Optimización

Dr. Félix Calderon Solorio

6 de septiembre de 2022

Índice general

Búsqueda uní–direccional de mínimos

Dada una función $f : \mathbb{R} \to \mathbb{R}$, deseamos calcular el mínimo de esta función. Una manera de hacerlo es hacer una búsqueda en una dirección (normalmente en la que decrece la función) utilizando una heurística, que puede ir desde la búsqueda exhaustiva, búsqueda de razón dorada, etc.

1.1. Búsqueda exhaustiva

Una manera de realizar esta tarea es proponer una sucesión dada por la ecuación 1.1

$$x^{(k+1)} = x^{(k)} + h \tag{1.1}$$

donde:h es un incremento con el cual nos movemos, de manera lo suficientemente lenta, para no pasar sobre el mínimo. El valor de h es positivo cuando tenemos evidencia que el mínimo esta a la derecha y será negativa cuando el mínimo esta a la izquierda (ver [Chapra and Canale, 2000]).

El algoritmo de búsqueda, para un mínimo será el algoritmo 1:

Algoritmo 1 Algoritmo de Búsqueda exhaustiva

Entrada: $x^{(0)}, h, f(x)$
Salida: $x^{(k)}$
  mientras $f(x^{(k+1)}) < f(x^{(k)})$ **hacer**
    $x^{(k+1)} = x^{(k)} + h$
    $k \leftarrow k + 1$
  fin mientras
  devolver $x^{(k)}$

1.1.1. Ejemplo

Utilizando la búsqueda exhaustiva, encuentre el mínimo de la función $f(x) = -2seno(x) + x^2/10$ en el intervalo[0,4]. En la figura 1.1 se muestra esta ecuación en el intervalo mencionado.

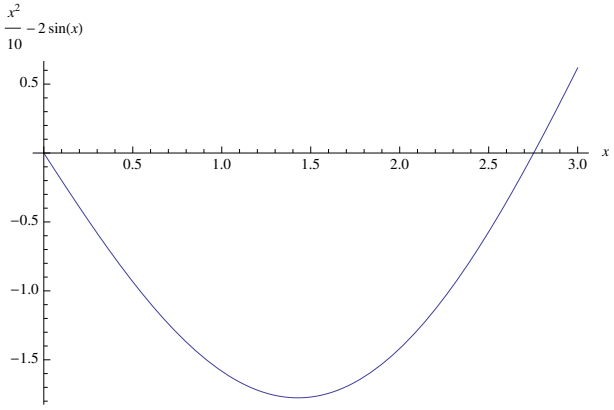


Figura 1.1: Función $-2sen(x) + x^2/10$

Para un incremento de búsqueda $h = 0.1$, de acuerdo con la tabla 1.1, el máximo es: $x = 1.4$. Note que la precisión de la solución esta limitada por el incremento y si se desea mayor precisión se necesitarán de un incremento más pequeño.

1.2. Búsqueda de la sección dorada

La búsqueda de la sección dorada es una técnica simple de búsqueda de una sola variable de propósito general. La clave para hacer eficiente este procedimiento es la mejor elección de los puntos intermedios. Esta meta se puede alcanzar al especificar que las siguientes dos condiciones se cumplan (ver [Chapra and Canale, 2000]).

$$l_0 = l_1 + l_2 \tag{1.2}$$

$$\frac{l_1}{l_0} = \frac{l_2}{l_1} \tag{1.3}$$

Sustituyendo la ecuación 1.2 en la ecuación 1.3 obtenemos:

$$\frac{l_1}{l_1 + l_2} = \frac{l_2}{l_1} \tag{1.4}$$

x	**f(x)**
0.0	0.
0.1	-0.1986
0.2	-0.3933
0.3	-0.5820
0.4	-0.7628
0.5	-0.9338
0.6	-1.0932
0.7	-1.2394
0.8	-1.3707
0.9	-1.4856
1.0	-1.5829
1.1	-1.6614
1.2	-1.7200
1.3	-1.7581
1.4	**-1.7749**
1.5	-1.7699
1.6	-1.7431
1.7	-1.6943
1.8	-1.6237
1.9	-1.5316

Cuadro 1.1: solución para la función $f(x) = -2seno(x) + x^2/10$

Definimos $R = \frac{l_2}{l_1}$ y sustituimos en la ecuación 1.4 para obtener $R^2 + R - 1$, cuya solución positiva es $R = \frac{\sqrt{5}-1}{2}$. R es definido como la razón dorada y fue un número ampliamente utilizado por los Griegos en su arquitectura.

El algoritmo de la razón dorada es:

Algoritmo 2 Algoritmo de Búsqueda con Razón Dorada

Entrada: $x_u, x_l, f(x)$
Salida: x
  mientras $x_u - x_l > Tol$ **hacer**
    $d = R(x_u - x_l)$
    $x_1 = x_l + d$
    $x_2 = x_u - d$
    $k = k + 1$
    si $f(x_1) < f(x_2)$ **entonces**
      $x_l \leftarrow x_2$
    si no
      $x_u \leftarrow x_1$
    fin si
  fin mientras
  si $f(x_1) < f(x_2)$ **entonces**
    devolver x_1
  si no
    devolver x_2
  fin si

1.2.1. Ejemplo

Use la búsqueda de la sección dorada para encontrar el mínimo de la función $f(x) = -2seno(x) + \frac{x^2}{10}$ en el intervalo [0,4].

Los resultados de este ejemplo se muestran en la tabla 1.2 y se puede observar que se logra mejor precisión que con razón dorada en menos iteraciones.

Según los resultados de la tabla 1.2, la solución es $x = 1.4267$ con $f(x) = -1.7757$.

k	x_l	f_l	x_2	f_2	x_1	f_1	x_u	f_u	d
0	0.0000	0.0000	1.5279	-1.7647	2.4721	-0.6300	4.0000	-3.1136	2.4721
1	0.0000	0.0000	0.9443	-1.5310	1.5279	-1.7647	2.4721	-0.6300	1.5279
2	0.9443	-1.5310	1.5279	-1.7647	1.8885	-1.5432	2.4721	-0.6300	0.9443
3	0.9443	-1.5310	1.3050	-1.7595	1.5279	-1.7647	1.8885	-1.5432	0.5836
4	1.3050	-1.7595	1.5279	-1.7647	1.6656	-1.7136	1.8885	-1.5432	0.3607
...	...	...	...	...	...	...	...	...	...
17	1.4267	-1.7757	1.4271	-1.7757	1.4274	-1.7757	1.4278	-1.7757	0.0007

Cuadro 1.2: solución para la función $f(x) = -2seno(x) + x^2/10$ utilizando razón dorada

Tarea: Comparación de Búsqueda exhaustiva y Razón Dorada

Dada la función $f(x) = x^2 - seno(x)$, determinar:

- Su gráfica en el intervalo $[0, 1]$ y verificar si el punto de inflexion es un mínimo o un máximo,
- Buscar el punto de inflexión utilizando búsqueda exhaustiva y
- Razón dorada calcular su mínimo
- Comparar los resultados

1.3. Método de Newton

Consideremos que la función $f : \mathbb{R} \to \mathbb{R}$ es de clase dos, es decir que la segunda derivada puede ser calculada. La idea consiste en reemplazar en la vecindad del punto $x^{(k)}$ de la función f por una aproximación cuadrática $q(x)$ dada por

$$q(x) = f(x^{(k)}) + f'(x^{(k)})(x - x^{(k)}) + \frac{1}{2}f''(x^{(k)})(x - x^{(k)})^2$$

llamaremos a $x^{(k+1)}$ el mínimo de $q(x)$. Para calcularlo, es necesario que la matriz $f''(x^{(k)})$ sea positivo. La función $q(x)$ es entonces estrictamente convexa y tiene un mínimo único en $x^{(k+1)}$ dado por

$$q'(x^{(k+1)}) = 0$$

Esto da lugar sistema lineal de ecuaciones:

$$f'(x^{(k)}) = -f''(x^{(k)})(x^{(k+1)} - x^{(k)})$$

$x^{(k)}$	$f(x^{(k)})$	$f'(x^{(k)})$	$f''(x^{(k)})$	$x^{(k+1)}$
0.5000	-0.9338	-1.6551	1.1588	1.9282
1.9282	-1.5017	1.0854	2.0735	1.4047
1.4047	-1.7751	-0.0495	2.1725	1.4275
1.4275	-1.7757	8.20126E-05	2.1795	1.4275
1.4275	-1.7757	2.02094E-10	2.1795	1.4275

Cuadro 1.3: solución para la función $f(x) = -2seno(x) + x^2/10$ utilizando el método de Newton

La solución para $x^{(k+1)}$ es el mínimo, lo cual da lugar a la recurrencia

$$x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{f''(x^{(k)})} \tag{1.5}$$

Algoritmo 3 Algoritmo de Newton Unidimensional

Entrada: $f(x), x^{(0)}$
Salida: $x^{(k)}$
  mientras $\|f'(x^{(k)})\| \geq \varepsilon$ **hacer**
    $x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{f''(x^{(k)})}$
    $k \leftarrow k + 1$
  fin mientras
  devolver $x^{(k)}$

1.3.1. Ejemplo

Calcular el máximo de la función $f(x) = -2seno(x) + \frac{x^2}{10}$ utilizando el método de Newton con un valor inicial $x^{(0)} = 0.5$.

Para resolver tenemos que calcular $f'(x)$ y $f''(x)$

$$f'(x) = \frac{x}{5} - 2cos(x)$$

$$f''(x) = \frac{1}{5} + 2sin(x)$$

Búsqueda basadas en gradiente

2.1. Derivadas direccionales y el gradiente

Sea una función $f : \mathbb{R}^2 \to \mathbb{R}$ de dos variables x y y y sea $P(x, y)$ un punto en el plano x,y. Definimos u como el vector unitario que forma un ángulo de θ radianes con el eje x. Entonces:

$$u = cos\theta\hat{i} + \sin\theta\hat{j} \tag{2.6}$$

La derivada direccional de f los podemos definir como:

$$\mathrm{D}_u f(x_1, x_2) = \lim_{h\to 0} \frac{f(x_1 + h\cos\theta, x_2 + h\sin\theta) - f(x_1, x_2)}{h}$$

La derivada direccional de la razón de cambio de los valores de la función $f(x_1, x_2)$ con respecto a la distancia en el plano x_1, x_2, medida en la dirección del vector unitario u.

Si queremos calcular la derivada en la dirección de x_1 hacemos $u = \hat{i}$ con $\theta = 0$

$$\mathrm{D}_{\hat{i}} f(x_1, x_2) = \lim_{h\to o} \frac{f(x_1 + h, x_2) - f(x_1, x_2)}{h}$$

Ahora si queremos valuar la deriva en la dirección x_2 hacemos $u = \hat{j}$

$$\mathrm{D}_{\hat{j}} f(x_1, x_2) = \lim_{h\to 0} \frac{f(x_1, x_2 + h) - f(x_1, x_2)}{h}$$

Ahora obtendremos una fórmula que nos permita calcular una derivada direccional en una forma más corta.

$$g(t) = f(x_1 + t\cos\theta, x_2 + t\sin\theta)$$

y para un vector unitario $u = \cos\theta\hat{i} + \sin\theta\hat{j}$. Entonces por definición de una derivada ordinaria tenemos:

$$\begin{aligned} g'(0) &= \lim_{h\to 0} \frac{f(x_1+(0+h)\cos\theta, x_2+(0+h)\sin\theta)}{h} - \\ &\quad \frac{f(x_1+0\cos\theta, x_2+0\sin\theta)}{h} \\ &= \lim_{h\to 0} \frac{f(x_1+h\cos\theta, x_2+h\sin\theta) - f(x_1,x_2)}{h} \\ g'(0) &= \mathrm{D}_u f(x_1, x_2) \end{aligned}$$

Calculamos $g'(t)$ aplicando la regla de la cadena:

$$\begin{aligned} g'(t) &= f_{x_1}(x_1 + t\cos\theta, x_2 + t\sin\theta)\frac{\partial(x_1+t\cos\theta)}{\partial t} \\ &\quad + f_{x_2}(x_1 + t\cos\theta, x_2 + t\sin\theta)\frac{\partial(x_2+t\sin\theta)}{\partial t} \end{aligned}$$

Por lo tanto:

$$g'(0) = f_{x_1}(x_1, x_2)cos\theta + f_{x_2}(x_1, x_2)\sin\theta$$

En general

$$\mathrm{D}_u f(x, y) = f_{x_1}(x_1, x_2)\cos\theta + f_{x_2}(x_1, x_2)\sin\theta$$

Para detalles ver [Leithold, 1981]

2.2. Gradiente

Si f es una función de dos variables x_1 y x_2, y f_{x_1} y f_{x_2} existen, entonces el gradiente de f denotado por ∇f está definido por:

$$\nabla f(x_1, x_2) = f_{x_1}(x_1, x_2)\hat{i} + f_{x_2}(x_1, x_2)\hat{j}$$

Dado que la derivada direccional puede escribirse como:

$$\mathrm{D}_u f(x_1, x_2) = \left(\cos\theta\hat{i} + \sin\theta\hat{j}\right)\left[f_{x_1}(x_1, x_2)\hat{i} + f_{x_2}(x_1, x_2)\hat{j}\right]$$

entonces:

$$\mathrm{D}_u f(x_1, x_2) = u \cdot \nabla f(x_1, x_2)$$

2.3. Reconociendo un mínimo local

Una herramienta utilizada para estudiar mínimos es la serie de Taylor, la cual puede ser descrita por:

$$\begin{aligned} f(x) &= f(a) + f'(a)(x-a) + \tfrac{1}{2!}f''(a)(x-a)^2 \\ &\quad \tfrac{1}{3!}f'''(a)(x-a)^3 + \tfrac{1}{4!}f^{\mathrm{IV}}(a)(x-a)^4 + \ldots \end{aligned}$$

La serie de Taylor es una aproximación polinomial dada por

$$f(x+p) = c_0 + c_1 p + c_2 p^2 + c_3 p^3 + c_4 p^4 + c_5 p^5 + \cdots$$

donde x es un punto y p es un incremento de x en alguna dirección. Si calculamos las derivadas de $f(x+p)$ tenemos

$$\begin{aligned} f^{i}(x+p) &= c_1 + 2c_2 p + 3c_3 p^2 + 4c_4 p^3 + 5c_5 p^4 + \cdots \\ f^{ii}(x+p) &= 2c_2 + 2\times 3c_3 p + 3\times 4c_4 p^2 + 4\times 5c_5 p^3 + \cdots \\ f^{iii}(x+p) &= 2\times 3c_3 + 2\times 3\times 4c_4 p + 3\times 4\times 5c_5 p^2 + \cdots \\ f^{iv}(x+p) &= 2\times 3\times 4c_4 + 2\times 3\times 4\times 5c_5 p + \cdots \\ f^{v}(x+p) &= 2\times 3\times 4\times 5\times c_5 + \cdots \end{aligned}$$

Si evaluamos las ecuaciones anteriores en $p = 0$

$$\begin{aligned} f(x+0) &= c_0 \\ f^{i}(x+0) &= c_1 \\ f^{ii}(x+0) &= 2c_2 \\ f^{iii}(x+0) &= 2\times 3c_3 \\ f^{iv}(x+0) &= 2\times 3\times 4c_4 \\ f^{v}(x+0) &= 2\times 3\times 4\times 5\times c_5 \end{aligned}$$

Con esto temos el conjunto de coeficientes c_i para la serie de Taylor

$$\begin{aligned} c_0 &= f(x) \\ c_1 &= f^i(x) \\ c_2 &= \frac{f^{ii}(x)}{2!} \\ c_3 &= \frac{f^{iii}(x)}{3!} \\ c_4 &= \frac{f^{iv}(x)}{4!} \\ c_5 &= \frac{f^{v}(x)}{5!} \end{aligned}$$

y una formula cerrada para la serie de Taylor dada como:

$$f(x+p) = f(x) + f^i(x)p + \frac{f^{ii}(x)}{2!}p^2 + \frac{f^{iii}(x)}{3!}p^3 + \cdots \tag{2.7}$$

En el caso de una función de varias variables:

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

Existen condiciones necesarias para determinar un mínimo, estas están relacionadas con el gradiente ∇f y el Hessiano $\nabla^2 f$.

2.3.1. Condición necesaria de primer orden

Si x^* es un mínimo local y f es diferenciable en una vecindad de x^* entonces $\nabla f(x^*) = 0$.
Prueba
Supongamos por contradicción que $\nabla f(x^*) \neq 0$. Definimos el vector $p = -\nabla f(x^*)$ y notamos que:

$$p^T \nabla f(x^*) = - \parallel \nabla f(x^*) \parallel^2 < 0$$

Dado que ∇f es continuo cerca de x^*, existe un escalar $T > 0$ tal que:

$$p^T \nabla f(x^* + tp) < 0 \quad \forall t \in [0, T]$$

Para algún $\bar{t} \in [0, T]$, tenemos que la serie de Taylor nos da:

$$\begin{aligned} f(x^* + \bar{t}p) &= f(x^*) + \bar{t}p^T \nabla f(x^*) \\ f(x^*) - \bar{t}p^T \nabla f(x^*) &> f(x^*) \end{aligned}$$

Por lo tanto $f(x^* + \bar{t}p) < f(x^*)$ para toda $\bar{t} \in (0, T]$. Tenemos una contradicción, hemos encontrado una dirección que se aleja de x^* y f decrece, por lo cual, x^* no es un mínimo local. Llamaremos x^* a un punto estacionario si $\nabla f(x^*) = 0$.

2.3.2. Condición necesaria de segundo orden

Si x^* es un mínimo local de f y $\nabla^2 f$ es continua en la vecindad de x^*, entonces $\nabla f(x^*) = 0$ y $\nabla^2 f^2(x^*)$ es definida semi positiva.

Por contradicción, asumimos que $\nabla^2 f(x^*)$ no es definido positivo. Entonces seleccionamos un vector p tal que $p^T \nabla^2 f(x^*) p < 0$ para todo $t \in [0, T]$

$$f(x^* + \bar{t}p) = f(x^*) + \bar{t}p^T \nabla f(x^*) + \frac{1}{2}\bar{t}^2 p^T \nabla^2 f(x^*) p \tag{2.8}$$

$$f(x^*) + \frac{1}{2}\bar{t}^2 p^T \nabla^2 f(x^*) p > f(x^*)$$

por lo tanto

$$f(x^* + \bar{t}p) < f(x^*) \tag{2.9}$$

por lo cual $f(x^*)$ no es un mínimo.

2.3.3. Ejemplo

Dada la función $f(x_1, x_2) = x_1^2 + x_2^2 + 5x_1x_2 + 2x_1 - 3x_2 - 20$ determinar:

1. La expresión del gradiente $\nabla f(x_1, x_2)$,
2. la expresión del Hessiano $\nabla^2 f(x_1, x_2)$ y
3. el punto de inflexion.

El vector gradiente queda como:

$$\nabla f(x_1, x_2) = \begin{bmatrix} 2x_1 + 5x_2 + 2 \\ 2x_2 + 5x_1 - 3 \end{bmatrix}$$

La matriz Hessiana es

$$\nabla^2 f(x_1, x_2) = \begin{pmatrix} 2 & 5 \\ 5 & 2 \end{pmatrix}$$

El punto extremo lo calculamos resolviendo el sistema de ecuaciones

$$\begin{pmatrix} 2 & 5 \\ 5 & 2 \end{pmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -2 \\ 3 \end{bmatrix}$$

La solución es $x_1 = \frac{19}{21}$ y $x_2 = \frac{-16}{21}$. El determinante del Hessiano $det(\nabla^2 f(x_1, x_2)) = -21$ por lo cual se trata de un máximo (ver [Nocedal and Wright, 1999]).

2.4. Dirección de búsqueda para métodos de búsqueda en línea

La dirección de descenso $-\nabla f_k$ es la opción más obvia como dirección de búsqueda. Es intuitivo que nos movamos en la dirección en la que más rápido decrece f. Para verificar esto, utilizaremos el teorema de Taylor:

$$f(x^{(k)} + \alpha p) = f(x^{(k)}) + \alpha p^T \nabla f(x^{(k)}) + \frac{1}{2}\alpha^2 p^T \nabla^2 f(x^{(k)}) p$$

El rango de cambio de f, en la dirección p en el punto x_k, es simplemente el coeficiente α. Por lo tanto la dirección unitaria p de más rápido descenso es la solución del problema:

$$\min_p p^T \nabla f(x^{(k)}) \qquad para \qquad \|p\| = 1 \tag{2.10}$$

Recordemos $p^T \nabla f(x^{(k)}) = \|p\| \|\nabla f_k\| \cos\theta$, donde θ es el ángulo entre p y $\nabla f(x^{(k)})$ tenemos que $\|p\| = 1$ por lo cual:

$$p^T \nabla f(x^{(k)}) = \|\nabla f(x^{(k)})\| \cos\theta$$

por lo tanto, la función objetivo es minimizada cuando $\cos\theta$ toma el valor de -1, lo cual sucede, cuando $\theta = \pi$ *radianes*. Sustituyendo en la ecuación tenemos:

$$\begin{aligned} pp^T \nabla f(x^{(k)}) &= p\|\nabla f(x^{(k)})\| \cos\pi \\ \nabla f(x^{(k)}) &= -p\|\nabla f(x^{(k)})\| \\ p &= -\frac{\nabla f(x^{(k)})}{\|\nabla f(x^{(k)})\|} \end{aligned}$$

El vector unitario que minimiza la ecuación 2.10 es:

$$p = -\frac{\nabla f(x^{(k)})}{\|\nabla f(x^{(k)})\|}$$

El método de descenso de gradiente es un método de búsquedalineal, que se mueve en dirección $p = -\nabla f(x^{(k)})$ en cada paso, tal que:

$$\min f(x^{(k)} + \alpha p^{(k)})$$

2.4.1. Curvas de Nivel y el vector Gradiente

El gradiente de una función f en un punto en particular es perperdicular a la curva de nivel de f en el mismo punto.

Prueba:

Consideremos que X_0 es el punto de interés. La curva de nivel que pasa a través de X_0 es una $X|f(X) = f(X_0)$. Consideremos la curva paramétrica $\gamma(t)$ es la curva de nivel que pasa a traves de X, por lo tanto podemos asumir que $\gamma(0) = X_0 = [x_1(t), x_2(t), \cdots, x_n(t)]^T$. Entonces tenemos

$$f(\gamma(0)) = f(X_0) = C$$

Ahora vamos a calcular la derivada de la función con respecto al parámetro t utilizando la regla de la cadena

$$\begin{aligned}
\frac{df(x_1(t), x_2(t), \cdots, x_n(t))}{dt} &= \frac{\partial f(X)}{\partial x_1}\frac{dx_1}{dt} + \frac{\partial f(X)}{\partial x_2}\frac{dx_2}{dt} + \cdots + \frac{\partial f(X)}{\partial x_n}\frac{dx_n}{dt} = 0 \\
&= \left[\frac{\partial f(X)}{\partial x_1}, \frac{\partial f(X)}{\partial x_2}, \cdots \frac{\partial f(X)}{\partial x_n}\right] \begin{bmatrix} \frac{dx_1}{dt} \\ \frac{dx_2}{dt} \\ \vdots \\ \frac{dx_n}{dt} \end{bmatrix} = 0 \\
&= [\nabla f(X_0)]^T \gamma'(X_0) = 0
\end{aligned}$$

De la formula anterior podemos ver que $f'(X(t))$ es el producto escalar del gradiente $\nabla f(X_0)]$ y la derivada de la curva de nivel $\gamma'(X_0)$. El producto escalar de vectores, que también puede ser escrito como $||\nabla f(x_0)||||\gamma'(X_0)||cos(\theta)$ y la única posibilidad de que este sea cero es que el ángulo entre ambos vectores sea 90°. Por lo tanto la el ángulo entre la tangente a la curva de nivel $\gamma'(X_0)$ forma un ángulo de 90° con la dirección del vector gradiente. En la figura 2.2 podemos ver las curvas de nivel en un punto y la dirección del menos gradiente.

Ver [Nocedal and Wright, 1999]

Tarea: Producto cruz y Serie de Taylor

1.- Determinar el ángulo entre los vectores $a = \widehat{i} + 3\widehat{j} - 2\widehat{k}$ y $b = 3\widehat{k}$
2.- Realizar la aproximación en serie de Taylor de tercer grado, de la función $f(x) = e^x + cos(x)$ en el punto $a = 0$

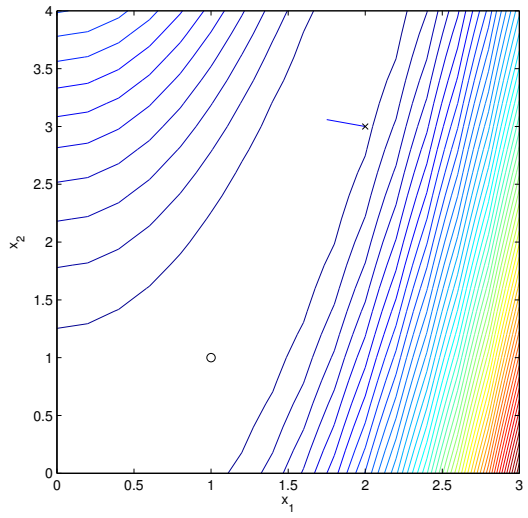


Figura 2.2: Curvas de nivel para la función de Rosenbrock

2.5. Métodos de búsqueda en una dirección

Los métodos de búsqueda en una sola dirección calculan una dirección $p^{(k)}$ y entonces deciden que tan lejos se deben mover en esta dirección. La iteración esta dada a por:

$$x^{(k+1)} = x^{(k)} + \alpha_k p$$

donde el escalar positivo α es llamada el tamaño de paso

De acuerdo a lo anterior la mejor distancia de descenso es:

$$p = -\frac{\nabla f(x^{(k)})}{\|\nabla f(x^{(k)})\|}$$

El algoritmo de búsqueda es 4:

2.5.1. Ejemplo

Este ejemplo muestra como se da la búsqueda en una dirección. Así dada la función en dos dimensiones $f(x_1, x_2) = x_1 e^{-x_1^2 - x_2^2}$ cuya imagen se muestra en la figura 2.3 y considerando como punto inicial $x^{(0)} = [-1, -1]^T$:

Algoritmo 4 Algoritmo de Búsqueda en la dirección del gradiente

Entrada: $x^{ini}, f(x), T$
Salida: $x^{(k)}$
  Hacemos $k = 0$, $\alpha = 0$ y $x^{(0)} = x^{ini}$
  Calculamos el valor del gradiente $\nabla f(x^{(k)})$
  Calculamos $p = -\frac{\nabla f(x^{(k)})}{\|\nabla f(^{(k)})\|}$
  repetir
    $\alpha \leftarrow \alpha + h$
    $x^{(k+1)} = x^{(k)} + \alpha * p$
    $k \leftarrow k + 1$
  hasta que $f(x^{(k+1)}) < f(x^{(k)})$
  devolver $x^{(k)}$

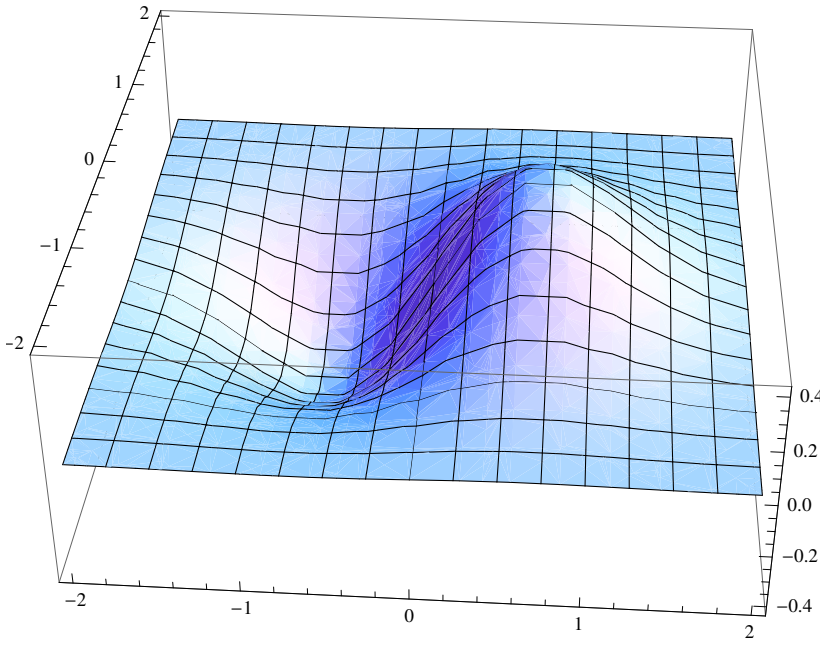


Figura 2.3: Función $f(x_1, x_2) = e^{-x_1^2 - x_2^2} x_1$

El vector gradiente esta dado por

$$g(x_1, x_2) = \begin{bmatrix} e^{-x_1^2-x_2^2} - 2e^{-x_1^2-x_2^2}x_1^2 \\ -2e^{-x_1^2-x_2^2}x_1x_2 \end{bmatrix}$$

La dirección del gradiente es $d = \left[-\frac{1}{\sqrt{5}}, -\frac{2}{\sqrt{5}}\right]$ y la dirección contraria al gradiente es $p = \left[\frac{1}{\sqrt{5}}, \frac{2}{\sqrt{5}}\right]$

La función a minimizar en la dirección del gradiente es

$$\phi(\alpha) = \left(-1 + \frac{\alpha}{\sqrt{5}}\right) e^{-\left[\left(-1+\frac{\alpha}{\sqrt{5}}\right)^2+\left(-1+\frac{2\alpha}{\sqrt{5}}\right)^2\right]}$$

cuya imagen se muestra en la figura 2.4, en esta podemos verificar que el mínimo $\phi(0.952194) = -0.403933$.

Los valores x y y se calculan como:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix} - 0.952194 \times \begin{bmatrix} -\frac{1}{\sqrt{5}} \\ -\frac{2}{\sqrt{5}} \end{bmatrix} = \begin{bmatrix} -0.574166 \\ -0.148331 \end{bmatrix}$$

Obviamente la función $f(-0.574166, -0.148331) = -0.403933$ y no es el mínimo global de la función. La función tiene un mínimo global en $x^* = [-\frac{1}{\sqrt{2}}, 0]^T$ con $f(-\frac{1}{\sqrt{2}}, 0) = -0.428882$ y el valor calculado esta lejos de la solución, sin embargo, nos hemos movido hacia abajo.

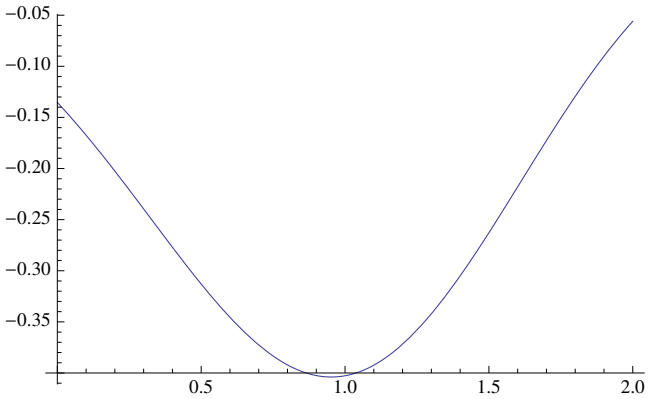


Figura 2.4: Función $\phi(x^{(0)})$

2.5.2. Ejemplo

Consideremos la función $f : R^2 \rightarrow R$ dada por la ecuación 2.11, queremos calcular el mínimo de la función utilizando la dirección de gradiente. Dado punto de partida $x_0 = [2, 3]^T$ y un tamaño de paso $\alpha = 0.05$ calculamos el mínimo de la función.

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \tag{2.11}$$

El gradiente de la función para cualquier valor de x es:

$$\nabla f(x) = \begin{bmatrix} -400(x_2 - x_1^2)x_1 + 2x_1 - 2 \\ 200x_2 - 200x_1^2 \end{bmatrix}$$

Si valuamos el gradiente en x_0

$$\nabla f(x) = \begin{bmatrix} 802 \\ -200 \end{bmatrix}$$

La dirección de búsqueda es

$$p = -\begin{bmatrix} 802/\sqrt{802^2 + 200^2} \\ -200/\sqrt{802^2 + 200^2} \end{bmatrix} = \begin{bmatrix} -0.97028 \\ +0.24197 \end{bmatrix}$$

En la siguiente tabla tenemos los resultados de la búsqueda lineal. Note que el mínimo de la función se localiza en el punto $x* = [1, 1]$ y el algoritmo no esta llegando a tal valor. øA que se debe esta situación?. En la figura 2.2 se presenta la solución calculada por este método. Con una x se señala el valor inicial y con una o el mínimo global

k	$\alpha * k$	x_1	x_2	f
0	0	2.00000	3.00000	101.000000
1	0.05	1.951486	3.0120985	64.2986276
2	0.10	1.902972	3.024197	36.46884822
3	0.15	1.854458	3.0362955	16.94835566
4	0.20	1.805944	3.048394	5.188138435
5	0.25	1.75743	3.0604925	0.652479811
6	0.30	1.708916	3.072591	2.81895777

Tarea: Modificar el algoritmo Búsqueda en la dirección de gradiente

Dado el algoritmo 4, hacer la modificaciones necesaria para que en lugar de realizar búsqueda exhaustiva, el algoritmo utilice la razón dorada. Utilice los datos del ejemplo anterior

2.6. Escalamiento

El desempeño de un algoritmo dependerá principalmente, de cómo el problema es formulado. En optimización sin restricciones, un problema es llamado pobremente escalado si producen más cambios en una dirección que en otra. Un ejemplo simple es el propuesto por la función:

$$f(x) = ax_1^2 + x_2^2$$

Al calcular el gradiente de la función tenemos

$$\nabla f(x) = \begin{bmatrix} 2ax_1 \\ 2x_2 \end{bmatrix}$$

Aplicando el método de búsqueda en una dirección, podremos notar que, es mas sensible a cambios en x_1 que a cambios en x_2.

Ejemplo

Consideremos la función $f(x) = x_1^2 + x_2^2$, la cual podemos ver gráficamente en la figura 2.5.

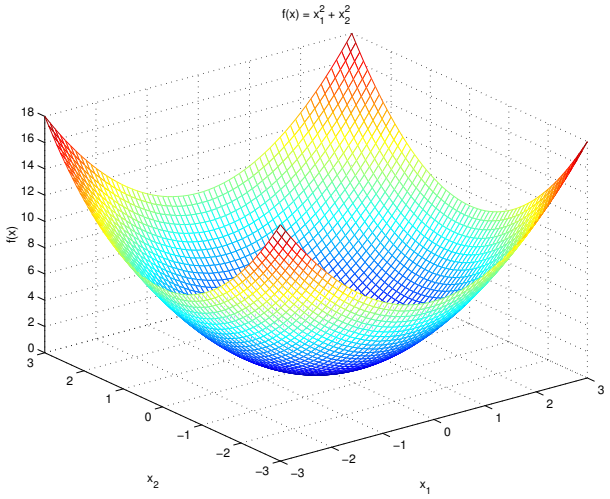


Figura 2.5: Función $f(x) = x_1^2 + x_2^2$

Con un punto inicial $x_0 = [1, 1]$ y calculando la dirección de búsqueda utilizando el gradiente, podemos ver la línea de búsqueda en la figura 2.6. Note que siguiendo la línea pasamos sobre el mínimo de la función.

