $\varphi(\alpha^*) \leq \varphi(\alpha_l)$ y $\varphi'(\alpha^*) = 0$.

Prueba 2.5 *La prueba de este resultado es casi inmediata. Si α^* es el mínimo global de φ en I, entonces la suposición de α_l y α_u garantiza que α^* está en el interior de I y así $\varphi'(\alpha^*) = 0$.*□

El intervalo especificado por el teorema (2.4) satisface las suposiciones del teorema (2.5), debido a que la derivada de φ tenía el signo apropiado. Asumimos que $\varphi'(\alpha_k) > 0$. Además, en el teorema (2.4) establecimos que $\alpha_l^{(k)} < \alpha_u^{(k)}$, y así la suposición (2.7) en los puntos finales de I_k implica que $\psi'(\alpha_l^{(k)}) < 0$. De aquí $\varphi'(\alpha_l^{(k)}) < 0$. Estos dos resultados muestran que I^* tiene las propiedades deseadas.

Ahora necesitamos modificar el algoritmo de actualización de tal forma que garantice terminar en una iteración que satisfaga la condición de decrecimiento suficiente (2.1) y la de curvatura (2.2). La modificación es simple; sólo reemplace ψ por φ en el algoritmo de actualización.

Algoritmo de Actualización Modificado. Dado un valor de prueba α_t en I, los puntos finales α_l^+ y α_u^+ del intervalo actualizado I_+ están determinados como sigue:

Caso a: Si $\varphi(a_t) > \varphi(a_l)$, entonces $a_l^+ = a_l$ y $a_u^+ = a_t$.

Caso b: Si $\varphi(a_t) \leq \varphi(a_l)$ y $\varphi'(a_t)(a_l - a_t) > 0$, entonces $a_l^+ = a_t$ y $a_u^+ = a_u$.

Caso c: Si $\varphi(a_t) \leq \varphi(a_l)$ y $\varphi'(a_t)(a_l - a_t) < 0$, entonces $a_l^+ = a_t$ y $a_u^+ = a_l$.

Hemos mostrado que el intervalo I^* especificado por el teorema (2.4) satisface las suposiciones del teorema (2.5). Además, un corto cálculo muestra que si I es cualquier intervalo que satisface las suposiciones del teorema (2.4), entonces el algoritmo de actualización modificado preserva estas suposiciones.

Nuestra implementación del algoritmo de búsqueda de la sección 2.2 usa el algoritmo de actualización modificado de una manea obvia: Si alguna iteración α_k satisface $\psi(\alpha_k) \leq 0$ y $\varphi'(\alpha_k) < 0$, entonces el algoritmo de actualización modificado es usado en esa iteración y en todas las otras iteraciones.

Teorema 2.6 *Suponga que los límites $\alpha_{\min}$ y $\alpha_{\max}$ satisfacen (2.12) y (2.13). Si el algoritmo de búsqueda modificado genera una iteración tal que $\psi(\alpha_k) \leq 0$ y $\varphi'(\alpha_k) > 0$, entonces el algoritmo de búsqueda modificado termina en un α_k que satisface (2.1) y (2.2).*

Prueba 2.6 *Si el algoritmo de búsqueda genera un α_k con $\psi(\alpha_k) \leq 0$ y $\varphi'(\alpha_k) > 0$, entonces el teorema (2.4) muestra que $\alpha_k > \alpha_l^{(k)}$, y así el algoritmo de actualización modificado pone*

$$I_{k+1} = \left[\alpha_l^{(k)}, \alpha_k\right]$$

debido a que el caso U2 no se sostiene. Además, el teorema (2.4) garantiza que cualquier $\alpha \in I_{k+1}$ con $\varphi(\alpha) \leq \varphi(\alpha_k)$ satisface (2.1). Esto implica que para cualquier iteración $j > k$ el punto final $\alpha_l^{(j)}$ satisface (2.1). También mostramos que cualquier sucesión $\{\theta_k\}$ con $\theta_k \in I_k$ debe converger a un límite común α^*. Ya que el teorema (2.5) muestra que hay un $\theta_k \in I_k$ tal que $\varphi'(\theta_k) = 0$, obtenemos que $\varphi'(\alpha^*) = 0$. De aquí, $\alpha_l^{(j)}$ satisface (2.2) para todo $j > k$ suficientemente grande. Esto prueba que el algoritmo de búsqueda modificado termina en una iteración que satisface (2.1) y (2.2).□

2.3 Selección de Valores de Prueba

Dados los puntos finales a_l y a_u de un intervalo I, y un valor de prueba a_t en I, el algoritmo de actualización descrito en la sección anterior produce un intervalo I_+ que

contiene puntos aceptables. Ahora especificaremos el nuevo valor de prueba α_t^+ en I_+.

Supongamos que conocemos los valores de la función f_l, f_u, f_t y sus derivadas g_l, g_u, g_t correspondientes a los puntos finales α_l y α_u, y al punto prueba α_t respectivamente. Los valores de la función f_l, f_u, f_t y sus derivadas g_l, g_u, g_t pueden ser obtenidos a través de la función φ o de la función auxiliar ψ. Los valores de la función y su derivada son obtenidos de la función auxiliar ψ hasta que alguna iteración satisfaga la prueba $\psi(\alpha_k) \leq 0$ y $\psi'(\alpha_k) \geq 0$. Una vez que esta prueba es satisfecha, φ es utilizada.

Hemos dividido los valores de prueba en cuatro casos. En los primeros dos casos elegimos α_t^+ interpolando los valores de la función entre α_l y α_t. Definimos α_t^+ en términos de α_c (el mínimo de la función cúbica que interpola f_l, f_t, g_l y g_t), α_q (el mínimo de la función que interpola f_l, f_t y g_l), y α_s (el mínimo de la función cuadrática que interpola g_l y g_t).

Caso 1: $f_t > f_l$. En este caso calculamos α_c, α_q, y ponemos

$$\alpha_t^+ = \begin{cases} \alpha_c \; Si \; |\alpha_c - \alpha_l| < |\alpha_q - \alpha_l| \\ \frac{1}{2}(\alpha_q + \alpha_c) \; en \; otro \; caso. \end{cases}$$

Ambos α_c y α_q se encuentran en I_+ así ellos son candidatos para α_t^+. Debemos hacer una elección que sea cercana a α_l ya que este es el punto con el valor más pequeño de la función. Ambos α_q y α_c están relativamente cercanos a α_l debido a

$$|\alpha_c - \alpha_l| \leq \frac{2}{3}|\alpha_c - \alpha_l|, \; |\alpha_q - \alpha_l| \leq \frac{1}{2}|\alpha_c - \alpha_l|.$$

Así, para la anterior elección de α_t^+,

$$\left|\alpha_t^+ - \alpha_l\right| \leq \frac{7}{12}|\alpha_u - \alpha_l|.$$

Una elección cercana a α_l cuando f_t es mucho más grande que f_l. En este caso el paso cuadrático es más cercano a α_l que α_c, pero generalmente, anormal. Efectivamente, si $\alpha_q(f_t)$ es el valor de α_q en función de f_t, entonces

$$\lim_{f_t \to \infty} \alpha_q(f_t) = \alpha_l.$$

Por otra parte, un cálculo muestra que

$$\lim_{f_t \to \infty} \alpha_c(f_t) = \alpha_l + \frac{2}{3}(\alpha_u - \alpha_l).$$

Así, el punto medio de α_c y α_t es un término medio razonable.

Caso 2: $f_t \leq f_l$ y $g_t g_l < 0$. En este caso calculamos α_c, α_s, y ponemos

$$\alpha_t^+ = \begin{cases} \alpha_c \text{ si } |\alpha_c - \alpha_t| \geq |\alpha_s - \alpha_t| \\ \alpha_s \text{ en otro caso.} \end{cases}$$

Ambos α_c y α_s ubicados en I_+, así son candidatos para α_t^+. Ya que $g_t g_l < 0$, existe un mínimo entre α_l y α_t. Eligiendo el paso que esté más lejano de α_t cuidando de generar un paso que encuadre un mínimo, y así es probable que el siguiente paso también caiga en este caso.

En el siguiente caso elegimos α_t^+ extrapolando los valores de la función en α_l y α_t, así el valor de prueba se encuentra fuera del intervalo con α_t y α_l como puntos finales. Definimos α_t^+ en términos de α_c (el mínimo de la función cúbica que interpola f_l, f_t, g_l y g_t) y α_s (el mínimo de la función cuadrática que interpola g_l y g_t).

Caso 3: $f_t \leq f_l$, $g_t g_l \geq 0$, y $|g_t| \leq |g_l|$. En este caso la función cúbica que interpola los valores de la función f_l y f_t y las derivadas g_l y g_t puede no tener mínimo. Además, incluso si el mínimo α_c existiese, esta en la dirección equivocada. Por ejemplo, tenemos $\alpha_t > \alpha_l$ pero $\alpha_c < \alpha_t$. Por otro lado, el paso secante α_s siempre existe y esté en la dirección correcta.

Si la función cúbica tiende a infinito en la dirección del paso y el mínimo de la función cúbica está más allá de α_t, ponemos

$$\alpha_t^+ = \begin{cases} \alpha_c \text{ si } |\alpha_c - \alpha_t| < |\alpha_s - \alpha_t| \\ \alpha_s \text{ en otro caso.} \end{cases}$$

En otro caso, ponemos $\alpha_t^+ = \alpha_s$. Esta elección está basada en la observación que durante la extrapolación es sensible a precauciones y elegir el paso que sea más cercano a α_t.

El valor de prueba α_t^+ definido anteriormente puede estar fuera del intervalo comprendido entre α_t y α_u, o puede estar en este intervalo pero cercano a α_u. Ambas situaciones son indeseables, así redefinimos α_t^+ colocando

$$\alpha_t^+ = \begin{cases} \min\left\{\alpha_t + \delta(\alpha_u - \alpha_t), \alpha_t^+\right\} \text{ si } \alpha_t > \alpha_l \\ \max\left\{\alpha_t + \delta(\alpha_u - \alpha_t), \alpha_t^+\right\} \text{ en otro caso} \end{cases}$$

para algún $\delta < 1$. Usamos en nuestro algoritmo $\delta = 0.66$.

En el último de caso la información disponible en α_l y α_t indica que la función decrece rápidamente en la dirección del paso, pero no parece ser una buena manera de elegir α_t^+.

Caso 4: $f_t \leq f_l, g_t g_l \geq 0$ y $|g_t| < |g_l|$. en este caso elegimos α_t^+ como el mínimo de la función cúbica que interpola f_u, f_t, g_u y g_t.

2.4 Resultados Numéricos

El conjunto de problemas de prueba que usamos para ilustrar las características del algoritmo de búsqueda contiene funciones generales y funciones convexas. Las primeras tres funciones tienen regiones de concavidad, mientras las últimas tres funciones son convexas. En todos los casos las funciones tienen un único mínimo. Los resultados fueron obtenidos con doble precisión.

La región de concavidad de la primera función en el conjunto prueba está a la derecha del mínimo, mientras la segunda función es cóncava a la izquierda del mínimo. La primera función está definida por

$$\varphi(\alpha) = -\frac{\alpha}{(\alpha^2+\beta)} \tag{2.15}$$

con $\beta = 2$, mientras la segunda función está definida por

$$\varphi(\alpha) = (\alpha+\beta)^5 - 2(\alpha+\beta)^4 \tag{2.16}$$

con $\beta = 0.004$. Las gráficas de estas dos funciones aparecen en las figuras 3 y 4.

La tercera función en el conjunto prueba fue sugerida por Paul Plassmann. Esta función es definida en términos de los parámetros l y β por

$$\varphi(\alpha) = \varphi_0(\alpha) + \frac{2(1-\beta)}{l\pi}\sin\left(\frac{l\pi}{2}\alpha\right), \tag{2.17}$$

donde

$$\varphi_0(\alpha) = \begin{cases} 1-\alpha & si\ \alpha \le 1-\beta \\ \alpha - 1 & si\ \alpha \ge 1+\beta \\ \frac{1}{2\beta}(\alpha-1)^2 + \frac{1}{2}\beta & si\ \alpha \in [1-\beta, 1+\beta]. \end{cases}$$

El parámetro β controla el tamaño de $\varphi'(0) = -\beta$. Este parámetro también controla el tamaño del intervalo donde (2.2) se sostiene debido a $|\varphi'(\alpha)| \ge \beta$ para $|\alpha - 1| \ge \beta$, y así (2.2) puede sostenerse sólo para $|\alpha - 1| < \beta$. El parámetro l controla el número de oscilaciones en la función para $|\alpha - 1| \ge \beta$ debido a que en este intervalo $\varphi''(\alpha)$ es un múltiplo de $\sin(\frac{l\pi}{2}\alpha)$. Observe que si l es impar, entonces $\varphi'(1) = 0$, y que si $l = 4k-1$ para algún entero $k \ge 1$, entonces $\varphi''(1) > 0$. También observe que φ es convexo para $|\alpha - 1| < \beta$ si

$$\beta(1-\beta)\frac{l\pi}{2} \le 1.$$

Elegimos $\beta = 0.01$ y $l = 39$. Una gráfica de esta función con tales parámetros aparece en la figura 5.

Las otra tres funciones están definidas en términos de los parámetros β_1 y β_2 por

$$\varphi(\alpha) = \gamma(\beta_1)[(1-\alpha)^2 + \beta_2^2]^{\frac{1}{2}} + \gamma(\beta_2)[\alpha^2 + \beta_1^2]^{\frac{1}{2}}, \tag{2.18}$$

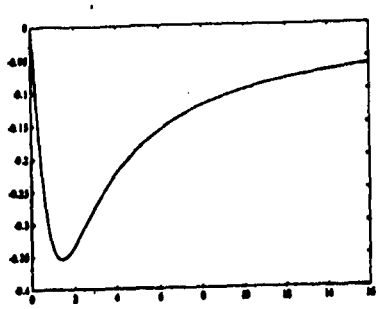


Figura 3

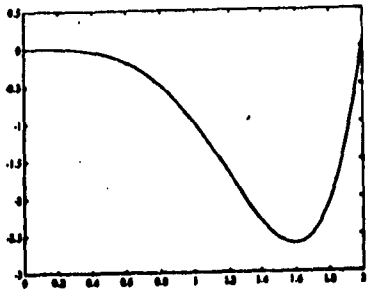


Figura 4

donde

$$\gamma(\beta) = (1+\beta^2)^{\frac{1}{2}} - \beta.$$

Estas funciones son convexas, pero diferentes elecciones de β_1 y β_2 llevan a funciones con características completamente diferentes. Esto puede verse en las figuras 7, 6 y 8.

En la tabla de abajo presentamos resultados numéricos para diferentes valores de α_0. Hemos usado $\alpha_0 = 10^i$ para $i = \pm 1, \pm 3$. Esto ilustra las características del algoritmo para diferentes puntos iniciales. Estamos particularmente interesados en las características de puntos iniciales lejanos tales como $\alpha_0 = 10^{\pm 3}$.

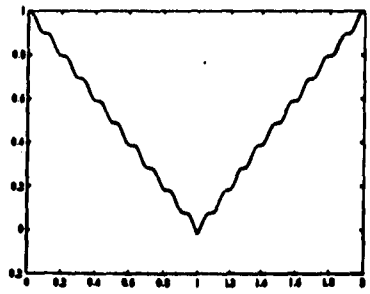


Figura 5

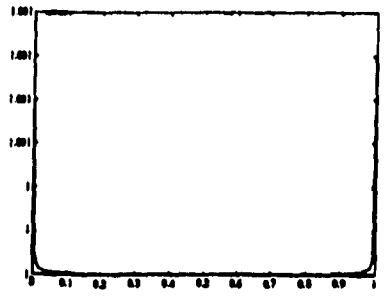


Figura 6

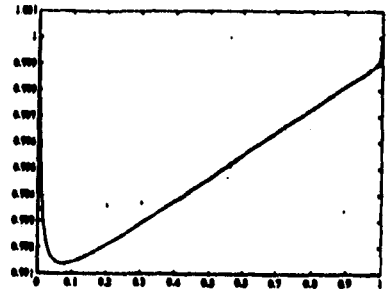


Figura 7

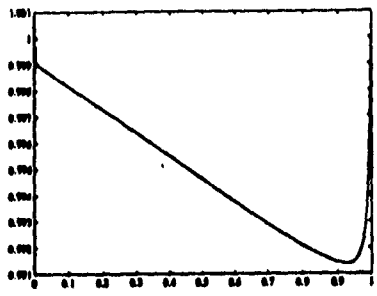


Figura 8

Tabla 2.1: Resultados para la función con $\mu = 0.001$, $\eta = 0.1$ y $\beta = 2$.

α_0	info	m	α_m	$\varphi'(\alpha_m)$
10^{-3}	1	6	1.4	$-9.2\,10^{-3}$
10^{-1}	1	3	1.4	$4.7\,10^{-3}$
10^{+1}	1	1	10	$9.4\,10^{-3}$
10^{+3}	1	4	37	$7.3\,10^{-4}$

En los resultados numéricos hemos usado diferentes valores de μ y η de modo de ilustrar las diferentes características de los problemas y el algoritmo de búsqueda. En muchos problemas hemos usado $\eta = 0.1$ debido a que este valor es típico de aquellos usados en optimización. Comentaremos qué sucede con otros valores de μ y ηfunción si μ disminuya o si η aumenta. La razón de esta tendencia es cuando μ decrece o η aumenta, la medida del conjunto de valores aceptables de α aumenta.

Una interesante característica de los resultados para la función 2.1 es que los valores de α que son mucho mayores que $\alpha^* \approx 1.4$ pueden satisfacer (2.1) y (2.2). Debería ser claro de los resultados de la figura 3. Estos resultados muestran que si usamos $\mu = 0.001$ y $\eta = 0.1$, entonces el punto inicial $\alpha_0 = 10$ satisface (2.1) y (2.2), y así el algoritmo de búsqueda subsiste con α_0. Similarmente, el algoritmo de búsqueda subsiste con $\alpha_4 \approx 37$ cuando el punto inicial es $\alpha_0 = 10^{+3}$.

Podemos evitar terminar en puntos lejanos del mínimo α^* incrementando μ o decrementando η. Si incrementamos μ y hacemos $\mu = \eta = 0.1$, entonces el algoritmo termina con $\alpha_3 \approx 1.6$ cuando $\alpha_0 = 10$ y con $\alpha_7 \approx 1.6$ cuando $\alpha_0 = 10^{+3}$. No hay ningún cambio en las características del algoritmo al iniciar en los otros dos puntos. Si decrementamos η colocándolo como $\eta = 0.001$ pero no cambiamos el valor de $\mu = 0.1$, entonces la iteración final α_m es cercana a α^* para todo punto inicial. Para $\eta = 0.001$ el algoritmo de búsqueda necesita evaluar la función seis veces para $\alpha_0 = 10$ y diez para $\alpha_0 = 10^{+3}$. El número de evaluaciones de la función para $\alpha_0 = 10^{-3}$ y $\alpha_0 = 10^{-1}$

son, 8 y 4 respectivamente. Este aumento en el número de evaluaciones es esperado debido a que ahora el conjunto de α aceptable es más pequeño.

Otra interesante característica de los resultados de la tabla 2.1 es que las seis evaluaciones de la función necesarias para $\alpha_0 = 10^{-3}$ podrían haber sido predichas a partir del proceso de extrapolación. Esto puede ser explicado observando que una situación típica del proceso de extrapolación genera iteraciones a partir de $\alpha_i^+ = \alpha_i + \delta(\alpha_i - \alpha_j)$ con $\delta = 4$, y así

$$\alpha_1 = 0.005,\ \alpha_2 = 0.021,\ \alpha_3 = 0.085,\ \alpha_4 = 0.341,\ \alpha_5 = 1.365,$$

Tabla 2.2: Resultados para la función con $\mu = \eta = 0.1$ y $\beta = 0.004$

α_0	info	m	α_m	$\varphi'(\alpha_m)$
10^{-3}	1	12	1.6	$7.1\ 10^{-9}$
10^{-1}	1	8	1.6	$1.0\ 10^{-10}$
10^{+1}	1	8	1.6	$-5.0\ 10^{-9}$
10^{+3}	1	11	1.6	$-2.3\ 10^{-8}$

hasta que el mínimo sea cercado, o hasta que una de las iteraciones satisfaga el criterio de terminación. Esto implica, por ejemplo, que si el mínimo está en $\alpha^* = 1.4$, entonces una de las iteraciones anteriores satisface (2.1) y (2.2), o al menos seis evaluaciones de la función son requeridas antes de que el algoritmo de búsqueda exista.

El número de evaluaciones de la función necesarias para encontrar un α aceptable depende generalmente de la medida del conjunto del α aceptable. Desde este punto de vista, los únicos problemas prueba difíciles son aquellos basados en las funciones de las figuras 4 y 5, debido a que para estas funciones el conjunto de α aceptables es pequeño. La elección de $\beta = 0.004$ para la función en la figura 4 garantiza que esta función tiene una gran región de concavidad, pero también forza a que $\varphi'(0)$ sea demasiado pequeña (aproximadamente $-5\ 10^{-7}$). A consecuencia de (2.2) es demasiado restrictiva para $\eta < 1$. Observaciones similares se aplican a los resultados numéricos de la función en la figura 5. Este es un problema prueba difícil debido a que la información obtenida en las derivadas es poco confiable como resultado de las oscilaciones de la función. Además, como ya observamos, (2.2) puede cumplirse sólo para $|\alpha - 1| < \beta$.

En la tabla 2.2 presentamos resultados numéricos para la función de la figura 4. En esta tabla usamos $\mu = \eta = 0.1$, pero estos resultados permanecen sin cambio si hacemos $\eta = 0.1$ y elegimos cualquier $\mu < \eta$.

El número de evaluaciones de la función en la tabla 2.2 se compara favorablemente con un algoritmo de búsqueda basado en bisección. Dado un valor inicial $\alpha_0 = 10$, un algoritmo de búsqueda basado en bisección requiere de 48 evaluaciones de la función para determinar un α aceptable, ya que en este problema el conjunto

de α aceptable es un intervalo de longitud aproximada de $2.5\ 10^{-9}$. La comparación es incluso más favorable para el punto inicial $\alpha_0 = 10^{+3}$ debido, en este caso, a que el algoritmo de bisección requiere de 107 evaluaciones.

Para la función en la figura 5 el conjunto de α aceptable es un intervalo de longitud 10^{-3}, así un algoritmo de bisección requiere de 10 evaluaciones de la función para el valor inicial de $\alpha_0 = 10$, y 30 evaluaciones de la función para $\alpha_0 = 10^{+3}$. Si comparamos esta información con los resultados numéricos de la tabla 2.3, veremos que el algoritmo de búsqueda se ejecuta mejor que un algoritmo basado en bisección. Esto es sorpresivo porque para esta función la información proporcionada por φ' es poco confiable.

Tabla 2.3: Resultados para la función con $\mu = \eta = 0.1$ y $\beta = 0.01, l = 39$.

α_0	info	m	α_m	$\varphi'(\alpha_m)$
10^{-3}	1	12	1.0	$-5.1\ 10^{-5}$
10^{-1}	1	12	1.0	$-1.9\ 10^{-4}$
10^{+1}	1	10	1.0	$-2.0\ 10^{-6}$
10^{+3}	1	13	1.0	$-1.6\ 10^{-5}$

Tabla 2.4: Resultados para la función con $\mu = \eta = 0.001$ y $\beta_1 = 0.001, \beta_2 = 0.001$.

α_0	info	m	α_m	$\varphi'(\alpha_m)$
10^{-3}	1	4	0.08	$-6.9\ 10^{-5}$
10^{-1}	1	1	0.10	$-4.9\ 10^{-5}$
10^{+1}	1	3	0.35	$-2.9\ 10^{-6}$
10^{+3}	1	4	0.83	$1.6\ 10^{-5}$

Los resultados numéricos basados en los problemas de la función (2.4) aparecen en las tablas 2.4, 2.5 y 2.6. En todas estas tablas elegimos $\mu = \eta = 0.001$. Aunque estas elecciones no son comunes a aquellos en el ambiente de optimización, ello nos lleva a interesantes resultados.

Si comparamos los resultados en las tres tablas, observamos que para diferentes puntos iniciales, el número de evaluaciones de la función a veces difiere considerablemente. Los resultados en la tabla 2.5 son típicos de aquellos que ocurren para $\eta = 0.001$. A examinar los resultados en la tabla 2.5, muchas concesiones deben ser hechas para que los puntos iniciales no se distribuyan simétricamente alrededor del mínimo $\alpha^* \approx 0.074$. En particular, un número pequeño de evaluaciones de la función para $\alpha_0 = 0.1$ es debido, principalmente, al hecho de que en este caso α_0 es cercano a α^*.

El número de evaluaciones de la función en la tabla 2.4 es bajo porque el conjunto de α aceptables es grande (este hecho no es normal). En particular, observe

que el valor de α_m regresado por el algoritmo no es cercano al mínimo $\alpha^* = \frac{1}{2}$ de la función en la figura 6.

El número de evaluaciones de la función en la tabla 2.6 es el más alto debido a que en este problema es difícil determinar una iteración α_k tal que $\varphi'(\alpha_k) > 0$ y α_k satisfaga la condición de decrecimiento suficiente. Recuerde que una vez que tal iteración es determinada, sabemos que el problema tiene un mínimo que satisface la condición de decrecimiento suficiente.

Tabla 2.5: Resultados para la función con $\mu = \eta = 0.001$ y con $\beta_1 = 0.01, \beta_2 = 0.001$.

α_0	info	m	α_m	$\varphi'(\alpha_m)$
10^{-3}	1	6	0.075	$1.9\,10^{-4}$
10^{-1}	1	3	0.078	$7.4\,10^{-4}$
10^{+1}	1	7	0.073	$-2.6\,10^{-4}$
10^{+3}	1	8	0.076	$4.5\,10^{-4}$

Tabla 2.6: Resultados para la función con $\mu = \eta = 0.001$ y con $\beta_1 = 0.001, \beta_2 = 0.01$.

α_0	info	m	α_m	$\varphi'(\alpha_m)$
10^{-3}	1	13	0.93	$5.2\,10^{-4}$
10^{-1}	1	11	0.93	$8.4\,10^{-5}$
10^{+1}	1	8	0.92	$-2.4\,10^{-4}$
10^{+3}	1	11	0.92	$-3.2\,10^{-4}$

En un escenario de optimización no se ocuparía de usar $\eta = 0.001$, y entonces el número de evaluaciones necesarias para obtener un α aceptable decrecerá considerablemente. Considere, por ejemplo, los resultados para la función en la figura 8 con $\mu = 0.001$ y $\eta = 0.1$. Para este escenario, el número de evaluaciones de la función necesarios para obtener un α aceptable a partir de un $\alpha_0 = 10^i$ para $i = \pm 1, \pm 3$ son, $2, 1, 3, 4$ respectivamente. Resultados similares serán obtenidos para las funciones en las figuras 6 y 7.

Capítulo 3
MÉTODO DE NEWTON

El método de Newton juega un papel central en el desarrollo de técnicas de optimización. Una de las razones de su importancia es que se origina de forma natural a partir de aproximar la función por la serie de Taylor, debido a su simplicidad y amplio campo de aplicación, el método de Newton constituye una importante herramienta para resolver muchos problemas de optimización. De hecho, muchos de los actuales métodos pueden ser vistos como variaciones del método de Newton (por ejemplo, los métodos Quasi-Newton).