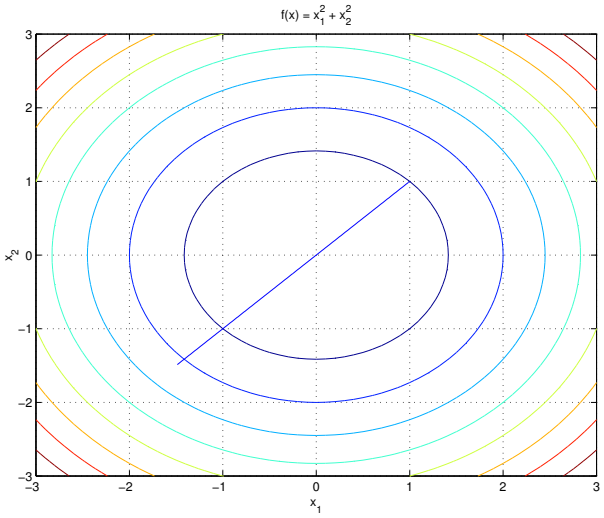


Figura 2.6: Curvas de nivel de la función $f(x) = x_1^2 + x_2^2$

Ahora si modificamos la función $f(x) = 2x_1^2 + x_2$, tendremos un escalamiento en la dirección x_1. Las curvas de nivel de esta se muestran en la figura 2.7 y podemos notar que la linea de búsqueda no pasará por el mínimo.

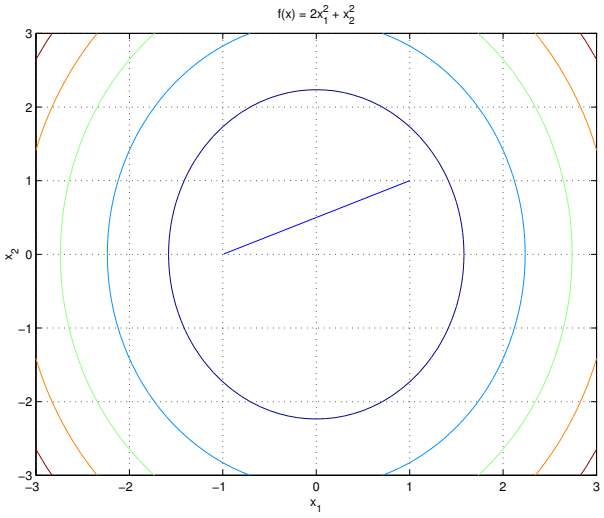


Figura 2.7: Curvas de nivel de la función $f(x) = 2x_1^2 + x_2^2$

Ver [Nocedal and Wright, 1999]

2.7. Algoritmos para seleccionar la longitud del paso

Ahora consideremos las técnicas para encontrar, de la mejor manera, el mínimo de la función $\phi(\alpha)$

$$\phi(\alpha) = f(x^{(k)} + \alpha p^{(k)})$$

2.7.1. Máximo descenso

Una manera precisa de calcular α, es utilizar las condiciones de Wolfe, sin embargo una aproximación cuadrática es una alternativa aproximada y rápida.

Si f es una función convexa cuadrática $f(x) = \frac{1}{2}x^T Ax + B^T x + c$ podemos calcular de manera exacta el valor de la longitud de paso utilizando la serie de Taylor:

$$\begin{aligned} f(x^{(0)} + \alpha^{(k)} p) &= f(x^{(0)} + \alpha^{(0)} p) \\ &+ (\alpha^{(k)} - \alpha^{(0)}) p^T \nabla f(x^{(0)} + \alpha^{(0)} p) \\ &+ \frac{1}{2}(\alpha^{(k)} - \alpha^{(0)})^2 p^T \nabla^2 f(x^{(0)} + \alpha^{(0)} p) p \end{aligned}$$

Dado que nuestra función depende solo de α, derivamos respecto a ella e igualamos a cero:

$$p^T \nabla f(x^{(0)} + \alpha^{(0)} p) + (\alpha^{(k)} - \alpha^{(0)}) p^T \nabla^2 f(x^{(0)} + \alpha^{(0)} p) p = 0$$

despejando, obtenemos:

$$\alpha^{(k)} = \alpha^{(0)} - \frac{p^T \nabla f(x^{(0)} + \alpha^{(0)} p)}{p^T \nabla^2 f(x^{(0)} + \alpha^{(0)} p) p}$$

Asumiendo que $\alpha^{(0)} = 0$ tenemos

$$\alpha^{(k)} = -\frac{p^T \nabla f(x^{(0)})}{p^T \nabla^2 f(x^{(0)}) p} \tag{2.12}$$

Utilizando esto, tenemos que una iteración de máximo descenso, esta dada por :

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$$

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

El el caso que $p^{(k)}$ sea un vector unitario con dirección opuesta al gradiente, el algoritmo se llamará, Máximo Descenso de Gradiente.

$$x^{(k+1)} = x^{(k)} - \left[\frac{\nabla f(x^{(k)})^T \nabla f(x^{(k)})}{\nabla f(x^{(k)})^T \nabla^2 f(x^{(k)}) \nabla f(x^{(k)})} \right] \nabla f(x^{(k)})$$

La implementación en Java es:

```
public double alpha_MD(Matriz x, double alpha_0)
{
    Matriz g = Gradiente(x);
    Matriz p = Direccion(g);
    Matriz x1 = x.mas(p.por(alpha_0));
    g = Gradiente(x1);
    Matriz H = Hessiano(x1);

    Matriz alfa = (p.T().por(g)).entre(p.T().por(H.por(p)));
    return (alpha_0-alfa.obten(0,0));
}
```

2.7.2. Interpolación

Suponga que tenemos un valor α_0, tal que cumple con la regla de Armijo (2.18):

$$\phi(\alpha^{(0)}) \leq \phi(0) + c_1 \alpha^{(0)} \phi'(0)$$

donde $\phi(\alpha^{(0)}) = f(x^{(k)} + \alpha^{(0)} p^{(k)})$ y $\phi'(0) = \nabla^T (f(x^{(k)} + \alpha^{(0)} p^{(k)})) p^{(k)}$

De ser así, tenemos que el valor que minimiza la función $\phi(\alpha)$, se encuentra en el intervalo $[\alpha^{(0)}, \alpha^{(1)}]$. Así pues hacemos una aproximación cuadrática de $\phi_q(\alpha)$ dada por:

$$\phi_q(\alpha) = A(\alpha - \alpha_0)^2 + B(\alpha - \alpha_0) + C \tag{2.13}$$

Dado que conocemos $\phi(\alpha_0)$, $\phi'(\alpha_0)$ tenemos la información suficiente para calcular los coeficientes de la ecuación cuadrática 2.13.

Para determinar C, valuamos la función 2.13, en α_0

$$\phi_q(\alpha^{(0)}) = C = \phi(\alpha^{(0)})$$

Para calcular B, derivamos la función 2.13 y la valuamos en α_0

$$\phi'_q(\alpha) = 2A(\alpha - \alpha^{(0)}) + B = \phi'(\alpha)$$

Sustituyendo $\alpha = \alpha^{(0)}$

$$\phi'_q(\alpha^{(0)}) = 2A(\alpha^{(0)} - \alpha^{(0)}) + B = \phi'(\alpha^{(0)})$$
$$\therefore B = \phi'(\alpha^{(0)}) = \nabla^T f(x^{(k)} + \alpha^{(0)} p^{(k)}) p^{(k)}$$

Finalmente el valor de A lo calculamos sustituyendo B y C en la ecuación 2.13

$$\phi_q(\alpha^{(1)}) = \phi(\alpha^{(1)}) = A(\alpha^{(1)} - \alpha^{(0)})^2 + \phi'(\alpha^{(0)})(\alpha^{(1)} - \alpha^{(0)}) + \phi(\alpha^{(0)})$$

$$A = \frac{\phi(\alpha^{(1)}) - \phi'(\alpha^{(0)})(\alpha^{(1)} - \alpha^{(0)}) - \phi(\alpha^{(0)})}{(\alpha^{(1)} - \alpha^{(0)})^2}$$

El valor de α que minimiza 2.13 se calcula haciendo

$$\phi'(\alpha) = 2 * A(\alpha - \alpha^{(0)}) + B = 0$$

$$\alpha = \alpha^{(0)} - \frac{B}{2A}$$

$$\alpha = \alpha^{(0)} - \frac{\phi'(\alpha^{(0)})(\alpha^{(1)} - \alpha^{(0)})^2}{2(\phi(\alpha^{(1)}) - \phi'(\alpha^{(0)})(\alpha^{(1)} - \alpha^{(0)}) - \phi(\alpha_0))} \tag{2.14}$$

Su implementación en Java es:

```
public double alpha_Int(Matriz x, Matriz p, double alpha_0, double alpha_1)
{
    double a;

    double dfi_0 = this.phi_p(x, alpha_0, p);
    double delta = alpha_1 -alpha_0;

    a = alpha_0 - (dfi_0*delta*delta)/ 2.0 /
        (phi(x, alpha_1, p) - dfi_0 * delta - phi(x, alpha_0, p));
    return a;
}
```

2.7.3. Razón Dorada

El valor de alfa también se puede calcular utilizando el método de Razón Dorada. La implementación en Java es:

```
public double alpha_RD(Matriz x, Matriz p, double alow, double aupper)
{
    double d, a1, a2, R = (Math.sqrt(5.0) -1.0)/2.0, a;

    do
    {
        d = (aupper - alow)*R;

        a1 = alow   + d;
        a2 = aupper - d;

        if(phi(x, a1, p) < phi(x, a2, p))
            alow = a2;
         else
           aupper = a1;

    } while (aupper - alow > 1e-6);

    a = phi(x,a1, p) < phi(x,a2,p) ? a1 : a2;
    a = phi(x, alow, p) < phi(x, a, p) ? alow: a;
    a = phi(x, aupper, p) < phi(x, a, p) ? aupper: a;

    return(a);
}
```

2.7.4. Método de Newton

Consideremos que la función $\phi : \mathbb{R} \to \mathbb{R}$ es de clase dos. Siguiendo con la idea de reemplazar en la vecindad del punto α_k de la función ϕ por una aproximación cuadrática $\phi(\alpha)$ dada por

$$\phi(\alpha^{(k+1)}) = \phi(\alpha^{(k)}) + \phi'(\alpha^{(k)})(\alpha^{(k+1)} - \alpha^{(k)}) + \frac{1}{2}\phi''(\alpha^{(k)})(\alpha^{(k+1)} - \alpha^{(k)})^2 \tag{2.15}$$

Comenzaremos por recordar que $\phi(\alpha^{(k)}) = f(x^{(0)} + \alpha^{(k)}p^{(0)})$ y procederemos a hacer la expansión en serie de Taylor para $f(x^{(k+1)})$

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

Dado un punto inicial $x^{(0)}$ y que la dirección de búsqueda es $p^{(0)}$, suponemos que tenemos diferentes valores de $x^{(k)}$ modificando la variable $\alpha^{(k)}$ como $x^{(k+1)} = x^{(k)} + p^{(k)}\alpha^{(k)}$. En

base a esto podemos decir que

$$x^{(k+1)} - x^{(k)} = (x^{(0)} + p^{(0)}\alpha^{(k+1)}) - (x^{(0)} + p^{(0)}\alpha^{(k)}) = p^{(0)}(\alpha^{(k+1)} - \alpha^{(k)})$$

$$\begin{aligned} f(x^{(0)} + p^{(0)}\alpha^{(k+1)}) &= f(x^{(0)} + p^{(0)}\alpha^{(k)}) + {p^{(0)}}^T \nabla f(x^{(0)} + p^{(0)}\alpha^{(k)})(\alpha^{(k+1)} - \alpha^{(k)}) \\ &+ \frac{1}{2}{p^{(0)}}^T \nabla^2 f(x^{(0)} + \alpha^{(k)}p^{(0)})p^{(0)}(\alpha^{(k+1)} - \alpha^{(k)})^2 \end{aligned} \tag{2.16}$$

Podemos notar que (2.15) y (2.16) son equivalentes si

$$\begin{aligned} \phi(\alpha^{(k+1)}) &= f(x^{(0)} + p^{(0)}\alpha^{(k+1)}) \\ \phi(\alpha^{(k)}) &= f(x^{(0)} + p^{(0)}\alpha^{(k)}) \\ \phi'(\alpha^{(k)}) &= {p^{(0)}}^T \nabla f(x^{(0)} + p^{(0)}\alpha^{(k)}) \\ \phi''(\alpha^{(k)}) &= {p^{(0)}}^T \nabla^2 f(x^{(0)} + \alpha^{(k)}p^{(0)})p^{(0)} \end{aligned}$$

El mínimo de la ecuación (2.15) lo podemos calcular como

$$\phi'(\alpha^{(k)}) + \phi''(\alpha^{(k)})(\alpha^{(k+1)} - \alpha^{(k)}) = 0$$

despejando para $\alpha^{(k+1)}$ tenemos

$$\begin{aligned} \alpha^{(k+1)} &= \alpha^{(k)} - \frac{\phi'(\alpha^{(k)})}{\phi''(\alpha^{(k)})} \\ \alpha^{(k+1)} &= \alpha^{(k)} - \frac{{p^{(0)}}^T \nabla f(x^{(0)} + p^{(0)}\alpha^{(k)})}{{p^{(0)}}^T \nabla^2 f(x^{(0)} + \alpha^{(k)}p^{(0)})p^{(0)}} \end{aligned} \tag{2.17}$$

El algoritmos de Newton para búsqueda del α, de máximo descenso se muestra, en el algoritmo 5.

La implementación en Java para este algoritmo es

```
public double alpha_Newton(Matriz x, Matriz p)
{
    double a_fin, alpha , phi, phi_ant;

    alpha = 0;
```

Algoritmo 5 Algoritmo de Newton Unidimensional

Entrada: $f(x), x_0$
Salida: $\alpha^{(k)}$

Dado una función diferenciable dos veces $f(x)$
una dirección de búsqueda $p^{(0)}$ y un valor inicial $x^{(0)}$
mientras $\|{p^{(0)}}^T \nabla f(x^{(0)} + \alpha^{(k)})\| \geq \varepsilon$ **hacer**
$\alpha^{(k+1)} = \alpha^{(k)} - \frac{{p^{(0)}}^T \nabla f(x^{(0)} + p^{(0)}\alpha^{(k)})}{{p^{(0)}}^T \nabla^2 f(x^{(0)} + \alpha^{(k)} p^{(0)}) p^{(0)}}$
$k = k + 1$
fin mientras
devolver $\alpha^{(k)}$

```
        phi = phi(x, alpha, p);

        do {
            phi_ant = phi;
            alpha += da;
            phi = phi(x,alpha,p);
        } while (phi < phi_ant);

        a_fin = alpha;

        Matriz g, H, delta;
        //System.out.println("   Alfa = " + alpha);

        for(int i=0; i<100; i++){
            //System.out.println("a = " + alpha);
            g = Gradiente(x.mas(p.por(alpha)));
            H = Hessiano(x.mas(p.por(alpha)));
            delta = (p.T().por(g)).entre(p.T().por(H.por(p)));
            alpha -= delta.obten(0, 0);
            //System.out.println("        Delta = " + delta.obten(0,0));
            if(Math.abs(delta.obten(0, 0)) < 1e-12) break;
        }
        //System.out.println("   Alfa = " + alpha);

        if(Checa_Wolfe(x, alpha, p) && alpha >= 0) return alpha;
        else return a_fin;
        //return alpha;
    }
```

2.7.5. Ejemplo

Para la función de Ronsenbrock determinar el valor del paso optimo α^* para un punto inicial $x^{(0)} = [2, 3]$ en un rango $\alpha \in [0, 0.3]$, utilizando máximo descenso, interpolación y Newton de máximo descenso. La función de Ronsenbrock es

$$f(x) = 100 * (x_2 - x_1^2)^2 + (1 - x_1)^2$$

El gradiente de la función para cualquier valor de x es:

$$\nabla f(x) = \begin{bmatrix} -400(x_2 - x_1^2)x_1 + 2x_1 - 2 \\ 200x_2 - 200x_1^2 \end{bmatrix}$$

El hessiano es

$$\nabla^2 f(x) = \begin{array}{|c|c|} \hline 1200x_1^2 - 400x_2 + 2 & -400x_1 \\ \hline -400x_1 & 200 \\ \hline \end{array}$$

Si valuamos el gradiente y el hessiano en $x_0 = [2, 3]^T$

$$\nabla f(2, 3) = \begin{bmatrix} 802 \\ -200 \end{bmatrix}$$

$$\nabla^2 f(2, 3) = \begin{bmatrix} 3602 & -800 \\ -800 & 200 \end{bmatrix}$$

La dirección de búsqueda es

$$p = - \begin{bmatrix} 802/\sqrt{802^2 + 200^2} \\ -200/\sqrt{802^2 + 200^2} \end{bmatrix} = \begin{bmatrix} -0.97028 \\ +0.24197 \end{bmatrix}$$

Máximo descenso

Para el calculo utilizamos la ecuación 2.12

$$\alpha^{(k)} = \frac{- \left[-0.97028, 0.24197\right] \begin{bmatrix} 802 \\ -200 \end{bmatrix}}{\left[-0.97028, 0.24197\right] \begin{bmatrix} 3602 & -800 \\ -800 & 200 \end{bmatrix} \begin{bmatrix} -0.97028 \\ +0.24197 \end{bmatrix}} = 0.2188$$

Interpolación cuadrática

De acuerdo con la ecuación 2.14 necesitamos calcular $\phi(0)$, $\phi'(0)$ y dado $\alpha_0 = 0$ $\alpha_0 = 0.3$. De la definición tenemos que

$$\phi(\alpha^{(k)}) = f(x^{(k)} + \alpha^{(k)} p^{(k)})$$

$$\phi(\alpha^{(0)}) = \phi(0) = f(2,3) = 101$$

$$\phi'(\alpha^{(k)}) = p^{(k)^T} \nabla f(x^{(k)} + \alpha^{(k)} p^{(k)})$$

$$\phi'(\alpha^{(0)}) = \phi'(0) = p^{(k)^T} \nabla f(x^{(k)}) = [-0.97028, 0.24197] \begin{bmatrix} 802 \\ -200 \end{bmatrix} = -826.5586$$

$$\phi(\alpha^{(1)}) = \phi(0.3) = 2.8189$$

El valor final para el paso es

$$\alpha^{(k)} = 0 - \frac{(-826.5586) * (0.3 - 0)^2}{2 * [2.8189 - 101 - (-826.5586) * (0.3 - 0)]} = 0.2483$$

Newton

La actualización del método de newton es

$$\alpha^{(k+1)} = \alpha^{(k)} - \frac{\phi'(\alpha^{(k)})}{\phi''(\alpha^{(k)})}$$

Las iteraciones para este método son:

$$\begin{aligned}
\alpha^{(1)} &= 0.000000 - \frac{802.000000}{3602.0} = 0.218756 \\
\alpha^{(2)} &= 0.218756 - \frac{103.904780}{2616.063257} = 0.257093 \\
\alpha^{(3)} &= 0.257093 - \frac{3.044904}{2454.412016} = 0.258247 \\
\alpha^{(4)} &= 0.258247 - \frac{0.102357}{2449.595032} = 0.258248 \\
\alpha^{(5)} &= 0.258248 - \frac{0.099721}{2449.590714} = 0.258248
\end{aligned}$$

Ejemplo

Para la función $f(x) = x_1 e^{-x_1^2 - x_2^2}$, un punto de coordenadas $x^{(0)} = [-0.5, -0.5]$ y una dirección $\theta = 30^0$ determinar:

1. La derivada direccional en la dirección dada,
2. el mínimo en la dirección dada utilizando interpolación cuadrática y
3. el mínimo utilizando Razón Dorada.

El vector gradiente para esta función es:

$$\nabla f(x) = \begin{bmatrix} (1-2x_1^2)e^{(-x_1^2-x_2^2)} \\ -2x_1x_2e^{(-x_1^2-x_2^2)} \end{bmatrix}$$

El Hessiano es

$$\nabla^2 f(x) = \begin{bmatrix} (-6x_1+4x_1^3)e^{(-x_1^2-x_2^2)} & (-2x_2+4x_1^2x_2)e^{(-x_1^2-x_2^2)} \\ (-2x_2+4x_1^2x_2)e^{(-x_1^2-x_2^2)} & (-2x_1+4x_1x_2^2)e^{(-x_1^2-x_2^2)} \end{bmatrix}$$

El vector unitario en la dirección dada es $u_{30} = [0.866, 0.5]^T$, el gradiente valuado en el punto dado es $\nabla f(-0.5, -0.5) = [0.303265, -0.303265]^T$ y la derivada direccional

$$D_{u_{30}} = [0.303265, -0.303265] \begin{bmatrix} 0.866 \\ 0.5 \end{bmatrix} = 0.11102$$

Con esto podemos definir $l(\alpha) = f(x) + \alpha D_{u_{30}} = -0.303265 + 0.11102\alpha$ y $\phi(\alpha) = f(x + \alpha D_{u_{30}})$. Suponiendo $\alpha_0 = 0.1$ tenemos:

$$x_1 = \begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix} + 0.1 * \begin{bmatrix} 0.866 \\ 0.5 \end{bmatrix} = \begin{bmatrix} -0.4134 \\ -0.45 \end{bmatrix}$$

Utilizando el algoritmo de interpolación cuadrática tenemos que

$$\alpha = \alpha^{(0)} - \frac{\phi'(\alpha^{(0)})(\alpha^{(1)} - \alpha^{(0)})^2}{2(\phi(\alpha_1) - \phi'(\alpha^{(0)})(\alpha^{(1)} - \alpha^{(0)}) - \phi(\alpha^{(0)}))}$$

$$\alpha = 0 - \frac{0.11102 * (0.1-0)^2}{2(-0.28458) - 0.11102 * (0.1-0) - 0.303265)}$$

$$\alpha = -0.073154$$

Con el algoritmo de razón dorada tenemos que $\alpha = 8.0123461723868 * 10^{-16}$.

Podemos notar que el valor *alpha* interpolado es negativo y el de razón dorada es casi cero. Esta condición se debe a que en la dirección dada la curvatura es negativa y por lo tanto en la dirección de u_{30^0} hay un máximo, tal como se puede confirmar en la figura 2.8

2.8. Las condiciones de Wolfe

La popular e inexacta búsqueda lineal estipula que α_k debe dar suficiente decrecimiento en la función objetivo f, como medida de esto se tiene la siguiente desigualdad:

$$f(x^{(k)} + \alpha p^{(k)}) \leq f(x^{(k)}) + c_1 \alpha \nabla f(x^{(k)})^T p^{(k)}$$

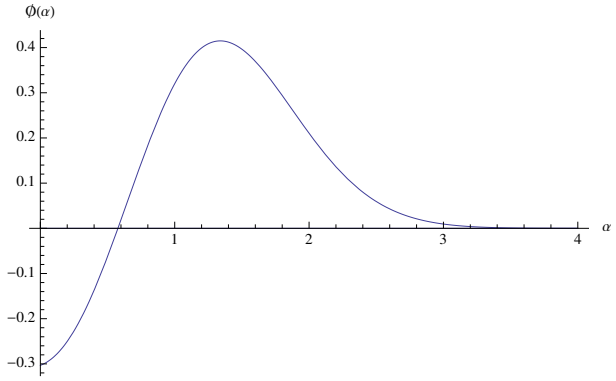


Figura 2.8: Función $\phi(\alpha)$ para la función $xe^{-x^2-y^2}$, en la dirección u_{30}

Si definimos $\phi(\alpha) = f(x^{(k)} + \alpha p^{(k)})$ y $l(\alpha) = f(x^{(k)}) + c_1\alpha\nabla f(x^{(k)})^T p^{(k)}$, podemos escribir

$$\phi(\alpha) \leq l(\alpha)$$

para alguna constante $c_1 \in (0, 1)$. En otras palabras, la reducción en f debe ser proporcional a la longitud de paso y la derivada direccional $\nabla f_k^T p_k$. Esta desigualdad es llamada *La condición de Armijo.*

La condición suficiente de decaimiento es ilustrada en la figura 2.9. El lado derecho de la ecuación la llamaremos $l(\alpha)$. La función $l(\alpha)$ tiene pendiente negativa c, $\nabla {f_K}^T P_K$ pero dada $c_1 \in (0, 1)$ esta miente con respecto a que tan pequeña deben de ser los valores de α. En la práctica podemos decir que $c_1 = 10^{-4}$.

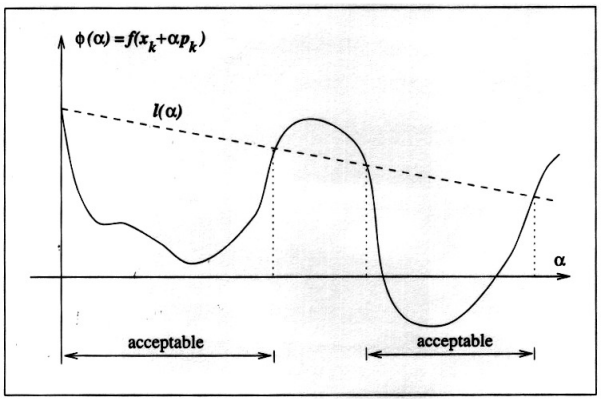


Figura 2.9: Condición de decrecimiento suficiente

La condición de Armijo no es suficiente por si misma para asegurar que el algoritmo tiene

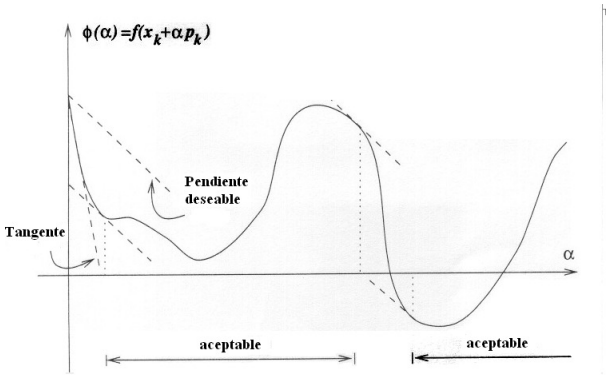


Figura 2.10: Condición de curvatura

progresos razonables. Por otro lado introduciremos un segundo requerimiento llamada la condición de curvatura, la cual requiere que α_k satisfaga:

$$\nabla f(x^{(k)} + \alpha p^{(k)})^T p_k \geq c_2 \nabla f(x^{(k)})^T p^{(k)} \tag{2.18}$$

donde: $c_2 \in (c_1, 1)$. Esta condición se observa en la figura 2.10.

Las condiciones de decrecimiento suficiente y pendiente son conocidas como las condiciones de Wolfe, las cuales están ilustradas en la figura 2.11 y se resumen en 2.19

$$f(x^{(k)} + \alpha p^{(k)}) \leq f(x^{(k)}) + c_1 \alpha \nabla f(x^{(k)})^T p^{(k)}$$

$$\nabla f(x^{(k)} + \alpha p^{(k)})^T p_k \geq c_2 \nabla f(x^{(k)})^T p^{(k)} \tag{2.19}$$

Ver [Nocedal and Wright, 1999]

Las condiciones fuertes de Wolfe están dada por las ecuaciones (2.20) y garantizan que tenemos las condiciones suficientes para tener un mínimo.

$$\begin{aligned} f(x^{(k)} + \alpha p^{(k)}) &\leq f(x^{(k)}) + c_1 \alpha \nabla f(x^{(k)})^T p^{(k)} \\ |\nabla f(x^{(k)} + \alpha p^{(k)})^T p^{(k)}| &\leq c_2 |\nabla f(x^{(k)})^T p^{(k)}| \end{aligned} \tag{2.20}$$

El código Java para comprobar la condiciones fuertes de wolfe es:

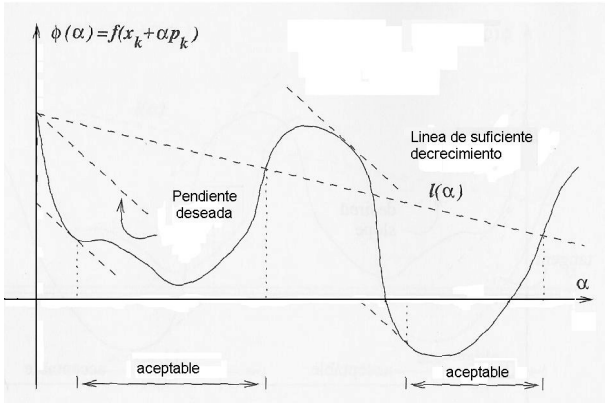


Figura 2.11: Condiciones de Wolfe

```
boolean Condiciones_Fuertes_Wolfe(Matriz x, double alfa, Matriz p) {
    boolean curvatura, armijo;
    double recta =  phi(x, 0, p) + c1*phi_p(x, alfa, p)*alfa;

    curvatura =
          (Math.abs(phi_p(x, alfa, p)) <=  c2*Math.abs(phi_p(x, 0, p)));
    armijo = (recta >= phi(x, alfa, p));

    return (curvatura && armijo);
}
```

Las funciones $\phi(x)$ y $\phi'(x)$ se implementaron en Java como

```
public double phi(Matriz x, double alfa, Matriz p)
{
    Matriz x1 = x.mas(p.por(alfa));
    return(funcion(x1));
}

public double phi_p(Matriz x, double alfa, Matriz p )
{
    Matriz g = Gradiente(x.mas(p.por(alfa)));

    return ((p.T().por(g)).obten(0,0));
}
```

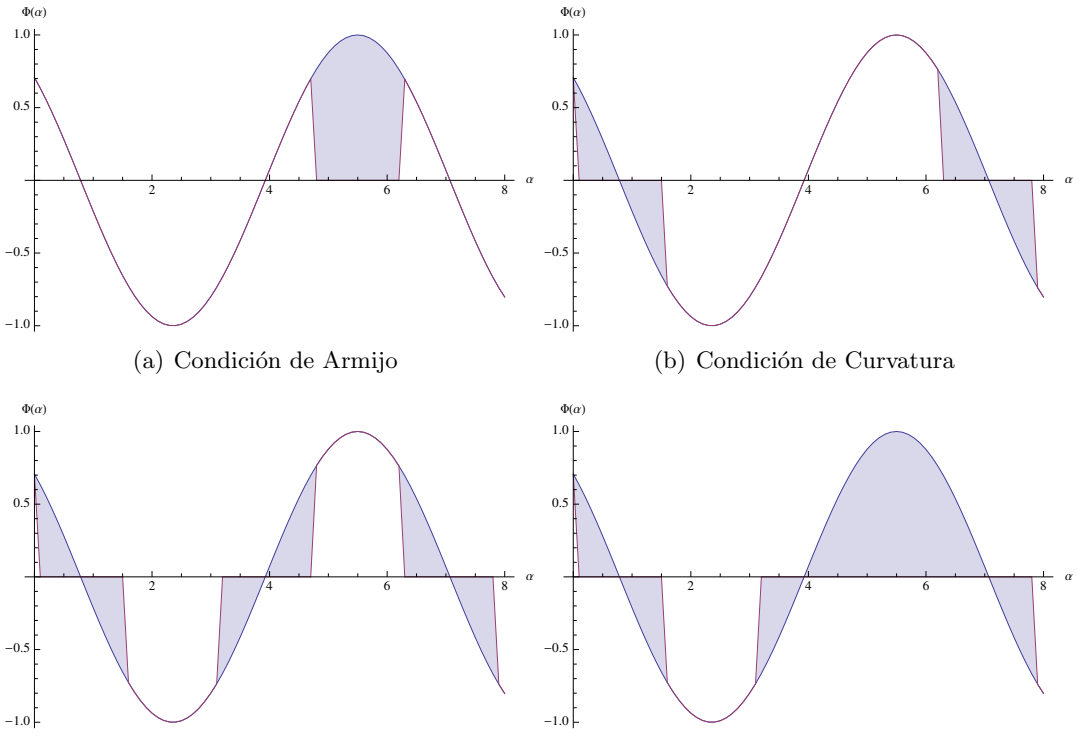
2.8.1. Ejemplo 1

Para una función $f(x)$ tenemos una función $\Phi(\alpha) = cos(\alpha + \frac{\pi}{4})$, calcular la gráfica $\Phi(\alpha)$ donde se cumplen:

1. las condiciones de armijo,
2. las condiciones de curvatura,
3. las condiciones fuertes de curvatura y
4. las condiciones fuentes de wolfe

considere $c_1 = 10^{-4}$ y $c_2 = 1$

En la figura 2.12 se muestran las condiciones de Wolfe para la función, en obscuro se muestra la región donde no se cumplen las condiciones. Las condiciones de Armijo, Curvatura, Curvatura fuerte y Wolfe fuerte se muestran en las figuras 2.12(a), 2.12(b), 2.12(c) y 2.12(d) respectivamente.



(a) Condición de Armijo

(b) Condición de Curvatura

(c) Condición de Curvatura fuerte

(d) Condiciones fuertes de Wolfe

Figura 2.12: Condiciones de Wolfe

2.8.2. Ejemplo 2

Para una función $f(x) : \mathbb{R}^{10} \to \mathbb{R}$ dada como

$$f(x) = \sum_{i=1}^{10} \left[(x_i - y_i)^2 + 2.5(x_i - x_{i-1})^2 \right]$$

Considere que el termino $2.5(x_i - x_{i-1})^2$ solo existe para valores $i - 1 > 0$.

Determinar el rango de valores en el cual se cumplen las condiciones de fuertes de Wolfe. Considere $c_1 = 10^{-4}$, $c_2 = 1.0$, $y = [1, 2, 3, 4, 5, 4, 3, 2, 1, 0]^T$ y $x^{(0)} = y$

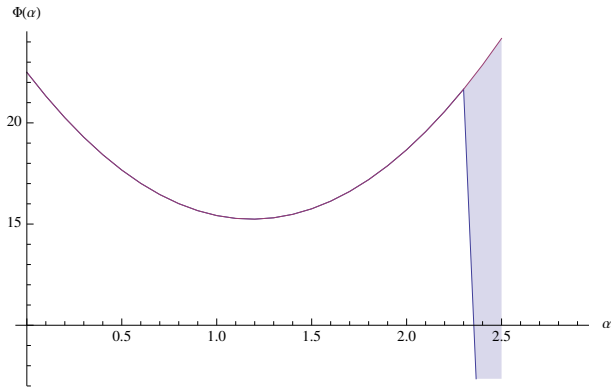


Figura 2.13: Condiciones fuertes de wolfe

La implementación de la función el Gradiente y el Hessiano se presenta a continuación

```
public double funcion(Matriz x) {
    double f = 0, y[] = {1,2,3,4,5,4,3,2,1,0};
    double aux;
    int N = y.length, i;
    for(i=0; i<N; i++) {
        aux = x.obten(i, 0) - y[i];
        f += aux*aux;
        if(i>0) {
            aux = x.obten(i, 0)-x.obten(i-1, 0);
            f += 2.5 * aux * aux;
        }
    }
    return f;
}
```

```
public Matriz Gradiente(Matriz x) {
    int i, N = 10;
    double g[] = new double[N], y[] = {1,2,3,4,5,4,3,2,1,0};

    for(i=0; i<N; i++){
        g[i] = 2.0*(x.obten(i, 0) - y[i]);
        if(i>0)   g[i] += 5.0*(x.obten(i, 0)  - x.obten(i-1, 0));
        if(i<N-1) g[i] += 5.0*(x.obten(i, 0)  - x.obten(i+1, 0));
    }


    Matriz grad = new Matriz(g);
    return grad;
}

public Matriz Hessiano(Matriz x) {
    double H[][] = {
    { 7, -5,  0,  0,  0,  0,  0,  0,  0,  0},
    { -5,12, -5,  0,  0,  0,  0,  0,  0,  0},
    {  0, -5,12, -5,  0,  0,  0,  0,  0,  0},
    {  0,  0, -5,12, -5,  0,  0,  0,  0,  0},
    {  0,  0,  0,-5, 12, -5,  0,  0,  0,  0},
    {  0,  0,  0, 0, -5, 12, -5,  0,  0,  0},
    {  0,  0,  0, 0,  0, -5, 12, -5,  0,  0},
    {  0,  0,  0, 0,  0,  0, -5, 12, -5,  0},
    {  0,  0,  0,  0, 0,  0,  0, -5, 12, -5},
    {  0,  0,  0,  0,  0, 0,  0,  0, -5,  7}};

    Matriz Hess = new Matriz(H);
    return Hess;
}
```

En la figura 2.13, se muestra el intervalo en el cual son validas las condiciones fuertes de wolfe. En obscuro se muestra la región donde ya no se cumplen dichas condiciones para la función dada. El código correspondiente a esta implementación es:

```
void Condiciones_Wolfe(Matriz x, Matriz p, double alfa_max){
    double alfa;

    for(alfa =0; alfa<=alfa_max; alfa+= 0.1) {
        if(Condiciones_Fuertes_Wolfe(x, alfa, p))
```

```
            System.out.printf("%4.2f \n ", alfa);
    }
}
```

Para detalles ver **Gradiente_filtro.java**

Tarea: Condiciones de Wolfe

Dada la función $100(x_2 - x_1^2)^2 + (1 - x_1^2)$, determinar según las condiciones de Wolfe, cuales son los valores de α aceptables. Considere valores para $c_1 = 10^{-4}$ y $c_2 = 1.0$ y como valor inicial $x_0 = [2, 3]^T$

2.8.3. Algoritmo de Suficiente decaimiento y muestreo hacia atrás

La condición de Armijo (ecuación (2.18)), también llamada condición de suficiente decaimiento es suficiente, no garantiza que el algoritmo seleccione una longitud de paso apropiadamente. Sin embargo si utilizamos el algoritmo de muestreo hacia atrás, la condición de Armijo será suficiente.