Uno de los propósitos de este capítulo es presentar y analizar las dos más importantes variaciones del método de Newton para minimización sin restricciones, que es la aproximación por búsqueda en línea y la aproximación a través de la región de confianza. Otro propósito es presentar algunos de los más recientes desarrollos en el campo de optimización relativos al método de Newton. En particular, exploraremos algunas variaciones del método de Newton, las cuales son apropiadas para problemas de gran escala, y también mostraremos cómo los métodos Quasi-Newton pueden ser derivados de manera completamente natural, a partir del método de Newton.

Dada una función $f : R^n \to R$ definida en un conjunto abierto D, el problema de minimización sin restricciones consiste en encontrar un punto $x^* \in D$ tal que

$$f(x^*) \leq f(x), \; \forall \, x \in N(x^*)$$

donde $N(x^*)$ es una vecindad de x^*.

Supondremos que f es dos veces continua y diferenciable, y así las propiedades de los mínimos locales pueden se expresadas en términos de la función cuadrática

$$\psi(w) = \nabla f(x)^T w + \frac{1}{2} w^T \nabla^2 f(x) w$$

ya que

$$f(x + w) = f(x) + \psi(w) + o(\|w\|^2) \tag{3.1}$$

donde ψ es el modelo cuadrático local en x de la reducción esperada de f, es decir

$$\psi(w) \approx f(x + w) - f(w).$$

Teorema 3.1 *Sea $f : R^n \to R$ dos veces continua y diferenciable en un conjunto abierto D. Si $x^* \in D$ es un mínimo local de f entonces $\nabla f(x^*) = 0$ y $\nabla^2 f(x^*)$ es semidefinido positivo. Si $\nabla f(x^*) = 0$ y $\nabla^2 f(x^*)$ es semidefinido positivo para algún $x^* \in D$, entonces x^* es un mínimo local aislado de f.*

Prueba 3.1 *Sea ψ el modelo cuadrático local en x^*. Si x^* es un mínimo local de f entonces (3.1) muestra que*

$$0 \leq \psi(\alpha p) + o(\alpha^2) = \alpha \nabla f(x^*)^t p + \alpha^2 p^t \nabla^2 f(x^*) p + o(\alpha^2),$$

para cada $p \in R^n$ y toda α suficientemente pequeña. Esto implica que $\nabla f(x^*)^t p = 0$ y que $p^t \nabla^2 f(x^*) \geq 0$. Ya que p es arbitraria, debemos concluir que $\nabla f(x^*) = 0$ y que $\nabla^2 f(x^*)$ es positivo semidefinido. Por otro lado, si $\nabla f(x^*) = 0$ y $\nabla^2 f(x^*)$ es positivo definido entonces

$$\psi(w) = \frac{1}{2} w^t \nabla^2 f(x^*) w \geq \frac{1}{2} \lambda \|w\|^2,$$

donde $\lambda > 0$ es el más pequeño valor propio de $\nabla^2 f(x^*)$. Se sigue de (3.1) que x^* debe ser un mínimo aislado de f.□

Un punto $x^* \in R^n$ tal que $\nabla f(x^*) = 0$ es un punto crítico de f. Los puntos críticos pueden ser divididos en mínimos locales, máximos locales y puntos silla. El teorema (3.1) muestra, en particular, que si x^* es un punto crítico de f y $\nabla^2 f(x^*)$ es indefinido entonces x^* es un punto silla de f. Sin embargo si, $\nabla^2 f(x^*)$ es positivo semidefinido y singular entonces el teorema no proporciona información alguna sobre la naturaleza del punto critico. Este vacío entre las condiciones necesarias y suficientes del teorema (3.1) es ilustrado por la siguiente función

$$f(x, y) = x^3 + y^2$$

observe que $(0, 0)$ es un punto critico de f y que la matriz Hessiana en $(0, 0)$ es positiva semidefinida. Sin embargo, $(0, 0)$ es un punto silla de f y no un punto mínimo.

Los algoritmos para minimizar una función $f : R^n \to R$ son usualmente métodos de descenso que generan una sucesión $\{x_k\}$ que aproxima un mínimo local con la propiedad

$$f(x_{k+1}) < f(x_k), \ \forall \, k \geq 0 \tag{3.2}$$

Esta sola condición no garantiza la convergencia a un mínimo local, por lo que se requiere de fuertes condiciones para forzar a la sucesión a una vecindad de un mínimo local. Una vez que las iteraciones están dentro de la vecindad, los métodos de descenso normalmente permiten una rápida convergencia. Las iteraciones del método de Newton son

$$x_{k+1} = x_k - \nabla^2 f(x_k)^{-1} \nabla f(x_k) \ \forall \, k \geq 0$$

Así el método de Newton toma una aproximación x_0 e intenta mejorarla a través de la iteración

$$x_{k+1} = x_k - \nabla^2 f(x_k)\nabla f(x_k) \; \forall \; k \geq 0. \tag{3.5}$$

Observe que en esta iteración la única restricción sobre el paso s_k es que satisfaga el sistema de ecuaciones lineales $\nabla\psi_k(w) = 0$. En otras palabras, requerimos que s_k sea un punto crítico de ψ_k. Como consecuencia, la iteración (3.5) tiene las mismas características en la vecindad de cualquier punto crítico de f. Esto parece indeseable ya que quisiéramos que nuestros algoritmos tuvieran predilección hacia los puntos mínimos locales.

Sin embargo, esta característica es consecuencia del hecho que la iteración (3.5) del método de Newton es la solución del sistema de ecuaciones lineales $\nabla f(x) = 0$. Ya que las propiedades locales de la iteración (3.5) únicamente dependen del mapeo $F(x) = \nabla f(x)$, consideremos el método de Newton de manera más general.

Sea $F : R^n \to R^n$ una función con rango y dominio en R^n, considere el problema de encontrar la solución al sistema de n ecuaciones con n incógnitas $F(x) = 0$, o equivalentemente,

$$f_i(\xi_1, \xi_2, \ldots, \xi_n) = 0, \; 1 \leq i \leq n.$$

donde f_i es la i-ésima componente de F. El método de Newton para este problema puede ser obtenido a partir de suponer que tenemos una aproximación x_k a la solución del sistema de ecuaciones no lineales $F(x) = 0$, y que en la vecindad de x_k la aproximación

$$F(x_k + w) \approx L_k(w) \equiv F(x_k) + F'(x_k)w$$

es apropiada donde $F'(x)$ es la matriz Hessiana de la función F en x. La siguiente aproximación $x_{k+1} = x_k + s_k$ puede ser obtenida requiriendo que el paso s_k satisfaga el sistema de ecuaciones lineales $L_k(w) = 0$. Así el método de Newton intenta mejorar x_0 a través de la iteración

$$x_{k+1} = x_k - F'(x_k)^{-1}F(x_k), \; k \geq 0 \tag{3.6}$$

Al comparar las iteraciones (3.5) y (3.6), observamos que (3.5) es un caso especial de la iteración de Newton (3.6) aplicada a la función $F(x) = \nabla f(x)$. Debido a esta relación es suficiente estudiar la características locales de la iteración (3.6). Los aspectos más importantes de estas características están resumidos en los siguientes dos teoremas.

Teorema 3.2 *Sea $F : R^n \to R^n$ continuamente diferenciable definida sobre un conjunto abierto D, suponga que $F(x^*) = 0$ para algún x^* en D y que $F'(x^*)$ es no singular. Entonces existe un conjunto abierto S tal que para cualquier x_0 en S, las iteraciones (3.6) del método de Newton están bien definidas, permanecen en S, y convergen a x^*.*

y nuestra preocupación aquí son las modificaciones a este método local, las que nos proporcionarán un algoritmo de propósito general.

Un algoritmo que es diseñado para uso general debe ser analizado tan perfectamente como sea posible. El propósito de analizar la convergencia es predecir el comportamiento de la sucesión producido por el algoritmo. Esto implica establecer propiedades de puntos límites y factores de convergencia, estas características, junto con requerimientos de almacenamiento y trabajo computacional, ayudan en la selección de un algoritmo para aplicaciones específicas. Al menos, esperamos un algoritmo de minimización que produzca sucesiones que satisfagan

$$\lim_{k\to\infty} \nabla f(x_k) = 0 \tag{3.3}$$

Esta condición garantiza que cualquier punto límite x^* de $\{x_k\}$ es un punto crítico de f. Esto es todo lo que podemos esperar de los algoritmos que solamente usan información del gradiente. Si un algoritmo requiere información del Hessiano, entonces debemos esperar que las condiciones necesarias de segundo orden del teorema (3.1) sean satisfechas. Esto puede ser hecho asegurando que

$$\lim_{k\to\infty} \inf \lambda_1(\nabla^2 f(x_k)) \geq 0 \tag{3.4}$$

donde $\lambda_1(A)$ es el más pequeño valor propio de una matriz simétrica A. Si (3.3) y (3.4) son satisfechas, entonces cualquier punto límite x^* de $\{x_k\}$ satisface las condiciones del teorema (3.1).

3.1 El Algoritmo Local

El método de Newton puede ser estudiado desde un punto de vista local en el cual suponemos que el punto inicial x_0 es cercano a un mínimo local, lo cual es útil debido a que proporciona información del comportamiento fundamental de este método.

Sea $f : R^n \to R$ una función dos veces continua y diferenciable. La idea del método de Newton consiste en suponer que tenemos una aproximación x_k al mínimo local de f, si aproximamos por la serie de Taylor al rededor de x_k tendremos

$$f(x_k + w) \approx f(x_k) + \psi_k(w)$$

donde

$$\psi_k(w) = \nabla f(x_k)^T w + \frac{1}{2} w^T \nabla^2 f(x_k) w$$

es el modelo cuadrático en x_k. Si esta aproximación es apropiada, entonces es una mejor aproximación $x_{k+1} = x_k + s_k$ puede ser encontrada pidiendo que el paso s_k minimice ψ_k, por lo que s_k debe satisfacer

$$\nabla \psi_k(s_k) = \nabla f(x_k) + \nabla^2 f(x_k) s_k = 0.$$

Prueba 3.2 *Sea α una constante fija en $(0,1)$. Ya que F' es continua en x^* y $F'(x^*)$ es no singular, existe una bola abierta S y una constante μ positiva tal que*

$$\left\|F'(x)^{-1}\right\| \leq \mu, \left\|F'(y) - F'(x)\right\| \leq \frac{\alpha}{\mu},$$

para todo x y y en S. Suponga que $x_k \in S$. Ya que x_{k+1} satisface (3.6) y $F(x^*) = 0$ tenemos que

$$x_{k+1} - x^* = -F'(x_k)^{-1}\left(F(x_k) - F(x^*) - F'(x_k)(x_k - x^*)\right),$$

y por lo tanto

$$\|x_{k+1} - x^*\| \leq \mu \left\|F(x_k) - F(x^*) - F'(x_k)(x_k - x^*)\right\|.$$

Ahora, el teorema fundamental del cálculo integral implica que

$$F(x_k) - F(x^*) - F'(x_k)(x_k - x^*) = \int_0^1 \left[F'(x^* + \xi(x_k - x^*)) - F'(x_k)\right](x_k - x^*)d\xi,$$

y por lo tanto

$$\|x_{k+1} - x^*\| \leq \mu \left\{\max_{0\leq\xi\leq 1} \left\|F'(x^* + \xi(x_k - x^*)) - F'(x_k)\right\|\right\} \|x_k - x^*\|, \qquad (3.7)$$

Así,

$$\|x_{k+1} - x^*\| \leq \alpha \|x_k - x^*\|$$

mientras $x_k \in S$. Ya que $\alpha < 1$, esta última desigualdad implica que si $x_0 \in S$ entonces $x_k \in S$ para toda $k = 1, 2, \ldots$, y que $\{x_k\}$ converge a x^*.□

El teorema (3.2) establece que el método de Newton es localmente convergente en el sentido que si el punto inicial x_0 es suficientemente cercano a la solución x^* entonces el método de Newton converge a x^*. Desafortunadamente, para muchos problemas importantes el dominio de atracción S garantizado por el teorema es demasiado pequeño, por lo que muchas técnicas de investigación han sido desarrolladas para cubrir esta debilidad del método de Newton. Para sistemas de ecuaciones no li-neales éste es un campo de investigación particularmente activo, ya que se ha generado mucho interés por los recientes métodos de globales de Newton. Para el problema de minimización, la situación toma una mejor forma, examinaremos las dos principales aproximaciones del método global de Newton.

Aunque el teorema (3.2) es de importancia innegable, no cuenta la historia completa, ya que no es suficiente saber que la sucesión converge, si la razón de convergencia es demasiado lenta, no podremos permitirnos verla converger. Normalmente, cuando analizamos métodos iterativos, también estamos interesados en decir, tanto como sea posible, sobre la razón esperada de convergencia de la sucesión producida

por el método. Un algoritmo de optimización razonable sería capaz de generar sucesiones $\{x_k\}$ linealmente convergentes en el sentido de que

$$\|x_{k+1} - x^*\| \leq \alpha \|x_k - x^*\|, \; k \geq 0, \tag{3.8}$$

para alguna constante α en $(0,1)$. Si α es pequeña entonces (3.8) es adecuada, pero si α es cercana a la unidad, por decir $\alpha \geq 0.9$, entonces (3.8) no es satisfactorio, por que la convergencia puede ser demasiado lenta.

Para muchos algoritmos de optimización que utilizan información de segundo orden, es posible establecer un resultado más fuerte que (3.8). Una sucesión converge cuadráticamente a x^* si

$$\|x_{k+1} - x^*\| \leq \beta \|x_k - x^*\|^2, \; k \geq 0, \tag{3.9}$$

para alguna constante $\beta > 0$. Ya que

$$\frac{\|x_{k+1} - x^*\|}{\|x^*\|} \leq (\beta \|x^*\|) \left(\frac{\|x_{k+1} - x^*\|}{\|x^*\|} \right)^2,$$

la convergencia cuadrática implica que el número de dígitos significativos de x_k cuando se aproxima a x^* se duplica en cada iteración. Típicamente, en cuanto dos dígitos significativos son obtenidos, las siguientes tres iteraciones producirán rigurosamente seis dígitos significativos.