Algoritmo 6 Búsqueda-lineal-con muestreo-hacia-atras

Entrada: $\overline{\alpha}, \rho, c_1$
Salida: $\alpha^{(k)}$
Dado un valor inicial $\overline{\alpha} > 0$, un factor de incremento ρ y un escalar $c_1 \in (0, 1)$
Hacemos $\alpha = \overline{\alpha}$
mientras $f(x^{(k)} + \alpha * p^{(k)}) <= f(x^{(k)}) + c_1 \alpha \nabla f(x^{(k)})^T p^{(k)}$ **hacer**
$\alpha = \rho * \alpha$
fin mientras
$\alpha^{(k)} = \alpha$
devolver $\alpha^{(k)}$

2.9. Método de Descenso de gradiente

El método del descenso de gradiente es un algoritmo de búsqueda lineal en la dirección de gradiente. Los pasos de este algoritmo son los que se muestran en el algoritmo (7) y su implementación en Java es::

```
public Matriz Descenso_de_Gradiente(Matriz x0)
{
```

Algoritmo 7 Método de descenso de Gradiente

Entrada: $f(x), x^{(0)}$
Salida: $x^{(k)}$

Dado una función diferenciable $f(x)$ y un valor inicial $x^{(0)}$
Hacemos $k = 0$
mientras $\|\nabla f(x^{(k)})\| \geq \varepsilon$ **hacer**
Evaluar la dirección de búsqueda $p^{(k)} = -\frac{\nabla f(x^{(k)})}{\|\nabla f(x^{(k)})\|}$
Calcular el valor de $\alpha^{(k)}$
Hacer la actualización $x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$
$k = k + 1$
fin mientras
devolver $x^{(k)}$

```
Matriz g, x = null, p;
double alpha;
int iter = 1;

x = Matriz.igual_a(x0);

do
{
    // calcula el gradiente de la funcion
    g = Gradiente(x);
    // Direccion opuesta al gradiente
    p = Direccion(g);
    // alfa minima
    alpha = alpha_Newton(x, p);

    System.out.println("X  = ");
    x.imprime();
    System.out.println("Direccion = ");
    p.imprime();
    System.out.println("alfa = " + alpha);

    // Actualizacion x = x + alpha*p
    x = x.mas(p.por(alpha));

    System.out.println(iter++ + " alpha = " +
    alpha + " f(x) = " +funcion(x));
```

```
            //x.imprime();
        }while (Magnitud(g) > T );
        return x;
    }
```

El valor de α_k, para el algoritmo 7, puede ser calculado utilizando:

- Búsqueda exhaustiva (1.1)
- Razón dorada (sección 2)
- Búsqueda lineal con muestreo hacia atrás (sección 2.8.3).
- Máximo descenso (eq. (2.12))
- Interpolación cuadrática (eq. (2.14))
- Método de Newton de Alfa de máximo descenso ((2.17))

2.9.1. Ejemplo 1

Dada la función $f(x) = 4x_1^2 + x_2^2$ y un punto inicial $x^{(0)} = [2, 2]^T$, calcular el mínimo utilizando el algoritmo de descenso de gradiente. En todos los casos calcular el ángulo entre direcciones consecutivas utilizando la ecuación $\cos\theta = \left[p^{(k+1)}\right]^T p^{(k)}$

Para la función tenemos que el gradiente es:

$$\nabla f(x) = \begin{bmatrix} 8x_1 \\ 2x_2 \end{bmatrix}$$

y su Hessiano es:

$$\nabla^2 f(x) = \begin{bmatrix} 8 & 0 \\ 0 & 2 \end{bmatrix}$$

Las iteraciones necesarias para que converja el algoritmo se muestran en la siguiente tabla

k	$x_1^{(k)}$	$x_2^{(k)}$	$p_1^{(k)}$	$p_2^{(k)}$	$\theta^{(k)}$
0	2.0000	2.0000	-0.9701	-0.2425	–
1	-0.0923	1.4769	0.2425	-0.9701	90
2	0.2215	0.2215	-0.9701	-0.2425	90
3	-0.0102	0.1636	0.2425	-0.9701	90
4	0.0245	0.0245	-0.9701	-0.2425	90
5	-0.0011	0.0181	0.2425	-0.9701	90
⋮	⋮	⋮	⋮	⋮	⋮
12	0.0000	0.0000	-0.9701	-0.2425	90
13	0.0000	0.0000	0.2425	-0.9701	90

La Figura 2.14, se muestra las iteraciones así como cuatro de los pasos que el algoritmo da para llegar a la solución. Note como los pasos se llevan a cabo en ángulos de noventa grados y las direcciones de descenso están a noventa grados de las curvas de nivel.

ver Gradiente_ej02.java

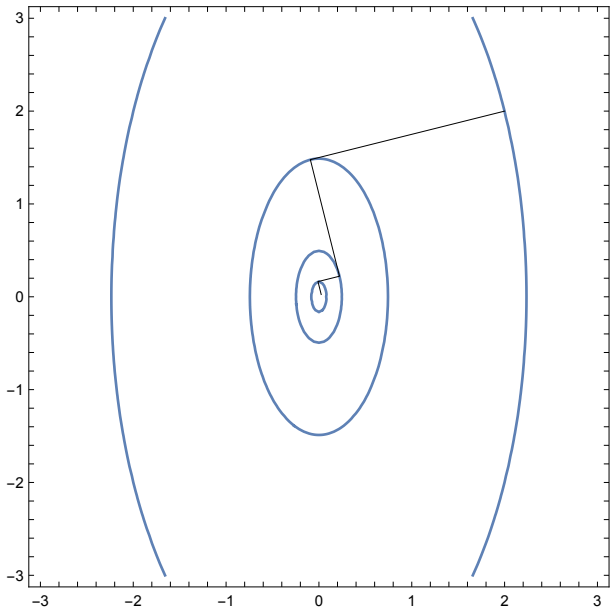


Figura 2.14: Curvas de nivel y direcciones de decenso

2.9.2. Ejemplo 2

Dada la función $f(x) = 100 * (x_2 - x_1^2)^2 + (1 - x_1)^2$ y un punto inicial $x^{(0)} = [2, 3]^T$, calcular el mínimo utilizando el algoritmo de descenso de gradiente. En todos los casos calcular el

ángulo entre direcciones consecutivas utilizando la ecuación $\cos\theta = \left[p^{(k+1)}\right]^T p^{(k)}$

La siguiente tabla muestra el proceso de convergencia del algoritmo de Descenso de Gradiente

k	$x_1^{(k)}$	$x_2^{(k)}$	$p_1^{(k)}$	$p_2^{(k)}$	$\theta^{(k)}$
0	2.0000	3.0000	-0.9703	0.2420	–
1	1.7494	3.0625	-0.2420	-0.9703	90
2	1.7220	2.9526	-0.9703	0.2420	90
3	1.7180	2.9536	-0.2420	-0.9703	90
4	1.6959	2.8648	-0.9703	0.2420	90
5	1.6923	2.8657	-0.2420	-0.9703	90
6	1.6735	2.7906	-0.9703	0.2420	90
7	1.6702	2.7914	-0.2420	-0.9703	90
⋮	⋮	⋮	⋮	⋮	⋮
2384	1.0000	1.0000	-0.9703	0.2420	90
2385	1.0000	1.0000	-0.2420	-0.9703	90

ver Gradiente_Rosenbrock.java para detalle de implementación

2.9.3. Convergencia del Método de descenso de gradiente

La principal desventaja del método de descenso de gradiente es el hecho de que la convergencia puede ser muy lenta para cierto tipo de funciones o valores iniciales. En general direcciones de búsqueda consecutivas están a 90 grados tal como se muestra en la figura 2.15.

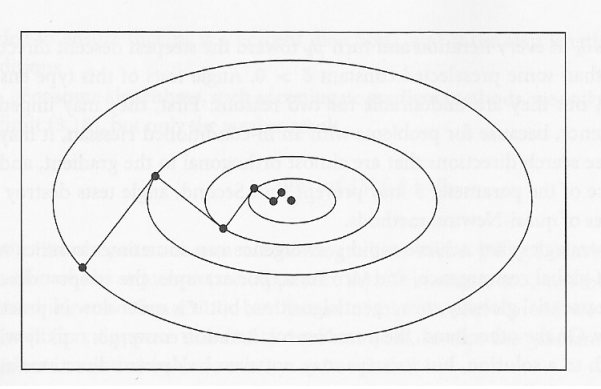


Figura 2.15: Direcciones de búsqueda para descenso de gradiente

Prueba
En los métodos de descenso de gradiente queremos calcular el valor α que minimiza

$$\phi(\alpha) = f(x^{(k)} + \alpha p^{(k)})$$

por lo tanto tenemos que calcular la derivada, de esta, respecto a $\alpha^{(k)}$ e igualar a cero

$$\frac{\mathrm{d}\phi(\alpha)}{\mathrm{d}\alpha} = {p^{(k)}}^T \nabla f(x^{(k)} + \alpha p^{(k)}) = {p^{(k)}}^T p^{(k+1)} = 0$$

De la definición del producto escalar de vectores, sabemos que:

$$\|p^{(k)}\| \, \|p^{(k+1)}\| \cos\theta = 0$$

la única manera de que esto se cumpla es que $\theta = 90^O$. Por lo tanto, las direcciones de búsqueda se encuentran a 90grados.

Tarea: Comparación de los métodos de búsqueda

Implementar el algoritmo de descenso de gradiente 7 y comparar su desempeño con las diferentes estrategias de búsqueda lineal para calcular el α_k

2.9.4. Método de Forsyte

Para eliminar la característica de zigzag (ver figura 2.16) de los métodos de descenso de gradiente cuando se aplica a una función mal condicionada el siguiente método de aceleración propuesto por Forsythe and Luenberger. En cada paso k, almacenamos (comenzando en $x^{(k)}$) m estados del método de descenso de gradiente, el cual producirá un punto al que llamaremos $y^{(k)}$. El punto $x^{(k+1)}$ es entonces calculado como la minimización en la dirección $d^{(k)} = (y^{(k)} - x^{(k)})/|y^{(k)} - x^{(k)}|$ en lugar de $p^{(k)}$ (dirección opuesta al gradiente).

Este algoritmo queda como se muestra en 8:

- Dado un $x^{(k)}$ y m estados,
- Calculamos los valores temporales de $x^{(k+1)}$, $x^{(k+2)}$, ..., $x^{(k+m)}$ utilizando el algoritmo de descenso de gradiente.
- hacemos $y^{(k)} = x^{(k+m)}$
- la nueva dirección de búsqueda es $d^{(k)} = (y^{(k)} - x^{(k)})/|y^{(k)} - x^{(k)}|$
- Buscamos el valor de $\alpha^{(k)}$ que minimiza $f(x^{(k)} + \alpha^{(k)} d^{(k)})$

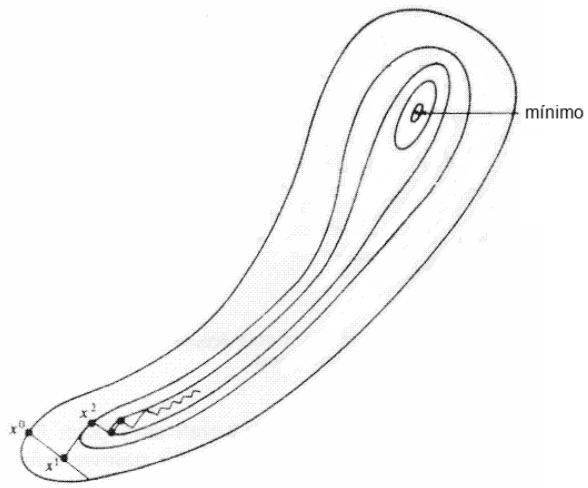


Figura 2.16: Efecto de zigzag creado por el método de descenso de gradiente

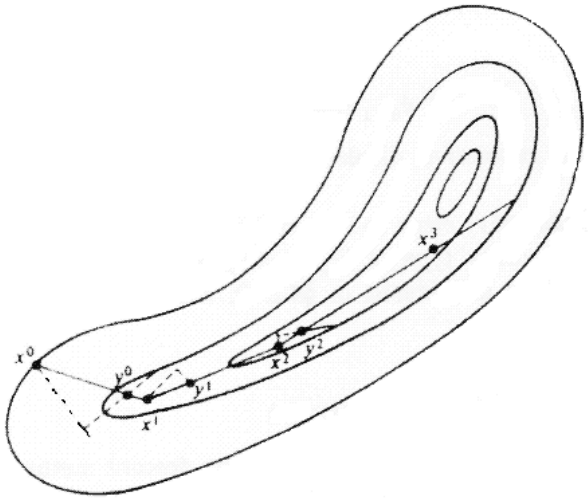


Figura 2.17: Algoritmo de Forsyte de dos pasos

Algoritmo 8 Algoritmo de Forsyte

Entrada: $f(x), x^{(0)}, m$

Dado una función diferenciable $f(x)$ y un valor inicial $x^{(0)}$
Hacemos $k \leftarrow 0$
mientras $\|\nabla f(x^{(k)})\| \geq \varepsilon$ **hacer**
Calcular los m estados haciendo
$x^{(k+1)} = \text{Descenso\_de\_Gradiente}(f(x),\ x^{(k)})$
$x^{(k+2)} = \text{Descenso\_de\_Gradiente}(f(x),\ x^{(k+1)})$
$\vdots$
$x^{(k+m)} = \text{Descenso\_de\_Gradiente}(f(x),\ x^{(k+m-1)})$
hacemos $y^{(k)} = x^{(k+m)}$
la nueva dirección de búsqueda es $d^{(k)} = (y^{(k)} - x^{(k)})/|y^{(k)} - x^{(k)}|$
Calcular el valor de $\alpha^{(k)}$ que minimiza $f(x^{(k)} + \alpha^{(k)} d^{(k)})$
Hacer la actualización $x^{(k+1)} = x^{(k)} + \alpha^{(k)} d^{(k)}$
$k = k + 1$
fin mientras
devolver $x^{(k)}$

- Hacemos la actualización $x^{(k+1)} = x^{(k)} + \alpha^{(k)} d^{(k)}$

El la figura 2.17, podemos ver el desempeño del algoritmo de Forsyte y Luenberger de dos pasos para la función mostrada en la figura 2.16. Note la mejora en el desempeño del algoritmo.

```
public Matriz Forsite(Matriz X)
{
    Matriz X1 = new Matriz(), X2 = new Matriz();
    Matriz Y = new Matriz();
    Matriz g = new Matriz(), p = new Matriz(), d = new Matriz();
    int i, m =2, k =0;
    double alpha;

    X1 = Matriz.igual_a(X);
    X2 = Matriz.igual_a(X);

    do
    {
        for (i = 0; i < m; i++) {
            g = Gradiente(X2);
            p = Direccion(g);
```

```
            //System.out.println("Direcc");
            //p.imprime();
            alpha = alpha_Newton(X2, p);
            X2 = X2.mas(p.por(alpha));
        }

        Y = X1.menos(X2);
        System.out.println("d = ");
        d = Direccion(Y);
        //X1.imprime();
        //X2.imprime();
        d.imprime();
        alpha = alpha_Newton(X1, d); //alpha_Direcccion(X2, Y);
        System.out.println("alpha = " + alpha);

        X1 = X1.mas(d.por(alpha));
        X2 = Matriz.igual_a(X1);
        System.out.println("X = ");
        X2.imprime();

        System.out.println("Iteracion " + k++ + ", f(x) = "+ funcion(X2));
    } while(Magnitud(Gradiente(X2)) > T);

    return X2;
}
```

2.9.5. Ejemplo 1

Implementar la mejora al método de descenso de gradiente propuesta por Forsythe y Luenberger. Los resultados de las iteraciones de búsqueda lineal con muestreo hacia atrás refinando con Razón Dorada y Forsythe y Luenberger (para $m = 2$), utilizando un valor de $\varepsilon = 1 \times 10^{-6}$.

La función a minimizar es $f(x) = x_1 e^{-x_1^2 - x_2^2}$, con un punto inicial $x_1^{(0)} = -0.5$, $x_2^{(0)} = -0.5$

Los resultados para la Búsqueda lineal con muestreo hacia atrás refinando con Razón Dorada se muestran en la tabla 2.4 y para el algoritmo mejorado de Forsythe y Luenberger en la tabla 2.5. Note que el algoritmo de Forsythe y Luenberger converge en dos iteraciones para considerando la misma precisión que el descenso de gradiente normal.

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$
1	-0.809017	-0.190982	-0.405384
2	-0.677837	-0.059803	-0.426608
3	-0.717333	-0.020307	-0.428615
4	-0.703751	-0.006725	-0.428852
5	-0.708231	-0.002246	-0.428878
6	-0.706732	-7.483517E-4	-0.428881
7	-0.707231	-2.495093E-4	-0.428881
8	-0.707065	-8.315781E-5	-0.428881
9	-0.707120	-2.772337E-5	-0.428881
10	-0.707102	-9.237318E-6	-0.42881
11	-0.707108	-3.081763E-6	-0.428881
12	-0.707106	-1.023345E-6	-0.428881
13	-0.707106	-3.402927E-7	-0.428881
14	-0.707106	-1.130393E-7	-0.428881

Cuadro 2.4: solución para la función $f(x) = x_1 e^{(-x_1^2 - x_2^2)}$ utilizando descenso de gradiente

k	$x_1^{(k)}$	$x_2^{(k)}$	$f(x^{(k)})$
1	-0.703257	0.0031191	-0.428865
2	-0.707106	-3.7149863E-7	-0.428881

Cuadro 2.5: solución para la función $f(x) = xe^{(-x^2 - y^2)}$ utilizando descenso de gradiente utilizando la mejora de Forsyte–Luenberger

Tarea: Algoritmo de Forsyte

Replicar la solución del ejemplo anterior (2.9.5)utilizando el algoritmo de descenso de gradiente 7 y de Forsyte 8.

2.9.6. Ejemplo 2

Para una función $f(x)$ dada como

$$f(x) = \sum_{i=1}^{10} \left[(x_i - y_i)^2 + 2.5(x_i - x_{i-1})^2\right]$$

considerando $y = [1, 2, 3, 4, 5, 4, 3, 2, 1, 0]^T$ y $x^{(0)} = y$

Determinar el número de iteraciones del método de Descenso de Gradiente y Forsite y graficar k vs $f(x^{(k)})$ para los diferentes valores calculados por ambos métodos.

El número de iteraciones para Descenso de Gradiente fue de 54 y para Forsyte de 14. En la Figura (2.18) se muestra la gráfica de la función $f(x^{(k)})$ para los dos métodos.

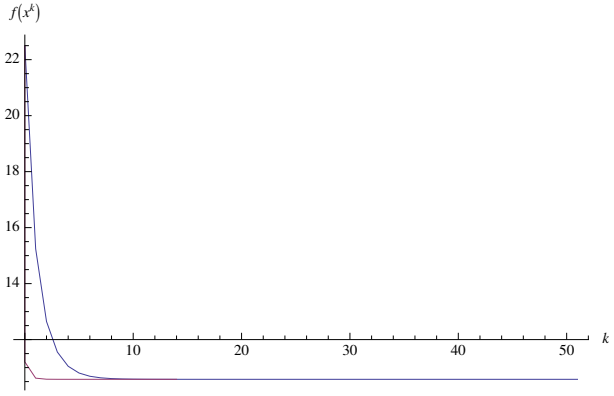


Figura 2.18: Comparativo del método de Descenso de Gradiente y Forsyte

2.10. Descenso de gradiente con búsqueda lineal basada en condiciones de Wolfe

Una especificación formal de la búsqueda lineal es la siguiente. Nos referiremos a la ecuación (2.18) como la condición de suficiente decrecimiento y a la ecuación (2.19) como la condición

de curvatura. El algoritmo de búsqueda lineal determina cual ancho de paso α^*, satisface las condición de Wolfe.

Algoritmo 9 Algoritmo de Búsqueda lineal basado en condiciones de Wolfe

Entrada: $f(x), x^{(k)}$
  Dado una función diferenciable $f(x)$ y un valor inicial x_0
  Hacer $\alpha_0 = 0$, seleccionar $\delta\alpha > 0$ y un α_{max}
  para $i = 1, \dots MAX$ **hacer**
    Evaluar $\phi(\alpha_i)$
    si $\phi(\alpha^{(i)}) > \phi(0) + c_1 * \alpha_i \phi'(0)$ o $\phi(\alpha^{(i)}) \geq \phi(\alpha^{(i-1)})$ **entonces**
      hacer $\alpha^* = RanzonDorada(\alpha_{i-1}, \alpha_i)$ y finalizar
    fin si
    Evalúa $\phi'(\alpha^{(i)})$
    si $\mid \phi'(\alpha^{(i)}) \mid > -c_2\phi'(0)$ **entonces**
      $\alpha^* = \alpha^{(i)}$ y termina
    fin si
    si $\phi'(\alpha_i) \geq 0$ **entonces**
      $\alpha^* = RazonDorada(\alpha^{(i-1)}, \alpha^{(i)})$ y finalizar
    fin si
    Seleccionar $\alpha^{(i+1)} = \alpha^{(i)} + \delta\alpha$
    hacer $i = i + 1$
  fin para
  devolver α^*

2.10.1. Ejemplo

Una comparación es necesaria, para evaluar las condiciones Wolfe. En este caso utilizaremos la función de Ronsebrock dada como $f(x) = 100 * (x_2 - x_1^2)^2 + (1 - x_1)^2$. Para que sea interesante veamos el comportamiento cuando queremos calcular un mínimo utilizando $f(x)$ directamente y un máximo con una función alterna $g(x) = -f(x)$.

En la figura 2.19 y la tabla 2.6 podemos ver el comportamiento de los algoritmos de máximo descenso utilizando tamaño de paso de máximo descenso y tamaño de paso con condiciones de Wolfe para un punto inicial $x_0 = [2, 3]^T$. Note que ambos algoritmos tienen un desempeño muy similar cuando minimizamos la función $f(x)$. En la tabla 2.6 se presenta en la primer columna el α de máximo descenso, el la segunda columna el valor de la función, el la tercer columna el α con condiciones de Wolfe y en la última columna el valor de la función.

En el caso de la figura 2.20 y la tabla 2.7 es muy diferente. Aquí calcular el tamaño de paso con máximo descenso lleva al algoritmo a un máximo mientras que calcular el tamaño de

Cuadro 2.6: Comparación de iteraciones entre tamaño de paso α de MD y Wolfe para minimizar $f(x)$

k	MD		Wolfe	
	α	$f(x)$	α	$f(x)$
0	0.0000	101.00	0.0000	101.0000
1	0.2187	2.6682	0.2582	0.5620
2	0.0383	0.5641	0.0750	0.5473
3	0.0011	0.5623	0.0032	0.5314
4	0.0094	0.5604	5.2786E-4	0.5313
5	0.0012	0.5584	5.2786E-4	0.5311
6	0.0106	0.5562	5.2786E-4	0.5310
7	0.0013	0.5540	5.2786E-4	0.5309
8	0.0123	0.5514	5.2786E-4	0.5308
9	0.0013	0.5489	5.2786E-4	0.5306

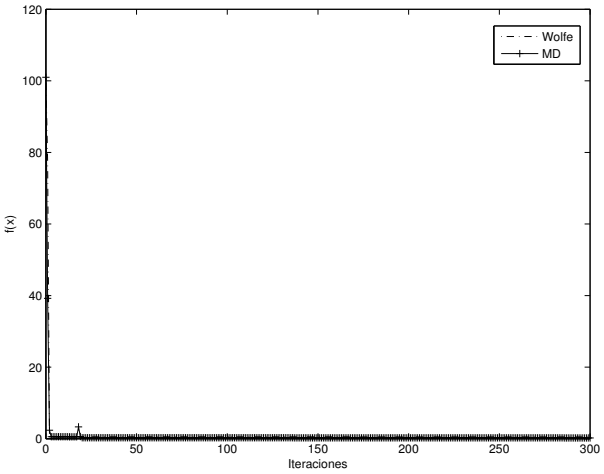


Figura 2.19: Comportamiento de Maximo descenso y Maximo descenso con condiciones de Wolfe

Cuadro 2.7: Comparación de iteraciones entre tamaño de paso utilizando α de MD y Wolfe para minimizar $-f(x)$

k	MD		Wolfe	
	α	$f(x)$	α	$f(x)$
0	0.0000	-101.0000	0.0000	-101.0000
1	-0.2187	-2.6682	0.0010	-101.8284
2	-0.0383	-0.5641	0.0010	-102.6606
3	-0.0011	-0.5623	0.0010	-103.4967
4	-0.0094	-0.5604	0.0010	-104.336
5	-0.0012	-0.5584	0.0010	-105.180
6	-0.0106	-0.5562	0.0010	-106.027
7	-0.0013	-0.5540	0.0010	-106.878
8	-0.0123	-0.5514	0.0010	-107.733
9	-0.0013	-0.5489	0.0010	-108.592

paso con condiciones de Wolfe va al mínimo.

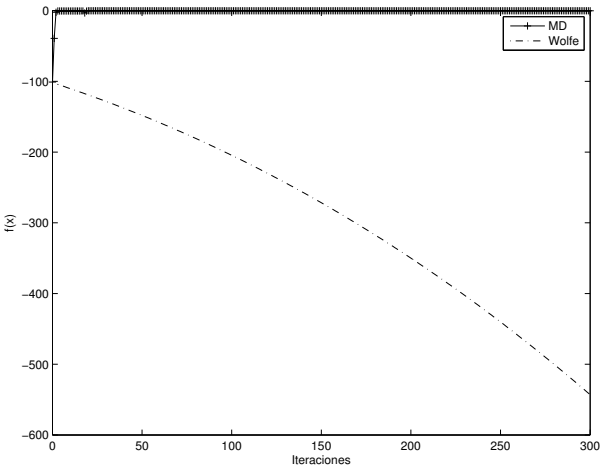


Figura 2.20: Comportamiento de Maximo descenso y Maximo descenso con condiciones de Wolfe

Método del Gradiente Conjugado

El método del gradiente conjugado es un método iterativo para solucionar un sistema lineal de ecuaciones de la forma $Ax = b$; donde A es una matriz definida positiva. Este problema es equivalente al siguiente problema de minimización

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

3.1. Método de direcciones conjugadas

Una de las propiedades más importantes del método de gradiente es su habilidad de generar direcciones conjugadas de una manera muy económica. Un conjunto de vectores $\{p^{(0)}, p^{(1)}, \dots p^{(l)}\}$ es llamado conjugado respecto a una matriz definida positiva A, si :

$$p^{(i)^T} Ap^{(j)} = 0 \qquad \forall i \neq j$$

La importancia de las direcciones conjugadas radica en el hecho de que podemos minimizar $f(x)$ en n pasos por la sucesiva minimización a lo largo de direcciones individuales en un conjunto conjugado. Para verificar esto, consideremos el siguiente método de direcciones conjugadas.

Dado un punto inicial $x^{(0)} \in \mathbb{R}^n$ y un conjunto de direcciones $\{p^{(0)}, p^{(1)}, \dots, p^{(n-1)}\}$. Consideraremos la siguiente secuencia generadora $\{x^{(k)}\}$

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$$

donde $\alpha^{(k)}$ es el minimizador de la función cuadrática $\phi(\cdot)$ a lo largo de $x^{(k)} + \alpha p^{(k)}$ dada

explícitamente por

$$\alpha^{(k)} = -\frac{{r^{(k)}}^T p^{(k)}}{{p^{(k)}}^T A p^{(k)}}$$

con $r^{(k)} = \nabla\phi(x^{(k)}) = Ax^{(k)} - b$

Teorema

Para cualquier $x^{(0)} \in \mathbb{R}^n$ la secuencia $\{x_k\}$ generada por el algoritmo de direcciones conjugadas converge a la solución x^* de un sistema lineal en no más de n pasos.

$$x^* - x^{(0)} = \sigma^{(0)}p^{(0)} + \sigma^{(1)}p^{(1)} + \ldots + \sigma^{(n-1)}p^{(n-1)}$$

para algún escalar $\sigma^{(k)}$. Pre–multiplicando la expresión por ${p^{(k)}}^T A$ y utilizando la propiedad del conjugado tenemos:

$$\sigma^{(k)} = \frac{{p^{(k)}}^T A(x^* - x^{(0)})}{{p^{(k)}}^T A p^{(k)}}$$

Ahora estableceremos el resultado mostrando que este coeficiente coincide con la longitud de paso α_k.

Si $x^{(k)}$ es generado por la sucesión $x^{(k+1)} = x^{(k)} + \sigma^{(k)}p^{(k)}$ tenemos:

$$x^{(k)} = x^{(0)} + \sigma^{(0)}p^{(0)} + \sigma^{(1)}p^{(1)} + \ldots + \sigma^{(k-1)}p^{(k-1)}$$

Pre multiplicando ésta expresión por ${p^{(k)}}^T A$ y utilizando la propiedad del conjugado tenemos:

$${p^{(k)}}^T A(x^{(k)} - x^{(0)}) = 0$$

Adicionalmente

$${p^{(k)}}^T A(x^* - x^{(k)}) = {p^{(k)}}^T A(\sigma^{(k)}p^{(k)} + \sigma^{(k+1)}p^{(k+1)} + \ldots + \sigma^{(n-1)}p^{(n-1)})$$

por la propiedad de conjugado

$${p^{(k)}}^T A(x^* - x^{(k)}) = \sigma^{(k)}{p^{(k)}}^T A p^{(k)}$$

tenemos que ${p^{(k)}}^T A(x^* - x^{(k)}) = {p^{(k)}}^T A(x^* - x^{(0)})$

y por lo tanto:

$$\frac{{p^{(k)}}^T A(x^* - x^{(0)})}{{p^{(k)}}^T A p^{(k)}} = \frac{{p^{(k)}}^T A(x^* - x^{(k)})}{{p^{(k)}}^T A p^{(k)}}$$

$${p^{(k)}}^T A(x^* - x^{(0)}) = {p^{(k)}}^T A(x^* - x^{(k)}) = {p^{(k)}}^T (b - Ax^{(k)}) = -{p^{(k)}}^T r^{(k)}$$

con lo que $\sigma^{(k)}$ queda:

$$\sigma^{(k)} = -\frac{{p^{(k)}}^T r^{(k)}}{{p^{(k)}}^T A p^{(k)}}$$

comparando ambas relaciones tenemos que $\alpha^{(k)} = \sigma^{(k)}$.

Hay una interpretación simple de las direcciones conjugadas. Si la matriz A es diagonal los contornos de la función $f(\cdot)$ son elipses cuyos ejes están alineados con los ejes coordenados. Podemos encontrar el mínimo de ésta función con una búsqueda lineal a lo largo de los ejes coordenados.

Cuando A no es una diagonal, sus contornos son elípticos, pero ellos no están alineados en la dirección de los ejes coordenados. La estrategia de minimizaciones sucesivas a lo largo de estas direcciones no da una solución en n iteraciones.

En la figura 3.21, podemos ver como se ejemplifica el uso de las direcciones conjugadas.

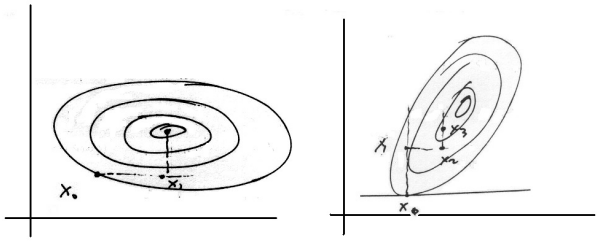


Figura 3.21: Ejemplo de como trabajan las direcciones conjugadas

3.2. El método del gradiente conjugado para funciones cuadráticas

Asumiremos que minimizaremos una función cuadrática de la forma:

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c \tag{3.21}$$

La idea del método es construir progresivamente direcciones $p^{(0)}, p^{(1)}, \ldots, p^{(K)}$, las cuales son conjugadas respecto a la matriz A de la forma cuadrática. En cada estado K la dirección $p^{(K)}$ es obtenida por la combinación lineal del gradiente $-\nabla q(x^{(K)})$ en cada posición $x^{(K)}$ y las direcciones previas $p^{(0)}, p^{(1)}, \ldots, p^{(K-1)}$.

Llamemos $g^{(k)} = \nabla q(x^{(k)})$ el gradiente de la función q en $x^{(k)}$, El método toma la siguiente forma que se muestra en

10.

Algoritmo 10 Algoritmo de Gradiente Conjugado preliminar

Entrada: Dado una función diferenciable $f(x)$ y un valor inicial $x^{(0)}$
Salida: $x^{(k)}$
  Calcular el gradiente $r^{(0)} = \nabla f(x^{(0)}) = Ax^{(0)} - b$
  Hacer hacer $p^{(0)} = -r^{(0)}$ y poner $k = 0$.
  mientras $\|\nabla f(x^{(k)})\| \geq \varepsilon$ **hacer**
    Calcular $\alpha^{(k)} = -\frac{{r^{(k)}}^T p^{(k)}}{{p^{(k)}}^T A p^{(k)}}$
    Hacer la actualización $x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$
    Calcular el valor del gradiente $r^{(k+1)} = Ax^{(k+1)} - b$
    determinar $\beta^{(k+1)} = \frac{{r^{(k+1)}}^T A p^{(k)}}{{p^{(k)}}^T A p^{(k)}}$
    Calcular la nueva dirección con $p^{(k+1)} = -r^{(k+1)} + \beta^{(k+1)} p^{(k)}$
    Actualizar $k \leftarrow k + 1$
  fin mientras
  devolver $x^{(k)}$

El algoritmo de gradiente conjugado, requiere de la siguiente sucesión para calcular las direcciones conjugadas $p^{(k+1)} = -r^{(k+1)} + \beta^{(k+1)} p^{(k)}$ con un parámetro $\beta^{(k+1)} = \frac{{r^{(k+1)}}^T A p^{(k)}}{{p^{(k)}}^T A p^{(k)}}$, pero debemos mostrar que estas expresiones crean direcciones conjugadas.

Prueba

Por definición tenemos que las direcciones conjugadas cumplen con (3.22):

$$p^{(k+1)^T} A p^{(k)} = 0 \tag{3.22}$$

y tenemos que:

$$p^{(k+1)} = -r^{(k+1)} + \beta^{(k)} p^{(k)} \tag{3.23}$$

Sustituyendo (3.23) en (3.22):

$$(-r^{(k+1)} + \beta^{(k+1)} p^{(k)})^T A p^{(k)} = 0$$

$$(-r^{(k+1)^T} + \beta^{(k+1)} p^{(k)^T}) A p^{(k)} = 0$$

$$-r^{(k+1)^T} A p^{(k)} + \beta^{(k+1)} p^{(k)^T} A p^{(k)} = 0$$

$$\beta^{(k+1)} = \frac{r^{(k+1)^T} A p^{(k)}}{p^{(k)^T} A p^{(k)}}$$

Con lo cual se muestra que la sucesión si produce direcciones conjugadas.

Una expresión equivalente para el ancho de paso es:

$$\alpha^{(k)} = \frac{r^{(k)^T} r^{(k)}}{p^{(k)^T} A p^{(k)}}$$

mostrar que es equivalente a:

$$\alpha^{(k)} = -\frac{r^{(k)^T} p^{(k)}}{p^{(k)^T} A p^{(k)}}$$

Prueba:

Tenemos que $p^{(k)} = -r^{(k)} + \beta^{(k)} p^{(k-1)}$ podemos escribir:

$$\alpha^{(k)} = -\frac{r^{(k)^T} p^{(k)}}{p^{(k)^T} A p^{(k)}}$$

$$\alpha^{(k)} = -\frac{{r^{(k)}}^T \left(-r^{(k)} + \beta^{(k)} p^{(k-1)}\right)}{{p^{(k)}}^T A p^{(k)}}$$

$$\alpha^{(k)} = \frac{{r^{(k)}}^T r^{(k)}}{{p^{(k)}}^T A p^{(k)}} - \beta^{(k)} \frac{{r^{(k)}}^T p^{(k-1)}}{{p^{(k)}}^T A p^{(k)}}$$

Dado que $(p^{(0)}, p^{(0)}, \cdots, p^{(k-1)})$ son mutuamente conjugados $x^{(k)}$ es el mínimo de $f(x^{(k)})$ en la dirección $p^{(k-1)}$ así que:

$${r^{(k)}}^T p^{(k-1)} = 0$$

Con lo cual

$$\alpha^{(k)} = \frac{{r^{(k)}}^T r^{(k)}}{{p^{(k)}}^T A p^{(k)}}$$

Una expresión equivalente para $\beta^{(k+1)}$ es:

$$\beta^{(k+1)} = \frac{{r^{(k+1)}}^T r^{(k+1)}}{{r^{(k)}}^T r^{(k)}} = \frac{{r^{(k+1)}}^T A p^{(k)}}{{p^{(k)}}^T A p^{(k)}}$$

Recordemos que

$$r^{(k)} = A x^{(k)} - b$$

Note que:

$$r^{(k+1)} - r^{(k)} = A\left(x^{(k+1)} - x^{(k)}\right) = \alpha^{(k)} A p^{(k)} \tag{3.24}$$

Si multiplicamos por ${r^{(k+1)}}^T$ la ecuación (3.24) y despejamos, tenemos:

$${r^{(k+1)}}^T A p^{(k)} = \frac{1}{\alpha^{(k)}} {r^{(k+1)}}^T \left(r^{(k+1)} - r^{(k)}\right)$$

Sustituyendo el valor de $\alpha^{(k)}$

$${r^{(k+1)}}^T A p^{(k)} = \left(\frac{{p^{(k)}}^T A p^{(k)}}{{r^{(k)}}^T r^{(k)}}\right) {r_{k+1}}^T \left(r_{k+1} - r_k\right)$$

reordenando términos tenemos

$$\frac{{r^{(k+1)}}^T A p^{(k)}}{{p^{(k)}}^T A p^{(k)}} = \frac{{r^{(k+1)}}^T \left[r^{(k+1)} - r^{(k)}\right]}{{r^{(k)}}^T r^{(k)}}$$

Lo cual finalmente demuestra que

$$\beta^{(k+1)} = \frac{{r^{(k+1)}}^T r^{(k+1)}}{{r^{(k)}}^T r^{(k)}}$$

3.2.1. Algoritmo de GC

El algoritmo 11 presenta los detalles para la implementación del Método de Gradiente Conjugado y a continuación se da la implementación en Java

Algoritmo 11 Algoritmo de Gradiente Conjugado

Entrada: Dado una función diferenciable $f(x)$ y un valor inicial $x^{(0)}$
Salida: $x^{(k)}$
Calcular el gradiente $r^{(0)} = \nabla f(x^{(0)}) = Ax^{(0)} - b$
Hacer hacer $p^{(0)} = -r^{(0)}$ y poner $k = 0$.
mientras $\|r^{(k)}\| \geq \varepsilon$ **hacer**
Calcular $\alpha^{(k)} = \frac{{r^{(k)}}^T r^{(k)}}{{p^{(k)}}^T A p^{(k)}}$
Hacer la actualización $x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$
Calcular el valor del gradiente $r^{(k+1)} = r^{(k)} + \alpha^{(k)} A p^{(k)}$
determinar $\beta^{(k+1)} = \frac{{r^{(k+1)}}^T r^{(k+1)}}{{r^{(k)}}^T r^{(k)}}$
Calcular la nueva dirección con $p^{(k+1)} = -r^{(k+1)} + \beta^{(k+1)} p^{(k)}$
Actualizar $k \leftarrow k + 1$
fin mientras
devolver $x^{(k)}$

```
public Matriz GC(Matriz x)
{
    double alpha, beta, rr0, rr1;
    int Nmax = x.nren + 1;
    Matriz A = Hessiano(x);

    Matriz rr = new Matriz(1, 1);
    Matriz pAp = new Matriz(1, 1);

    Matriz Ap = new Matriz(A.nren, 1);
    Matriz r = new Matriz(A.nren, 1);
    Matriz p = new Matriz(A.nren, 1);
    Matriz aux1 = new Matriz(1, A.nren);
    Matriz aux2 = new Matriz(A.nren, 1);
```

```
    Ap = A.por(x);
    r = Gradiente(x); //Ap.menos(b);
    rr = (r.T()).por(r);
    rr0 = rr.obten(0, 0);
    p = r.por( -1);

    for (int k = 0; k < Nmax; k++) {
        Ap = A.por(p);
        pAp = (p.T()).por(Ap);
        alpha = rr0 / pAp.obten(0, 0);
        x = x.mas(p.por(alpha));
        r = r.mas(Ap.por(alpha));
        rr = (r.T()).por(r);
        beta = rr.obten(0, 0) / rr0;
        p = (p.por(beta)).menos(r);
        rr0 = rr.obten(0, 0);
    }
    return x;
}
```

3.2.2. Ejemplo 1

Muestre que la siguiente sistema de ecuaciones tiene una matriz definida positiva. Determine la solución del sistema de ecuaciones, utilizando gradiente conjugado.

$$\begin{pmatrix} 3 & -1 & -2 \\ -1 & 4 & -3 \\ -2 & -3 & 6 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 3 \end{pmatrix}$$

Para determinar si se trata de una matriz definida positiva, calculamos su determinante

$$\det \begin{pmatrix} 3 & -1 & -2 \\ -1 & 4 & -3 \\ -2 & -3 & 6 \end{pmatrix} = 11$$

$\therefore$ tenemos una matriz definida positiva y se puede aplicar el método de Gradiente conjugado. La solución aplicando el proceso iterativo es:

Iteración 1

$r^{(0)} = [-1, 0, -3]^T$

$p^{(0)} = [1, 0, 3]^T$

${r^{(0)}}^T r^{(0)} = 10$

$Ap^{(0)} = [-3, -10, 16]^T$

$\alpha^{(0)} = 0.2222$

$x^{(1)} = [0.2222, 0, 0.6667]^T$

$r^{(1)} = [-1.6667, -2.2222, 0.5556]^T$

${r^{(1)}}^T r^{(1)} = 8.0247$

$\beta^{(1)} = 8.0247/10 = 0.80247$

$p^{(1)} = [2.4691, 2.2222, 1.8519]^T$

Iteración 2

$Ap^{(1)} = [1.4815, 0.8642, -0.4938]^T$

$\alpha^{(2)} = 1.7206$

$x^{(2)} = [4.4706, 3.8235, 3.8529]^T$

$r^{(2)} = [0.8824, -0.7353, -0.2944]^T$

${r^{(2)}}^T r^{(2)} = 1.4057$

$\beta^{(2)} = 1.4057/10 = 0.1752$

$p^{(2)} = [-0.4498, 1.1246, 0.6185]^T$

Iteración 3

$Ap^{(2)} = [-3.7111, 3.0926, 1.2370]^T$

$\alpha^{(3)} = 0.2378$

$x^{(3)} = [4.3636, 4.0909, 4]^T$

$r^{(3)} = [0.0111, 0.0555, -0.1443] \times 10^{-14}$

${r^{(3)}}^T r^{(3)} = 2.4036 \times 10^{-30}$

$\beta^{(3)} = 1.7099 \times 10^{-30}$

$p^{(3)} = [0.0111, 0.0555, -0.1443] \times 10^{-14}$

Finalmente la solución es:

$x = [4.3636, 4.0909, 4]$

3.2.3. Ejemplo 2

Para una función $f(x) : \mathbb{R}^{10} \to \mathbb{R}$ dada como

$$f(x) = \sum_{i=1}^{10} \left[(x_i - y_i)^2 + 2.5(x_i - x_{i-1})^2\right]$$

donde $x = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}]^T$ con $y = [1, 2, 3, 4, 5, 4, 3, 2, 1, 0]^T$ y $x^{(0)} = y$. Considere que el término $2.5(x_i - x_{i-1})^2$ existe si $i - 1 \geq 0$.

Calcular el mínimo de la función y el valor de $f(x^{(k)})$ en cada una de las iteraciones.

El mínimo del sistema es

$$x = [1.9725, 2.3615, 2.8951, 3.3868, 3.6332, 3.3329, 2.7657, 2.1049, 1.4859, 1.0614]^T$$

y la siguiente tabla muestra las 7 iteraciones necesarias para llegar al óptimo

k	$f(x^{(k)})$
0	22.5000
1	15.2419
2	11.2041
3	10.6239
4	10.5910
5	10.5838
6	10.5821
7	10.5813

Para detalles ver **Gradiente_filtro.java**

3.2.4. Ejemplo 3

Para una función $f(x) : \mathbb{R}^n \to \mathbb{R}$ dada como

$$f(x) = \sum_{i=1}^{n} \left[(x_i - y_i)^2 + 2.5(x_i - x_{i-1})^2\right]$$

donde $x = [x_1, x_2, x_3, \cdots, x_n]^T$ y $x_k^{(0)} = y_k$.

El gradiente y el Hessiano pueden calcularse de manera directa haciendo:

$$f(x) = (x_1 - y_1)^2 + (x_2 - y_2)^2 + 2.5(x_2 - x_1)^2 + (x_3 - y_3)^2 + 2.5(x_3 - x_2)^2 + \cdots$$

Calculamos la derivada $\partial f(x)/\partial x_1$

$$\frac{\partial f(x)}{\partial x_1} = 2(x_1 - y_1) - 5(x_2 - x_1)$$

Calculamos la derivada $\partial f(x)/\partial x_2$

$$\frac{\partial f(x)}{\partial x_2} = 2(x_2 - y_2) + 5(x_2 - x_1) - 5(x_3 - x_2)$$

sustituyendo k por 2 tenemos el k-ésimo término del Gradiente

$$\frac{\partial f(x)}{\partial x_k} = 2(x_k - y_k) + 5(x_k - x_{k-1}) - 5(x_{k+1} - x_k)$$

donde los términos $(x_k - x_{k-1})$ y $(x_{k+1} - x_k)$ existen si $k - 1 \geq 0$ y $k + 1 \leq n$ respectivamente.

Para determinar el vector b hacemos

$$\frac{\partial f(x)}{\partial x_k} = 2(x_k - y_k) + 5(x_k - x_{k-1}) - 5(x_{k+1} - x_k) = 0$$

reordenando téminos

$$12x_k - 5x_{k-1} - 5x_{k+1} - 2y_k = 0$$
$$12x_k - 5x_{k-1} - 5x_{k+1} = 2y_k$$

de donde tenemos que el k-ésimo término $b_k = 2y_k$

El Hessiano tendrá por renglón tres términos correspondientes a las variables x_k, x_{k-1} y x_{k+1} por lo que para el k-ésimo renglón solo existen tres términos

$$\frac{\partial^2 f(x)}{\partial^2 x_k} = \frac{\partial(12x_k - 5x_{k-1} - 5x_{k+1} - 2y_k)}{\partial x_k} = \begin{cases} 12 & \text{Si } 1 < k < n \\ 7 & \text{de lo contrario} \end{cases}$$

$$\frac{\partial^2 f(x)}{\partial x_k \partial x_{k-1}} = \frac{\partial(12x_k - 5x_{k-1} - 5x_{k+1} - 2y_k)}{\partial x_{k+1}} = \begin{cases} -5 & \text{Si } 1 < k \\ 0 & \text{de lo contrario} \end{cases}$$

$$\frac{\partial^2 f(x)}{\partial x_k \partial x_{k+1}} = \frac{\partial(12x_k - 5x_{k-1} - 5x_{k+1} - 2y_k)}{\partial x_{k+1}} = \begin{cases} -5 & \text{Si } k < n \\ 0 & \text{de lo contrario} \end{cases}$$

siempre que $k - 1 \geq 0$ y $k + 1 \leq n$.

3.3. Precondicionamiento

Partiremos de de función $f(x) = \frac{1}{2}x^T Ax - b^T x + c$ la cual remplazaremos por una nueva función $\hat{f}(x)$ la cual esta multiplicada en algunos términos por la matriz M^{-1}. Esta condición no altera el mínimo de la función.

$$\hat{f}(x) = \frac{1}{2}x^T M^{-1} Ax - M^{-1} b^T x + c$$

El gradiente de la función $\hat{f}(x)$ esta dada por

$$\begin{aligned} \nabla \hat{f}(x) &= M^{-1}(Ax - b) \\ y &= M^{-1} r \\ My &= r \end{aligned}$$

y el mínimo de esta función es exactamente el mismo que la función $f(x)$ en virtud de que

$$M^{-1}(Ax - b) = 0$$

y es equivalente a

$$(Ax - b) = 0$$

La manera de calcular el $y^{(k)} = \nabla \hat{f}(x^{(k)})$ será simplemente resolviendo el sistema de ecuaciones $Mr^{(k)} = y^{(k)}$ donde $r^{(k)} = \nabla f(x^{(k)}) = Ax^{(k)} - b$

Note que si $M \equiv A$ entonces la solución para x es:

$$\begin{aligned} \nabla \hat{f}(x) &= M^{-1}(Ax-b) = 0 \\ \nabla \hat{f}(x) &= A^{-1}(Ax-b) = 0 \end{aligned}$$

$$\begin{aligned} A^{-1}(Ax-b) &= 0 \\ A^{-1}Ax &= A^{-1}b \end{aligned}$$

$$x = A^{-1}b$$

y si $M \approx A$ el algoritmo de Gradiente Conjugada Precondicionado deberá hacer muchas menos iteraciones que el algoritmo de Gradiente Conjugado.

Cálculo de las direcciones:

Para hacer las actualizaciones de las direcciones haremos una combinación lineal entre el residuo precondicionado $y^{(k+1)}$ y la dirección anterior $p^{(k)}$, así la nueva dirección la calcularemos como

$$p^{(k+1)} = -y^{(k+1)} + \beta^{(k+1)} p^{(k)} \tag{3.25}$$

Tamaño de Paso:

El tamaño de paso para el método de Gradiente Conjugado Precondicionado para una función cuadrática $\hat{\phi}(\alpha) = M^{-1}(\frac{1}{2}(x^{(k)} + \alpha^{(k)} p^{(k)})^T A(x^{(k)} + \alpha^{(k)} p^{(k)}) - b^T(x^{(k)} + \alpha^{(k)} p^{(k)}))$ es

$$\alpha^{(k)} = -\frac{{r^{(k)}}^T p^{(k)}}{{p^{(k)}}^T A p^{(k)}}$$

sustituimos el valor de (3.25) en el tamaño de paso $\alpha^{(k)}$ obtenemos

$$\alpha^{(k)} = -\frac{{r^{(k)}}^T(-y^{(k)} + \beta^{(k)} p^{(k+1)})}{{p^{(k)}}^T A p^{(k)}}$$

$$\alpha^{(k)} = \frac{{r^{(k)}}^T y^{(k)}}{{p^{(k)}}^T A p^{(k)}} - \beta^{(k)} \frac{{r^{(k)}}^T p^{(k-1)}}{{p^{(k)}}^T A p^{(k)}}$$

Dado que ${r^{(k)}}^T p^{(k-1)} = 0$ tenemos que el tamaño de paso es:

$$\alpha^{(k)} = \frac{{r^{(k)}}^T y^{(k)}}{{p^{(k)}}^T A p^{(k)}}$$

Cálculo del gradiente:

Nuestro nuevo vector de gradiente precondicionado $y^{(k)}$ esta definido como $y^{(k)} = M^{-1} r^{(k)}$, para el calculo de $r^{(k)}$ hacemos

$$r^{(k+1)} = Ax^{(k+1)} - b = A(x^{(k)} + \alpha^{(k)} p^{(k)}) - b = (Ax^{(k)} - b) + \alpha^{(k)} A p^{(k)}$$

finalmente tenemos que

$$r^{(k+1)} = r^{(k)} + \alpha^{(k)} A p^{(k)}$$

$$M y^{(k+1)} = r^{(k)} + \alpha^{(k)} A p^{(k)}$$

Cálculo del tamaño de paso para las direcciones:

A partir de la ecuación (3.25) y asumiendo que las direcciones están conjugadas podemos escribir

$$(-y^{(k+1)} + \beta^{(k+1)} p^{(k)})^T A p^{(k)} = 0$$
$$-{y^{(k+1)}}^T A p^{(k)} + \beta^{(k+1)} {p^{(k)}}^T A p^{(k)} = 0$$

despejando tenemos

$$\beta^{(k+1)} = \frac{{y^{(k+1)}}^T A p^{(k)}}{{p^{(k)}}^T A p^{(k)}}$$

A partir de la ecuación $r^{(k+1)} - r^{(k)} = \alpha^{(k)} A p^{(k)}$, si multiplicamos por ${y^{(k+1)}}^T$ y despejamos, tenemos:

$${y^{(k+1)}}^T A p^{(k)} = \frac{1}{\alpha^{(k)}} {y^{(k+1)}}^T \left(r^{(k+1)} - r^{(k)} \right)$$

Sustituyendo el valor de $\alpha^{(k)}$

$${y^{(k+1)}}^T A p^{(k)} = \left(\frac{{p^{(k)}}^T A p^{(k)}}{{r^{(k)}}^T y^{(k)}} \right) {y^{(k+1)}}^T \left(r^{(k+1)} - r^{(k)} \right)$$

reordenando términos tenemos

$$\frac{y^{(k+1)^T} A p^{(k)}}{p^{(k)^T} A p^{(k)}} = \frac{y^{(k+1)^T} \left[r^{(k+1)} - r^{(k)} \right]}{r^{(k)^T} y^{(k)}} = \frac{r^{(k+1)^T} y^{(k+1)}}{r^{(k)^T} y^{(k)}} - \frac{y^{(k+1)^T} r^{(k)}}{r^{(k)^T} y^{(k)}}$$

y dado que $y^{(k)^T} r^{(k)} = M^{-T} r^{(k+1)^T} r^{(k)} = 0$ finalmente tenemos

$$\beta^{(k+1)} = \frac{r^{(k+1)^T} y^{(k+1)}}{r^{(k)^T} y^{(k)}}$$

El método de gradiente conjugado utilizando precondicionado se presenta en el algoritmo 12

Algoritmo 12 Algoritmo de Gradiente Conjugado Precondicionado

Dado una función diferenciable $f(x)$, un valor inicial $x^{(0)}$
y una matriz de precondicionamiento M
Calcular el gradiente $r^{(0)} = \nabla f(x^{(0)}) = Ax^{(0)} - b$
Resolver $My^{(0)} = r^{(0)}$ para $y^{(0)}$
Hacer hacer $p^{(0)} = -y^{(0)}$ y $k = 0$.
mientras $\|r^{(k)}\| \geq \varepsilon$ **hacer**
  Calcular $\alpha^{(k)} = \frac{r^{(k)^T} y^{(k)}}{p^{(k)^T} A p^{(k)}}$
  Hacer la actualización $x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$
  Calcular el valor del gradiente $r^{(k+1)} = r^{(k)} + \alpha^{(k)} A p^{(k)}$
  Resolver para $y^{(k+1)}$, el sistema de ecuaciones $My^{(k+1)} = r^{(k+1)}$
  determinar $\beta^{(k+1)} = \frac{r^{(k+1)^T} y^{(k+1)}}{r^{(k)^T} y^{(k)}}$
  Calcular la nueva dirección con $p^{(k+1)} = -y^{(k+1)} + \beta^{(k+1)} p^{(k)}$
  Actualizar $k \leftarrow k + 1$
fin mientras
devolver $x^{(k)}$

Observe que en el caso del que $M = I$ el algoritmo 12 es equivalente al método del Gradiente Conjugado. Por esta razón es importante calcular una matriz C que acerque a la solución final, reduciendo las iteraciones y/o tiempo necesario para que el algoritmo de Gradiente conjugado converga

3.3.1. Precondicionadores Prácticos

El algoritmo de Cholesky, se basa en el hecho de que una matriz simétrica y definida positiva se puede factoriza en dos matrices L tal que

$$A = LL^T$$

donde los factores triangulares de A son L y L^T.

La formula para calcular la descomposición de Cholesky en el caso de los elementos fuera de la diagonal es:

$$l_{k,i} = \frac{a_{k,i} - \sum_{j=1}^{i-1} l_{i,j} l_{k,j}}{l_{i,i}} \tag{3.26}$$

$$\forall k = 0, 1, 2, \cdots, N-1 \tag{3.27}$$
$$\forall i = 0, 1, 2, \cdots, k-1$$

y para los elementos en la diagonal

$$l_{k,k} = \sqrt{a_{k,k} - \sum_{j=1}^{k-1} l_{k,j}^2} \tag{3.28}$$

El código Java para realizar esta implementación se muestra a continuación

```
public Matriz Cholesky3() {
  int i, j, k, n, s;
  double fact, suma = 0;

  if (this.nren != this.ncol || this == null) {
    System.out.println("Matriz mal condicionada");
    return null;
  }
  n = this.nren;
  for (k = 0; k < n; k++) {
    for (i = 0; i <= k - 1; i++) {
      suma = 0;
      for (j = 0; j <= i - 1; j++)
        suma += this.datos[i][j] * this.datos[k][j];

      this.datos[k][i] = (this.datos[k][i] - suma) / this.datos[i][i];
    }
    suma = 0;
    for (j = 0; j <= k - 1; j++)
```

```
        suma += this.datos[k][j] * this.datos[k][j];
      this.datos[k][k] = Math.sqrt(this.datos[k][k] - suma);
    }
    return this;
  }
```

Choleski incompleto es probablemente la estrategia de precondicionamiento mas efectiva. La idea básica es muy simple. Seguiremos el procedimiento de Choleski, pero en lugar de calcular el valor exacto de L que satisface $A = LL^T$, calcularemos un valor de $\hat{L}$ que es tan dispersa como A. Tendremos entonces $A \approx \hat{L}\hat{L}^T$ y seleccionando $C = \hat{L}^T$, obtendremos $A = \hat{L}\hat{L}^T$ y

$$C^T A C^{-1} = \hat{L}^{-1} A \hat{L}^{-T} \approx I$$

En el caso de el Cholesky incompleto aplicaremos las formulas (3.27) y (3.28) solo en el caso que el $a_{i,j} \neq 0$. El código Java para esta implementación es:

```
public static void CholeskyI(double A[][]) {
    int i, j, k, n, s;
    double fact, suma = 0;
    n = A.length;

    for (i = 0; i < n; i++) {
        for (j = 0; j <= i - 1; j++) {
            if (A[i][j] != 0) {
                System.out.print("A"+i+"," + j +"= ");
                suma = 0;
                for (k = 0; k <= j - 1; k++) {
                    suma += A[i][k] * A[j][k];
                    if(A[i][k]*A[j][k] != 0)
                    System.out.print ("A"+i+"," + k + " * A" +j +","+k  + " ");
                }
                A[i][j] = (A[i][j] - suma) / A[j][j];
                System.out.println(" ");
            }
        }
        suma = 0;
        for (k = 0; k <= i - 1; k++) {
            suma += A[i][k] * A[i][k];
            if(A[i][k]*A[i][k] != 0) System.out.println("si");
        }
        A[i][i] = Math.sqrt(A[i][i] - suma);
    }
```

```
    for (j = 0; j < n; j++) {
        for (k = 1 + j; k < n; k++)
            A[j][k] = 0;
    }
}
```

La solución del sistema factorizado $LL^T x = b$ lo realizaremos en dos pasos, para ello hacemos $L^T x = y$. El el primer paso resolvemos $Ly = b$ para y y en el segundo paso resolvemos el sistema $Lx = y$ para x. La solución de estos sistemas la haremos utilizando Sustitución hacia adelante (3.29) y sustitución hacia atrás (3.30)

$$y_i = \frac{b_i - \sum_{k=0}^{k=i-1} l_{i,k} y_i}{l_{i,i}} \qquad i = 0, 1, \cdots, N-1 \tag{3.29}$$

$$x_i = \frac{y_i - \sum_{k=0}^{k=i-1} l_{i,k} y_i}{l_{i,i}} \qquad i = N-1, \cdots, 1, 0 \tag{3.30}$$

3.3.2. Ejemplo

Dado el sistema de de ecuaciones $Ax = b$ calcular la solución utilizando

$$\begin{bmatrix} 3 & -1 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 3 & 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 & 3 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 2 \\ 1 \\ 2 \\ 3 \\ 2 \end{bmatrix}$$

a) Utilizando la factorizacón de Choleky

La matriz L factorizada queda como

$$L = \begin{bmatrix} 1.7321 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ -0.5774 & 1.9149 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & -0.5222 & 1.9306 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & -0.5180 & 1.6528 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ -0.5774 & -0.1741 & -0.0471 & -0.0148 & 1.6229 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & -0.5222 & -0.1413 & -0.0443 & -0.6767 & 1.8021 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & -0.5180 & -0.1623 & -0.0165 & -0.6057 & 1.8271 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & -0.6050 & -0.0055 & -0.0169 & -0.6067 & 1.5052 \end{bmatrix}$$

Aplicando sustitución hacia atras solucionamos el sistema $Ly = b$, para y

$$y = [0.5774, 1.2185, 1.8835, 1.8004, 1.0233, 2.0391, 3.0211, 3.297]^T$$

y finalmente resolvemos el sistema $L^T x = y$ para x

$$x = [1.4762, 1.9524, 2.381, 2.1905, 1.4762, 1.9524, 2.381, 2.1905]^T$$

b) La solución utilizando Cholesky incompleto.

Primero calculamos la factorización incompleta de Cholesky dada como

$$\widehat{L} = \begin{bmatrix} 1.7321 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.5774 & 1.9149 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & -0.5222 & 1.9306 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & -0.518 & 1.6528 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.5774 & 0.0 & 0.0 & 0.0 & 1.633 & 0.0 & 0.0 & 0.0 \\ 0.0 & -0.5222 & 0.0 & 0.0 & -0.6124 & 1.8309 & 0.0 & 0.0 \\ 0.0 & 0.0 & -0.518 & 0.0 & 0.0 & -0.5462 & 1.8529 & 0.0 \\ 0.0 & 0.0 & 0.0 & -0.605 & 0.0 & 0.0 & -0.5397 & 1.5306 \end{bmatrix}$$

Aplicando sustitución hacia atras solucionamos el sistema $\widehat{L}y = b$ para y

$$y = [0.5774, 1.2185, 1.8835, 1.8004, 0.8165, 1.7130, 2.6505, 2.9529]^T$$

y finalmente resolvemos el sistema $\widehat{L}^T x = y$ para x

$$x = [1.2235, 1.5969, 1.9919, 1.7955, 1.0737, 1.5299, 1.9923, 1.9293]^T$$

Note que la solución es próxima a la solución del inciso a).

c) Calcular la solución utilizando el método de Gradiente Conjugado con un vector inicial $X^{(0)} = [10, 0, 0, 0, 0, 0, 0, 0]^T$. La solución en 7 iteraciones utilizando el método de GC fue

$$X_{GC} = [1.4762, 1.9524, 2.3810, 2.1905, 1.4762, 1.9524, 2.3810, 2.1905]^T$$

d) Calcular la solución utilizando el método de Gradiente Conjugado precondicionado con un vector inicial $X^{(0)} = [10, 0, 0, 0, 0, 0, 0, 0]^T$.

Utilizando GC precondicionado con Cholesky incompleto, en 4 iteraciones fue:

$$X_{GC-Pre} = [1.4762, 1.9524, 2.3810, 2.1905, 1.4762, 1.9524, 2.3810, 2.1905]^T$$

e) Haga una comparación entre el desempeño del GC y el GC precondicionado.

En la tabla 3.8 y en la figura 3.23 podemos ver un comparativo del desempeño de estos dos algoritmos para los datos del problema. Note como el método de Gradiente conjugado precondicionado alcanza el mínimo en menos iteraciones.

Cuadro 3.8: Comparación entre el gradiente Conjugado y el Gradiente Conjugado Precondicionado

iteracion	G Conj	G Conj. Pre
0	140	140
1	2.1854	-16.3758
2	-12.8598	-16.9042
3	-16.0625	-16.9048
4	-16.8210	-16.9048
5	-16.8815	-16.9048
6	-16.9027	-16.9048
7	-16.9048	-16.9048
8	-16.9048	-16.9048

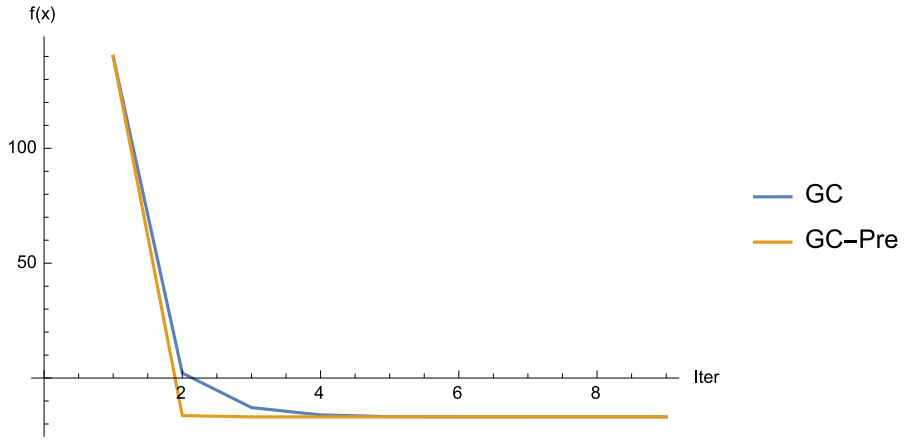


Figura 3.22: Comparación entre el método de GC y GC precondicionado.

ver Gradiente_ej04.java

3.4. Manejo de Dispersidad

En algunas ocaciones el minimizar una función, dado el número de dimensiones, implica que no es posible ni siquiera reservar una cantidad de memoria para almacenar el Hessiano. Sin embargo dado el número de ceros en el Hessiano es preferible no reservar una cantidad de memoria y trabajar con alguna técnica para el manejo de dispersidad.

Consideremos el caso de minimizar la función dada por

$$E(f) = \sum_{i=0}^{i=N-1} \sum_{j=0}^{j=M-1} \left[(f_{i,j} - g_{i,j})^2 + \lambda (f_{i,j} - f_{i-1,j})^2 + \lambda (f_{i,j} - f_{i,j-1})^2 \right]$$

Con un valor de $\lambda = 5$ y una senãl bidimensional g dada

$$g = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

El sistema lineal por resolver es $Af = b$ con

$$A = \begin{bmatrix} 11.0 & -5.0 & 0.0 & -5.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -5.0 & 16.0 & -5.0 & 0.0 & -5.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & -5.0 & 11.0 & 0.0 & 0.0 & -5.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -5.0 & 0.0 & 0.0 & 16.0 & -5.0 & 0.0 & -5.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & -5.0 & 0.0 & -5.0 & 21.0 & -5.0 & 0.0 & -5.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & -5.0 & 0.0 & -5.0 & 16.0 & 0.0 & 0.0 & -5.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & -5.0 & 0.0 & 0.0 & 16.0 & -5.0 & 0.0 & -5.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & -5.0 & 0.0 & -5.0 & 21.0 & -5.0 & 0.0 & -5.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -5.0 & 0.0 & -5.0 & 16.0 & 0.0 & 0.0 & -5.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -5.0 & 0.0 & 0.0 & 11.0 & -5.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -5.0 & 0.0 & -5.0 & 16.0 & -5.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -5.0 & 0.0 & -5.0 & 11.0 \end{bmatrix}$$

y $b = [0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0]^T$. La solución es:

$$f = [0.1475, 0.1595, 0.1475, 0.165, 0.2155, 0.165, 0.165, 0.2155, 0.165, 0.1475, 0.1595, 0.1475]^T$$

Para este sistema podemos notar que muchos de los valores son cero.

La factorización de la matriz A es:

$$L = \begin{bmatrix} 3.3166 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -1.5074 & 3.705 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & -1.3494 & 3.0296 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -1.5074 & 0.0 & 0.0 & 3.705 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & -1.3494 & 0.0 & -1.3494 & 4.1662 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & -1.6502 & 0.0 & -1.2 & 3.4403 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & -1.3494 & 0.0 & 0.0 & 3.7654 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & -1.2 & 0.0 & -1.3277 & 4.2185 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -1.4532 & 0.0 & -1.1851 & 3.5331 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -1.3277 & 0.0 & 0.0 & 3.0392 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -1.1851 & 0.0 & -1.645 & 3.4479 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -1.415 & 0.0 & -1.45 & 2.6257 \end{bmatrix}$$

La formula para calcular la descomposicón de Cholesky sobre los elementos fuera de la diagonal es

$$l_{k,i} = \frac{a_{k,i} - \sum_{j=0}^{i-1} l_{i,j} l_{k,j}}{l_{i,i}}$$

Podemos notar, dado que es una matriz bandada que los productos $l_{i,j} l_{k,j} = 0$ siempre son cero y los elementos $a_{k,i} = -\lambda$ por tal razón la ecuación para modificar los elementos fuera de las diagonales se puede expresar como

$$l_{k,i} = \frac{-\lambda}{l_{i,i}}$$

Dado la simplicidad de los elementos fuera de la diagonal, no los almacenaremos y en su lugar realizaremos este simple cálculo y simplimente almacenaremos los datos de la diagonal en una matriz

$$D = \begin{bmatrix} 11.0 & 16.0 & 11.0 \\ 16.0 & 21.0 & 16.0 \\ 16.0 & 21.0 & 16.0 \\ 11.0 & 16.0 & 11.0 \end{bmatrix}$$

Los elementos en la diagonal son actualizados por la ecuación :

$$l_{i,i} = \sqrt{a_{i,i} - \sum_{k=0}^{i-1} l_{i,k}^2}$$

Sustituyendo la expresion para los elementos fuera de la diagonal

$$l_{i,i} = \sqrt{a_{i,i} - \sum_{k=0}^{i-1} \frac{\lambda^2}{l_{k,k}^2}}$$

En nuestra representación matricial

$$D_{i,j} = \sqrt{D_{i,j} - \frac{\lambda^2}{D_{i-1,j}^2} - \frac{\lambda^2}{D_{i,j-1}^2}}$$

Aplicando la ecuaciones anteriores, los elementos de la diagonal factorizados son

$$LD = \begin{bmatrix} 3.3166 & 3.705 & 3.0296 \\ 3.705 & 4.1662 & 3.4403 \\ 3.7654 & 4.2185 & 3.5331 \\ 3.0392 & 3.4479 & 2.6257 \end{bmatrix}$$

Para realizar la solución del sistema tenemos que calcular la sustitución hacia atras y hacia adelante.