Hay un punto medio entre (3.8) y (3.9). Una sucesión $\{x_k\}$ converge superlinealmente a x^* si

$$\|x_{k+1} - x^*\| \leq \beta_k \|x_k - x^*\|^2, \; k \geq 0, \tag{3.10}$$

para alguna sucesión $\{\beta_k\}$ que converga a cero. Resulta claro que una sucesión que converge superlinealmente también convergerá linealmente, y que una sucesión que converga cuadráticamente convergerá superlinealmente. También observe que ya que

$$|\|x_{k+1} - x^*\| - \|x_k - x^*\|| \leq \|x_{k+1} - x^*\|,$$

se sigue que

$$\lim_{k \to +\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 1$$

donde $\{x_k\}$ converge superlinealmente a x^*. Esta es una propiedad importante debido a que $\|x_{k+1} - x^*\|$ puede usarse para estimar la distancia de x_k a x^*.

A un método iterativo se le asigna una razón de convergencia, si es posible mostrar que toda sucesión convergente producirá al menos esta razón. Normalmente algunas restricciones razonables son impuestas sobre el dominio de aplicación para el método de modo de obtener una útil valoración de esta razón. El método de Newton converge normalmente de forma cuadrática.

Teorema 3.3 *Sea $F : R^n \to R^n$ satisface las hipótesis del teorema (3.2). Entonces la sucesión $\{x_k\}$ producida por la iteración (3.6) converge superlinealmente a x^*. Además, si*

$$\left\|F'(x) - F'(x^*)\right\| < \kappa \|x - x^*\|, \; x \in D, \tag{3.11}$$

para alguna $\kappa > 0$, entonces la sucesión converge cuadráticamente a x^*.

Prueba 3.3 *La convergencia de la sucesión $\{x_k\}$ fue establecida en el teorema (3.2). sólo resta establecer la razón de convergencia. Para este fin definimos*

$$\beta_k \equiv \mu \left\{ \max_{0 \le \xi \le 1} \left\|F'(x^* + \xi(x_k - x^*)) - F'(x_k)\right\| \right\},$$

y suponemos que $x_0 \in S$ con μ y S definidos en la demostración del teorema (3.2). La hipótesis sobre F' en x^* y la convergencia de la sucesión a x^* implica que $\{\beta_k\}$ converge a cero. Ya que la desigualdad (3.7) muestra que

$$\|x_{k+1} - x^*\| \le \beta_k \|x_k - x^*\|,$$

esto prueba que $\{x_k\}$ converge superlinealmente a x^*. Además, si (3.11) se cumple, entonces

$$\beta_k \le 2\mu\kappa \|x_k - x^*\|,$$

y por lo tanto $\{x_k\}$ converge cuadráticamente a x^*.□

Observe que la condición de Lipschistz (3.11) es necesaria para garantizar que el método de Newton sea cuadráticamente convergente. Por ejemplo, el método de Newton aplicado a un problema unidimensional definido por

$$f(\xi) = \xi \left[1 + \left(\frac{1}{\log(|\xi|)}\right)\right], \; \xi \neq 0, \; f(0) = 0,$$

es justamente superlinealmente convergente a $\xi^* = 0$; esto es, si $\{\xi_k\}$ es una sucesión generada por el método de Newton, entonces el radio

$$\frac{|\xi_{k+1}|}{|\xi_k|^{1+\rho}}$$

no está acotado para $\rho > 0$.

Los resultados de la razón de convergencia son utilizados para comparar algoritmos, exigiendo que el algoritmo considerado como superior tenga la mas alta razón de convergencia. Exigencias de este tipo deben hacerse con cuidado debido a que estos resultados son asintóticos. Generalmente no es posible establecer la magnitud de las constantes que aparecen en las expresiones como (3.8), (3.9) y (3.10). Además,

la razón de convergencia no mide necesariamente el trabajo invertido en calcular x_{k+1} a partir de x_k, y en muchos casos esta información es decisiva para la elección del algoritmo. Por ejemplo, considere la clase de métodos Quasi-Newton descritos en la sección (3.5). Se sabe que las sucesiones generadas por estos métodos convergen superlinealmente, y no cuadráticamente. Sin embargo, ya que no requieren del cálculo de la matriz Hessiana, los métodos Quasi-Newton son considerados a menudo superiores a los métodos Newton que convergen cuadráticamente.

3.2 Propiedades de las Funciones Cuadráticas

Las funciones cuadráticas juegan un papel importante en el desarrollo de algoritmos para problemas de optimización. Por ejemplo, hemos visto que en una vecindad de un mínimo local de una función $f : R^n \to R$, el método de Newton puede ser obtenido tomando el paso que minimice el modelo cuadrático local

$$\psi_k(w) = \nabla f(x_k)^t w + \frac{1}{2} w^t \nabla^2 f(x_k) w \tag{3.12}$$

de la reducción esperada de la función f, por lo que es importante entender las propiedades de las funciones cuadráticas y proporcionar algoritmos numéricamente estables que las minimicen.

El siguiente resultado describe la minimización sin restricciones de las funciones cuadráticas.

Lema 3.1 *Sea $\psi : R^n \to R$ una función cuadrática de la forma*

$$\psi(w) = g^t w + \frac{1}{2} w^t B w \tag{3.13}$$

donde $g \in R^n$ y $B \in R^{n \times n}$ es una matriz simétrica.

1.) La función cuadrática ψ tiene un mínimo si y B es semidefinida positiva y g pertenece al rango de B.
2.) La función cuadrática ψ tiene un único mínimo si y sólo si B es positiva definida.
3.) Si B positiva semidefinida entonces toda solución a la ecuación $Bp = -g$ es un mínimo global.

Prueba 3.1 *Suponga que B es positiva semidefinida con g en el rango de B. Entonces $Bp = -g$ tiene una solución p, y así*

$$\psi(p + w) = \psi(p) + (Bp + g)^t w + \frac{1}{2} w^t B w = \psi(p) + \frac{1}{2} w^t B w \geq \psi(p) \tag{3.14}$$

para toda $w \in R^n$. Por otro lado, si p es un mínimo de ψ entonces el teorema (3.1) implica que $Bp + g = \nabla\psi(p) = 0$, y que $B = \nabla^2\psi(p)$ es semipositivo definido. Para establecer (2) y (3) observamos que (3.14) es válida cuando $Bp = -g$ y B es positiva semidefinida, y que la desigualdad estricta se sostiene para $w \neq 0$ si y sólo si B es positiva definida.□

Dada la función cuadrática ψ, existe un procedimiento numérico excelente para encontrar su mínimo. Primero intentamos calcular la factorización de Cholesky de B. Esta factorización existe si y sólo si B es positiva semidefinida, y en este caso tenemos una matriz triangular superior tal que

$$B = R^t R.$$

Si un elemento negativo es encontrado en la diagonal durante la factorización, entonces B no es positiva semidefinida y por lo tanto el lema (3.1) muestra que la función cuadrática ψ no tiene mínimo. Si la factorización es exitosa y R es no singular, entonces el mínimo es calculado resolviendo el sistema $Bp = -p$, o equivalentemente,

$$R^t v = -g, \; Rp = v.$$

Si la factorización es exitosa pero R es singular entonces B es positiva semidefinida y singular. Todavía es posible calcular una solución p, pero desde el punto de vista numérico, este cálculo es inestable debido a que pequeñas perturbaciones pueden transformar a B en una matriz definida positiva o en una matriz indefinida.

El teorema (3.1) muestra que en una vecindad de un mínimo local de f, podemos esperar que la matriz Hessiana sea positiva definida y entonces el lema (3.1) muestra que el modelo local cuadrático (3.12) tiene un único mínimo. Así, en este caso, el mínimo del modelo cuadrático es un paso razonable para el algoritmo de minimización. Sin embargo, fuera del mínimo local la matriz Hessiana $\nabla^2 f(x)$ puede tener valores propios negativos y entonces el lema (3.1) nos dice que este modelo cuadrático local no tienen mínimo. De hecho, el modelo no está acotado inferiormente. Hay va-rios remedios para este problema. Una posibilidad es modificar el modelo agregándole una matriz positiva semidefinida $E(x)$ tal que

$$\nabla^2 f(x) + E(x)$$

sea positiva definida. Cuando B es reemplazado por esta matriz el cálculo del paso puede hacerse como se describió anteriormente.

Otro posible remedio es restringir la región para la cual el modelo es apropiado. Localmente el modelo todavía proporciona una aproximación excelente a la reducción esperada de f, es razonable restringir ψ a la bola $\{w : \|w\| \leq \Delta\}$ para algún $\Delta > 0$, y calcular el paso como el mínimo de ψ en esta bola. Los siguientes resultados caracterizan las soluciones globales al problema de minimización de una función cuadrática en esta región restringida.

Lema 3.2 *Dados $\psi : R^n \to R$ la función cuadrática definida por (3.13) y $\Delta > 0$. Un punto $p \in R^n$ resuelve el problema*

$$\min\{\psi(w) : \|w\| \leq \Delta\} \tag{3.15}$$

si y sólo si $\|p\| \leq \Delta$ y existe un $\lambda \geq 0$ tal que

$$(B + \lambda I)p = -g, \ \lambda(\Delta - \|p\|) = 0, \tag{3.16}$$

con $B + \lambda I$ positiva semidefinida.

Prueba 3.2 *Suponga que λ y p satisfacen (3.16) con $\|p\| \leq \Delta$ y $B + \lambda I$ positiva semidefinida. Entonces el lema (3.1) implica que p minimiza la función*

$$\hat{\psi}(w) = g^t w + \frac{1}{2} w^t (B + \lambda I) w.$$

Así $\hat{\psi}(w) \geq \hat{\psi}(p)$, lo cual implica que

$$g^t w + \frac{1}{2} w^t B w \geq g^t p + \frac{1}{2} p^t B p + \frac{\lambda}{2}(p^t p - w^t w) \tag{3.17}$$

para toda $w \in R^n$. Ya que $\lambda p^t p = \lambda \Delta^2$ y $\lambda \geq 0$, se sigue de (3.17) que $\psi(w) \geq \psi(p)$ cuando $\|w\| \leq \Delta$, así p debe resolver (3.15).
Ahora suponga que p resuelve (3.15). Si $\|p\| < \Delta$ entonces p es un mínimo de ψ, así el lema (3.1) implica que (3.16) se cumple con $\lambda = 0$, y que B es positiva semidefinida. Si $\|p\| = \Delta$ entonces p debe también resolver el problema sujeto a restricciones de $\min\{\psi(w) : \|w\| = p\}$. Por lo tanto, el método de Lagrange asegura la existencia de λ tal que

$$\nabla L(p) = 0, \text{ donde } L(w) = \psi(w) + \frac{\lambda}{2}(w^t w - \Delta^2).$$

Esto implica que (3.16) se cumple para λ y p. Además, ya que p resuelve (3.15) tenemos que (3.17) es válida para estos λ y p cuando $\|w\| = \|p\|$. Usamos (3.16) para reemplazar g y entonces rearreglar los términos en (3.17) tenemos que

$$\frac{1}{2}(w - p)^t (B + \lambda I)(w - p) \geq 0$$

para todo w con norma $\|p\|$. Se sigue de esta desigualdad que $B + \lambda I$ es positiva semidefinida. Para mostrar que $\lambda \geq 0$, observe que el lema (3.1) implica que (3.17) es válido para toda $w \in R^n$. Ahora, si λ no es positiva entonces (3.17) implica que $\psi(w) \geq \psi(p)$ cuando $\|w\| \geq \|p\|$. Ya que p resuelve (3.15) debemos tener que p es un mínimo sin restricciones de ψ y entonces el lema (3.1) implica que $\lambda = 0$. De aquí, $\lambda \geq 0$ como pedimos.□

El cálculo de la aproximación numérica de la solución para (3.15) debe ser hecho con cuidado. Una complicación inmediata es que, debido a las restricciones no lineales, no puede haber ningún método general directo para resolver (3.15). De hecho, si $g = 0$ y B tiene un valor propio negativo entonces una solución p para (3.15) debe ser un vector propio de norma Δ correspondiente al más pequeño valor propio de B. Por lo tanto, un método general para resolver (3.15) debe resolver un problema de valores propios simétricos, en este caso especial. Debido a que estamos preocupados por encontrar soluciones globales para (3.15) puede parecer que un algoritmo, el cual resuelva (3.15), está destinado a ser demasiado costoso. Sin embargo, Mostraremos que existe un algoritmo el cual produce una solución casi óptima para (3.15) en todos los casos y sólo usa unas pocas iteraciones.

La solución de (3.15) es lineal si no hay soluciones en la frontera de $\{w : \|w\| \leq \Delta\}$. De hecho, no es difícil probar que (3.15) no tiene solución cuando $\|p\| = \Delta$ si y sólo si B es positiva definida y $\|B^{-1}g\| < \Delta$.

Si (3.15) tiene una solución en la frontera de $\{w : \|w\| \leq \Delta\}$, entonces el lema (3.2) muestra que es razonable esperar que la ecuación no lineal

$$\|p_\alpha\| = \Delta, \tag{3.18}$$

donde

$$p_\alpha = -(B + \alpha I)^{-1} g$$

tenga una solución $\alpha^* = \lambda \geq 0$ en $(-\lambda, \infty)$, donde $\lambda_1 \leq 0$ es el más pequeño valor propio de B. Observe que (3.18) es un problema de cero de una función en α que puede ser resuelto, por ejemplo, por el método de Newton. Sin embargo, ya que cada evaluación de p_α requiere la solución de un sistema de ecuaciones lineales, es importante resolver (3.18) con pocas evaluaciones de p_α.