3.4.1. Sustitución hacia adelante

Como primer paso debemos resolver el sistema $Ly = b$ utilizando sustitución hacia adelante. La ecuación para llevar a cabo al sustitución hacia adelante esta dada por

$$y_i = \frac{b_i - \sum_{k=0}^{k=i-1} l_{i,k} y_k}{l_{i,i}}$$
$$i = 0, 1, 2, \cdots, N-1$$

Para nuestro sistema de ecuaciones tenemos:

$$y_{i,j} = \frac{b_{i,j} + \lambda * y_{i-1,j}/D_{i-1,j} + \lambda y_{i,j-1}/D_{i,j-1}}{D_{i,j}}$$

La implementación en Java queda como:

```
static public void SAdelante(double A[][], double x[][],
   double b[][], double lambda, int nr, int nc)
   {
     int i, j;
     double suma;

     for(i=0; i<nr; i++)
       for(j=0; j<nc; j++)
       {
         suma = 0;
         if(i-1 >= 0) suma += -lambda * x[i - 1][j] / A[i - 1][j];
         if(j-1 >= 0) suma += -lambda * x[i][j - 1] / A[i][j - 1];

         x[i][j] = (b[i][j] - suma)/A[i][j];
       }
   }
```

3.4.2. Sustitución hacia atras

Como segundo paso debemos resolver el sistema $L^T x = y$ utilizando sustitución hacia atras. La ecuación para llevar a cabo al sustitución hacia atras esta dada por

$$x_k = \frac{y_k - \sum_{i=0}^{i=k-1} l_{k,i} x_i}{l_{k,k}}$$

$$k = N-1, N-2, \cdots, 2, 1, 0$$

Para nuestro sistema de ecuaciones tenemos:

$$x_{i,j} = \frac{b_{i,j} + \lambda * y_{i+1,j}/D_{i+1,j} + \lambda y_{i,j+1}/D_{i,j+1}}{D_{i,j}}$$

La implementación en Java queda como:

```
static public void SAtras(double A[][], double x[][],
  double b[][], double lambda, int nr, int nc)
{
int i, j;
double suma;

for(i=nr-1; i>=0; i--)
  for(j=nc-1; j>=0; j--)
  {
    suma = 0;
    if(i+1 < nr) suma += -lambda*x[i+1][j]/A[i][j];
    if(j+1 < nc) suma += -lambda*x[i][j+1]/A[i][j];
    x[i][j] = (b[i][j] - suma)/A[i][j];
  }
}
```

El resultado depués de aplicar el procedimiento completo es:

$$f = \begin{bmatrix} 0.0457 & 0.0467 & 0.0267 \\ 0.0538 & 0.1015 & 0.0491 \\ 0.0462 & 0.1033 & 0.0588 \\ 0.0278 & 0.0515 & 0.0501 \end{bmatrix}$$

3.4.3. Ejemplo de Filtrado de Imágenes

Como ejemplo considere que para una imágen queremos encontra un imagen filtrada. Para esto utilizaremos la función de energía

$$E(f) = \sum_{i=0}^{i=N-1} \sum_{j=0}^{j=M-1} \left[(f_{i,j} - g_{i,j})^2 + \lambda (f_{i,j} - f_{i-1,j})^2 + \lambda (f_{i,j} - f_{i,j-1})^2 \right]$$

En el caso de una imagen pequenã las dimensiones pueden ser 256×256. Considerando que es el número de variables de nuestro sistema tendremos que el Hessiano tiene que ser una matriz de dobles de tamaño $256^2 \times 256^2$, dando un total de memoria de 128 Giga Bytes. Una cantindad de memoria muy alta si consideramos que solamente necesitamos almacenar menos $0.007\,\%$ de variables diferentes de cero.

Los resultados aplicando Gradiente conjudado y Gradiente conjugado se muestran en la siguiente figura:

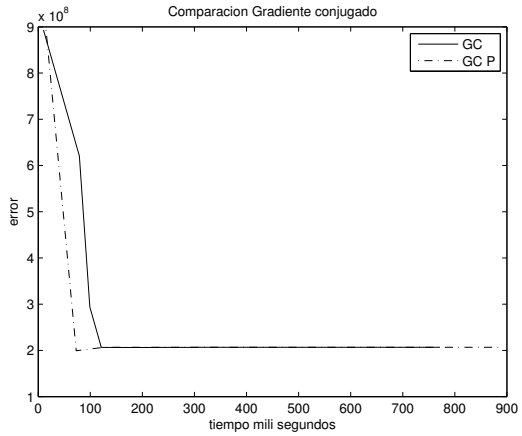


Figura 3.23: Comparación entre el método de GC y GC precondicionado para el caso del filtrado de imagenes.

Tarea

Características de precondicionamiento Para el ejemplo anterior:

1. Comparar la solución para los métodos de descenso de gradiente, Gradiente Conjugado y Gradiente conjugado precondicionado

2. Mostrar en una gráfica el número de iteraciones de cada uno
3. Comparar la matriz de precondicionamiento calculada con Choleski Incompleto y la matriz Inversa.

ver (regulariza.java)

3.5. Gradiente conjugado no lineal

Hemos notado que le método de G.C., puede ser visto como un algoritmo de minimización de funciones cuadráticas convexas. Es natural preguntar que pasa cuando no tenemos una función no lineal.

Flecher y Reaves mostraron que una extensión de este es posible haciendo cambios simples al algoritmo de G.C. (ver [Dennis and Schnabel, 1996, Nocedal and Wright, 1999]

Primero la longitud de paso

$$\alpha^{(k)} = \frac{{r^{(k)}}^T r^{(k)}}{{p^{(k)}}^T A p^{(k)}}$$

debe ser reemplazada por una búsqueda en una dirección.

Segundo la manera de calcular el residuo debe ser reemplazada por el cálculo del gradiente.

Los cambios dan el siguiente algoritmo:

Algoritmo 13 Algoritmo de Gradiente Conjugado no lineal Fletcher Reaves

Dado una función diferenciable $f(x)$ y un valor inicial $x^{(0)}$
Hacer hacer $p^{(0)} = -\nabla f(x^{(0)})$ y poner $k = 0$.
mientras $\|\nabla f(x^{(k)})\| \geq \varepsilon$ **hacer**
  Calcular $\alpha^{(k)}$ que minimize $\phi(\alpha^{(k)})$ en la dirección $p^{(k)}$
  Hacer la actualización $x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}$
  Calcular el valor del gradiente $\nabla f(x^{(k+1)})$
  determinar $\beta_{FR}^{(k+1)} = \frac{\nabla f(x^{(k+1)})^T \nabla f(x^{(k+1)})}{\nabla f(x^{(k)})^T \nabla f(x^{(k)})}$
  Calcular la nueva dirección con $p^{(k+1)} = -\nabla f(x^{(k+1)}) + \beta_{FR}^{(k+1)} p^{(k)}$
  Actualizar $k = k + 1$
fin mientras
devolver $x^{(k)}$

Para calcular α debemos considerar las condiciones de Wolfe

$$f(x^{(k)} + \alpha^{(k)} p^{(k)}) \leq f(x^{(k)}) + c_1 \alpha^{(k)} \nabla f(x^{(k)})^T p^{(k)}$$
$$|\nabla f(x^{(k)} + \alpha^{(k)} p^{(k)})^T p^{(k)}| \leq c_2 |\nabla f(x^{(k)})^T p^{(k)}|$$

3.5.1. Algunas Variantes

Existen muchas variantes del método de Flecher-Reeves que difieren principalmente en la selección del parámetro $\beta^{(k)}$. La más importante de estas variantes propuesta por Polak y Ribiere, define este parámetro como:

$$\beta_{PR}^{(k+1)} = \frac{\nabla f(x^{(k+1)})^T (\nabla f(x^{(k+1)}) - \nabla f(x^{(k)}))}{||\nabla f(x^{(k)})||^2}$$

Un hecho sorprendente acerca del algoritmo PR-GR es que las condiciones de Wolfe no garantizan que $p^{(k)}$ es siempre descendente en la dirección $p^{(k)}$. Si nosotros definimos el parámetro β como:

$$\beta_{PR^+}^{(k+1)} = max \left\{ \beta_{PR}^{(k+1)}, 0 \right\}$$

Da lugar a un algoritmo que llamaremos PR+, entonces una simple adaptación de las condiciones de Wolfe encierra que la propiedad descendente se mantenga.

Si revisamos los valores de β podemos notar que las direcciones no son conjugadas. Una manera de ligar es utilizando la formula de Hestenes-Stiefel dada por

$$\beta_{HS}^{(k+1)} = \frac{\nabla f(x^{(k+1)})(\nabla f(x^{(k+1)}) - \nabla f(x^{(k)}))}{(\nabla f(x^{(k+1)}) - \nabla f(x^{(k)}))^T p^{(k)}}$$

Adicionalmente el único valor de β que garantiza que las direcciones sean conjugada es aquella calculada como

$$\beta^{(k+1)} = \frac{\nabla f(x^{(k+1)}) A(x^{(k+1)}) p^{(k)}}{p^{(k)} A(x^{(k+1)}) p^{(k)}}$$

La implementación en Java para este algoritmo se muestra a continuación, note que se pasa como parámetro unaBeta que sirve para seleccionar cualquiera de los valores dados:

```
public Matriz GC_NoLineal(Matriz x, String unaBeta)
{
    double alpha, beta=0, tol;
    int k =0;

    Matriz g0 = null, g1 = null;
    Matriz p = null, A=null;

    g0 = Gradiente(x);

    p = g0.por( -1);

    System.out.print("Iteracion \t" + (k) + "\t f(x) = \t" + funcion(x) + "\t");
    x.T().imprime();
    k = 1;
    do {
        alpha = alpha_Newton(x, p);
        //alpha = alpha_RD(x, p, 0, alpha_max);

        x = x.mas(p.por(alpha));

        g1 = Gradiente(x);

        if(unaBeta.equals("FR")) {
            beta = (((g1.T())).por(g1)).obten(0, 0)/
                    (((g0.T())).por(g0)).obten(0, 0);
        }
        else if(unaBeta.equals("PR")) {
            beta = ((g1.T()).por(g1.menos(g0))).obten(0,0) /
                    (g0.T().por(g0)).obten(0,0);
            if(beta < 0) beta = 0;
        }
        else if(unaBeta.equals("HS")) {
            beta = ((g1.T()).por(g1.menos(g0))).obten(0,0)/
                    (((g1.menos(g0)).T()).por(p)).obten(0,0);
        }

        else {
            A = Hessiano(x);
            beta = (((g1.T()).por(A)).por(p)).obten(0, 0)/
                    (((p.T()).por(A)).por(p)).obten(0, 0);
```

```
        }

        p = (p.por(beta)).menos(g1);

        System.out.print("Iteracion \t" + (k) + "\t f(x) = \t" + funcion(x) + "\t");
        x.T().imprime();
        tol = Magnitud(g1);
        g0 = Matriz.igual_a(g1);
        k++;
    } while(tol > T);
    return x;
}
```

3.5.2. Ejemplo

Implementar el algoritmo de GC no lineal de Fletcher–Reeves FR, la mejora de Polak–Ribiere y Hestenes-Stiefel, para la función $f(x) = x_1 e^{(-x_1^2 - x_2^2)}$ comparar los resultados. Considere un punto inicial $x^{(0)} = [-0.5, -0.5]^T$

El vector gradiente para esta función es:

$$\nabla f(x) = \begin{bmatrix} (1 - 2x_1^2)e^{(-x_1^2 - x_2^2)} \\ -2x_1 x_2 e^{(-x_1^2 - x_2^2)} \end{bmatrix}$$

El Hessiano es

$$\nabla^2 f(x) = \begin{bmatrix} (-6x_1 + 4x_1^3)e^{(-x_1^2 - x_2^2)} & (-2x_2 + 4x_1^2 x_2)e^{(-x_1^2 - x_2^2)} \\ (-2x_2 + 4x_1^2 x_2)e^{(-x_1^2 - x_2^2)} & (-2x_1 + 4x_1 x_2^2)e^{(-x_1^2 - x_2^2)} \end{bmatrix}$$

Solución utilizando el método de GC-FR

Inicialmente el Gradiente es

$$r^{(0)} = \begin{bmatrix} 0.3033 \\ -0.3033 \end{bmatrix}$$

La magnitud del gradiente es

$$r^{(0)^T} r^{(0)} = 0.183939$$

y la dirección inicial es

$$p^{(0)} = \begin{bmatrix} -0.3033 \\ 0.3033 \end{bmatrix}$$

Iteración 1

Calculamos el tamaño de paso $\alpha^{(1)}$

$$\alpha^{(1)} = 1.0189657832675953$$

Hacemos la actualización

$$x^{(1)} = \begin{bmatrix} -0.5000 \\ -0.5000 \end{bmatrix} + 1.018965 \begin{bmatrix} -0.3033 \\ 0.3033 \end{bmatrix} = \begin{bmatrix} -0.809 \\ -0.191 \end{bmatrix}$$

El Gradiente en $x^{(1)}$ es

$$r^{(1)} = \begin{bmatrix} -0.1548 \\ -0.1548 \end{bmatrix}$$

y la magnitud es:

El valor de $\beta^{(1)}$

$$\beta^{(1)} = \frac{{r^{(1)}}^T r^{(1)}}{{r^{(0)}}^T r^{(0)}} = \frac{0.047952}{0.183939} = 0.260698$$

La nueva dirección de búsqueda es:

$$p^{(1)} = -\begin{bmatrix} -0.1548 \\ -0.1548 \end{bmatrix} + 0.260698 \times \begin{bmatrix} -0.3033 \\ 0.3033 \end{bmatrix} = \begin{bmatrix} 0.0758 \\ 0.2339 \end{bmatrix}$$

Iteración 2

Calculamos el tamaño de paso $\alpha^{(2)}$

$$\alpha^{(1)} = 0.9064563938228329$$

Hacemos la actualización

$$x^{(2)} = \begin{bmatrix} -0.809 \\ -0.191 \end{bmatrix} + 0.906456 \begin{bmatrix} 0.0758 \\ 0.2339 \end{bmatrix} = \begin{bmatrix} -0.7403 \\ 0.0210 \end{bmatrix}$$

El Gradiente en valor $x^{(2)}$ es

$$r^{(2)} = \begin{bmatrix} -0.0556 \\ 0.0180 \end{bmatrix}$$

y la magnitud es

$$r^{(2)^T} r^{(2)} = 0.003410$$

El valor de $\beta^{(2)}$

$$\beta^{(2)} = \frac{r^{(2)^T} r^{(2)}}{r^{(1)^T} r^{(1)}} = \frac{0.003410}{0.047952} = 0.071132$$

La nueva dirección de búsqueda es:

$$p^{(2)} = -\begin{bmatrix} -0.0556 \\ 0.0180 \end{bmatrix} + 0.071132 \times \begin{bmatrix} 0.0758 \\ 0.2339 \end{bmatrix} = \begin{bmatrix} 0.0610 \\ -0.0014 \end{bmatrix}$$

Las soluciones utilizando los tres métodos es $x = [-0.707107, 0.000000]^T$. En la siguientes tablas, se muestra el proceso de convergencia de los los tres métodos.

Descenso de Gradiente			
k	$f(x^{(k)})$	$x_1^{(k)}$	$x_2^{(k)}$
0	-0.303265	-0.5000	-0.5000
1	-0.405384	-0.809017	-0.190983
2	-0.426608	-0.677837	-0.059803
3	-0.428615	-0.717333	-0.020307
4	-0.428852	-0.703752	-0.006726
5	-0.428878	-0.708231	-0.002247
6	-0.428881	-0.706733	-7.48E-4
7	-0.428881	-0.707232	-2.5E-4
8	-0.428881	-0.707065	-8.3E-5
9	-0.428881	-0.707121	-2.8E-5
10	-0.42888	-0.707102	-9.0E-6

GC Fletcher–Reeves FR			
k	$f(x^{(k)})$	$x_1^{(k)}$	$x_2^{(k)}$
0	-0.303265	-0.5000	-0.5000
1	-0.405384	-0.8090	-0.1910
2	-0.427761	-0.7403	0.0210
3	-0.428705	-0.7069	0.0203
4	-0.428854	-0.7019	0.0030
5	-0.428880	-0.7064	-0.0015
6	-0.428881	-0.7074	-8.0E-4
7	-0.428881	-0.7073	0.0000
8	-0.428881	-0.7071	1.0E-4
9	-0.428881	-0.7071	0.0000
10	-0.42888	-0.7071	0.0000

GC Polak–Ribiere			
k	$f(x^{(k)})$	$x_1^{(k)}$	$x_2^{(k)}$
0	-0.303265	-0.5000	-0.5000
1	-0.405384	-0.8090	-0.1910
2	-0.427761	-0.7403	0.0210
3	-0.428838	-0.7055	0.0098
4	-0.428881	-0.7071	0.0000
5	-0.428881	-0.7071	0.0000

GC Hestenes-Stiefel			
k	$f(x^{(k)})$	$x_1^{(k)}$	$x_2^{(k)}$
0	-0.303265	-0.5000	-0.5000
1	-0.405384	-0.8090	-0.1910
2	-0.427761	-0.7403	0.0210
3	-0.428880	-0.7066	0.0017
4	-0.428881	-0.7071	0.0000
5	-0.428881	-0.7071	0.0000

GC Normal			
k	$f(x^{(k)})$	$x_1^{(k)}$	$x_2^{(k)}$
0	-0.303265	-0.5000	-0.5000
1	-0.405384	-0.8090	-0.1910
2	-0.428860	-0.7032	-0.0044
3	-0.428881	-0.7071	0.0000

note que para este ejemplo, el método de GC-PR+ tiene un mejor desempeño.

Métodos de Newton

4.1. Método de Newton

Consideremos que la función f es de clase dos, es decir que la segunda derivada puede ser calculada. La idea consiste en reemplazar en la vecindad del punto $x^{(k)}$ de la función f por una aproximación cuadrática $q(x)$ dada por

$$q(x) = f(x^{(k)}) + \nabla f^T(x^{(k)})(x - x^{(k)}) + \frac{1}{2}(x - x^{(k)})^T \nabla^2 f(x^{(k)})(x - x^{(k)})$$

llamaremos a $x^{(k+1)}$ el mínimo de $q(x)$. Para calcularlo, es necesario que la matriz $\nabla^2 f(x^{(k)})$ sea una matriz definida positiva. La función $q(x)$ es entonces estrictamente convexa y tiene un mínimo único en $x^{(k+1)}$ dado por

$$\nabla_q(x^{(k+1)}) = 0$$

Esto da lugar sistema lineal de ecuaciones:

$$\nabla f(x^{(k)}) = -\nabla^2 f(x^{(k)})(x^{(k+1)} - x^{(k)})$$

La solución para $x^{(k+1)}$ es el mínimo, lo cual da lugar a la recurrencia

$$x^{(k+1)} = x^{(k)} - \left[\nabla^2 f(x^{(k)})\right]^{-1} \nabla f(x^{(k)}) \tag{4.31}$$

Comparando las ecuación 4.31 con la sucesión para el descenso de gradiente, podemos obsérvese que la dirección y el tamaño de paso es obtenido de manera óptima.

Una propiedad importante de este método es que converge en un solo paso cuando se tiene una función estrictamente cuadrática.

Demostración:

Dada una función cuadrática

Algoritmo 14 Algoritmo de Newton

Entrada: Dado una función diferenciable dos veces $f(x)$ y un valor inicial $x^{(0)}$
Salida: $x^{(k)}$
  mientras $(\|\nabla f(x^{(k)})\| \geq \varepsilon)$ **hacer**
    $x^{(k+1)} = x^{(k)} - [\nabla^2 f(x^{(k)})]^{-1} * \nabla f(x^{(k)})$
    $k \leftarrow k + 1$
  fin mientras
  devolver $x^{(k)}$

$$f(x) = \frac{1}{2}x^T Ax + x^T b + c$$

el gradiente lo podemos calcular como $\nabla f(x) = Ax + b$ y el Hessiano es $\nabla^2 f(x) = A$. Si sustituimos en la ecuación 4.31 tenemos

$$= x^{(k)} - A^{-1}(Ax^{(k)} + b)$$

$$x^{(k+1)} = x^{(k)} - x^{(k)} - A^{-1}b$$

Lo que finalmente da como resultado

$$x^{(k+1)} = -A^{-1}b$$

4.1.1. Ejemplo

Dada la función $f(x) = 10x_1^2 + x_2^2$ y un punto inicial $x^{(0)} = [1, 2]^T$, calcular el mínimo utilizando el algoritmo de Newton.

Para esta función el gradiente y el Hessiano es:

$$\nabla f(x) = \begin{array}{|c|}\hline 20x_1 \\ \hline 2x_2 \\ \hline \end{array}$$

y el Hessiano es:

$$A(x) = \begin{array}{|c|c|}\hline 20 & 0 \\ \hline 0 & 2 \\ \hline \end{array}$$

Aplicando la ecuación 4.31 tenemos

$$x^{(k+1)} = \begin{array}{|c|}\hline 1 \\ \hline 2 \\ \hline \end{array} - \begin{array}{|c|c|}\hline 20 & 0 \\ \hline 0 & 2 \\ \hline \end{array}^{-1} \begin{array}{|c|}\hline 20 \\ \hline 4 \\ \hline \end{array} = \begin{array}{|c|}\hline 0 \\ \hline 0 \\ \hline \end{array}$$

4.1.2. Ejemplo

Minimizar es $f(x) = x_1 e^{-x_1^2 - x_2^2}$, con un punto inicial $x_1^{(0)} = -0.5$, $x_2^{(1)} = -0.1$ aplicando el Método de Newton.

k	Newton	DG	GC-PR+	GC-FR
0	-0.385526	-0.385526	-0.385526	-0.385526
1	-0.428652	-0.427442	-0.427442	-0.427442
2	-0.428882	-0.428855	-0.428879	-0.428879
3	-0.428882	-0.428882	-0.428882	-0.428882
4	-0.428882	-0.428882	-0.428882	-0.428882
5	-0.428882	-0.428882	-0.428882	-0.428882

Note que el mejor desempeño lo tiene el método de Newton para este problema.

4.2. Problemas de convergencia del Método de Newton

Si uno desea aplicar el método a una función arbitraria encontraremos algunas dificultades, esencialmente debido al hecho de que no encontraremos convergencia global. Si el punto inicial $x^{(0)}$ esta muy lejos de x^* el método no podrá converger.

4.2.1. Ejemplo

Minimizar es $f(x) = x_1 e^{-x_1^2 - x_2^2}$, con un punto inicial $x_1^{(0)} = -0.5$, $x_2^{(0)} = -0.5$ aplicando el Método de Newton.

k	Newton	DG	GC-PR+	GC-FR
0	-0.303265	-0.303265	-0.303265	-0.303265
1	-0.135335	-0.405385	-0.405385	-0.405385
2	-0.012255	-0.426609	-0.427761	-0.428705
3	-0.003497	-0.428616	-0.428839	-0.428855
4	-0.000345	-0.428853	-0.428882	-0.428881
5	-0.000113	-0.428879	-0.428882	-0.428882

Podemos ver que todos los algoritmos convergen, pero el caso de Newton la solución a la que llega es $x^{(20)} = [-2.33182, 4.39075]^T$, mientras que la solución con los otros métodos fue $x^{(20)} = [-0.70711, 0.00000]^T$. Esta condición se debe a que el punto inicial para el método de Newton esta muy lejos y la aproximación cuadrática, resulta in apropiada.La figura 4.24 muestra los valores de las iteraciones de la función $f(x)$ y la figura 4.25 se muestra la función y se puede ver que esta tiene solamente un mínimo.

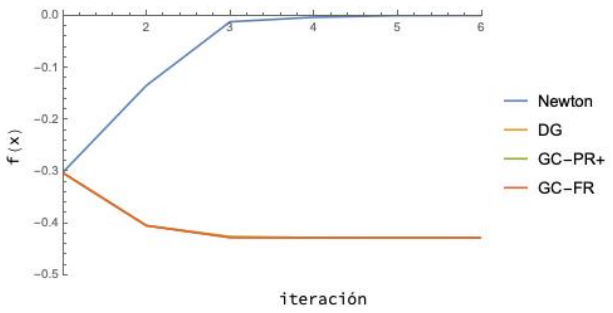


Figura 4.24: Comportamiento de las iteraciones al minimizar la función.

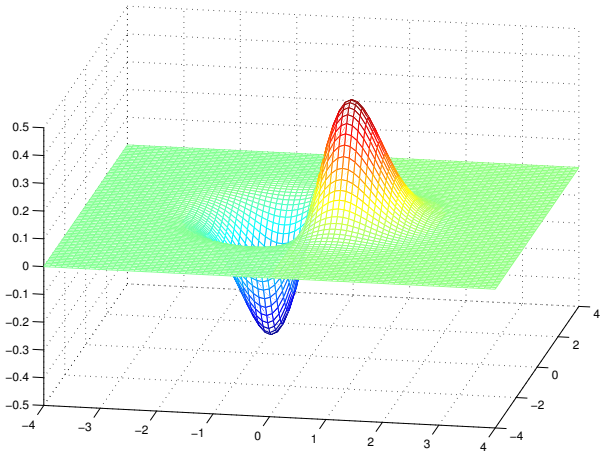


Figura 4.25: Función $f(x) = x_1 e^{-x_1^2 - x_2^2}$.

Para comenzar, la aproximación de $f(x)$ dada por $q(x)$ es solamente valida a la vecindad de x_k, el tamaño de paso puede ser controlado a través de una formula iterativa de tipo:

$$x^{(k+1)} = x^{(k)} - \lambda^{(k)} \left[\nabla^2 f(x^{(k)})\right]^{-1} \nabla f(x^{(k)})$$

donde $\lambda^{(k)}$ es un escalar, seleccionado por ejemplo, de tal manera que $||x^{(k+1)} - x^{(k)}||$ no sea muy largo. También podemos seleccionarlo tal que $\lambda^{(k)}$ minimice $\phi(\lambda^{(k)}) = f(x^{(k)} + \lambda^{(k)} d^{(k)})$ en la dirección de:

$$d^{(k)} = -\left[\nabla^2 f(x^{(k)})\right]^{-1} \nabla f(x^{(k)})$$

4.2.2. Ejemplo

Minimizar $f(x) = (x-3)^4$ utilizando el método de Newton y la modificación planteada utilizando un valor $\lambda^{(k)} = 2.9$. Considere un punto inicial $x^{(0)} = 2$.

k	Newton	Newton con $\lambda = 2.9$
0	2.00000	2.00000
1	2.33333	2.96667
2	2.55556	2.99889
3	2.70370	2.99996
4	2.80247	3.00000
5	2.86831	3.00000
6	2.91221	3.00000
7	2.94147	3.00000
8	2.96098	3.00000
9	2.97399	3.00000
10	2.98266	3.00000
11	2.98844	3.00000

4.3. Algoritmo de Levenberg-Marquardt

Una variación del algoritmo de Newton consiste en sumar en la diagonal un valor μ tal que nos garantice que la matriz Hessiana es una matriz definida positiva. Al sumar un valor constante a la matriz Hessiana, esta tenderá a ser diagonal dominante y definida positiva. Esta variante es conocida como el algoritmo de Levenberg-Marquardt y se presenta en el algoritmo 15. El algoritmo calcula automáticamente el valor de μ de manera automática con un mecanismo de aumentar y disminuir su valor hasta encontrar el mínimo.

Algoritmo 15 Algoritmo de Levenberg-Marquardt

Entrada: Dado una función $f(x)$ y un valor inicial $x^{(0)}$
Salida: $x^{(k)}$
  Hacer $\mu^{(0)} = 0.001$ y $\nu = 10$
  repetir
    Poner $\mu^{(k)} = \frac{\mu^{(k)}}{\nu}$
    repetir
      Resolver $\left[\nabla^2(x^{(k)}) + \mu^{(k)} I\right] \delta^{(k)} = \nabla(x^{(k)})$
      $x^{(k+1)} = x^{(k)} - \delta^{(k)}$
      $\mu^{(k)} = \mu^{(k)} \times \nu$
    hasta que $(F(x^{(k+1)}) < F(x^{(k)}))$
    Hacer $\mu^{(k+1)} = \mu^{(k)}$
    $k \leftarrow k + 1$
  hasta que $(\|\nabla f(x^{(k)})\| \leq \varepsilon)$
  devolver $x^{(k)}$

4.4. Método de Newton Modificado

Una alternativa propuesta por Ralston y Rabinowitz (1978) es la de definir una nueva función $u(x)$, que es el cociente del gradiente de $g(x)$ y su derivada (en una dimensión)

$$u(x) = \frac{g(x)}{g'(x)}$$

Se puede mostrar que esta función tiene las mismas raíces que $g(x)$ y que la multiplicidad de raíces no afectará. La formulación del método de Newton es:

$$x^{(k+1)} = x^{(k)} - \frac{u(x^{(k)})}{u'(x^{(k)})}$$

La derivada de u(x) es:

$$u'(x) = \frac{g'(x)g'(x) - g(x)g''(x)}{[g'(x)]^2}$$

Sustituyendo esta, tenemos la formulación final del Método de Newton modificado.

$$x^{(k+1)} = x^{(k)} - \frac{g'(x^{(k)})g(x^{(k)})}{g'(x^{(k)})g'(x^{(k)}) - g(x^{(k)})g''(x^{(k)})}$$

Para el ejemplo anterior tenemos que con un valor inicial $x^{(0)} = 2$ la solución es:

$$x^{(k+1)} = 2 - \frac{12 * (-4)}{12 * 12 - (-4) * (-24)} = 3$$

Note que la solución se da en una sola iteración.

4.5. Método de Broyden's o secante

Para este método consideraremos que los valores de x serán calculados utilizando:

$$x^{(k+1)} = x^{(k)} - \left[A^{(k)}\right]^{-1} \nabla f(x^{(k)})$$

donde $\nabla f(x^{(k)})$ es el gradiente, pero en lugar de calcular $A^{(k)}$ como la matriz Hessiana realizaremos una aproximación con secantes. Así pues:

$$A^{(k)} = \frac{\nabla f(x^{(k)}) - \nabla f(x^{(k-1)})}{x^{(k)} - x^{(k-1)}} \tag{4.32}$$

La restricción de la secante la podemos dar como

$$A^{(k)}(x^{(k)} - x^{(k-1)}) = \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$$

$$A^{(k)} s^{(k)} = y^{(k)}$$

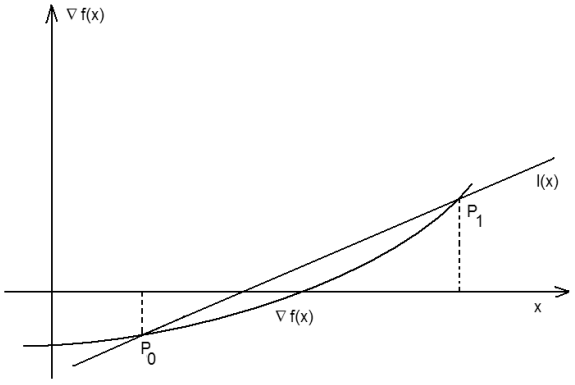


Figura 4.26: Gradiente con una aproximación de secantes

donde $s^{(k)} = x^{(k)} - x^{(k-1)}$ y $y^{(k)} = \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$

Para determinar una forma iterativa de calcular $A^{(k)}$, calculamos para la figura 4.26, la ecuación de la linea recta que pasa en el punto P_0

$$l_{k-1}(x) = \nabla f(x^{(k-1)}) + A^{(k-1)}(x - x^{(k-1)})$$

de igual manera en el punto P_1

$$l_k(x) = \nabla f(x^{(k)}) + A^{(k)}(x - x^{(k)})$$

Es deseable que las dos líneas sean iguales y que con esto se cumpla que esta linea es secante. Así :

$$l_k(x) - l_{k-1}(x) = 0$$

sustituyendo los valores respectivos

$$\nabla f(x^{(k)}) + A^{(k)}(x - x^{(k)}) - \nabla f(x^{(k-1)}) - A^{(k-1)}(x - x^{(k-1)}) = 0$$

re agrupando términos tenemos

$$\nabla f(x^{(k)}) - \nabla f(x^{(k-1)}) + A^{(k)}(x - x^{(k)}) - A^{(k-1)}(x - x^{(k-1)}) = 0$$

Aplicando la definición de secante dada por la ecuación 4.32 tenemos:

$$A^{(k)}(x^{(k)} - x^{(k-1)}) + A^{(k)}(x - x^{(k)}) - A^{(k-1)}(x - x^{(k-1)}) = 0$$

reagrupando términos

$$A^{(k)}(x^{(k)} - x^{(k-1)}) + A^{(k)}(x - x^{(k)}) - A^{(k-1)}(x - x^{(k-1)}) = (A^{(k)} - A^{(k-1)})(x - x^{(k-1)})$$

suponiendo $x = x^{(k)}$ tenemos

$$(A^{(k)} - A^{(k-1)})(x^{(k)} - x^{(k-1)}) = A^{(k)}(x^{(k)} - x^{(k-1)}) + A^{(k)}(x^{(k)} - x^{(k)}) - A^{(k-1)}(x^{(k)} - x^{(k-1)})$$

$$(A^{(k)} - A^{(k-1)})s^{(k)} = A^{(k)}(x^{(k)} - x^{(k-1)}) - A^{(k-1)}(x^{(k)} - x^{(k-1)})$$

$$(A^{(k)} - A^{(k-1)})s^{(k)} = y^{(k)} - A^{(k-1)}(x^{(k)} - x^{k-1)})$$

$$(A^{(k)} - A^{(k-1)})s^{(k)} = y^{(k)} - A^{(k-1)}s^{(k)}$$

por lo tanto

$$A^{(k)} = A^{(k-1)} + \frac{(y^{(k)} - A^{(k-1)}s^{(k)})s^{(k)^T}}{s^{(k)^T}s^{(k)}}$$

En general, para cualquier iteración, podemos calcular la pendiente del hiperplano de la secante utilizando la ecuación

$$A^{(k+1)} = A^{(k)} + \frac{(y^{(k+1)} - A^{(k)}s^{(k+1)})s^{(k+1)^T}}{s^{(k+1)^T}s^{(k+1)}} \tag{4.33}$$

4.5.1. Algoritmo de Broyden

El algoritmo para el método de Broyden se presenta en 16:

Algoritmo 16 Algoritmo de Broyden

Entrada: Dado una función diferenciable $f(x)$ y un valor inicial $x^{(0)}$
Salida: $x^{(k)}$

Hacer $k = 0$ y una estimación del Hessiano $A^{(0)}$
mientras $(\|\nabla f(x^{(k)})\| \geq \varepsilon)$ **hacer**
Resolver $A^{(k)}s^{(k+1)} = -\nabla f(x^{(k)})$ para $s^{(k+1)}$
$x^{(k+1)} = x^{(k)} + s^{(k+1)}$
$y^{(k+1)} = \nabla f(x^{(k+1)}) - \nabla f(x^{(k)})$
$A^{(k+1)} = A^{(k)} + \frac{(y^{(k+1)} - A^{(k)}s^{(k+1)})s^{(k+1)^T}}{s^{(k+1)^T}s^{(k+1)}}$
$k \leftarrow k + 1$
fin mientras
devolver $x^{(k)}$

4.5.2. Ejemplo

Minimizar es $f(x) = x_1 e^{-x_1^2 - x_2^2}$, con un punto inicial $x_1^{(0)} = -0.5$, $x_2^{(0)} = -0.5$ aplicando el Método de Broyden con una matriz Hessiana inicial igual a la identidad. Comparar el resultado con el método de Newton.

Para la función $f(x)$ el Gradiente es:

$$\nabla f(x) = \begin{bmatrix} e^{-x_1^2 - x_2^2} - 2e^{-x_1^2 - x_2^2}x_1^2 \\ -2e^{-x_1^2 - x_2^2}x_1 x_2 \end{bmatrix}$$

Primer iteración

$$A^{(0)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

el gradiente es

$$\nabla f(x^{(0)}) = \begin{bmatrix} 0.303265 \\ -0.303265 \end{bmatrix}$$

$$s_1 = -A_0^{-1}\nabla f(x_0) = \begin{bmatrix} -0.303265 \\ 0.303265 \end{bmatrix}$$

$$x^{(1)} = x^{(0)} + s^{(1)} = \begin{bmatrix} -0.803265 \\ -0.196735 \end{bmatrix}$$

$$\nabla f(x^{(1)}) = \begin{bmatrix} -0.146579 \\ -0.159492 \end{bmatrix}$$

$$y^{(1)} = \nabla f(x^{(1)}) - \nabla f(x^{(0)}) = \begin{bmatrix} -0.146579 \\ -0.159492 \end{bmatrix} - \begin{bmatrix} 0.303265 \\ -0.303265 \end{bmatrix} = \begin{bmatrix} -0.449844 \\ 0.143773 \end{bmatrix}$$

La actualización de A es:

$$A^{(1)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \frac{\left(\begin{bmatrix} -0.449844 \\ 0.143773 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -0.303265 \\ 0.303265 \end{bmatrix} \right) [-0.303265, 0.303265]}{[-0.303265, 0.303265] \begin{pmatrix} -0.303265 \\ 0.303265 \end{pmatrix}}$$

$$A^{(1)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \frac{\begin{bmatrix} -0.146579 \\ -0.159492 \end{bmatrix} [-0.303265, 0.303265]}{0.18394}$$

$$A^{(1)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0.241667 & -0.241667 \\ 0.262958 & -0.262958 \end{bmatrix} = \begin{bmatrix} 1.24167 & -0.241667 \\ 0.262958 & 0.737042 \end{bmatrix}$$

Segunda Iteración

$$s^{(2)} = -A^{-1}\nabla f(x^{(1)}) = \begin{bmatrix} 0.149767 \\ 0.162961 \end{bmatrix}$$

$$x^{(2)} = x^{(1)} + s^{(2)} = \begin{bmatrix} -0.653498 \\ -0.0337732 \end{bmatrix}$$

Cuadro 4.9: Solución con el método de Broyden

k	$f(x)$	$x^{(k)}$
0	-0.303265	[-0.5, -0.5]
1	-0.303265	[-0.803265, -0.196735]
2	-0.303265	[-0.653498, -0.0337732]
3	-0.303265	[-0.718588, 0.0288411]
4	-0.303265	[-0.708339, -0.00062164]
5	-0.303265	[-0.707067, 4.9495e-05]

Cuadro 4.10: Solución con el método de Newton

k	$f(x)$	$x^{(k)}$
0	-0.303265	[-0.5, -0.5]
1	-0.135335	[-1, 1]
2	-0.0122546	[-1.25, 1.75]
3	-0.00349727	[-1.35353, 2.03136]
4	-0.00107748	[-1.44164, 2.26287]
5	-0.000345221	[-1.52062, 2.46539]
6	-0.000113329	[-1.59325, 2.64812]
7	-3.78311e-05	[-1.66108, 2.81614]
8	-1.27846e-05	[-1.72507, 2.97266]
9	-4.36133e-06	[-1.78589, 3.11981]
10	-1.49898e-06	[-1.84405, 3.25917]
11	-5.18346e-07	[-1.89988, 3.39188]
12	-1.80152e-07	[-1.95369, 3.51884]
13	-6.28805e-08	[-2.0057, 3.64077]

$$\nabla f(x^{(2)}) = \begin{bmatrix} 0.0950675 \\ -0.0287662 \end{bmatrix}$$

$$y^{(2)} = \begin{bmatrix} 0.241646 \\ 0.130726 \end{bmatrix}$$

$$A^{(2)} = \begin{bmatrix} 1.24167 & -0.241667 \\ 0.262958 & 0.737042 \end{bmatrix}$$

En las tablas 4.9 y 4.10 se presenta el resumen de las iteraciones. Note que el algoritmo de Broyden tiene un mejor desempeño debido a que la matriz Hessiana inicial (identidad) tiene mejores condiciones que la matriz Hessiana.

4.5.3. Ejemplo

Dado

$$\nabla f(x) = \begin{bmatrix} x_1 + x_2 & -3 \\ x_1^2 + x_2^2 & -9 \end{bmatrix}$$

la cual tiene raíces $(0,3)^T$ y $(3,0)^T$. Dado $x_0 = (1,5)^T$ aplicando el método de la secante calcular la solución y comparar con el método de Newton.

Primer iteración

$$A^{(0)} = \nabla^2(x^{(k)}) = \begin{bmatrix} 1 & 1 \\ 2 & 10 \end{bmatrix}$$

el gradiente es

$$\nabla f(x^{(0)}) = \begin{bmatrix} 3 \\ 17 \end{bmatrix}$$

$$s_1 = -A_0^{-1}\nabla f(x_0) = \begin{bmatrix} -1.625 \\ -1.375 \end{bmatrix}$$

$$x^{(1)} = x^{(0)} + s^{(1)} = \begin{bmatrix} -0.625 \\ 3.625 \end{bmatrix}$$

$$\nabla f(x^{(1)}) = \begin{bmatrix} 0 \\ 4.53125 \end{bmatrix}$$

La actualización de A es:

$$A^{(1)} = A^{(0)} + \begin{bmatrix} 0 & 0 \\ -1.625 & -1.375 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0.375 & 8.625 \end{bmatrix}$$

Segunda Iteración

$$s^{(1)} = -A^{-1}\nabla f(x^{(1)}) = \begin{bmatrix} 0.549 \\ -0.549 \end{bmatrix}$$

$$x^{(2)} = x^{(1)} + s^{(2)} = \begin{bmatrix} -0.076 \\ -3.076 \end{bmatrix}$$

$$\nabla f(x^{(1)}) = \begin{bmatrix} 0 \\ 0.466 \end{bmatrix}$$

$$A^{(2)} = A^{(1)} + \begin{bmatrix} 1 & 1 \\ -0.799 & 8.201 \end{bmatrix}$$

4.6. Método de Secante con actualización BFGS

Al resolver un problema de minimización sin restricciones por algún Quasi algoritmo de Newton, se tienen muchas mas ventajas y estabilidad numérica si se tiene un pseudo Hessiano simétrico y definido positivo. En este caso utilizaremos la actualización del Hessiano propuesto por Broyden, Fletcher, Goldfarb and Shanno en 1970. El método se aplica de manera similar, pero el calculo de la matriz Hessiana se hace de la siguiente forma

$$A^{(k+1)} = A^{(k)} + \frac{y^{(k+1)}y^{(k+1)^T}}{y^{(k+1)^T}s^{(k+1)}} - \frac{A^{(k)}s^{(k+1)}s^{(k+1)^T}A^{(k)}}{s^{(k+1)^T}A^{(k)}s^{(k+1)}}$$

Algoritmo 17 Algoritmo BFGS

Entrada: Dado una función diferenciable $f(x)$ y un valor inicial $x^{(0)}$
Salida: $x^{(k)}$
  Hacer $k = 0$ y una estimación $A^{(0)}$
  mientras $(\|\nabla f(x^{(k)})\| \leq \varepsilon)$ **hacer**
    Resolver $A^{(k)}s^{(k+1)} = -\nabla f(x^{(k)})$ para $s^{(k+1)}$
    $x^{(k+1)} = x^{(k)} + s^{(k+1)}$
    $y^{(k+1)} = \nabla f(x^{(k+1)}) - \nabla f(x^{(k)})$
    $A^{(k+1)} = A^{(k)} + \frac{y^{(k)}y^{(k+1)^T}}{y^{(k+1)^T}s^{(k+1)}} - \frac{A^{(k)}s^{(k+1)}s^{(k+1)^T}A^{(k)}}{s^{(k+1)^T}A^{(k)}s^{(k+1)}}$
    $k \leftarrow k + 1$
  fin mientras
  devolver $x^{(k)}$

La implementación en Java de este método es:

```
public Matriz BFGS(Matriz x) {
  int k = 0;
  double tol, ss;
  Matriz A = Calcula_Hessiano(x);
  Matriz s = null, y = null, g1 = null, g0 = Gradiente(x);
  Matriz aux = null;
  Matriz x1, x0;
  x0 = Matriz.igual_a(x);

  do {
      System.out.print("iteracion \t" + k  + " \t f(x) = \t " + funcion(x0) + "\t");
      x0.T().imprime();
      s = (g0.entre(A)).por(-1);
```

```
        x1 = x0.mas(s);

        g1 = Gradiente(x1);
        y = g1.menos(g0);

        ss = 1.0/(s.T().por(s)).obten(0,0);

        aux = s.T().por(A).por(s);

        A = A.menos(((A.por(s)).por(s.T())).por(A).por(1.0/aux.obten(0,0)));

        aux = y.T().por(s);
        A = A.mas((y.por(y.T())).por(1.0/aux.obten(0,0)));

        k++;
        tol = Magnitud(g1);
        g0 = Matriz.igual_a(g1);
        x0 = Matriz.igual_a(x1);
        //System.out.println("Tol = " + tol);
    } while ( tol > T);
    System.out.println("La solucion en "+ k + " iteraciones es x* = " );
    x1.imprime();
    return x1;
}
```

Una manera de hacer la estimación inicial de una matriz $A^{(0)}$ es

```
Matriz Calcula_Hessiano(Matriz x) {
    int i, N = x.nren;
    Matriz A = new Matriz(N, N);
    Matriz g = Gradiente(x);
    Matriz p = Direccion(g);
    double mag = Magnitud(g);
    double alpha , phi, phi_ant;

    alpha = 0;

    phi = phi(x, alpha, p);

    do {
        phi_ant = phi;
        alpha += da;
```

```
            phi = phi(x,alpha,p);
        } while (phi < phi_ant);
        alpha = this.alpha_RD(x, p, alpha-da, alpha);
        for(i=0; i<N; i++)
            A.inserta(i, i, mag/(alpha));

        return A;
    }
```

4.7. Mínimos cuadrados no lineales

El problema de mínimos cuadrados puede ser escrito como

$$F(x) = \sum_{i=1}^{m} f_i^2(x) = f_1^2(x) + f_2^2(x) + \cdots + f_m^2(x)$$

donde

$$f^T(x) = [f_1(x) + f_2(x) + \cdots + f_m(x)]^T$$

Entonces la función objetivo que deseamos minimizar puede ser expresada de forma equivalente por

$$F(x) = f^T(x) f(x)$$

Aplicando un esquema de Newton, debemos estimar el gradiente y Hessiano. Para obtener una expresión para el vector gradiente, las primeras derivadas parciales respecto a x_j esta dada por:

$$\nabla F(x) = \begin{bmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_n(x) \end{bmatrix} = \begin{bmatrix} \frac{\partial F(x)}{\partial x_1} \\ \frac{\partial F(x)}{\partial x_2} \\ \vdots \\ \frac{\partial F(x)}{\partial x_n} \end{bmatrix} = 2 \begin{bmatrix} f_1(x)\frac{\partial f_1(x)}{\partial x_1} + f_2(x)\frac{\partial f_2(x)}{\partial x_1} + \cdots + f_n(x)\frac{\partial f_m(x)}{\partial x_1} \\ f_1(x)\frac{\partial f_1(x)}{\partial x_2} + f_2(x)\frac{\partial f_2(x)}{\partial x_2} + \cdots + f_n(x)\frac{\partial f_m(x)}{\partial x_2} \\ \vdots \\ f_1(x)\frac{\partial f_1(x)}{\partial x_n} + f_2(x)\frac{\partial f_2(x)}{\partial x_n} + \cdots + f_n(x)\frac{\partial f_m(x)}{\partial x_n} \end{bmatrix}$$

En general el j-ésimo termino del gradiente $g_j(x)$ se puede calcular como;

$$g_j(x) = \frac{\partial F(x)}{\partial x_j} = 2\sum_{i=1}^{m} f_i(x)\frac{\partial f_i(x)}{\partial x_j}$$

Podemos de manera equivalente escribir el gradiente en forma compacta como

$$\nabla F(x) = \begin{bmatrix} \frac{\partial f_1(x)}{\partial x_1} & \frac{\partial f_2(x)}{\partial x_1} & \cdots & \frac{\partial f_m(x)}{\partial x_1} \\ \frac{\partial f_1(x)}{\partial x_2} & \frac{\partial f_2(x)}{\partial x_2} & \cdots & \frac{\partial f_m(x)}{\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_1(x)}{\partial x_n} & \frac{\partial f_2(x)}{\partial x_n} & \cdots & \frac{\partial f_m(x)}{\partial x_n} \end{bmatrix} \begin{bmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_m(x) \end{bmatrix}$$

Si definimos la matriz Jacobiana J como

$$J = \begin{bmatrix} \frac{\partial f_1(x)}{\partial x_1} & \frac{\partial f_1(x)}{\partial x_2} & \cdots & \frac{\partial f_1(x)}{\partial x_m} \\ \frac{\partial f_2(x)}{\partial x_1} & \frac{\partial f_2(x)}{\partial x_2} & \cdots & \frac{\partial f_2(x)}{\partial x_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m(x)}{\partial x_1} & \frac{\partial f_m(x)}{\partial x_2} & \cdots & \frac{\partial f_m(x)}{\partial x_m} \end{bmatrix}$$

entonces podemos ver que el vector gradiente puede ser escrito como

$$\nabla F(x) = g(x) = 2J^T(x)f(x)$$

Para calcular el Hessiano $H(x)$ hacemos:

$$\nabla^2 F(x) = H(x) = \begin{bmatrix} \frac{\partial g_1(x)}{\partial x_1} & \frac{\partial g_1(x)}{\partial x_2} & \cdots & \frac{\partial g_1(x)}{\partial x_n} \\ \frac{\partial g_2(x)}{\partial x_1} & \frac{\partial g_2(x)}{\partial x_2} & \cdots & \frac{\partial g_2(x)}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_n(x)}{\partial x_1} & \frac{\partial g_n(x)}{\partial x_2} & \cdots & \frac{\partial g_n(x)}{\partial x_n} \end{bmatrix}$$

$$H(x) = 2\sum_{i=1}^{m} \left[\begin{array}{c|c|c} \frac{\partial f_i(x)}{\partial x_1}\frac{\partial f_i(x)}{\partial x_1} + f_i(x)\frac{\partial^2 f_i(x)}{\partial^2 x_1} & \frac{\partial f_i(x)}{\partial x_2}\frac{\partial f_i(x)}{\partial x_1} + f_i(x)\frac{\partial^2 f_i(x)}{\partial x_1 \partial x_2} & \cdots \\ \hline \frac{\partial f_i(x)}{\partial x_1}\frac{\partial f_i(x)}{\partial x_2} + f_i(x)\frac{\partial^2 f_i(x)}{\partial x_1 \partial x_2} & \frac{\partial f_i(x)}{\partial x_2}\frac{\partial f_i(x)}{\partial x_2} + f_i(x)\frac{\partial^2 f_i(x)}{\partial^2 x_2} & \cdots \\ \hline \vdots & \vdots & \ddots \end{array}\right]$$

Entonces el término del Hessiano en el k-ésimo renglo y j-ésima columna es:

$$H_{kj}(x) = 2\sum_{i=1}^{m}\left\{\frac{\partial f_i(x)}{\partial x_k}\frac{\partial f_i(x)}{\partial x_j} + f_i(x)\frac{\partial^2 f_i(x)}{\partial x_k \partial x_j}\right\}$$

La matriz Hessiana de $F(x)$ puede reescribirse es forma compacta como

$$\nabla^2 F(x) = H(x) = 2J^T(x)J(x) + 2\sum_{i=1}^{m} f_i(x)T_i(x)$$

definiendo $S(x) = \sum_{i=1}^{m} f_i(x)T_i(x)$ tenemos

$$H(x) = 2J^T(x)J(x) + 2S(x)$$

La actualización de Newton queda como

$$\left[J^T(x^{(k)})J(x^{(k)}) + S(x^{(k)})\right]\delta^{(k)} = -J^T(x^{(k)})f(x^{(k)})$$

$$x^{(k+1)} = x^{(k)} + \delta^{(k)}$$

4.7.1. Método de Gauss-Newton

Despreciando el termino $S(x^{(k)})$ de la actualización de Newton tenemos la definición del método de Newton

$$\left[J^T(x^{(k)})J(x^{(k)})\right]\delta^{(k)} = -J^T(x^{(k)})f(x^{(k)})$$

$$x^{(k+1)} = x^{(k)} + \delta^{(k)}$$

Lo que resulta interesante de este método, es que el hecho de despreciar la parte de segundas derivadas del Hessiano, tenemos una mejora en la convergencia del algoritmo ya que $J^T(x_k)J(x_k)$ es una matriz definida positiva. Para mostrar esto podemos hacer $y = J(x_k)z$ podemos ver que

$$z^T J^T(x)J(x)z = y^T y \geq 0$$

Algoritmo 18 Algoritmo de Gauss Newton

Entrada: Dado una función diferenciable $f(x)$ y un valor inicial $x^{(0)}$
Salida: $x^{(k)}$
  Hacer $k = 0$
  mientras $(\|\nabla f(x^{(k)})\| \geq \varepsilon)$ **hacer**
    $A(x^{(k)}) = J(x^{(k)})^T J(x^{(k)})$
    Resolver $A(x^{(k)})s^{(k)} = -\nabla f(x^{(k)})$
    $x^{(k+1)} = x^{(k)} + s^{(k)}$
    $k \leftarrow k + 1$
  fin mientras
  devolver $x^{(k)}$

4.7.2. Método de Levenberg-Marquardt

El método de Levenberg-Marquardt incorpora una técnica para resolver el problema relacionado con la singularidad de la matriz $J^T(x^{(k)})J(x^{(k)})$ y es un algoritmo efectivo para problema de residuos pequeños. La actualización en este caso esta dada por

$$\left[J^T(x^{(k)})J(x^{(k)}) + S(x^{(k)}) + \mu^{(k)}I\right]\delta^{(k)} = -J^T(x^{(k)})f(x^{(k)})$$

Una estrategia para seleccionar $\mu^{(k)}$, debe ser dada. El siguiente algoritmo proponer como realizar este procedimiento

Algoritmo 19 Algoritmo de Levenberg-Marquardt

Entrada: Dado una función diferenciable $f(x)$ y un valor inicial $x^{(0)}$
Salida: $x^{(k)}$
  Hacer $\mu^{(0)} = 0.001$ y ,$\nu = 10$
  repetir
    Poner $\mu^{(k)} = \frac{\mu^{(k)}}{\nu}$
    repetir
      Resolver $\left[J^T(x^{(k)})J(x^{(k)}) + S(x^{(k)}) + \mu^{(k)}I\right]\delta^{(k)} = J^T(x^{(k)})f(x^{(k)})$
      $x^{(k+1)} = x^{(k)} - \delta^{(k)}$
      $\mu^{(k)} = \mu^{(k)} \times \nu$
    hasta que $(F(x^{(k+1)}) < F(x^{(k)}))$
    Hacer $\mu^{(k+1)} = \mu^{(k)}$
    $k \leftarrow k + 1$
  hasta que $(\|\nabla f(x^{(k)})\| \leq \varepsilon)$
  devolver $x^{(k)}$

4.7.3. Ejemplo

En la tabla 4.11 se presentan los datos $u_k \rightarrow v_k$ correspondientes a los valores para ajustar la función $\phi(u) = x_1 Sin(x_2 u) + x_3$. El vector de parámetros $x = [x_1, x_2, x_3]^T$ se calcula mediante la minimización de la función $F(x)$

$$F(x) = \sum_{k=1}^{15} (x_1 Sin(x_2 u_k) + x_3 - v_k)^2$$

Tomando como valores iniciales $x^{(0)} = [1, 2, 1.3]^T$, calcular el vector parametros utilizando:

- El método de Gauss Newton
- El método de Newton y
- El método de Levenberg-Marquard

El k-esimo renglón de la función de error y del Jacobiano de nuestra función esta dado por

$$f_k(x) = x_1 sin(x_2 u_k) + x_3 - v_k$$

$$J_k(x) = [sen(x_2 u_k), u_k x_1 cos(x_2 u_k), 1]$$

Para el método de Gauss-Newton el gradiente esta dado por $g(x) = J(x)^T f(x)$ y el hessiano $H(x) = J(x)^T J(x)$

$$g(x) = \sum_{k=1}^{15} \begin{bmatrix} sen(x_2 u_k) \\ u_k x_1 cos(x_2 u_k) \\ 1 \end{bmatrix} (x_1 sin(x_2 u_k) + x_3 - v_k)$$

$$H(x) = \sum_{k=1}^{15} \begin{bmatrix} sen(x_2 u_k) \\ u_k x_1 cos(x_2 u_k) \\ 1 \end{bmatrix} \begin{bmatrix} sen(x_2 u_k) & u_k x_1 cos(x_2 u_k) & 1 \end{bmatrix}$$

Los términos de segundas derivadas son:

$$T_k(x) = \begin{bmatrix} 0 & u_k cos(x_2 u_k) & 0 \\ u_k cos(x_2 u_k) & -x_1 u_k^2 sin(x_2 u_k) & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Cuadro 4.11: Valores para ajuntar la función por mínimos cuadrados

k	u	v
1	0.1	4.74
2	0.2	5.12
3	0.3	5.39
4	0.4	5.49
5	0.5	5.43
6	0.6	5.21
7	0.7	4.85
8	0.8	4.42
9	0.9	3.97
10	1.0	3.56
11	1.1	3.26
12	1.2	3.11
13	1.3	3.13
14	1.4	3.31
15	1.5	3.63

y el Hessiano completo es

$$H(x) = \sum_{k=1}^{15} \left\{ \begin{bmatrix} sen(x_2 u_k) \\ u_k x_1 cos(x_2 u_k) \\ 1 \end{bmatrix} \begin{bmatrix} sen(x_2 u_k) & u_k x_1 cos(x_2 u_k) & 1 \end{bmatrix} + T_k(x) f_k(x) \right\}$$

La solución para el método de Gauss-Newton se presenta a continuaciónla función

$$\begin{bmatrix} x^{(0)} \\ x^{(1)} \\ x^{(2)} \\ x^{(3)} \\ x^{(4)} \\ x^{(5)} \\ x^{(6)} \\ x^{(7)} \end{bmatrix} = \begin{bmatrix} 1 & 2 & 1.3 \\ -0.93568 & 3.49736 & 5.39809 \\ 1.14539 & 3.08263 & 4.32988 \\ 0.807206 & 3.95417 & 4.54756 \\ 1.18849 & 3.72984 & 4.30198 \\ 1.19729 & 3.80013 & 4.29694 \\ 1.19974 & 3.79842 & 4.29545 \\ 1.19974 & 3.79842 & 4.29545 \end{bmatrix} \tag{4.34}$$

La Figura 4.27 muestra la gráfica correspondientes a los puntos de la tabla 4.11 y la función ajustada $\phi(u)$.

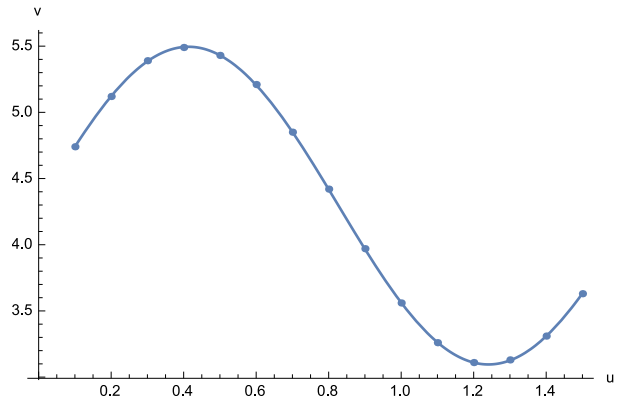


Figura 4.27: Función $\phi(u) = x_1 Sin(x_2 u) + x_3$.

N	Método	Iteraciones	f(x)	x1	x2	x3	tiempo (ms.)
1	Newton	127	11.84324	-9.10E-05	-1.00E-06	4.308005	373
2	Broyden	76	11.84324	7.00E-06	15.476981	4.308	89
3	BFGS	28	11.366203	0.263729	-6.508707	4.355711	75
4	LM	499	2.864619	-87.49974	0.020468	5.740672	113
5	Forsyte	7	1.03E-04	1.199743	3.798423	4.295445	201
6	DG	22	1.03E-04	1.199743	3.798423	4.295445	129
7	GC-FR	40	1.03E-04	1.199743	3.798423	4.295445	87
8	GC-HS	16	1.03E-04	1.199743	3.798423	4.295445	48
9	GC-PR	13	1.03E-04	1.199743	3.798423	4.295445	39
10	GN	6	1.03E-04	1.199742	3.798421	4.295446	26

Cuadro 4.12: Comparativo de Métodos para calcular el problema de mínimos cuadrados

La tabla 4.12 muestra el comparativo para 10 de los métodos vistos a lo largo del curso

Optimización con Restricciones

5.1. Teoría de la optimización con restricciones

La formulación general del problema, de optimización de una función con restricciones puede escribirse como:

$$\begin{aligned} &\text{mín} && f(x) && x \in \mathbb{R} \\ &s.a \\ &&& C_i(x) = 0 : i \in \mathcal{E} \\ &&& C_i(x) \geq 0 : i \in \mathcal{I} \end{aligned} \tag{5.35}$$

donde $f(x)$ es la función objetivo, C_i son las restricciones de igualdad y desigualdad, i es el índice en el conjunto de números los cuales indican la cantidad de restricciones de igualdad $\mathcal{E}$ y desigualdad $\mathcal{I}$, y x es un vector de variables.

La región de factibilidad es aquella que contiene elementos (x) tales que cumplen con las restricciones C_i. En un lenguaje matemático se pueden expresar de la siguiente manera.

$$\Omega = \{x | C_i(x) = 0, i \in \mathcal{E}; C_i(x) \geq 0, i \in \mathcal{I}\}$$

de manera compacta podemos escribir (5.35) como:

$$\min_{x \in \Omega} f(x)$$

5.2. Una simple restricción de desigualdad

En los problemas de optimización sin restricciones, el problema de minimización es sencillo, solo deben encontrar los ceros del vector gradiente $\nabla f(x) = 0$ tal que la matriz

Hessina $H(x)$ sea definida positiva. El problema de minimización restringida es un poco más complejo, e iniciaremos por analizar dicho problema considerando restricciones de igualdad.

A manera de introducción consideraremos el siguiente ejemplo;

$$\begin{aligned} min\ f(x) &= x_1 + x_2 \\ s.a &\quad x_1^2 + x_2^2 - 2 = 0 \end{aligned}$$

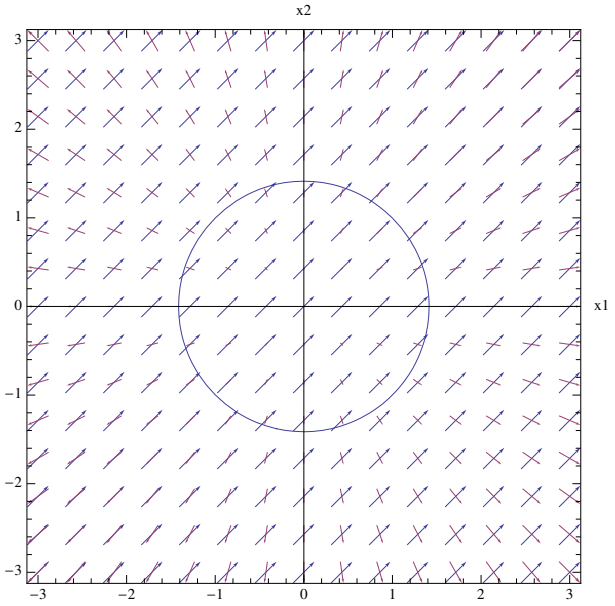


Figura 5.28: Función lineal con restricción cuadrática

Se puede observar que $f(x)$ es un plano recto, por lo tanto su mínimo será $x_{min} = \infty$. Sin embargo, al considerar la restricción, la cual es un circulo con centro en el origen y radio $\sqrt{2}$, el mínimo para este problema ya no se encuentra en $x_{min} = \infty$. Entonces, la solución del problema estará acota a valores de x que pertenezcan a la única restricción $C_1 = x_1^2 + x_2^2 - 2$ (perímetro de la circunferencia). La solución obvia es x^* es $[-1, -1]$, ya que este valor es el único que cumple con C_1 y a la vez no existe otro valor sobre $f(x)$ que minimice $f(x)$ y satisfaga $C(x)$.

De la figura 5.28 se puede observar que existen un número infinito de valores de x tales que $C(x) = 0$, sin embargo estos no minimizan $f(x)$, por lo tanto no son mejores que $[-1, -1]$.

Ahora, observemos como se encuentran direccionados los gradientes de $f(x)$ y $c(x)$; el gradiente de la función es un vector $\nabla f(x) = [1, 1]^T$ y el gradiente de la única restricción

es $\nabla C(x) = [2x_1, 2x_2]^T$

De la figura 5.28, se puede observar que x^* se encuentra en el punto en el que el gradiente de la función ∇f y el gradiente de las restricciones ∇C están alineados, pero con sentidos opuesta. Observe que en el óptimo $x^* = [-1, -1]^T$, la dirección del gradiente $\nabla f(x^*) = [1, 1]^T$ tiene la misma dirección que el vector gradiente de la restricciones $\nabla C(x^*) = [-2, -2]^T$, es decir:

$$\nabla f(x^*) = \lambda_1 \nabla C(x^*) \tag{5.36}$$

donde λ_1 es una constante de proporcionalidad. De esta ecuaciones podemos concluir que el gradiente de la función y el gradiente de las restricciones son vectores paralelos. En nuestro caso el valor de λ_1, puede ser calculado sustituyendo los valores del gradiente de la función y el gradiente de la restricción

$$\begin{aligned} \begin{bmatrix} 1 \\ 1 \end{bmatrix} &= \lambda \begin{bmatrix} -2 \\ -2 \end{bmatrix} \\ \lambda &= -\frac{1}{2} \end{aligned}$$

lo cual da como resultado $\lambda = -\frac{1}{2}$. Otro punto que cumple con dicha condición es $x = [1, 1]$, sin embargo es obvio que este punto no minimiza $f(x)$. Es necesario mostrar de manera formal este hecho

Otro ejemplo que podemos ver es el caso de la siguiente función cuadrática con una restricción lineal.

$$\begin{aligned} min\ f(x) &= 2x_1^2 + 3x_2^2 \\ s.a &\quad 3x_1 + x_2 - 3 = 0 \end{aligned}$$

El mínimo en este caso es $f(x^*) = 1.86207$ con $x^* = [0.931034, 0.206897]$, podemos notar que en el óptimo los gradientes de la función y la restricción tienen la misma dirección tal como se muestra en la figura 5.29.

El gradiente de la función valuado en el óptimo es

$$\nabla f(x^*) = \begin{bmatrix} 4x_1^* \\ 6x_2^* \end{bmatrix} = \begin{bmatrix} 4 \times 0.931034 \\ 6 \times 0.206897 \end{bmatrix}$$

El gradiente de la restricción queda como

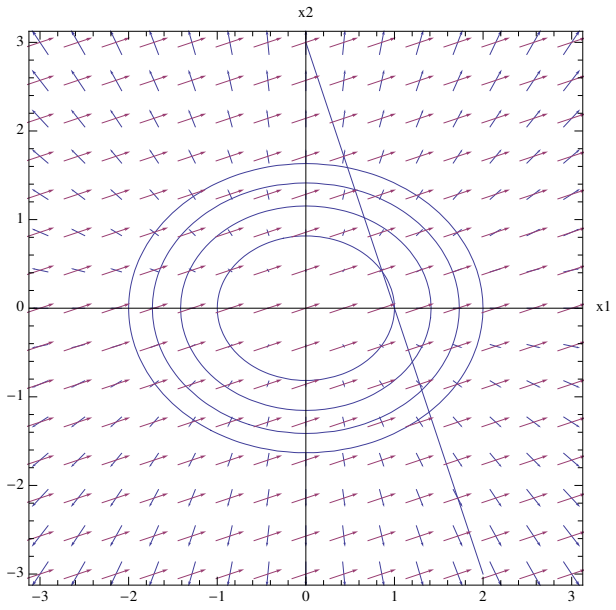


Figura 5.29: Función cuadrática con restricción lineal

$$\nabla C(x^*) = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

Con esto podemos notar que $\nabla f(x^*) = \lambda \nabla C(x^*)$ con $\lambda = 1.241382$

$$\begin{bmatrix} 3.724136 \\ 1.241382 \end{bmatrix} = 1.241382 \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

Un último ejemplo que podemos ver es el caso de de una función cuadrática con una restricción cuadratica dada por.

$$\begin{aligned} min\ f(x) &= 2x_1^2 + 3x_2^2 \\ s.a &\quad x_1^2 + x_2^2 = 3 \end{aligned}$$

Para este caso tenemos 2 mínimos localizados en $[\sqrt{3}, 0]$ y $[-\sqrt{3}, 0]$ con $f(x^*) = 6$, en la figura 5.30 podemos ver como en este punto los gradientes de la restricción y de la función son paralelos.