Para resolver (3.18), Reinsch, y Hebden observaron independientemente que pueden obtenerse grandes ventajas a partir del hecho de que la función $\|p_\alpha\|^2$ es una función racional en α con polos de segundo orden en el subconjunto de los valores propios negativos de la matriz simétrica B. Para ver esto considere la descomposición

$$B = Q\Lambda Q^t \; con \; \Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_n) \; y \; Q^tQ = I,$$

y observe que

$$\|p_\alpha\|^2 = \left\|Q(\Lambda + \alpha I)^{-1} Q^t g\right\|^2 = \sum_{j=1}^{n} \frac{\gamma_j^2}{(\lambda_j + \alpha)^2} \tag{3.19}$$

donde γ_i es la i-ésima componente de Q^tg. El conocimiento de la forma funcional (3.19) muestra que el método de Newton no puede ser muy eficiente si es aplicado a la función

$$\varphi_1(\alpha) = \|p_\alpha\| - \Delta.$$

Una razón para esto es que φ_1 tiene un polo en λ_1, y así el método de Newton tiende a ejecutarse pobremente cuando la solución de (3.18) es cercana a $-\lambda_1$. Reinsch y Hebden sugieren que es más eficiente aplicar el método de Newton a la función

$$\varphi_2 = \frac{1}{\Delta} - \frac{1}{\|p_\alpha\|}.$$

Una de las ventajas de utilizar esta función es que no tiene polos, y es casi lineal cerca de la solución de (3.18). Las iteraciones del método de Newton aplicadas para encontrar un cero de φ_2 tiene la siguiente forma.

ALGORITMO 1.0

Sea λ_0 y $\Delta > 0$.

Para $k = 0, 1, \ldots$

1. Factorizar $B + \lambda_k I = R_k^t R_k$.
2. Resolver $R_k^t R_k p_k = -g$.
3. Resolver $R_k^t q_k = p_k$.
4. $\lambda_{k+1} = \lambda_k + \left(\frac{\|p_k\|}{\|q_k\|}\right)^2 \left(\frac{\|p_k\| - \Delta}{\Delta}\right)$.

Si tomamos ciertas precauciones, entonces estas iteraciones básicas pueden ser usadas para resolver (3.15) en muchos casos. Sin embargo, cuando B es indefinido hay casos en los cuales la ecuación (3.18) no tienen solución en $(-\lambda_1, \infty)$, y entonces el algoritmo (1.0) falla. Esto sucede, por ejemplo, cuando $g = 0$ y B es indefinido. Puede también suceder cuando $g \neq 0$, como lo ilustra el siguiente ejemplo. Si

$$B = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}, \quad g = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

entonces $\lambda_1 = -1$, y si $\alpha > 1$ entonces $\|p_\alpha\|^2 < \frac{1}{2}$. En nuestros ejemplos g es ortogonal al espacio propio de B correspondiente al más pequeño valor propio. Esto es típico; g debe ser ortogonal al espacio propio

$$S_1 \equiv \{z : Bz = \lambda_1 z,\ z \neq 0\}$$

correspondiente al más pequeño valor propio de B cuando (3.18) no tiene solución en $(-\lambda_1, \infty)$. Para ver esto basta observar que si g no es ortogonal a S_1, entonces $\gamma_1 \neq 0$ en (3.19), y por lo tanto

$$\lim_{\alpha \to -\lambda_1} \|p_\alpha\| = \infty, \quad \lim_{\alpha \to +\infty} \|p_\alpha\| = 0.$$

Ya que pequeñas perturbaciones de g conducen a γ_1 diferente de cero, es tentador ignorar el caso en el que g es ortogonal a S_1. Sin embargo, en muchos casos g es casi ortogonal a S_1, y en estos casos un algoritmo basado completamente en el método de Newton pudiera requerir de un gran número de iteraciones. Esto no es aceptable ya que se requiere factorizar una matriz en cada iteración.

Varios algoritmos han sido propuestos para la solución numérica de (3.15), pero Gay fue el primero en mostrar que su algoritmo produce una solución casi óptima. El algoritmo de Gay, sin embargo, puede requerir de un gran número de iteraciones cuando g es ortogonal a S_1, y fracasa cuando $g = 0$ y B es indefinida. Moré y Sorensen han mejorado el algoritmo de Gay, y sus resultados numéricos muestran que es posible producir una solución casi óptima para (3.15) en todos los casos y con sólo unas pocas iteraciones.

3.3 Métodos de la Región de Confianza

En el método de Newton con búsqueda en línea el Hessiano es modificado cuando no es suficientemente definido positivo. Esta modificación al modelo cuadrático garantiza convergencia, pero parece ignorar el papel del modelo cuadrático como una aproximación local a la función objetivo, por lo que una alternativa es que este modelo no sea modificado, sino que sea sólo considerado en una vecindad que llamaremos región de confianza; esta técnica fue brevemente mencionada como motivación al lema (3.2), su uso para la globalización del método de Newton ha generado métodos seguros con propiedades fuertes de convergencia.

Sea $f : R^n \to R$ una función dos veces diferenciable y continua. En el método de Newton con la estrategia de región de confianza, cada iteración x_k tiene una cota Δ_k tal que

$$f(x_k + w) \approx f(x_k) + \psi_k(w), \ \|w\| \leq \Delta_k,$$

donde

$$\psi_k(w) = \nabla f(x_k)^t w + \frac{1}{2} w^t \nabla^2 f(x_k) w$$

es el modelo cuadrático de f dentro de una vecindad de la iteración x_k. Esto sugiere que puede ser conveniente calcular un paso s_k el cual resuelva aproximadamente el problema

$$\min\{\psi_k(w) : \|w\| \leq \Delta_k\}. \tag{3.20}$$

Si el paso es satisfactorio en el sentido que $x_k + s_k$ produce una reducción suficiente de f, entonces Δ_k puede ser incrementado, si el paso es insatisfactorio entonces Δ_k podría ser decrementado. El siguiente algoritmo expresa estas ideas con más detalle.

ALGORITMO 2.0

Sea $0 < \mu < \eta < 1$ y sean $0 < \gamma_1 < \gamma_2 < 1 < \gamma_3$ constantes especificadas.

Sean $x_0 \in R^n$ y $\Delta_0 > 0$.

Para $k = 0, 1, \ldots$ hasta "converger".

1. Calcule $\nabla f(x_k)$ y $\nabla^2 f(x_k)$.
2. Determine una solución aproximada s_k para el problema (3.20).
3. Calcule $\rho_k = \frac{f(x_k+s_k)-f(x_k)}{\psi_k(s_k)}$.
4. Si $\rho_k \leq \mu$ entonces reemplace Δ_k por un número en el intervalo $[\gamma_1\Delta_k, \gamma_2\Delta_k]$ y regrese a (2).
5. Calcule $x_{k+1} = x_k + s_k$.
6. Si $\rho_k \leq \eta$ entonces elija $\Delta_{k+1} \in [\gamma_2\Delta_k, \Delta_k]$, de otro modo elija $\Delta_{k+1} \in [\Delta_k, \gamma_3\Delta_k]$.

Esta es una forma básica del método de Newton de región de confianza. Una variación interesante de este algoritmo incluye una matriz de escala para las variables. Con esta variación el subproblema (3.20) es reemplazado por

$$\min\{\psi_k(w) : \|D_k w\| \leq \Delta_k\},$$

donde D_k es una matriz no singular. Sin embargo, es importante observar que todos los resultados se sostienen para esta variación si $\{D_k\}$ está uniformemente acotada por números condición. Tal modificación puede ser muy importante en la práctica cuando las unidades de las variables tienen escalas muy diferentes.

No estamos interesados en resolver el problema (3.20) con una gran exactitud, sino que, estamos interesados en proporcionar condiciones relajadas para aceptar una solución aproximada s_k al problema (3.20),dichas condiciones deben ser suficientes para forzar a la sucesión $\{x_k\}$ generada por el algoritmo (2.0) a converger. De hecho,

existen condiciones las cuales garantizan mucho más que convergencia. Si ψ_k^* es el valor óptimo de (3.20), y si la solución aproximada s_k para (3.20) satisface

$$-\psi_k(s_k) \geq \beta_1 |\psi_k^*|, \quad \|s_k\| \leq \beta_2 \Delta_k, \tag{3.21}$$

para $\beta_1 > 0$ y $\beta_2 > 0$ constantes especificadas, entonces es posible probar que bajo ciertas condiciones aceptables sobre f, la sucesión $\{x_k\}$ es convergente al punto x^* con $\nabla f(x^*) = 0$ y $\nabla^2 f(x^*)$ positivo semidefinido.

No es difícil obtener un vector s_k que satisfaga (3.21), aunque como se mencionó en secciones anteriores, esto requiere de atender un gran número de detalles. Dado un σ en $(0,1)$, el algoritmo de Moré y Sorensen, por ejemplo, encuentra un vector s_k tal que

$$\psi_k(s_k) - \psi_k^* \leq \sigma(2-\sigma)|\psi_k^*|, \quad \|s_k\| \leq (1+\sigma)\Delta_k,$$

para $\psi_k^* \neq 0$. Por supuesto, si $\psi_k^* = 0$, entonces $\nabla f(x_k) = 0$ y $\nabla^2 f(x_k)$ positivo semidefinido, así el algoritmo (2.0) termina en x_k. También es importante mencionar que si $\sigma = 0.1$ entonces el costo del algoritmo es bastante bajo. La solución aproximada de cada problema requiere en promedio de menos de dos factorizaciones de una matriz simétrica definida positiva de orden n.

La condición (3.21) puede ser expresada de forma alternativa la cual es más conveniente para pruebas de convergencia. Si $p_k \in R^n$ es una solución para el problema (3.20) entonces el lema (3.2) implica que existe un parámetro $\lambda_k \geq 0$ tal que

$$(\nabla^2 f(x_k) + \lambda_k I)p_k = -\nabla f(x_k), \qquad \lambda_k(\Delta_k - \|p_k\|) = 0.$$

Sea $R_k^t R_k$ la factorización de Cholesky de $\nabla^2 f(x_k) + \lambda_k I$. Entonces

$$|\psi_k^*| = \frac{1}{2}\left(\|R_k p_k\|^2 + \lambda_k \Delta_k^2\right). \tag{3.22}$$

Esta expresión para ψ_k^* muestra que si (3.21) se cumple entonces

$$-\psi_k(s_k) \geq \frac{1}{2}\beta_1\left(\|R_k p_k\|^2 + \lambda_k \Delta_k^2\right), \tag{3.23}$$

y así las iteraciones generadas por el algoritmo (2.0) satisfacen

$$f(x_k) - f(x_{k+1}) \geq \frac{1}{2}\mu\beta_1\left(\|R_k p_k\|^2 + \lambda_k \Delta_k^2\right). \tag{3.24}$$

Estas dos desigualdades son esenciales para probar el siguiente resultado.

Teorema 3.4 *Sea $f : R^n \to R$ una función dos veces diferenciable y continua sobre un conjunto abierto D, y suponga que el punto inicial x_0 es tal que el conjunto de nivel*

$$\Omega = \{x \in D : f(x) \leq f(x_0)\}$$

es compacto. Si la sucesión $\{x_k\}$ es producida por el algoritmo (2.0) donde s_k satisface (3.21), entonces el algoritmo termina en $x_l \in \Omega$ debido a que $\nabla f(x_l) = 0$ y $\nabla^2 f(x_l)$ es positivo semidefinido, o $\{x_k\}$ tiene un punto límite x^* en Ω con $\nabla f(x^*) = 0$ y $\nabla^2 f(x^*)$ positiva semidefinida.

Omitimos la demostración de este teorema debido a su complejidad. Si desea estudiar el contenido de la demostración por favor revise Moré y Sorensen.

Los resultados que hemos establecido son sólo una muestra de los resultados de convergencia disponibles para el algoritmo (2.0) bajo la suposición (3.21) para s_k. Este teorema extiende el resultado de Fletcher y Sorensen. Los siguientes resultados adicionales son conocidos.

a) La sucesión $\{\nabla f(x_k)\}$ converge a cero.

b) Si x^* es un punto límite aislado de $\{x_k\}$ entonces $\nabla^2 f(x^*)$ es positivo semidefinido.

c) Si $\nabla^2 f(x^*)$ es no singular para algún punto límite x^* de $\{x_k\}$ entonces $\{x_k\}$ converge a x^*.

Thomas probó a), mientras Moré y Sorensen establecieron b) y c) como extensión del resultado debido a Sorensen. De estos resultados, b) es característico de la aproximación a la región de confianza, y es el único resultado que no se sostiene para el método de Newton con búsqueda en línea. Esta diferencia entre las dos aproximaciones es de importancia teórica. Desde un punto de vista practico, sin embargo, es posible argumentar que una diferencia más importante es que con una aproximación de búsqueda en línea la búsqueda del valor más pequeño de la función ocurre en un subespacio unidimensional, mientras que con la aproximación por búsqueda de la región de confianza la búsqueda no se restrìnge a un subespacio de dimensión baja.

3.4 Aproximación al Gradiente

Puede verse que muchas de las consideraciones son sólo relevantes cuando el método conoce de manera explicita o es capaz de calcular el vector $\nabla f(x_k)$. También resulta en la práctica que los métodos con primera derivada son más confiables que los métodos sin derivadas, en los cuales $\nabla f(x_k)$ no está disponible. En realidad los mejores métodos sin derivadas parecen ser aquellos que estiman las derivadas por diferencias finitas:

$$\nabla f_i(x) \simeq (f(x + he_i) - f(x))/h \tag{3.25}$$

$$\nabla f_i(x) \simeq (f(x + he_i) - f(x - he_i))/2h \tag{3.26}$$

para los métodos con primeras derivadas es muy fácil para el usuario equivocarse al programar la fórmula de la derivada y es por lo tanto muy sabio verificar la fórmula sis-

tematicamente usando 3.25 o 3.26. En nuestra implementación utilizamos la fórmula 3.26 debido a que proporciona una mayor exactitud.