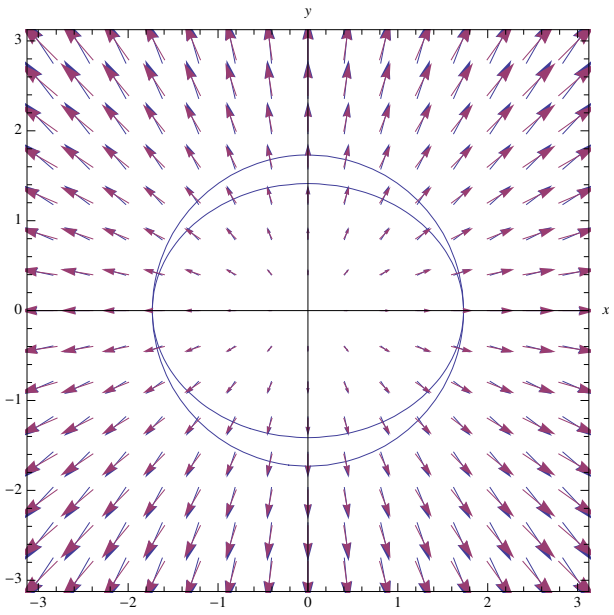


Figura 5.30: Función cuadrática con restricción cuadrática

El gradiente de la función valuado en el óptimo es

$$\nabla f(x^*) = \begin{bmatrix} 4x_1^* \\ 6x_2^* \end{bmatrix} = \begin{bmatrix} 4 \times \sqrt{3} \\ 6 \times 0 \end{bmatrix}$$

El gradiente de la restricción queda como

$$\nabla C(x^*) = \begin{bmatrix} 2x_1^* \\ 2x_2^* \end{bmatrix} = \begin{bmatrix} 2 \times \sqrt{3} \\ 2 \times 0 \end{bmatrix}$$

Con esto podemos notar que $\nabla f(x^*) = \lambda \nabla C(x^*)$ con $\lambda = 2$

$$\begin{bmatrix} 4\sqrt{3} \\ 0 \end{bmatrix} = 2 \begin{bmatrix} 2\sqrt{3} \\ 0 \end{bmatrix}$$

5.3. Multiplicadores de Lagrange

El método de los multiplicadores de Lagrange es una herramienta útil para encontrar el mínimo o máximo, para una función $f(x)$ dada, sujeta a una a una restricción $C(x) = 0$. El

método puede ser también usado en casos de mas de dos variables o mas de una restricción. Note que solamente para este método se aplican las restricciones de igualdad.

Restricción de Igualdad

La forma general del problema es

$$\begin{aligned} \text{mín} \quad & f(x) \\ s.a \quad & \\ & c_i(x) = 0 \end{aligned}$$

El método de multiplicadores de Lagrange se basa en una simple observación de que el gradiente de la restricción es paralelo al gradiente de la función, en el punto de solución eq. (5.35).

Prueba

Podemos derivar la expresión (5.36) aplicando la Serie de Taylor sobre la restricción de la función objetivo. Para mantener la restricción $c_1(x)$, si minimizamos en una dirección d, cualquiera se debe mantener que $c_1(x+d) = 0$ esto significa que

$$\begin{aligned} 0 &= c_1(x+d) \approx c_1(x) + \nabla c_1(x)^T d \\ 0 &= \nabla c_1(x)^T d \end{aligned} \tag{5.37}$$

Similarmente, una dirección apropiada que produzca un decrecimiento en $f(x)$, es

$$\begin{aligned} 0 &\geq f(x+d) - f(x) \\ 0 &\geq f(x) + \nabla f(x)^T d - f(x) \end{aligned}$$

con lo cual tenemos

$$\nabla f(x)^T d \leq 0 \tag{5.38}$$

Existe un sinfín de números que cumplen con ambas restricciones (5.37) y (5.38) y solamente existe una dirección que no satisface ambas restricciones. En el mínimo tenemos que $\nabla_x f(x)^T d = 0$ y debe de cumplirse que $\nabla_x c_1(x)^T d = 0$, por lo tanto en el mínimo el gradiente de la función debe ser paralelo al gradiente de la restricción

$$\nabla_x f(x) = \lambda \nabla_x C(x) \tag{5.39}$$

La ecuación (5.39) indica que si los ∇f y ∇C están alineados en un punto x, x es x^*, recuerde que esta condición es necesaria pero no suficiente, λ es denominado Multiplicador de Lagrange. La ecuación (5.39) da lugar a la llamada función Lagrangiana;

$$\mathcal{L}(x, \lambda) = f(x) - \lambda C(x) \tag{5.40}$$

de la ecuación (5.40) podemos ver que el gradiente de esta $\nabla\mathcal{L}(x, \lambda)$ debe ser igual a cero para cumplir con la ecuación (5.39). Con esto tenemos que el gradiente de la función lagrangiana (5.40) como:

$$\begin{bmatrix} \nabla_x \mathcal{L}(x, \lambda) \\ \nabla_\lambda \mathcal{L}(x, \lambda) \end{bmatrix} = \begin{bmatrix} \nabla_x f(x) - \lambda \nabla_x C(x) \\ C(x) \end{bmatrix} = 0 \tag{5.41}$$

5.3.1. Ejemplo 1

En este ejemplo se muestra una función de dos variables con una sencilla restricción de igualdad. En la figura 5.28 se muestran los vectores gradiente de la restricción y de la función.

$$\begin{aligned} \text{mín} \quad & f(x) = x_1 + x_2 \\ s.a. \quad & x_1^2 + x_2^2 - 2 = 0 \end{aligned}$$

La función Lagrangiana para este ejemplo es

$$\mathcal{L}(x, \lambda) = x_1 + x_2 - \lambda(x_1^2 + x_2^2 - 2)$$

y el gradiente del Lagrangiano es

$$\nabla\mathcal{L}(x, \lambda) = \left[\begin{array}{c} \nabla\mathcal{L}_x(x, \lambda) \\ \hline \nabla\mathcal{L}_\lambda(x, \lambda) \end{array}\right] = \left[\begin{array}{c} 1 - 2\lambda x_1 \\ 1 - 2\lambda x_2 \\ \hline x_1^2 + x_2^2 - 2 \end{array}\right] = \left[\begin{array}{c} 0 \\ 0 \\ \hline 0 \end{array}\right]$$

Para resolver multiplicamos $(1 - 2\lambda x_1) \times x_2 = 0$ y $(1 - 2\lambda x_2) \times x_1 =$ y sumamos las ecuaciones resultantes

$$x_2 - 2\lambda x_1 x_2 = x_1 - 2\lambda x_1 x_2 = 0$$

con lo cual obtenemos que $x_1 = x_2$, sustituyendo este valor tenemos

$$\begin{aligned} x_1^2 + x_2^2 &= 2 \\ x_1^2 + x_1^2 &= 2 \\ 2x_1^2 &= 2 \\ x_1 &= \pm 1 \end{aligned}$$

La solución del sistema es $x_1 = -1$ y $x_2 = -1$ dado que la otra solución es un máximo. Sustituyendo esto en cualquiera de los las ecuaciones restantes tendremos que $\lambda = 0.5$

La solución X^* de acuerdo con la figura 5.28 es $[-1, 1]^T$ con $\lambda = 0.5$ y podemos verificar que para estos valores $\nabla\mathcal{L} = 0$

Sin embargo cuando la solución se calcula minimizando la función Lagrangiana $\mathcal{L}(x, \lambda)$, resulta que no tiene solución por alguno, de los métodos basados en gradiente. Esto se debe a que la convexidad de la función no se puede garantizar para diferentes valores de λ.

Una alternativa para solucionar el problema es suponer que el valor de λ es fijo en un valor tal que garantice la convexidad de la función o utilizar funciones de penalización.

5.3.2. Ejemplo 2

Considere la función dada como:

$$\begin{aligned} min\ f(x) &= 2x_1^2 + 3x_2^2 \\ s.a &\quad 3x_1 + x_2 - 3 = 0 \end{aligned}$$

El Lagrangiano de esta función es:

$$\mathcal{L}(x, \lambda) = 2x_1^2 + 3x_2^2 - \lambda(3x_1 + x_2 - 3)$$

El gradiente de la función Lagrangiana es:

$$\nabla\mathcal{L}(x, \lambda) = \begin{bmatrix} \nabla\mathcal{L}_x(x, \lambda) \\ \hline \nabla\mathcal{L}_\lambda(x, \lambda) \end{bmatrix} = \begin{bmatrix} 4x_1 - 3\lambda \\ 6x_2 - \lambda \\ \hline 3x_1 + x_2 - 3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \hline 0 \end{bmatrix}$$

Para calcular la solución se plantea el siguiente sistema de ecuaciones:

$$\begin{bmatrix} 4 & 0 & -3 \\ 0 & 6 & -1 \\ 3 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 3 \end{bmatrix}$$

La solución del sistema de ecuaciones es $x_1 = 27/29$, $x_2 = 6/29$ y $\lambda = 36/29$

5.3.3. Ejemplo 3

Consideremos el ejemplo de la función dada por

$$\begin{array}{ll} \text{mín} & f(x) = 2x_1^2 + 3x_2^2 \\ s.a. & x_1^2 + x_2^2 = 3 \end{array}$$

El Lagrangiano de la función esta dado por

$$\mathcal{L}(x, \lambda) = x_1^2 + x_2^2 - \lambda(x_1^2 + x_2^2 - 3)$$

El gradiente de la función Lagrangiana es:

$$\nabla\mathcal{L}(x, \lambda) = \left[\begin{array}{c} \nabla\mathcal{L}_x(x, \lambda) \\ \hline \nabla\mathcal{L}_\lambda(x, \lambda) \end{array}\right] = \left[\begin{array}{c} 4x_1 - 2\lambda x_1 \\ 6x_2 - 2\lambda x_2 \\ \hline x_1^2 + x_2^2 - 3 \end{array}\right] = \left[\begin{array}{c} 0 \\ 0 \\ \hline 0 \end{array}\right]$$

Haciendo arreglos a las ecuaciones tenemos

$$\begin{array}{r} (4x_1 - 2\lambda x_1) \times x_2 = 0 \\ - \\ (6x_2 - 2\lambda x_2) \times x_1 = 0 \end{array}$$

Con lo cual tenemos que $-2x_1x_2 = 0$ si multiplicamos por x_2 y sustituimos $x_2^2 = 3 - x_1^2$ tenemos

$$\begin{aligned} (2x_1x_2) \times x_2 &= 0 \\ 2x_1x_2^2 &= 0 \\ 2x_1(3 - x_1^2) &= 0 \end{aligned}$$

Podemos notar que esta ecuación tiene tres raíces $x_1 = 0$, $x_1 = \sqrt{3}$ y $x_1 = -\sqrt{3}$, con lo cual tenemos cuatro soluciones que satisfacen el gradiente del Laplaciano

$$x_1 = 0, x_2 = -\sqrt{3}, \lambda = 3, f(x) = 9$$
$$x_1 = 0, x_2 = +\sqrt{3}, \lambda = 3, f(x) = 9$$
$$x_1 = -\sqrt{3}, x_2 = 0, \lambda = 2, f(x) = 6$$
$$x_1 = \sqrt{3}, x_2 = 0, \lambda = 2, f(x) = 6$$

pero solamente las dos últimas son mínimos.

5.4. Restricciones de desigualdad

Consideremos una simple modificación a los ejemplos anteriores donde ya no solamente tenemos una restricción de igualdad. En este caso queremos minimizar

$$\begin{aligned} f(x) = \quad & x_1 + x_2 \\ s.a \quad & \\ & 2 - x_1^2 - x_2^2 \geq 0 \end{aligned}$$

la región de factibilidad es el interior de un circulo de radio $\sqrt{2}$, de acuerdo como se muestra en la figura 5.28. Para determinar la solución tenemos que cumplir la restricción, cualquier movimiento en dirección d que hagamos debe cumplir también con esta restricción

$$0 \leq c_1(x + d) \approx c_1(x) + \nabla c_1(x)^T d$$

por lo que la factibilidad de primer orden se mantiene si

$$c_1(x) + \nabla c_1(x)^T d \geq 0 \tag{5.42}$$

En este caso también deben existir una dirección que satisfaga esta restricción y $\nabla f(x)^T d \leq 0$. Consideremos los siguientes casos

Caso 1: Considere primero el caso en el cual x vive dentro de la circunferencia, por lo cual se da de manera estricta la restricción $c_1(x) > 0$. En este caso, algún vector d satisface la condición (5.42), siempre que la longitud sea pequeña. En particular, siempre que $\nabla f(x^*) \neq 0$, podemos obtener una dirección d que satisface ambas restricciones (5.42) y (5.38) haciendo

$$d = -c_1(x) \frac{\nabla f(x)}{\|\nabla f(x)\|}$$

Caso 2: Consideremos ahora el caso en el cual x vive en la frontera de el circulo, por lo cual $c_1(x) = 0$. Las condiciones entonces son

$$\begin{aligned} \nabla f(x)^T d &< 0 \\ \nabla c_1(x)^T d &\geq 0 \end{aligned}$$

las cuales son posibles cuando los vectores gradientes son paralelos, así que

$$\nabla f(x)^T d = \lambda_1 \nabla c_1(x)$$

para algún $\lambda_1 > 0$

Las condiciones de optimalidad para ambos casos pueden ser sumarizados con la función Lagrangiana. Así tenemos

$$\nabla_x \mathcal{L}(x^*, \lambda_1^*) = 0$$

para algún $\lambda_1 > 0$ y también requeriremos que

$$\lambda_1^* c_1(x^*) = 0$$

Esta condición es conocida como la condición complementaria. Note que en el caso I, cuando $c_1(x^*) > 0$ requiere que $\lambda_1^* = 0$, de ahí que $\nabla f(x^*) = 0$. En el caso II tenemos que λ_1^* tenga un valor positivo con lo cual se cumple que $\nabla f(x)^T d = \lambda_1 \nabla c_1(x)$.

5.5. Condiciones Necesarias de Primer Orden

Suponga que x^* es una solución de una función con restricciones. Entonces existe un multiplicador de Lagrange λ^*, con componentes $\lambda_i^*, i \in \mathcal{E} \bigcup \mathcal{I}$, tal que las siguientes condiciones son satisfechas en el punto (x^*, λ^*)

$$\begin{aligned} \nabla_x \mathcal{L}(x^*, \lambda_1^*) &= 0 \\ c_i(x^*) &= 0 \text{ para todo } i \in \mathcal{E} \\ c_i(x^*) &\geq 0 \text{ para todo } i \in \mathcal{I} \\ \lambda_i^* &\geq 0 \text{ para todo } i \in \mathcal{E} \\ \lambda_i^* c_i(x^*) &= 0 \text{ para todo } i \in \mathcal{E} \bigcup \mathcal{I} \end{aligned}$$

Estas son conocidas como las condiciones de Karush-Kuhn-Tucker o KKT.

5.6. Programación Cuadrática

A los problemas de optimización, cuya función objetivo es cuadrática y sus restricciones son lineales, se le denominan problema de *Programación Cuadrática (QP)*. La formulación general del problema es el siguiente;

$$\text{mín } q(x) = \tfrac{1}{2}x^T Hx + x^T d \tag{5.43}$$

$s.a$

$$a_i^T x = b_i,\ i \in \varepsilon \tag{5.44}$$

$$a_i^T x \geq b_i,\ i \in \mathrm{I} \tag{5.45}$$

donde H es una matriz Hessiana simétrica definida positiva de tamaño $n \times n$, y $\mathcal{E}$ e $\mathcal{I}$ son conjuntos de indices correspondientes a las restricciones de igualdad y desigualdad, respectivamente.

Los problemas de QP pueden ser resueltos, o puede mostrarse, que no son factibles, en un número finitos de iteraciones, el esfuerzo de solución depende de las características de la función objetivo y del número de restricciones de desigualdad.

Si H es positiva entonces el problema es denominado QP convexa

5.6.1. Restricciones de igualdad QP

Consideramos que el número de restricciones de igualdad es m, y que $m \geq 0$, escribimos el problema de QP como;

$$\text{mín} \quad q(x) \overset{def}{=} \frac{1}{2}x^T Hx + x^T d \tag{5.46}$$

$s.a.$

$$Ax = b$$

donde A es de $m \times n$ y es el Jacobiano de las restricciones de igualdad, definido por:

$$A = [a_i] \qquad \forall i \in \mathcal{E} \tag{5.47}$$

Consideraremos, que A es de rango completo (rango = m) y que las restricciones son consistentes. La solución de (5.46) requiere del cumplimiento de las condiciones de KKT, las cuales son:

$$\begin{gathered} \nabla_x \mathcal{L}(x^*, \lambda^*) = 0 \\ C_i(x) = 0 \end{gathered} \tag{5.48}$$

Para expresar completamente la ecuación (5.46) en términos de KKT, formulamos la $\mathcal{L}$ del problema;

$$\mathcal{L}(x,\lambda) = \frac{1}{2}x^T Hx + x^T d - \lambda^T(Ax - b) \tag{5.49}$$

donde el $\nabla\mathcal{L}$ es;

$$\nabla_x \mathcal{L}(x^*,\lambda^*) = Hx^* + d - A^T\lambda^* = 0 \tag{5.50}$$

Además, las restricciones de igualdad es la ecuación (5.46) en términos de KKT, son:

$$c(x^*) = Ax^* - b = 0 \tag{5.51}$$

Por medio de (5.50) y (5.51) hemos expresado las condiciones de KKT y en x^* estas deben ser satisfecha simultáneamente, de tal manera que las podemos expresar como sigue.

$$\begin{bmatrix} H & -A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} -d \\ b \end{bmatrix} \tag{5.52}$$

Entonces el problema ha sido reducido a la solución de (5.52). La ecuación (5.52) pude reescribirla, para que esta sea usada para cuestiones computacionales. Se puede expresar x^* como;

$$x^* = x + p \tag{5.53}$$

donde x es un valor estimado de la solución y p es el incremento deseado. Sustituyendo (5.53) en (5.50) y (5.51) obtenemos:

$$\begin{bmatrix} H & -A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} p \\ \lambda^* \end{bmatrix} = \begin{bmatrix} -Hx - d \\ b - Ax \end{bmatrix} \Rightarrow \begin{bmatrix} H & -A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} p \\ \lambda^* \end{bmatrix} = \begin{bmatrix} -\nabla f(x^{(0)}) \\ -c(x^{(0)}) \end{bmatrix} \tag{5.54}$$

donde $\nabla f(x^{(0)})$ es el gradiente de la funcion $f(x)$ y $c(x^{(0)}) = Ax^{(0)} - b$ valuados en el punto inicial $x^{(0)}$

Entonces el problema se ha reducido a solucionar la ecuación (5.46), (5.52) o (5.54) y la matriz $\left|\begin{matrix} H & -A^T \\ A & 0 \end{matrix}\right|$ es denominada Matriz de Karush - Kum - Tucker.

5.6.2. Ejemplo 1

Dada la función con restricciones:

$$f(x) = 2x_1^2 + 3x_2^2$$
$$\text{sujeto a}$$
$$3x_1 + x_2 = 3$$

y un punto inicial $x^{(0)} = [1,2]^T$ determinar:

1. El mínimo de la función sin restricciones,
2. La región de factibilidad,
3. la solución del sistema utilizando la formulación de Programación Cuadrática y
4. si la solución cumple con las condiciones de KKT

Solución sin restricciones

El gradiente de la función es:

$$\nabla f(x) = \begin{bmatrix} 4x_1 \\ 6x_2 \end{bmatrix}$$

y Hessiano

$$H = \begin{bmatrix} 4 & 0 \\ 0 & 6 \end{bmatrix}$$

En este caso la solución es simplemente el método de Newton. Considerando un valor inicial $x^{(0)} = [1, 2]^T$ el gradiente es $\nabla f(x^{(0)}) = [4, 12]^T$.

$$Hp = -\nabla f(x^{(0)})$$

$$\begin{bmatrix} 4 & 0 \\ 0 & 6 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} -4 \\ -12 \end{bmatrix}$$

$$\begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} -1 \\ -2 \end{bmatrix}$$

$$x^* = x^{(0)} + p = \begin{bmatrix} 1 \\ 2 \end{bmatrix} + \begin{bmatrix} -1 \\ -2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Región de Factibilidad

En la Figura 5.31 se muestra la región de factibilidad en azul. La línea negra muestra el vector diferencia entre el punto inicial y la solución sin restricciones.

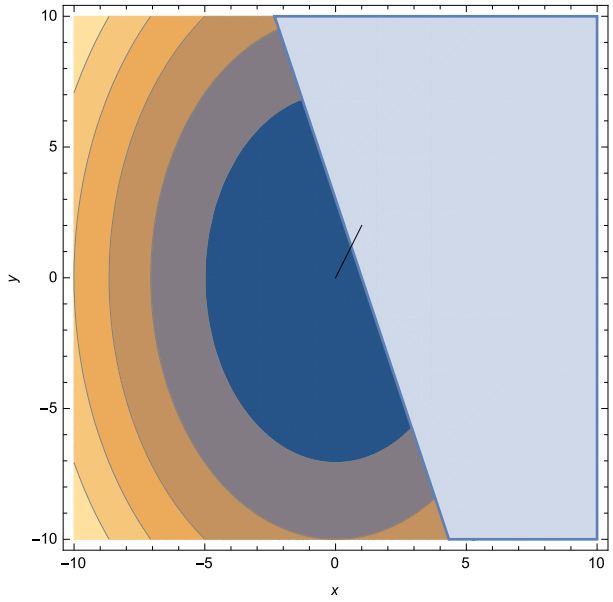


Figura 5.31: Región de factibilidad para el ejemplo 1.

La solución utilizando programación cuadrática

Considerando un valor inicial $x^{(0)} = [1, 2]^T$ el gradiente es $\nabla f(x^{(0)}) = [-4, -12]^T$ y $c(x^{(0)}) = Ax^{(0)} - b = 2$. Por lo tanto la formulación queda:

$$\begin{bmatrix} H & -A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} p \\ \lambda^* \end{bmatrix} = \begin{bmatrix} -\nabla f(x^{(0)}) \\ -c(x^{(0)}) \end{bmatrix}$$

$$\left[\begin{array}{cc|c} 4 & 0 & -3 \\ 0 & 6 & -1 \\ \hline 3 & 1 & 0 \end{array}\right] \left[\begin{array}{c} p_1 \\ p_2 \\ \hline \lambda^* \end{array}\right] = \left[\begin{array}{c} -4 \\ -12 \\ \hline -2 \end{array}\right]$$

La solución del sistema es:

$$\left[\begin{array}{c} x_1 \\ x_2 \\ \hline \lambda^* \end{array}\right] = \left[\begin{array}{c} x_1^{(0)} \\ x_2^{(0)} \\ \hline 0 \end{array}\right] + \left[\begin{array}{c} p_1 \\ p_2 \\ \hline \lambda^* \end{array}\right]$$

$$\left[\begin{array}{c} x_1 \\ x_2 \\ \hline \lambda^* \end{array}\right] = \left[\begin{array}{c} 1 \\ 2 \\ \hline 0 \end{array}\right] + \left[\begin{array}{c} -2/29 \\ -52/29 \\ \hline 36/29 \end{array}\right] = \left[\begin{array}{c} 27/29 \\ 6/29 \\ \hline 36/29 \end{array}\right]$$

Condiciones de KKT

1. Gradiente del Laplaciano
$$\nabla_x \mathcal{L}(x^*, \lambda) = Hx^* + d - A^T\lambda = \nabla f(x^*) - A^T\lambda = 0$$
$$\nabla_x \mathcal{L}(x^*, \lambda) = \begin{bmatrix} 4 \times 27/29 \\ 6 \times 6/29 \end{bmatrix} - \begin{bmatrix} 3 \\ 1 \end{bmatrix} \times 36/29 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$
2. Cumplimento de las Igualdades
$$c(x^*) = Ax^* - b = 3 \times 27/29 + 6/29 - 3 = 0$$
3. Cumplimiento de Desigualdades

   No existen desigualdades en el planteamiento
4. Multiplicadores de Lagrange positivos

   El multiplicador de Lagrange es $\lambda^* = 36/29 > 0$, por lo cual si se cumple
5. Condición Complementaria

   Si se cumple la condición complementaria dado que $c(x^*)\lambda^* = 0 \times 36/29 = 0$

5.6.3. Ejemplo 2

Dada la función con restricciones:

$$f(x) = 4x_1^2 + 2x_1x_2 + x_2^2 + x_1 + 3x_2$$

sujeto a

$$-x_1/4 - x_2 \geq 15/2$$

$$-3x_1/2 - x_2 \geq 10$$

y un punto inicial $x^{(0)} = [-10, -10]^T$ determinar:

1. El mínimo de la función sin restricciones,
2. La región de factibilidad,
3. la solución del sistema utilizando la formulación de Programación Cuadrática y
4. si la solución cumple con las condiciones de KKT

Solución sin restricciones

El gradiente de la función es:

$$\nabla f(x) = \begin{bmatrix} 8x_1 + 2x_2 + 1 \\ 2x_1 + 2x_2 + 3 \end{bmatrix}$$

y Hessiano

$$H = \begin{bmatrix} 8 & 2 \\ 2 & 2 \end{bmatrix}$$

En este caso la solución es simplemente el método de Newton. Considerando un valor inicial $x^{(0)} = [-10, -10]^T$ el gradiente es $\nabla f(x^{(0)}) = [-99, -37]^T$.

$$Hp = -\nabla f(x^{(0)})$$

$$\begin{bmatrix} 8 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} 99 \\ 37 \end{bmatrix}$$

$$\begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} 31/3 \\ 49/6 \end{bmatrix}$$

$$x^* = x^{(0)} + p = \begin{bmatrix} -10 \\ -10 \end{bmatrix} + \begin{bmatrix} 31/3 \\ 49/6 \end{bmatrix} = \begin{bmatrix} 1/3 \\ -11/6 \end{bmatrix}$$

Región de Factibilidad

En la Figura 5.32 se muestra la región de factibilidad en azul. La línea negra muestra el vector diferencia entre el punto inicial y la solución sin restricciones.

La solución utilizando programación cuadrática

Para resolver consideraremos que las restricciones de desigualdad son igualdades. Considerando un valor inicial $x^{(0)} = [-10, -10]^T$ el gradiente es $\nabla f(x^{(0)}) = [-99, -37]^T$ y

$$c(x^{(0)}) = Ax^{(0)} - b = \begin{bmatrix} -1/4 & -1 \\ -3/2 & -1 \end{bmatrix} \begin{bmatrix} -10 \\ -10 \end{bmatrix} - \begin{bmatrix} 15/2 \\ 10 \end{bmatrix} = \begin{bmatrix} 5 \\ 15 \end{bmatrix}$$

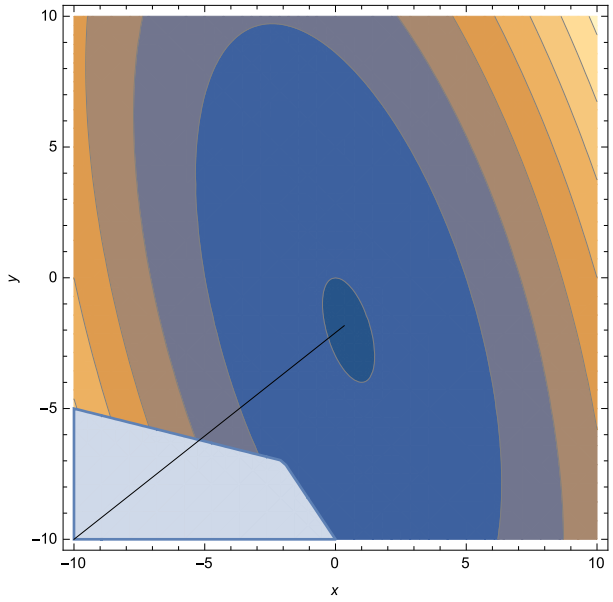


Figura 5.32: Región de factibilidad para el ejemplo 2.

por lo tanto la formulación queda:

$$\begin{bmatrix} H & -A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} p \\ \lambda^* \end{bmatrix} = \begin{bmatrix} -\nabla f(x^{(0)}) \\ -c(x^{(0)}) \end{bmatrix}$$

$$\left[\begin{array}{cc|cc} 8 & 2 & 1/4 & 3/2 \\ 2 & 2 & 1 & 1 \\ \hline -1/4 & -1 & 0 & 0 \\ -3/2 & -1 & 0 & 0 \end{array}\right] \left[\begin{array}{c} p_1 \\ p_2 \\ \hline \lambda_1^* \\ \lambda_2^* \end{array}\right] = \left[\begin{array}{c} 99 \\ 37 \\ \hline -5 \\ -15 \end{array}\right]$$

La solución del sistema es $[p|\lambda] = [8, 3| - 26/5, 101/5]^T$ y el óptimo x^* es:

$$\left[\begin{array}{c} x_1 \\ x_2 \\ \hline \lambda_1^* \\ \lambda_2^* \end{array}\right] = \left[\begin{array}{c} x_1^{(0)} \\ x_2^{(0)} \\ \hline 0 \\ 0 \end{array}\right] + \left[\begin{array}{c} p_1 \\ p_2 \\ \hline \lambda_1^* \\ \lambda_2^* \end{array}\right]$$

$$\left[\begin{array}{c} x_1 \\ x_2 \\ \hline \lambda_1^* \\ \lambda_2^* \end{array}\right] = \left[\begin{array}{c} -10 \\ -10 \\ \hline 0 \\ 0 \end{array}\right] + \left[\begin{array}{c} 8 \\ 3 \\ \hline -26/5 \\ 101/5 \end{array}\right] = \left[\begin{array}{c} -2 \\ -7 \\ \hline -26/5 \\ 101/5 \end{array}\right]$$

Condiciones de KKT

1. Gradiente del Laplaciano

$$\nabla_x \mathcal{L}(x^*, \lambda) = \nabla f(x^*) - A^T \lambda = 0$$

$$\nabla_x \mathcal{L}(x^*, \lambda) = \left(\begin{bmatrix} 8 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} -2 \\ -7 \end{bmatrix} + \begin{bmatrix} 1 \\ 3 \end{bmatrix} \right) - \begin{bmatrix} -1/4 & -3/2 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} -26/5 \\ 101/5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

2. Cumplimento de las Igualdades

$$c(x^*) = Ax^* - b = \begin{bmatrix} -1/4 & -1 \\ -3/2 & -1 \end{bmatrix} \begin{bmatrix} -2 \\ 7 \end{bmatrix} - \begin{bmatrix} 15/2 \\ 10 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

3. Cumplimiento de Desigualdades

   No existen desigualdades en el planteamiento

4. Multiplicadores de Lagrange positivos

   El multiplicador de Lagrange es $\lambda_1^* = -26/5 \not> 0$ y $\lambda_2^* = 101/5$ por lo cual no se cumple esta condición

5. Condición Complementaria

   Si se cumple la condición complementaria dado que $c(x^*)^T \lambda^* = 0$

En conclusión el problema esta mal planteado en los términos dados. La solución es sacar una de las restricciones y ver si el óptimo cumple con todas las condiciones de KKT.

5.7. Conjunto Activo

El problema a minimizar es el siguiente;

$$\begin{aligned} \text{mín } q(x) &= \frac{1}{2} x^T H x + x^T d \\ s.a. &: a_i^T x = b \end{aligned} \tag{5.55}$$

El método comienza por hacer una estimación $x^{(0)}$ factible, y después se aseguran de que todas las subsecuentes iteraciones sigan siendo factibles. Se encuentra un paso a partir de la primera iteración resolviendo un sub-problema cuadrático en el cual un subconjunto (5.44),

(5.45) se impongan como igualdades. Este subconjunto es nuestro *conjunto de trabajo* y es denotado como $W^{(k)}$ en la iteración k. Este consiste en todas las igualdades existentes en $\mathcal{E}$ (5.44) junto con algunos pero no necesariamente todas las desigualdades.

Dada un $x^{(k)}$ en una iteración y un conjunto de trabajo $W^{(k)}$, primero observamos si el $x^{(k)}$ minimiza a $q(x)$ en el subespacio definido por el conjunto de trabajo. Si no, calculamos un p_k resolviendo un problema QP.

Para resolver (5.55) planteamos el Gradiente del Laplaciano de $q(x)$ como:

$$\nabla\mathcal{L}(x,\lambda) = Hx + d - A^T\lambda = 0$$

si sustituimos $x = x^{(k)} + p(k)$ encontramos que;

$$\begin{aligned} H(x^{(k)} + p^{(k)}) + d - A^T\lambda = 0 \\ Hp^{(k)} - A^T\lambda = -Hx^{(k)} - d = -\nabla f(x^{(k)}) \end{aligned}$$

en el caso de las restricciones de igualdad tenemos

$$\begin{aligned} A(x^{(k)} + p^{(k)}) = b \\ Ax^{(k)} + Ap^{(k)} = b \\ Ap^{(k)} = 0 \end{aligned}$$

Las ecuaciones anteriores la podemos escribir como

$$\begin{bmatrix} H & -A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} p^{(k)} \\ \lambda \end{bmatrix} = \begin{bmatrix} -Hx^{(k)} - d \\ 0 \end{bmatrix} = \begin{bmatrix} -\nabla f(x^{(k)}) \\ 0 \end{bmatrix} \tag{5.56}$$

donde $\nabla f(x^{(k)}) = Hx^{(k)} + d$. Las actualizaciones de x las llevaremos a cabo mediante

$$x^{(k+1)} = x^{(k)} + p^{(k)}$$

Suponga que tenemos un $p^{(k)}$ que es diferente de cero. Necesitamos decidir como movernos a lo largo de esta dirección. Si $x^{(k)} + p^{(k)}$ es factible con respecto a todas las restricciones, nuestro conjunto $x^{(k+1)} = x^{(k)} + p^{(k)}$. Sino ocurre esto fijamos.

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)}p^{(k)}$$

Donde el valor del parámetro $\alpha^{(k)}$ es elegido en un rango entre $[0, 1]$ para el cual se cumplen todas las restricciones. Siempre que $a_i^T p^{(k)} < 0$ para algún $i \notin W^{(k)}$, sin embargo, nosotros tendremos que $a_i^T(x^{(k)} + \alpha^{(k)}p^{(k)}) \geq b_i$ solo si

$$a_i^T x^{(k)} + a_i \alpha^{(k)} p^{(k)} \geq b_i$$
$$\alpha^{(k)} a_i^T p^{(k)} \geq b_i - a_i^T x^{(k)}$$

$$\alpha^{(k)} \leq \frac{b_i - a_i^T x^{(k)}}{a_i^T p^{(k)}}$$

Puesto que quisiéramos que el $\alpha^{(k)}$ fuera tan grande como sea posible dentro del rango $[0, 1]$ conforme la factibilidad de retención, tenemos la definición siguiente:

$$\alpha^{(k)} \stackrel{def}{=} \text{mín} \left(1, \min_{W_k, a_i^T p_k < 0} \frac{b_i - a_i^T x^{(k)}}{a_i^T p_k} \right) \tag{5.57}$$

Activamos aquellas restricciones para las cuales el mínimo (5.57) se cumplió (Si $\alpha^{(k)} = 1$ y no hay nuevas restricciones activas en el $x^{(k)} + \alpha^{(k)} p^{(k)}$ entonces no hay restricciones de bloqueo en esta iteración)

Si $\alpha_k < 1$, esto es, el paso a lo largo de p_k fue bloqueado por alguna restricción que no esta en $W^{(k)}$, donde un nuevo conjunto de trabajo $W^{(k+1)}$ es construido agregando una de las restricciones de bloqueo a $W^{(k)}$.

Ahora examinamos las muestras de los multiplicadores que corresponden a las restricciones en el conjunto de trabajo, es decir, los índices $i \in W$. Si estos multiplicadores no son negativos, la cuarta condición de KKT ($\lambda_i^* \geq 0$, para toda $i \in \mathcal{I} \cap A(x^*)$) también está se satisface, Concluimos que x' es un punto de KKT para el problema original. Puesto que G es definida semi-positiva, podemos demostrar que el x' es un minimizar. Cuando G es definida positiva, x' se minimiza.

Si por otra parte unos de los multiplicadores λ_j, $j \in W \cap \mathcal{I}$, son negativos, la condición cuatro de KKT no se satisface y la función objetivo $q(\cdot)$ puede ser decrementada por la disminución de las restricciones. Después quitamos un índice j que corresponde a uno de los multiplicadores negativos del conjunto de trabajo y solucionamos un nuevo problema para la siguiente iteración.