3.5 Aproximaciones a la Matriz Hessiana

Todos los métodos descritos en las secciones anteriores requieren del cálculo de la matriz Hessiana. Esta puede ser una tarea difícil y propensa a errores, y en algunos casos las expresión analítica de la matriz puede no estar disponible. ¿Qué podemos hacer en estos casos?

Una manera obvia de vencer estas dificultades es aproximar la matriz Hessiana a través de diferencias de gradientes. Sin embargo, hay varias cosas que debemos considerar. ¿Qué aproximación por diferencias deberemos usar? ¿Qué tan grande debiera ser el parámetro de diferencia? ¿Cómo es afectada la ejecución del método de minimización cuando la aproximación por diferencias es ejecutada?

Los dos más comunes tipos de aproximaciones por diferencias usados son el de diferencias delanteras y el de diferencias centrales. La aproximación por diferencias delanteras está basada en la expansión de la serie de Taylor

$$\left(\frac{1}{\alpha}\right)[\nabla f(x+\alpha p) - \nabla f(x)] = \nabla^2 f(x)p + O(\alpha), \tag{3.27}$$

mientras la aproximación por diferencias centrales está basada en

$$\left(\frac{1}{2\alpha}\right)[\nabla f(x+\alpha p) - 2\nabla f(x) + \nabla f(x-\alpha p)] = \nabla^2 f(x)p + O(\alpha^2).$$

En trabajos de optimización, (3.27) es más común, debido a que requieren pocas evaluaciones del gradiente y normalmente proporcionan la exactitud necesaria. Si las diferencias centrales son usadas, una aproximación $A(x)$ a la matriz Hessiana en algún $x \in R^n$ puede ser obtenida haciendo

$$A(x)e_j = \left(\frac{1}{\alpha_j}\right)[\nabla f(x+\alpha e_j) - \nabla f(x)],\ 1 \le j \le n$$

para algún parámetro de diferencia $\alpha_j \neq 0$. Desafortunadamente, esta aproximación no necesariamente proporciona una matriz simétrica. Esta importante caraterística de la matriz Hessiana puede ser obtenida usando la matriz simétrica

$$\frac{1}{2}\left[A(x) + A(x)^t\right]$$

como la aproximación a la matriz Hessiana en x.

La elección de diferentes parámetros presenta un dilema. De hecho para preservar la razón de convergencia superlineal disfrutada por el método de Newton es necesario forzar el parámetro de diferencia a cero. Sin embargo, cuando el parámetro

de diferencia α_j se vuelve pequeño, la diferencia pierde significado debido a la cancelación. Para prevenir esta pérdida de significado, el parámetro de diferencia debe quedar por encima de cierto valor umbral. Este dilema puede resolverse normalmente en la práctica, debido a que no es necesario proporcionar una aproximación a la matriz Hessiana de gran exactitud. Si la aproximación a la matriz Hessiana tiene una exactitud comparable a la exactitud deseada en la solución al problema de optimización, entonces la convergencia normalmente (en la práctica) es cuadrática. Aproximaciones menos exactas hacen decrecer la razón de convergencia pero impiden la convergencia. Estas observaciones suponen que el gradiente es evaluado exactamente; si éste no es el caso, podemos no ser capaces de calcular la dirección de descenso.

Las técnicas de elegir el parámetro de diferencias en (3.27) requiere de información acerca de ∇f en una vecindad de x, la cual es obtenida evaluando ∇f en varios puntos cercanos a x. Para muchos problemas prácticos sería demasiado costoso obtener esta información de forma exacta en cada iteración. Una estrategia sensible para un algoritmo de optimización consiste en elegir el parámetro de diferencias en un x común (posiblemente el punto inicial x_0), y esta elección es utilizada hasta que se juzga inapropiada. El parámetro de diferencias es sólo calculado otra vez cuando la calidad de la aproximación comienza a degradarse.

Existen varios algoritmos para seleccionar el parámetro de diferencias en un punto. Mencionaremos algunas ideas generales de estos algoritmos. En el caso unidimensional las principales ideas son muy claras, así que consideraremos una función diferenciable $\varphi : R^n \to R$, donde $\varphi_c(\alpha)$ denota el valor calculado de $\varphi(\alpha)$, y sea

$$\varepsilon(\alpha) = \varphi_c(\alpha) - \varphi(\alpha)$$

el error (absoluto) en el valor calculado. La suavidad de φ_c depende del método usado para evaluar φ en la computadora, pero en todos los casos φ_c es una función escalonada. Una razón para esto es que una computadora con l dígitos decimales de exactitud no distingue entre números con los primeros l dígitos iguales. Mencionamos este hecho debido a que implica que φ_c no es diferenciable. Con estas observaciones en mente, observe que el problema es determinar un α tal que

$$\left(\frac{1}{\alpha}\right)[\varphi_c(\alpha) - \varphi_c(0)] \tag{3.28}$$

sea cercano a $\varphi'(0)$. Si asumimos que tenemos una vecindad abierta I de $\alpha = 0$, y una cota ε_0 tal que

$$|\varepsilon(\alpha)| \leq \varepsilon_0, \quad \alpha \in I,$$

entonces no es difícil determinar el parámetro de diferencias. Observe que la expansión de la serie de Taylor de φ muestra que

$$\varphi_c(\alpha) - \varphi_c(0) - \alpha\varphi'(0) = \frac{1}{2}\varphi''(\xi)\alpha^2 + [\varepsilon(\alpha) - \varepsilon(0)]$$

para algún ξ con $|\xi| < |a|$, y por lo tanto

$$\left|\left(\frac{1}{a}\right)[\varphi_c(a) - \varphi_c(0)] - \varphi'(0)\right| \leq \frac{1}{2}\eta_0 |a| + \frac{2\varepsilon_0}{|a|},$$

donde η_0 es una cota para φ'' en I. Esta cota del error entre (3.28) y $\varphi'(0)$ tiene un comportamiento correcto cualitativo. Si a es demasiado pequeña entonces el error es dominado por ε_0, mientras que si a es demasiado grande entonces el error es determinado por la curvatura de φ. Es razonable elegir a de manera que esta cota sea mínima, y esto nos lleva a elegir

$$a = 2\left(\frac{\varepsilon_0}{\eta_0}\right)^{\frac{1}{2}}. \tag{3.29}$$

Los algoritmos para determinar ε_0 y η_0 pueden basarse en el trabajo de Hamming. La idea básica es que la 4ta y 5ta diferencia de orden de φ_c son una medida de ε_0 y que la 2da deferencia de orden puede usarse para estimar η_0. Es necesario tomar algunas precauciones, pero en general encontramos que un algoritmo basado en estas ideas y en (3.29) es muy efectivo.

Capítulo 4
MÉTODOS QUASI-NEWTON

Para algunos problemas la función objetivo y su gradiente son muy costosos de calcular, de forma que no estamos dispuestos a calcular la aproximación a la matriz Hessiana por diferencias. Estos no son necesariamente problemas de grandes dimensiones, por ejemplo, el problema puede ser minimizar la norma-L_2 de la solución para una ecuación diferencial que depende de pocos parámetros. En este caso cada evaluación de la función requerida por el método de optimización normalmente requiere de la solución de una ecuación diferencial.

En un esfuerzo por reducir los requerimientos computacionales del método de Newton, Davidon introdujo una idea revolucionaria la cual proporciona un camino para aproximar la matriz Hessiana usando sólo la información del gradiente reunida en cada iteración. Esta idea ha llevado a una clase de métodos sumamente exitosos, los cuales hoy en día son llamados métodos Quasi-Newton. Existe una enorme cantidad de literatura acerca de los métodos Quasi-Newton; en esta sección proporcionaremos una breve introducción a los dos miembros más poderosos de esta clase y compararemos métodos Quasi-Newton con métodos en la familia de Newton.

En términos muy simples un método Quasi-Newton puede ser llamado un método de "gana mientras aprendes". Puede ser comparado con métodos de la familia de Newton gracias a la manera de aproximar el Hessiano. En métodos Quasi-Newton, la aproximación al Hessiano debe satisfacer la ecuación Quasi-Newton. Para obtener esta ecuación, suponga que tenemos una aproximación definida positiva B_k al Hessiano de f en x_k.

El método de Newton es importante porque da un estándar para comparar métodos de convergencia rápida para resolver el problema de minimización de una función. Una forma de caracterizar la convergencia superlineal es que el vector de descenso calculado debe aproximarse al de Newton en magnitud y dirección. Los aspectos positivos y negativos del método de Newton se pueden resumir como sigue: En el lado positivo, el algoritmo es localmente y cuadráticamente convergente para f suficientemente suave, esto es, para x_0 suficientemente cercana a un mínimo local x^*,

existe una constante k tal que

$$\|x_{k+1} - x^*\| \leq k\|x_k - x^*\|^2$$

Sus desventajas son, sin embargo:

1. No es globalmente convergente, es decir, no converge a partir de cualquier punto inicial, independientemente de cuan lejos esté la solución.
2. No está definido en puntos donde el Hessiano es singular.
3. Para problemas no convexos, no genera necesariamente una sucesión de direcciones de descenso.
4. Debe resolverse en cada iteración un sistema lineal n dimensional.
5. Debe darse la expresión analítica del Hessiano.

Los métodos de Quasi-Newton son una variación del método de Newton, que conservan sus ventajas y evitan las dificultades que se presentan con el Hessiano al construir una aproximación al mismo según avanza el algoritmo, pero que resulta, a la vez fácil de calcular, mediante una actualización que se obtiene de la iteración precedente, ya que esta es definida positiva siempre, con lo que se solucionan los problemas 2, 3 y 5.

Es posible construir una aproximación al inverso de la matriz Hessiana en vez del Hessiano, con lo que se resuelve el problema (4). Supóngase que x_0 es el punto inicial y H_0 la correspondiente aproximación de la inversa de la matriz Hessiana, entonces, en la iteración k de un método Quasi-Newton se tiene que

$$p_k = -H_k \nabla f(x_k)$$

y

$$x_{k+1} = x_k + \alpha_k p_k$$

donde α_k se encuentra con el procedimiento de búsqueda en línea y

$$H_{k+1} = H_k + E_k$$

donde E_k es una matriz de rango dos escogida de tal forma que aproxima el inverso de la matriz Hessiana en x_{k+1}.

Las condiciones lógicas que se imponen para determinar H_k son:

1. $H_{k+1} y_k = s_k$, donde $s_k = x_{k+1} - x_k$; $y_k = g_{k+1} - g_k$, la que se conoce como *condición Quasi-Newton.*

2. H_{k+1} es definida positiva si H_k lo es y si $s_k^t y_k > 0$.

Davidon propuso el primer método Quasi-Newton, que más tarde fue modificado por Fletcher y Powell, quienes usaron lo que se conoce como la aproximación de Davidon-Fletcher-Powell (DFP) y cuya formulación es:

$$H_{k+1} = H_k + \frac{s_k s_k^t}{s_k^t y_k} - \frac{H_k y_k y_k^t H_k}{y_k^t H_k y_k} \tag{4.1}$$

Esta aproximación tiene la propiedad de que si la búsqeuda en línea es perfecta y la función es cuadrática, entonces $s_0, s_1, \ldots, s_{n-1}$ son conjugados y el mínimo se obtiene en n iteraciones a lo sumo. Además, se tiene que

$$H_n = Q^{-1}$$

El problema es construir aproximaciones H_{k+1} que satisfagan las condiciones (1) y (2), donde existen muchas soluciones. A principios de la década de los 70's, se encontró que la familia de soluciones de Broyden era tan eficiente como la DFP y en algunos casos se reportaron resultados mejores. La familia en cuestión está dada por:

$$H_{k+1} = H_k + \frac{s_k s_k^t}{s_k^t y_k} - \frac{H_k y_k y_k^t H_k}{y_k^t H_k y_k} + \zeta \tau_k v_k v_k^t \tag{4.2}$$

donde

$$\tau_k = \frac{y_k^t H_k y_k}{s_k^t y_k};\ 0 \leq \zeta \leq 1$$

y

$$v_k = s_k - \frac{1}{\tau_k} H_k y_k$$

y para el valor de $\zeta = 1$ tenemos la familia BFGS que tiene la siguiente forma:

$$H_{k+1} = H_k + \frac{H_k y_k s_k^t + s_k y_k^t H_k}{s_k^t y_k} + \left[1 + \frac{y_k^t H_k y_k}{s_k^t y_k}\right] \frac{s_k s_k^t}{s_k^t y_k} \tag{4.3}$$

Es justo decir que el entendimiento del método de Newton y los métodos Quasi-Newton es suficientemente bueno como para proporcionar software confiable para problemas generales de tamaño pequeño a mediano de minimización sin restricciones.

Dado que los métodos Quasi-Newton son muy similares al método de Newton con búsqueda en línea, excepto que la inversa de la matriz Hessiana $\nabla^2 f(x_k)^{-1}$ es aproximada por una matriz simétrica positiva definida H_k, la cual es corregida o actualizada después de cada iteración. Así, los métodos Quasi-Newton tienen la siguiente estructura:

ALGORITMO

Dado x_0 y $H_0 = I$.

Mientras $\|\nabla f(x_k)\| \geq \epsilon$.

1. Definir la dirección de búsqueda como:

$$p_k = -H_k \nabla f_k.$$

en vez de

$$p_k = -\nabla^2 f_k^{-1} \nabla f_k. \tag{4.4}$$

2. Hallar $\alpha = \min_{\alpha>0} f(x_k + \alpha p_k)$.
3. Hacer $x_{k+1} = x_k + \alpha_k p_k$.
4. Actualizar H_k como en (4.1) o (4.3).

El punto (4.4) remarca la diferencia fundamental entre el método de Newton y los método Quasi-Newton.

Actualmente, las investigaciones están enfocadas a problemas de gran escala. Las reglas de fondo que constituyen un algoritmo efectivo pueden cambiar drásticamente cuando el número de variables se vuelve grande. Ya que hemos supuesto que la solución del sistema lineal de orden n es comparable en costo a la evaluación del gradiente y Hessiano. Esta suposición puede no ser válida en problemas a gran escala, y entonces será necesario tomar ventaja de la estructura especial del problema. Con modificaciones razonables, el método de Newton puede aun ser una herramienta efectiva para problemas de gran escala. Pueden hacerse modificaciones a los algoritmos para determinar la dirección de búsqueda. Por ejemplo, ya que el método de Newton sólo requiere la descomposición de Cholesky de la matriz simétrica para problemas esparcidos es posible reducir la cantidad de trabajo y almacenamiento requerido para esta descomposición. Otra posibilidad es determinar una aproximación a la dirección de Newton.

El problema en el que la matriz Hessiana no puede ser almacenada en la memoria (rápida), es actualmente atacado por el método de direcciones conjugadas.

Capítulo 5
MÉTODO DE DIRECCIONES CONJUGADAS

El método de direcciones conjugadas fue propuesto inicialmente para resolver sistemas de ecuaciones lineales y posteriormente, fue extendido por Fletcher y Reeves para la solución de mínimos de funciones no lineales. Es muy economico en cuanto a requerimientos de memoria interna, pues las direcciones de descenso dependen sólo del gradiente en el punto actual y la dirección de descenso anterior.

Veremos primero la descripción del método para funciones cuadráticas y luego su extensión para el caso general de una función no lineal.

Sea la función cuadrática

$$\phi(x) = x^t Q x - b^t x \tag{5.1}$$

donde Q es una matriz definida positiva ($Q > 0$).

Definición 5.1 *Dada $Q > 0$ un conjunto de vectores $\{p_1, p_2, ..., p_k\}$ se dice que es conjugado con respecto a Q si*

$$p_i^t Q p_j = 0 \quad \textit{para} \quad i \neq j \tag{5.2}$$

Esta definición es una generalización del concepto de ortogonalidad de un conjunto de vectores si se pone $Q = I$.

Es fácil mostrar que los vectores Q-conjugados son linealmente independientes y que, por consiguiente, para $x \in R^n$ y $\{p_1, p_2, ..., p_k\}$ un conjunto de vectores Q-conjugados, no nulos, es posible expresar a x como una combinación lineal de estos

$$x = \sum_{k=0}^{n-1} \alpha_k p_k \tag{5.3}$$

donde

$$\alpha_k = \frac{p_k^t Q x}{p_k^t Q p_k} \tag{5.4}$$

Si x^* es el mínimo de $f(x)$ y $x_0 \in R^n$ es arbitrario, como $Qx^* = b$, entonces si se sustituye x en (5.3) y (5.4) por $x^* - x_0$ se obtiene

$$x^* - x_0 = \sum_{k=0}^{n-1} \alpha_k p_k \tag{5.5}$$

$$\alpha_k = \frac{p_k^t Q(x^* - x_0)}{p_k^t Q p_k} = \frac{-p_k^t g_0}{p_k^t Q p_k} \tag{5.6}$$

La idea del método de direcciones conjugadas consiste en que la expresión para el mínimo x^* puede calcularse de manera iterativa según el siguiente

ALGORITMO

Dados $x_0 \in R^n$ y $p_1, p_2, \ldots, p_k$ Q-conjugados,

Para $k = 0, 1, \ldots, n-1$ hacer:

1. $x_{k+1} = x_k + \alpha_k p_k$
2. $$\alpha_k = \frac{-p_k^t g_k}{p_k^t Q p_k}$$

Este método tiene las siguientes propiedades:

1. Si $x_1, x_2, \ldots, x_n$ son los vectores generados por el método de direcciones conjugadas, entonces
$$g_{k+1}^t p_i = 0 \quad para\ i = 0, 1, \ldots, k \tag{5.7}$$
2. x_{k+1} es el mínimo de f restringido a
$$M_k = \{z : z = x_0 + \gamma_0 p_0 + \cdots + \gamma_k p_k,\ \gamma_i \in R\} \tag{5.8}$$
3. Los gradientes de las iteraciones son mutuamente ortogonales, i.e,
$$g_{k+1}^t g_i = 0, \quad i = 0, \ldots, k \tag{5.9}$$

5.1 Método de Gradientes Conjugados

La forma usual de generar un conjunto de direcciones conjugadas es a partir de vectores linealmente independientes, a los que se aplica el método de ortogonalización de Gram-Schmidt.

El método de Gradiente Conjugado se obtiene generando direcciones conjugadas a partir de los gradientes en los puntos de cada iteración, de esta forma las direcciones p_k dependen sólo del gradiente actual, la dirección y el gradiente en la iteración anterior. El método tiene entonces la siguiente iteración :

ALGORITMO

Dados $x_0 \in R^n$ y $p_0 = -g_0$

Para $k = 0, 1, ..., n-1$ hacer

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

2. $$\alpha_k = \frac{-p_k^t g_k}{p_k^t Q p_k}$$

3. $$\beta_k = \frac{g_{k+1}^t (g_{k+1} - g_k)}{p_k^t (g_{k+1} - g_k)}$$

4. $p_{k+1} = -g_{k+1} + \beta_k p_k$

Este método tuvo su origen en la solución de sistemas de ecuaciones lineales con matrices definidas positivas y para este caso se tiene que ya que el mínimo en la línea es exacto, $\nabla f(x_k)^t p_k = 0$, y por la propiedad (5.7) se tiene que $\nabla f(x_{k+1})^t \nabla f(x_k) = 0$, lo que implica que β_k puede ser definido como:

$$\beta_k^{FR} = \frac{\|\nabla f(x_{k+1})\|^2}{\|\nabla f(x_k)\|^2} \tag{5.10}$$

El supraindice FR usado en la expresión anterior, hace referencia a Fletcher-Reeves, que fueron los que propusieron esta formulación para el caso lineal.

Fletcher y Reeves también fueron los que propusieron la extensión del método de gradientes conjugados a funciones no cuadráticas, observando que la matriz Hessiana que aparece en la aproximación de la función no lineal f mediante un modelo cuadrático, sólo hace falta para calcular el tamaño del paso α_k que sea mínimo en la dirección de descenso, que podría sustituirse por un proceso que lo calculara usando los valores de la función y el gradiente, también habría que tener en cuenta que como con esto se pierde la propiedad de convergencia en n iteraciones, sería necesario reiniciar la dirección de descenso después de n iteraciones, para garantizar que las direcciones que se generan sean linealmente independientes.

Polak y Ribiere consideran que cuando la función no es cuadrática, no tiene por qué satisfacer la condición (1.4) de ortogonalidad de los gradientes, y por ello proponen un nuevo β_k como:

$$\beta_k^{PR} = \frac{\nabla f(x_{k+1})^t (\nabla f(x_{k+1}) - \nabla f(x_k))}{\nabla f(x_k)^t \nabla f(x_k)} \tag{5.11}$$

La parte más costosa del método de gradientes conjugados es la búsqueda del óptimo en la dirección de descenso, para simplificar esto, la búsqueda se hace inexacta, esto es, el cálculo del mínimo a lo largo de la línea se sustituye por una aproximación que satisfaga las condiciones generales de descenso como son las condiciones impuestas por el conjunto $T(\mu)$. Sin embargo, ésto puede ocasionar que la dirección p_{k+1} dada por

$$p_{k+1} = -g_{k+1} + \beta_k p_k$$

no sea de descenso, ya que

$$p_{k+1}^t g_{k+1} = -\|g_{k+1}\|^2 + \beta_k p_k^t g_{k+1}$$

y si x_{k+1} es tal que

$$\left|\beta_k p_k^t g_{k+1}\right| > \|g_{k+1}\|^2$$

significaría que p_{k+1} no fuera una dirección de descenso. Esto puede resolverse usando un recomienzo, es decir, haciendo $p_{k+1} = -g_{k+1}$, aunque frecuentes recomienzos pueden afectar la eficiencia del método, por lo que a pesar de que la búsqueda en línea sea imperfecta debe ser, no obstante, una buena aproximación al mínimo real.

La búsqueda en línea requiere además, un buen valor inicial para α_k, lo que es crucial para obtener un algoritmo eficiente. Fletcher obtuvo experimentalmente que en el gradiente conjugado el decrecimiento de la función de x_{k-1} a x_k era del mismo orden de magnitud que el de x_k a x_{k+1}. Suponiendo que f es cuadrática, se tiene que el tamaño del paso en la iteración anterior α_{k-1} y la actual α_k están relacionados por la expresión

$$\alpha_k = \alpha_{k-1} \frac{p_{k-1}^t \nabla f(x_{k-1})}{p_k \nabla f(x_k)}$$

y este es el valor inicial a tomar para comenzar la búsqueda en línea, en el caso del método de gradientes conjugados.

El método de gradientes conjugados mejora mucho su efectividad si se recomienza cada n iteraciones, pero debido a que la búsqueda en línea no es perfecta, es necesario evitar que los gradientes consecutivos no se alejen mucho de la ortogonal.

Powell encontró que en la práctica cuando

$$\left|\nabla f(x_{k+1})^t \nabla f(x_k)\right| > \gamma \|\nabla f(x_k)\|^2$$

donde γ es un número positivo menor que uno, es conveniente recomenzar la búsqueda. El valor $\gamma = 0.2$ es ampliamente recomendado.

Otro criterio para hacer un recomienzo, también recomendado por Powell, es cuando la dirección p_{k+1} obtenida no es de descenso suficiente, esto es, si

$$\frac{\nabla f(x_{k+1})^t p_{k+1}}{\|\nabla f(x_{k+1})\|^2} \notin [-1.2, -0.8] \tag{5.12}$$

La implementación del algoritmo de gradientes conjugados que se tiene en el software tiene la siguiente estructura:

ALGORITMO DE GRADIENTES CONJUGADOS

Escoger $x_0 \in R^n$, $\epsilon > 0$.

$\alpha_0 = \frac{1}{\|\nabla f(x_0)\|}$; $p_0 = -\nabla f(x_0)$;

Mientras $\|\nabla f(x_k)\| \geq \epsilon$ has

1. Búsqueda en línea. Hallar $\alpha_k \approx \min_{\alpha>0} f(x_k + \alpha p_k)$.
2. Hacer $x_{k+1} = x_k + \alpha_k p_k$.
3. Calcular la dirección p_{k+1} como:

   Si $k \equiv 0 \bmod n$ o se cumple (5.12) hacer $p_{k+1=-\nabla f(x_{k+1})}$
   Si no $p_{k+1} = -\nabla f(x_{k+1}) + \beta_k p_k$ donde
   $\beta_k = \begin{cases} \beta_k^{FR} \text{ (5.10) si es Fletcher-Reeves} \\ \beta_k^{PR} \text{ (5.11) si es Polak-Ribiere} \end{cases}$

4. $\alpha_{k+1} = \alpha_k \frac{p_k^t \nabla f(x_k)}{p_{k+1}^t \nabla f(x_{k+1})}$

Capítulo 6
SISTEMA DE OPTIMIZACIÓN NO LINEAL

La hipótesis de que dada una función $f : R^n \rightarrow R$ continua y diferenciable, es utilizada en diferentes teoremas y algoritmos que han sido presentados y discutidos a lo largo de todo el trabajo, sin embargo, resulta ser particularmente difícil comprobar la validez de esta hipótesis cuando se tratan de implementar los teoremas o algoritmos en la práctica. A este respecto, la computadora presenta un reto de enorme dificultad debido a la codificación especial a que debe ser sometida una función f, para que ésta pueda ser reconocida por la computadora, sin llegar a tener aun idea sobre su continuidad o derivabilidad.

El propósito fundamental de este trabajo es la implementación de varios métodos de optimización no lineal sin restricciones, no obstante la aportación no estriba en la implementación de los métodos, sino en hacer que tales métodos sean interactivos, esto es, que el usuario pueda interactuar con ellos, siendo para ello indispensable la creación de un lenguaje especial para la edición de funciones en la computadora, de modo que sea fácil para el usuario, escribir la función y para la computadora el poder traducirla e interpretarla.

El sistema S.O.N, fue diseñado para resolver el problema de optimización no lineal, por lo que contiene los métodos vistos en los capítulos anteriores y una serie de módulos, que hacen al sistema interactivo. Así, en este capítulo, describiremos los módulos que hacen interactivo al sistema.

El sistema se escribió en lenguaje de programación c, ya que éste permite una gran flexibilidad en el manejo de localidades de memoria, lo que a su vez da la posibilidad implementar rutinas que son capaces de procesar y evaluar una gran gama de funciones, las cuales pueden ser leídas desde el mismo teclado de la computadora.

6.1 Descripción General del Sistema

Para tener una descripción completa del sistema de optimización no lineal S.O.N, presentaremos un diagrama donde podremos distinguir cada uno de los módulos, su ubicación y la forma en que interactúa con los demás módulos del sistema, así como

la forma en que es procesada la información y los posibles modos de interacción entre los módulos.

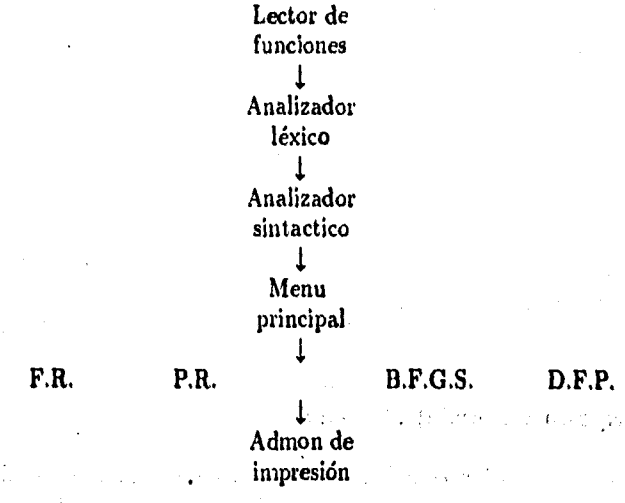
Cada módulo del sistema se encuentra implementado en una o varias rutinas o funciones del sistema, sin embargo la modularidad presentada en el esquema (6.1) es respetada.

6.2 Reconocimiento de Funciones

Antes de poder evaluar la función en un punto, la computadora debe traducir la función a su propio lenguaje, para esto la función debe pasar por diferentes fases de análisis que son el análisis léxico y el análisis sintáctico, en esta fase se revisa que la función no contenga errores de escritura, se identifican cada uno de sus componentes, la correcta concatenación de sus componentes, etcétera, para después proceder a traducir la función a un lenguaje que la computadora pueda entender.

A continuación se explica el análisis léxico, sintáctico y la traducción así como la generación de código intermedio, en la siguiente parte trataremos la fase de evaluación.

Tabla 6.1



6.2.1 Análisis Léxico

El analizador léxico conforma la primera parte de la fase de análisis. Su principal función consiste en leer los caracteres de la entrada, en este caso del teclado y elaborar como salida una secuencia de componentes léxicos (tokens) que utilizará el analizador sintáctico para trabajar. En esta fase se eliminan los espacios en blanco innecesarios y se procesan las posibles macro instrucciones que puedan existir.

Se entenderá por componentes léxicos a los diferentes símbolos que se encuentran en una expresión como pueden ser palabras, números, cadenas y caracteres especiales. En el sistema se define una tabla (como la que es presentada a continuación Tabla (6.2) donde se encuentran todos los símbolos que se consideran como válidos para ser reconocidos dentro de las expresiones.

Tabla 6.2

	Simbolo		Simbolo
1	(	14	asn
2	)	15	acs
3	+	16	atn
4	−	17	hsn
5	*	18	hcs
6	/	19	htn
7	∧	20	log
8	∼	21	exp
9	ln	22	cte
10	abs	23	var
11	sen	24	sum
12	cos	25	pro
13	tan		

Para reconocer un símbolo se debe recorrer la expresión carácter por carácter agrupándolos hasta encontrar uno que sirva de separador. Se empieza a reconocer el *tipo* de símbolo por su primer carácter: si es una letra, se tiene una palabra, y si es un dígito, se tiene un número, y si es cualquier otro carácter, se tiene un símbolo especial.

Todos los símbolos que son reconocidos en una expresión se comparan con los existentes en la Tabla 6.2, ya que sobre la tabla se realiza el reconocimiento de los componentes léxicos. Una vèz reconocidos, éstos se incorporan codificados en una tabla llamada tabla de símbolos.

Las siguientes instrucciones definen los pasos necesarios para reconocer los diferentes símbolos en una expresión:

1. Saltar los caracteres blancos hasta el primero no blanco, este carácter es el primer carácter del símbolo a extraer, e indica el *tipo* de símbolo.
2. Traer caracteres de la expresión hasta extraer un carácter que no pertenezca al símbolo.
3. Ahora se tiene el primer carácter del siguiente símbolo. Se procesa el símbolo que se ha extraído.
4. Repetir los pasos desde el (1) para extraer el siguiente símbolo.

Para tener éxito en el reconocimiento de símbolos es necesario eliminar los problemas que surgen al comparar los distintos elementos, ya que pueden aparecer escritos en distintos modos (por ejemplo: Sen, sen, sEn, etc.), esto es a veces en minúsculas, otras en mayúsculas, o en combinaciones de ámbas, por lo que en esta primera fase es conveniente convertir todos los caracteres a minúsculas o mayúsculas.

El analizador léxico se encuentra implementado en dos rutinas llamadas *leefuncion* y *codifica*, cuya implementación se presenta en el anexo 1.

6.2.2 Análisis Sintáctico

Toda expresión tiene una sintaxis, un conjunto de reglas gramaticales que especifican cuando una declaración está correctamente escrita. La sintaxis juega un papel central en la traducción de una función.

Una gramática describe de forma natural la estructura jerárquica de muchas de sus construcciones. El análisis sintáctico es el proceso que determina si una cadena de componentes léxicos, pertenece o no a una gramática, esto es si puede ser generada por ésta.

La parte de un compilador que conoce la sintaxis de un programa fuente es el parser. El analizador sintáctico (parser) controla el proceso de traducción, ya que analiza el programa fuente, de acuerdo a la sintaxis; llama al analizador léxico para obtener un token, y después llama a las rutinas de la tabla de símbolos para identificar y meter los tokens. Cuando el parser ha reconocido una entidad sintáctica, tal como una expresión aritmética, llama a la rutina de ejecución para que realice las operaciones adecuadas, aquí se van a interpretar expresiones.

Antes de reconocer una expresión, se debe ser capaz de describir la sintaxis de un programa fuente. En la figura 1 se muestra la descripción sintáctica de un número usando un diagrama de sintaxis. Este diagrama dice que un dígito es cualquier

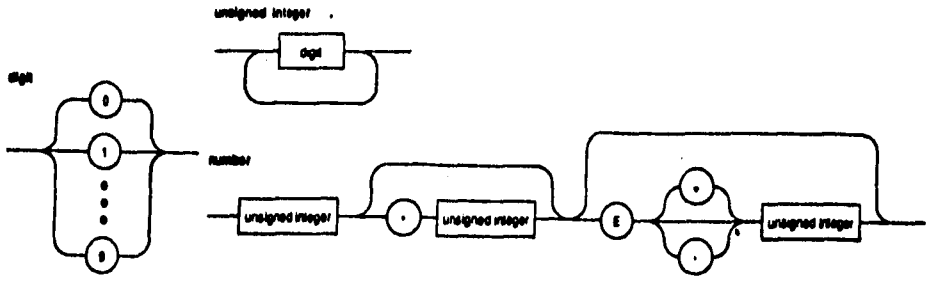


Figura 1

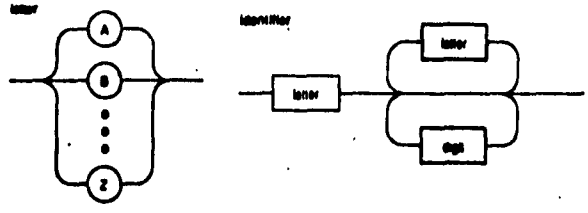


Figura 2

combinación de caracteres del 0 al 9. La figura 2 muestra el diagrama de sintaxis para un identificador. Así el diagrama no sólo describe la sintaxis, sino que también ayuda a describir el parser. El diagrama representa las entidades en el más bajo nivel sintáctico, como identificadores y símbolos numéricos.

La figura 3 muestra el diagrama de sintaxis que se utilizó para construir el parser. Este reconoce expresiones simples y términos. El primer diagrama dice que una expresión es una simple expresión. El segundo diagrama dice que una simple expresión es un simple término o varios términos separados por los operadores + o -. El tercer diagrama dice que un término es un simple factor o varios factores separados por los operadores * o /. El cuarto diagrama dice que un factor es un identificador, un número o una expresión entre paréntesis.

Todos los diagramas juntos muestran como las definiciones están anidadas; las expresiones, son expresiones simples, las expresiones simples se convierten en términos y los términos en factores. Estos diagramas, hacen también referencia a las reglas de precedencia de operadores tales como: * y / tienen mayor precedencia que + y -, y las expresiones entre paréntesis son evaluadas independientemente.

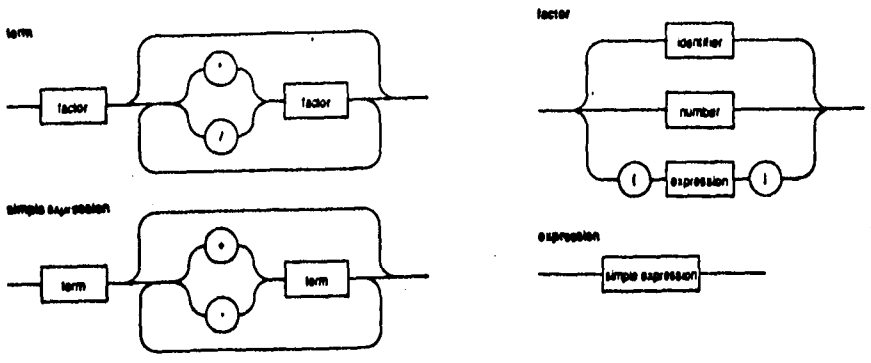


Figura 3

Ambigüedad

Se debe tener cuidado al considerar la estructura de una cadena segun una gramática, ya que si no se especifica bien se pueden tener ambigüedades como en la siguiente expresión 9-5+2, ya que esta puede agruparse de dos formas distintas (9-5)+2 y 9-(5+2). Esta segunda forma de agrupamiento da un valor de 2, en lugar del valor acostumbrado 6.

Asociatividad de Operadores

Por convención, 9+5+2 es equivalente a (9+5)+2, y 9-5-2 es equivalente a (9-5)-2. Cuando un operando como 5 tiene operadores a su izquierda y derecha, se necesitan convenciones para decidir qué operador debe considerar ese operando. Se dice que el operador + asocia a la izquierda, porque un operando que tenga un signo más a ambos lados es tomado por el operador que esté a su izquierda. En la mayoría de los lenguajes de programación, los cuatro operadores aritméticos, adición, sustracción, multiplicación y división son asociativos por la izquierda. Algunos operadores comunes como la exponenciación, son asociativos por la derecha.

Precedencia de Operadores

Considérese la expresión 9+5*2. Tiene dos posibles interpretaciones: (9+5)*2 o 9+(5*2). La asociatividad de + y * no resuelve esta ambigüedad. Por esta razón, se necesita conocer la precedencia en el orden de evaluación de los operadores.

Se dice que * tiene mayor precedencia que + si * considera sus operandos antes de que lo haga +. En aritmética elemental la división y la multiplicación tienen mayor precedencia que la adición y la sustracción. Por tanto, 5 es considerado por * en 9+5*2 y en 9*5+2; es decir, las expresiones son equivalentes a 9+(5*2) y (9*5)+2, respectivamente.

La siguiente tabla resume las reglas de precedencia y asociatividad de todos los operadores. Los operadores situados en la misma línea tienen la misma precedencia, los renglones están colocados por orden descendente en la precedencia.

Tabla de Precedencia.

Operador	*Asociatividad*
(,)	*izq – der*
*, \	*izq – der*
+, –	*izq – der*
^	*izq – der*
~	*der – izq*
s. d.	*izq – der*

6.2.3 Cambio de Notación

En notación algebraica normal, las expresiones son escritas en notación infija, esto es cuando los operandos están entre los operadores. Por ejemplo, a+b.

En notación postfija, los operadores se escriben antes que sus operandos, como en ab+. Esta notación también es llamada notación polaca inversa, y fue popularizada por las calculadoras científicas de bolsillo Hewlett-Packard. Esta notación es útil cuando realizamos cálculos sobre una pila (la pila es una estructura lineal definida sobre memoria). Si leemos la expresión postfija de izquierda a derecha, cada operando tiene un valor en la pila. Cada operador binario saca del tope de la pila dos operandos, ejecuta la operación en ellos, y pone el valor resultante otra vez en el tope de la pila. Un operador unario toma un valor del tope de la pila, ejecuta la operación en él, y lo pone de nuevo otra vez en el tope de la pila. La siguiente es una expresión infija complicada

$$((\sim 17 + 49)/4 - 2 * 3) * (9 - 3 + 2)$$

cuando la convertimos a postfijo tenemos:

$$17 \sim 49 + 4/23 * -93 - 2 + *$$

(utilizamos el operador $\sim$ menos unario, para distinguirlo de - que es el operador de sustracción binaria). Preservamos la forma de los operandos, pero reordenamos los operadores de acuerdo a las reglas de precedencia. Los paréntesis no son necesarios.

ALGORITMO

La notación postfija de una expresión E se puede definir de manera inductiva como sigue:

1. Si E es una variable o constante, entonces la notación postfija de E es también E.
2. Si E es una expresión de la forma E1 op E2, donde op es cualquier operador binario, entonces la notación postfija de E es Ep1Ep2 op, donde Ep1 y Ep2 son las notaciones postfijas de E1 y E2 respectivamente.
3. Si E es una expresión de la forma (E1), entonces la notación postfija de E1 es también la notación post fija de E.

6.3 Evaluación

El convertir de infijo a postfijo es mucho más que revisar la expresión, ya que traduce una expresión dentro de un código objeto para una calculadora. Nuestro pocedimiento de evaluación es un intérprete que ejecuta las operaciones especificadas en la pila por la notación postfija.

El procedimiento de calculadora interpreta un lenguaje que consiste en declaraciones de asignación, éstas declaraciones contenidas y codificadas en la pila contienen el código de las variables que corresponden a los registros de una calculadora. La pila también guarda la ruta de las variables así como la dirección de sus valores en la tabla de símbolos. Así, al seguir el código marcado en la pila evaluamos la expresión.

Bibliografía

Aho A, Sethi R, Ullman J. *Compiladores, principios, técnicas y herramientas*, Addison-Wesley Iberoamericana.

Barrera P, Castellanos J. L, *Métodos de optimización de gran escala para el problema de la generación de redes óptimas*, Facultad de ciencias, UNAM, México.

Dennis J. E. Jr, Schnabel R. B, *Numerical Methods for unconstrained optimization and nonlinear equations*, Prentice-Hall.

Fletcher R, *Practical methods of optimization*, John Wiley and Sons.

Gill P. E, Murray W, Wright M, *Practical optimization*, Academic Press.

Jacoby S, Kowalik J, Pizzo J, *Iterative methods for nonlinear optimization problems*, Prentice-Hall.