5.7.1. Algoritmo Conjunto Activo

El algoritmo del conjunto Activo se presenta en 20

Algoritmo 20 Algoritmo de Conjunto Activo

Entrada: $x^{(0)}$
Salida: x^*

Calcular un punto factible inicial $x^{(0)}$
Poner $W^{(0)}$ como el sub-conjunto activo de las restricciones en el punto $x^{(0)}$
para k=0,1,2. . . . **hacer**
Resolver el sistema de ecuaciones (5.56) para $p^{(k)}$
si $p^{(k)} = 0$ **entonces**
Calcular los Multiplicadores de Lagrange $\hat{\lambda}_i$ que satisfacen $A^T\lambda = Hx^{(k)} + d$ donde el conjunto $\hat{W} = W^{(k)}$
si $\hat{\lambda}_i \geq 0$ para toda $i \in \mathrm{W} \cap \mathrm{I}$ **entonces**
Parar, con la solución $x^* = x^{(k)}$
si no
Determinar el indice de la restricción bloqueda $j = argmin_{j \in W^{(k)} \cap \mathrm{I}} \hat{\lambda}_j$;
Dejar la solución igual $x^{(k+1)} = x^{(k)}$
Eliminar la restricción j del conjunto activo $W^{(k+1)} \longleftarrow W^{(k)} - \{j\}$
fin si
si no
Calculamos $\alpha^{(k)}$ para mín $\left(1, \underset{i \notin W^{(k)}, a_i^T p^{(k)} < 0}{\text{mín}} \frac{b_i - a_i^T x^{(k)}}{a_i^T p^{(k)}}\right)$
si alguna restricción se viola $\exists \alpha_i^{(k)} < 1.0$ **entonces**
Se obteniendo $W^{(k+1)}$ agregando la i-esima restricción violada a $W^{(k+1)} \longleftarrow W^{(k)} + \{i\}$
si no
El conjunto Activo permanece sin cambios
$W^{(k+1)} \longleftarrow W^{(k)}$
fin si
fin si
fin para

5.7.2. Ejemplo 1

Utilizando el método del conjunto activo, encuentre los valores de las incógnitas que minimizan la función objetivo dada, considerando para ello, las ecuaciones de restricción a las cuales se sujetan dichas funciones.

Minimizar:

$$q(x) = (x_1 - 2)^2 + (x_2 - 2)^2$$

sujeto a:

$$\begin{aligned} x_1 + x_2 &\geq 4 \\ 2x_1 - x_2 &\geq 0 \end{aligned}$$

considere la aproximación inicial al vector solución: $x^{(0)} = [2, 4]^T$

Para este problema tenemos

$$\nabla f(x) = \begin{bmatrix} 2(x_1 - 2) \\ 2(x_2 - 2) \end{bmatrix}$$

Iteración 1

Con $x^{(0)} = [2, 4]^T$, $W^{(0)} = \{2\}$ y $\nabla f(x^{(0)}) = [0, 4]^T$ tenemos que resolvemos el sistema de ecuaciones

$$\begin{pmatrix} 2.0 & 0.0 & -2.0 \\ 0.0 & 2.0 & 1.0 \\ 2.0 & -1.0 & 0.0 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ \lambda_2 \end{pmatrix} = \begin{pmatrix} 0 \\ -4 \\ 0 \end{pmatrix}$$

La solución es $[-0.8, -1.6, -0.8]^T$ y por lo tanto $p^{(0)} = [-0.8 - 1.6]^T$ con $\lambda^{(0)} = -0.8$. Calculamos $Ap^{(0)} = [-2.4, 0.0]^T$ y como el primer valor es negativo es posible que se este violando una restricción por ello calculamos los valores de α donde $Ap^{(0)}$ sea negativo.

$$\alpha_1^{(0)} = \frac{b_1 - a_1 x^{(0)}}{a_1 p_1^{(0)}} = \frac{4 - [1, 1][2, 4]^T}{-2.4} = 0.8333$$

$\alpha^{(0)} = [0.83333, No]^T$, por lo tanto las actualizaciones son

$$\begin{aligned} x_1^{(1)} &= 2 + (-0.8 * 0.8333) = 1.3333 \\ x_2^{(1)} &= 4 + (-1.6 * 0.8333) = 2.6667 \end{aligned}$$

Dado que $\alpha_1^{(0)} < 1.0$, entra al conjunto activo la restricción 1. El nuevo conjunto activo queda $W^{(1)} = \{1, 2\}$

Iteración 2

Con $x^{(1)} = [1.3333, 2.6667]^T$, $W_1 = \{1, 2\}$ y $\nabla f(x^{(1)}) = [-1.3333, +1.3333]^T$, resolvemos el sistema de ecuaciones

$$\begin{pmatrix} 2.0 & 0.0 & -1.0 & -2.0 \\ 0.0 & 2.0 & -1.0 & 1.0 \\ 1.0 & 1.0 & 0.0 & 0.0 \\ 2.0 & -1.0 & 0.0 & 0.0 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ \lambda_1 \\ \lambda_2 \end{pmatrix} = \begin{pmatrix} 1.3333 \\ -1.3333 \\ 0.0 \\ 0.0 \end{pmatrix}$$

La solución es $[0.0, 0.0, +0.4444, -0.8889]^T$ por lo tanto $p^{(1)} = [0, 0]$ y $\lambda^{(1)} = [0.4444, -0.8889]$. Dado que tenemos $\lambda_2^{(1)} < 0$ entonces tenemos que sacar la restricción 2 del conjunto activo. El conjunto activo queda $W^{(2)} = \{1\}$. La solución de la iteración 2 $x^{(2)}$ es:

$$x_1^{(2)} = x_1^{(1)} = 1.3333$$
$$x_2^{(2)} = x_2^{(1)} = 2.6667$$

Iteración 3

Con $x^{(2)} = [1.3333, 2.6667]^T$, $W^{(2)} = \{1\}$ y $\nabla f(x^{(2)}) = [-1.3333, 1.3333]^T$ resolvemos el sistema de ecuaciones

$$\begin{pmatrix} 2.0 & 0.0 & -1.0 \\ 0.0 & 2.0 & -1.0 \\ 1.0 & 1.0 & 0.0 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ \lambda_1 \end{pmatrix} = \begin{pmatrix} 1.3333 \\ -1.3333 \\ 0.0 \end{pmatrix}$$

La solución del sistema de ecuaciones es $[0.6667, -0.6667, 0.0]$, por lo tanto $p^{(2)} = [0.6667, -0.6667]$ y $\lambda_1^{(2)} = 0$. El vector $Ap^{(2)} = [0.0, 2.0]$ por lo cual no se viola ninguna restricción y $\alpha^{(2)} = 1.0$. Las actualizaciones de $x^{(3)}$ quedan como

$$x_1^{(3)} = 1.3333 + 0.6667 = 2.0$$
$$x_2^{(3)} = 2.6667 - 0.6677 = 2.0$$

con un conjunto activo $W^{(3)} = \{1\}$

Iteración 4

Con $x^{(3)} = [2.0, 2.0]^T$, $W^{(3)} = \{1\}$ y $\nabla f(x^{(2)}) = [0, 0]^T$ resolvemos el sistema de ecuacines

$$\begin{pmatrix} 2.0 & 0.0 & -1.0 \\ 0.0 & 2.0 & -1.0 \\ 1.0 & 1.0 & 0.0 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ \lambda_1 \end{pmatrix} = \begin{pmatrix} 0.0 \\ 0.0 \\ 0.0 \end{pmatrix}$$

La solución del sistema de ecuaciones es $[0.0, 0.0, 0.0]^T$ por lo cual $p^{(3)} = [0.0, 0.0]^T$ y $\lambda_1^{(3)} = 0$. Esto significa que hemos llegado a la solución y damos por terminado el proceso iterativo con:

$$x_1^{(4)} = 2.0 + 0.0 = 2.0$$
$$x_2^{(4)} = 2.0 + 0.0 = 2.0$$

Los resultados obtenidos del algoritmo del conjunto activo para el problema planteado se muestran en la figura 5.33.

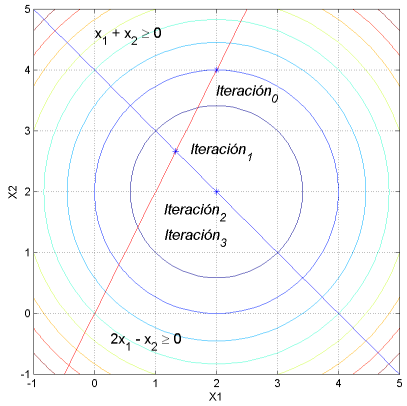


Figura 5.33: Solución del ***Método del Conjunto Activo*** para $q(x) = (x_1 - 2)^2 + (x_2 - 2)^2$; $x^{(0)} = [2, 4]^T$.

Ver QP_ejemplo_01.java

5.7.3. Ejemplo 2

Minimizar la función tomada del libro de Jorge Nocedal para la función $q(x)$ dada como

$$q(x) = (x_1 - 1)^2 + (x_2 - 2.5)^2$$

sujeto a:

$$\begin{aligned} x_1 - 2x_2 + 2 &\geq 0 \\ -x_1 - 2x_2 + 6 &\geq 0 \\ -x_1 + 2x_2 + 2 &\geq 0 \\ x_1 &\geq 0 \\ x_2 &\geq 0 \end{aligned}$$

Considere la aproximación inicial al vector solución: $x^{(0)} = [1, 1]^T$

En este caso tenemos que el Hessiano H y el vector d es

$$H = \begin{bmatrix} 2.0 & 0.0 \\ 0.0 & 2.0 \end{bmatrix}$$

$$d = \begin{bmatrix} -2.0 \\ -5.0 \end{bmatrix}$$

$$\nabla f(x) = H(x) + d = \begin{bmatrix} 2(x_1 - 1) \\ 2(x_2 - 2.5) \end{bmatrix}$$

Las restricciones pueden expresarse como

$$Ax \geq b$$

con

$$\begin{bmatrix} 1 & -2 \\ -1 & -2 \\ -1 & 2 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \geq \begin{bmatrix} -2 \\ -6 \\ -2 \\ 0 \\ 0 \end{bmatrix}$$

Iteración 1

Dado el punto inicial $x^{(0)} = [1, 1]^T$, $\nabla f(x^{(0)}) = [0.0, -3.0]$ y el conjunto activo es $W^{(0)} = \{\}$. Por lo tanto, resolvemos el sistema de ecuaciones dado por

$$\begin{bmatrix} 2.0 & 0.0 \\ 0.0 & 2.0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 3 \end{bmatrix}$$

La solución es $p^{(0)} = [0.0, 1.5]^T$, y $Ap^{(0)} = [-3.0, -3.0, 3.0, 0.0, 1.5]^T$ por lo que las restricciones 1 y 2 se violan dado que $Ap^{(0)} < 0$. Calculamos los valores de $\alpha^{(0)} = [0.33333, 1.0, *, *, *]^T$ y el valor mínimo de $\alpha^{(0)}$ es

$$\alpha_1 = \frac{b_1 - a_1 x^{(0)}}{a_1 p^{(0)}} = \frac{2 - [1, -2][1, 1]^T}{[1, -2][0, 1.5]^T} = \frac{1}{3}$$

$$\alpha_2 = \frac{b_2 - a_2 x^{(0)}}{a_2 p^{(0)}} = \frac{-6 - [-1, -2][1, 1]^T}{[-1, -2][0, 1.5]^T} = 1$$

$$\alpha_{min}^{(0)} = min\{1, [0.3333, 1.0, *, *, *]^T\} = 0.3333$$

En cálculo de las variables en la siguiente iteración queda como

$$x^{(1)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} + 0.33333 \times \begin{bmatrix} 0 \\ 1.5 \end{bmatrix} = \begin{bmatrix} 1 \\ 1.5 \end{bmatrix}$$

El conjunto activo de restricciones para esta iteración queda $W^{(1)} = \{1\}$, dado que $x^{(1)}$ esta sobre la restricción 1.

Iteración 2

Dado el punto inicial $x^{(1)} = [1.0, 1.5]^T$, $\nabla f(x^{(1)}) = [0.0, -2.0]^T$ y el conjunto activo es $W^{(1)} = \{1\}$, resolvemos el sistema de ecuaciones

$$\begin{bmatrix} 2.0 & 0.0 & -1.0 \\ 0.0 & 2.0 & 2.0 \\ 1.0 & -2.0 & 0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ \lambda_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2.0 \\ 0 \end{bmatrix}$$

La solución es $[p^{(1)}, \lambda_1^{(1)}]^T = [0.4, 0.2, 0.8]^T$ y $Ap^{(1)} = [0.0, -0.8, 0.0, 0.4, 0.2]^T$. Para ello calculamos los valores de $\alpha^{(1)} = [*, 2.5, *, *, *]^T$ dado que hay valores negativos en $Ap^{(0)}$ y el valor mínimo de α es:

$$\alpha_2 = \frac{b_2 - a_2 x^{(1)}}{a_2 p^{(1)}} = \frac{-6 - [-1, -2][1, 1.5]^T}{[-1, -2][0.4, 0.2]^T} = 2.5$$

$$\alpha_{min}^{(1)} = min\{1, [*, 2.5, *, *, *]^T\} = 1.0$$

En cálculo de las variables en la siguiente iteración queda como:

$$x^{(2)} = \begin{bmatrix} 1 \\ 1.5 \end{bmatrix} + 1 \times \begin{bmatrix} 0.4 \\ 0.2 \end{bmatrix} = \begin{bmatrix} 1.4 \\ 1.7 \end{bmatrix}$$

El conjunto activo de restricciones queda $W^{(2)} = \{1\}$ dado que la restricción 2 no se viola.

Iteración 3

Dado el punto inicial $x^{(2)} = [1.4, 1.7]^T$, $\nabla f(x^{(2)}) = [0.8, -1.6]$ y el conjunto activo es $W^{(2)} = \{1\}$, resolvemos el sistema de ecuaciones

$$\begin{bmatrix} 2.0 & 0.0 & -1.0 \\ 0.0 & 2.0 & 2.0 \\ 1.0 & -2.0 & 2.0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ \lambda_1 \end{bmatrix} = \begin{bmatrix} -0.8 \\ 1.6 \\ 0 \end{bmatrix}$$

La solución es $[p^{(2)}, \lambda_1^{(2)}]^T = [0.0, 0.0, 0.8]^T$ y el cálculo de $Ap^{(1)} = [0.0, 0.0, 0.0, 0.0, 0.0]^T$ por lo que no se viola restricción alguna, por lo tanto la solución es:

$$x^{(3)} = \begin{bmatrix} 1.4 \\ 1.7 \end{bmatrix} + 1 \times \begin{bmatrix} 0.0 \\ 0.0 \end{bmatrix} = \begin{bmatrix} 1.4 \\ 1.7 \end{bmatrix}$$

con esto se da por terminado el procedimiento siendo la solución $x^{(*)} = [1.4, 1.7]^T$.

Ver QP_ejemplo_02.java

5.7.4. Ejemplo 3

Minimizar la función $q(x)$ del ejemplo anterior pero considerando como valor inicial $x^{(0)} = [2, 0]^T$.

$$q(X) = (x_1 - 1)^2 + (x_2 - 2.5)^2$$

sujeto a:

$$\begin{aligned} x_1 - 2x_2 + 2 &\geq 0 \\ -x_1 - 2x_2 + 6 &\geq 0 \\ -x_1 + 2x_2 + 2 &\geq 0 \\ x_1 &\geq 0 \\ x_2 &\geq 0 \end{aligned}$$

Iteración 1

Dado el punto inicial $x^{(0)} = [2, 0]^T$, el conjunto activo es $W^{(0)} = \{3, 5\}$ en virtud de que el punto $x^{(0)}$ está en las intersección de ambas desigualdades. El sistema de ecuaciones a resolver es:

$$\begin{bmatrix} 2.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 2.0 & -2.0 & -1.0 \\ -1.0 & 2.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ \lambda_3 \\ \lambda_5 \end{bmatrix} = \begin{bmatrix} -2.0 \\ 5.0 \\ 0.0 \\ 0.0 \end{bmatrix}$$

La solución es $[p^{(0)}, \lambda_3, \lambda_5] = [0.0, 0.0, -2.0, -1.0]^T$, $\lambda_3 = -2$, $\lambda_5 = -1$. En este caso tenemos valores de p en cero pero las λ's están violando la condición de ser mayores a cero, esto se debe a que no estamos en el mínimo de la función. Por tal motivo tenemos que eliminar del conjunto activo la restricción con el valor de lambda menor.

$$\lambda_{min} = min[-2, -1] = -2$$

por lo tanto el conjunto activo queda $W^{(1)} = \{3, 5\} - \{3\} = \{5\}$ y $x^{(1)} = x^{(0)} = [2, 0]$

Iteración 2

Dado el punto $x^{(1)} = [2, .0]^T$ y el conjunto activo $W^{(1)} = \{5\}$, tenemos que solucionar el sistema de ecuaciones

$$\begin{bmatrix} 2.0 & 0.0 & 0.0 \\ 0.0 & 2.0 & -1.0 \\ 0.0 & 1.0 & 0.0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ \lambda_5 \end{bmatrix} = \begin{bmatrix} -2.0 \\ 5.0 \\ 0.0 \end{bmatrix}$$

La solución es $[p^{(1)}, \lambda_5]^T = [-1.0, 0.0, -5.0]^T$, $\lambda_5 = -5$ y el cálculo de $Ap^{(1)} = [-1.0, 1.0, 1.0, -1.0, 0.0]^T$. Dado que hay valores $Ap^{(0)}$ negativos, calculamos los valores de $\alpha^{(1)} = [4, *, *, 2, *]$ y el valor mínimo es:

$$\alpha_1 = \frac{b_1 - a_1 x^{(1)}}{a_1 p^{(1)}} = \frac{-2 - [1, -2][2, 0]^T}{[1, -2][-1, 0]^T} = 4$$

$$\alpha_4 = \frac{b_4 - a_4 x^{(1)}}{a_4 p^{(1)}} = \frac{-0 - [1, 0][2, 0]^T}{[1, 0][-1, 0]^T} = 2$$

$$\alpha^{(1)}_{min} = min\{1, [4, *, *, 2, *]^T\} = 1.0$$

En cálculo de las variables en la siguiente iteración queda como:

$$x^{(2)} = \begin{bmatrix} 2.0 \\ 0.0 \end{bmatrix} + 1 \times \begin{bmatrix} -1.0 \\ 0.0 \end{bmatrix} = \begin{bmatrix} 1.0 \\ 0.0 \end{bmatrix}$$

El conjunto activo de restricciones queda $W^{(2)} = \{5\}$ dado los valores de $x^{(2)}$ no violan las restricciones y que λ_5 es positivo.

Iteración 3

Dado el punto $x^{(2)} = [1.0, 0.0]^T$, $\nabla f(x^{(2)}) = [0.0, -5.0]^T$ y el conjunto activo es $W^{(2)} = \{5\}$, tenemos que solucionar el sistema de ecuaciones dado como:

$$\begin{bmatrix} 2.0 & 0.0 & 0.0 \\ 0.0 & 2.0 & -1.0 \\ 0.0 & 1.0 & 0.0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ \lambda_5 \end{bmatrix} = \begin{bmatrix} 0.0 \\ 5.0 \\ 0.0 \end{bmatrix}$$

La solución es $[p^{(2)}, \lambda_5]^T = [0.0, 0.0, -5.0]^T$, con lo cual tenemos $p^{(2)} = 0$ y $\lambda_5 = -5.0$. Los valores de $x^{(3)} = x^{(2)}$ y sacamos la restricción 5 del conjunto activo $W^{(3)} = \{\}$.

Iteración 4

Dado el punto $x^{(3)} = [1.0, 0.0]^T$, $\nabla f(x^{(3)}) = [0.0, -5.0]^T$ y el conjunto activo es $W^{3)} = \{\}$ y tenemos que solucionar el sistema

$$\begin{bmatrix} 2.0 & 0.0 \\ 0.0 & 2.0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} 0.0 \\ 5.0 \end{bmatrix}$$

La solución es $[p^{(3)}]^T = [0.0, 2.5]^T$ y los valores de $Ap^{(3)} = [-5.0, -5.0, 5.0, 0.0, 2.5]^T$. Del vector $Ap^{(3)}$ podemos ver que se violan las restricciones 1 y 2 por lo cual calculamos los valores de $\alpha^{(3)} = [0.6, 1.0, *, *, *]$ y el valor mínimo es:

$$\alpha_1 = \frac{b_1 - a_1 x^{(3)}}{a_1 p^{(3)}} = \frac{-2 - [1, -2][1, 0]^T}{[1, -2][0, 2.5]^T} = 4 = \frac{-3}{-5} = 0.6$$

$$\alpha_2 = \frac{b_2 - a_2 x^{(3)}}{a_2 p^{(3)}} = \frac{-6 - [-1, -2][1, 0]^T}{[-1, -2][0, 2.5]^T} = 1$$

$$\alpha_{min}^{(3)} = min\{1, [0.6, 1.0, *, *, *]\} = 0.6$$

En cálculo de las variables en la siguiente iteración queda como:

$$x^{(4)} = \begin{bmatrix}1.0\\0.0\end{bmatrix} + 0.6 \times \begin{bmatrix}0.0\\2.5\end{bmatrix} = \begin{bmatrix}1.0\\1.5\end{bmatrix}$$

y el conjunto activo de restricciones queda $W^{(4)} = \{1\}$

Iteración 5

Dado el punto $x^{(4)} = [1.0, 1.5]^T$, el conjunto activo es $W^{(4)} = \{1\}$ y tenemos que solucionar el sistema

$$\begin{bmatrix}2.0 & 0.0 & -1.0\\0.0 & 2.0 & 2.0\\1.0 & -2.0 & 0.0\end{bmatrix} \begin{bmatrix}p_1\\p_2\\\lambda_1\end{bmatrix} = \begin{bmatrix}0.0\\2.0\\0.0\end{bmatrix}$$

La solución es $[p^{(4)}, \lambda_1]^T = [0.4, 0.2, 0.8]^T$ por lo cual $\lambda_1 = 0.8$ y el cálculo de $Ap^{(4)} = [0.0, -0.8, 0.0, 0.4, 0.2]^T$ nos dice que es posible que se viole una restricción. El valor mínimo de las α es:

$$\alpha_2 = \frac{b_2 - a_2 x^{(4)}}{a_2 p^{(4)}} = \frac{-6 - [-1, -2][1.0, 1.5]^T}{[-1, -2][0.4, 0.2]^T} = 2.5$$

$$\alpha^{(4)}_{min} = min\{1, [*, 2.500, *, *, *]\} = 1.0$$

En cálculo de las variables en la siguiente iteración queda como:

$$x^{(5)} = \begin{bmatrix}1.0\\1.5\end{bmatrix} + 1.0 \times \begin{bmatrix}0.4\\0.2\end{bmatrix} = \begin{bmatrix}1.4\\1.7\end{bmatrix}$$

El conjunto activo de restricciones queda $W^{(5)} = \{1\}$ dado que $x^{(5)}$ esta sobre el borde de la restricción 1.

Iteración 6

Dado el punto $x^{(5)} = [1.4, 1.7]^T$, el conjunto activo es $W^{(4)} = \{1\}$ y tenemos que solucionar el sistema

$$\begin{bmatrix}2.0 & 0.0 & -1.0\\0.0 & 2.0 & 2.0\\1.0 & -2.0 & 0.0\end{bmatrix} \begin{bmatrix}p_1\\p_2\\\lambda_1\end{bmatrix} = \begin{bmatrix}-0.8\\1.6\\0.0\end{bmatrix}$$

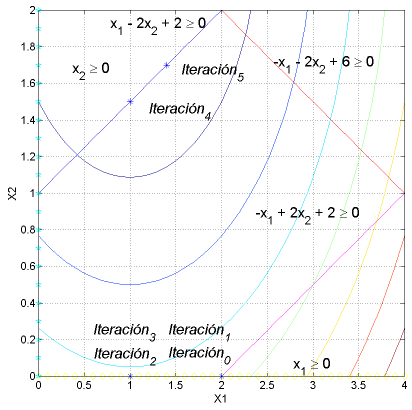


Figura 5.34: Solución del ***Método del Conjunto Activo*** para $q(X) = (x_1 - 1)^2 + (x_2 - 2.5)^2$; $x^{(0)} = [2, 0]^T$.

La solución es $[p^{(5)}, \lambda_1]^T = [0.0, 0.0, 0.8]^T$ por lo cual $\lambda_1 = 0.8$. Como los valores de $p^{(5)} = 0$ y el valor de $\lambda > 0$ terminamos con solución $x^{(6)} = [1.4, 1.7]^T$,

ver QP_ejemplo_03.java

5.7.5. Ejemplo 4

Resolver la función $q(x)$ para las restricciones dadas utilizando el método de conjunto activo.

Minimizar:

$$q(x) \quad = \quad (x_1 - 2)^2 + (x_2 - 2)^2$$

sujeto a:

$$\begin{aligned} x_1 + x_2 &\geq 6 \\ 2x_1 - x_2 &\geq 0 \end{aligned}$$

considere la aproximación inicial al vector solución: $x^{(0)} = [4, 3]^T$

Los resultados obtenidos del algoritmo del conjunto activo para el problema planteado se muestran en la tabla 5.13. Además, en la figura 5.35 se muestra gráficamente el proceso de solución.

k	x		p		λ		Notas
0	4.0	3.0	-2.0	-1.0	0.0	0.0	Activar λ_0
$\alpha = 0.3333$ $x^{(k+1)} = [3.3333, 2.6667]^T$							
1	3.3333	2.6667	-0.3333	0.3333	2.0	0.0	
$\alpha = 1.0$ $x^{(k+1($ $= [3.0, 3.0]^T$							
2	3.0	3.0	0.0	0.0	2.0	0.0	
$F(x^*) = 2.0$ $x^* = [3.0, 3.0]^T$							

Cuadro 5.13: Resultados del ***Método del Conjunto Activo*** para $q(x) = (x_1 - 2)^2 + (x_2 - 2)^2$; $x^{(0)} = [4, 3]^T$.

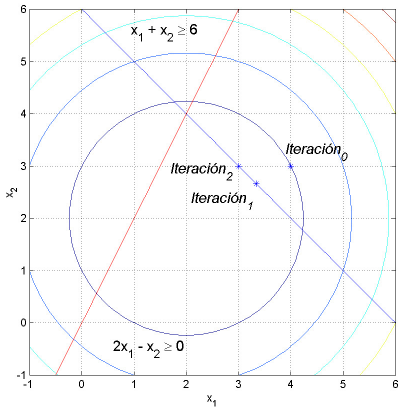


Figura 5.35: Solución del ***Método del Conjunto Activo*** para $q(X) = (x_1 - 2)^2 + (x_2 - 2)^2$; $x^{(0)} = [4, 3]^T$.

5.8. Funciones de Penalización

Una alternativa a las funciones de penalización en los casos como el ejemplo anterior es utilizar una función alterna en la cual se deja el valor λ fijo y en un valor tal que garantice la convexidad de la función así la función Lagrangiana para a ser una función con los valores de λ lo suficientemente grande para garantizar la convexidad de la función.

$$F(x) = f(x) - \sum_{i=0}^{N} \lambda_i C_i(x)$$

Así por ejemplo para el ejemplo anterior si suponemos que nos dan $\lambda = 0.5$ tenemos que minimizar la función

$$F(x, y) = x + y - 0.5(x^2 + y^2 - 2)$$

cuya solución en $x = -1$ y $y = -1$.

5.9. El método de Barrera Logaritmica

Comenzaremos por describir el concepto de función de barrera en términos de un problema con restricciones de desigualdad. Dado el problema

$$\begin{aligned} \min_x & f(x) \\ & s.a \\ c_i(x) & \geq 0 \\ i & \in \mathcal{I} \end{aligned}$$

con una region de factibilidad definida como

$$\mathcal{F} \stackrel{def}{=} \{x \in \mathcal{R} | c_i(x) \geq 0 \text{ para todo } i \in \mathcal{I}\}$$

Asumiremos que $\mathcal{F}$ es un conjunto no vacío.

La mas importante función de barrera es la función de barrera logarítmica, la cual es utilizada para el conjunto de restricciones $c_i \geq 0$, y tiene la forma

$$-\sum_{i \in \mathcal{I}} \log c_i(x)$$

donde log(.) denota el logaritmo natural. Para un problema de optimización con restricciones de desigualdad, la función objetivo/barrera combinada esta dada por

$$P(x;\mu) = f(x) - \mu \sum_{i\in\mathcal{I}} \log c_i(x) \tag{5.58}$$

donde μ es conocido como el parámetro de barrera. Desde este momento, nos referiremos a $P(x;\mu)$ como la función de barrera logarítmica.

El gradiente de la nueva funcón $P(x,\mu)$ se calcula como:

$$\nabla P(x,\mu) = \nabla f(x) - \mu \sum_{i=1}^{M} \frac{\nabla C_i(x)}{C_i(x)}$$

y el Hessiano como:

$$\nabla^2 P(x,\mu) = \nabla^2 f(x) - \mu \sum_{i=1}^{M} \frac{\nabla^2 C_i(x) C_i(x) - \nabla^T C_i(x) \nabla C_i(x)}{C_i^2(x)}$$

Calculo de Gradiente y Hessiano con Restricciones Lineales

En el caso de tener restricciones lineales dada como

$$\begin{bmatrix} c_1(x) \\ c_2(x) \\ c_3(x) \\ \vdots \\ c_M(x) \end{bmatrix} = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,N} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,N} \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,N} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{M,1} & a_{M,2} & a_{M,3} & \cdots & a_{M,N} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_N \end{bmatrix} \geq \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_M \end{bmatrix}$$

El gradiente lo podemos calcula como sigue

$$\nabla P(x;\mu) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} - \mu \left(\frac{\partial log(c_1(x))}{\partial x_1} + \frac{\partial log(c_2(x))}{\partial x_1} + \frac{\partial log(c_3(x))}{\partial x_1} + \cdots \frac{\partial log(c_M(x))}{\partial x_1} \right) \\ \frac{\partial f(x)}{\partial x_2} - \mu \left(\frac{\partial log(c_1(x))}{\partial x_2} + \frac{\partial log(c_2(x))}{\partial x_2} + \frac{\partial log(c_3(x))}{\partial x_2} + \cdots \frac{\partial log(c_M(x))}{\partial x_2} \right) \\ \vdots \\ \frac{\partial f(x)}{\partial x_N} - \mu \left(\frac{\partial log(c_1(x))}{\partial x_N} + \frac{\partial log(c_2(x))}{\partial x_N} + \frac{\partial log(c_3(x))}{\partial x_N} + \cdots \frac{\partial log(c_M(x))}{\partial x_N} \right) \end{bmatrix}$$

$$\nabla P(x;\mu) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} - \mu\left(\frac{a_{1,1}}{c_1(x)} + \frac{a_{2,1}}{c_2(x)} + \frac{a_{3,1}}{c_3(x)} \cdots \frac{a_{M,1}}{c_M(x)}\right) \\ \frac{\partial f(x)}{\partial x_2} - \mu\left(\frac{a_{1,2}}{c_1(x)} + \frac{a_{2,2}}{c_2(x)} + \frac{a_{3,2}}{c_3(x)} \cdots \frac{a_{M,2}}{c_M(x)}\right) \\ \frac{\partial f(x)}{\partial x_3} - \mu\left(\frac{a_{1,3}}{c_1(x)} + \frac{a_{2,3}}{c_2(x)} + \frac{a_{3,3}}{c_3(x)} \cdots \frac{a_{M,3}}{c_M(x)}\right) \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} - \mu\left(\frac{a_{1,N}}{c_1(x)} + \frac{a_{2,N}}{c_2(x)} + \frac{a_{3,N}}{c_3(x)} \cdots \frac{a_{M,N}}{c_M(x)}\right) \end{bmatrix}$$

$$\nabla P(x;\mu) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \frac{\partial f(x)}{\partial x_2} \\ \frac{\partial f(x)}{\partial x_3} \\ \vdots \\ \frac{\partial f(x)}{\partial x_N} \end{bmatrix} - \begin{bmatrix} a_{1,1} & a_{2,1} & a_{3,1} & \cdots & a_{M,1} \\ a_{1,2} & a_{2,2} & a_{3,2} & \cdots & a_{M,2} \\ a_{1,3} & a_{2,3} & a_{3,3} & \cdots & a_{M,3} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ a_{1,N} & a_{2,N} & a_{3,N} & \cdots & a_{M,N} \end{bmatrix} \begin{bmatrix} 1/c_1(x) \\ 1/c_2(x) \\ 1/c_3(x) \\ \vdots \\ 1/c_M(x) \end{bmatrix}$$

En forma compacta podemos representar el Gradiente como

$$\nabla P(x;\mu) = \nabla f(x) - \mu A^T C_1$$

donde

$$C_1 = \begin{bmatrix} \frac{1}{c_1(x)} \\ \frac{1}{c_2(x)} \\ \frac{1}{c_3(x)} \\ \vdots \\ \frac{1}{c_M(x)} \end{bmatrix}$$

Para el calculo del Hessiano procedemos a derivar nuevamente el vector Gradiente, llegando a una forma matricial como la siguiente

$$\nabla^2 P(x;\mu) = \nabla^2 f(x) + \mu A^T C_2 A$$

donde $\widehat{C}$ esta dada por la expresión

$$C_2 = \begin{bmatrix} \frac{1}{c_1^2(x)} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{c_2^2(x} & 0 & \cdots & 0 \\ 0 & 0 & \frac{1}{c_3^2(x)^2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{1}{c_M^2(x)} \end{bmatrix}$$

5.9.1. Método

De acuerdo con el ejemplo anterior una estrategia, para encontrar la solución utilizando el método de Barrera, es comenzar con un valor μ_0 grande y a medida que se alcanza un mínimo en la función de Barrera (5.58), reducirlo. Esto debe ser repetido hasta alcanzar convergencia. El valor inicial del parámetro de Barrera debe ser lo suficientemente grande de acuerdo con las características de la función a minimizar. Un criterio de paro, recomendable, será cuando $\mu \leq \tau$ y $\| \nabla F(x;\mu) \| \leq \tau$. En el algoritmo 21

Algoritmo 21 Método de Barrera

Entrada: $f(x), x^{(0)}, Ax - b, \mu$
Dado una función diferenciable $f(x)$, un valor inicial $x^{(0)}$, A y μ_0
para $k = 1 \ldots MAX$ **hacer**
Encontrar una minimización aproximada de $x^{(k)}$ de $P(.;\mu)$,
comenzando en un punto $x^{(k)}$ y finalizar cuando $\| \nabla P(x;\mu) \leq \tau_k \|$
si la prueba de de convergencia es satisfactoria **entonces**
Parar con una solución aproximada $x^{(k)}$
fin si
Seleccionar un nuevo parámetro de Barrera $\mu_{k+1} \in (0, \mu_k)$
Seleccionar un nuevo punto de arranque $x^{(k+1)}$
fin para
devolver $x^{(k)}$

5.9.2. Ejemplo 1

Considere el siguiente problema con una simple variable x

$$\begin{aligned} \min_x \quad & x \\ & s.a. \\ x & \geq 0 \\ 1 - x & \geq 0 \end{aligned}$$

la función de barrera para este caso es

$$P(x;\mu) = f(x) - \mu \log x - \mu \log(1-x) \tag{5.59}$$

La gráfica correspondiente a la función de barrera dada por la ecuación (5.59), se muestra en la figura 5.36, en esta se presentan diferentes valores de μ dados como 1, 0.4, 0.1 y 0.01. Note que a medida que se reduce el parámetro de barrera μ, el mínimo de la función se acerca a la realidad y las barreras laterales, que impiden que el método salga de la región de factibilidad, se reducen.

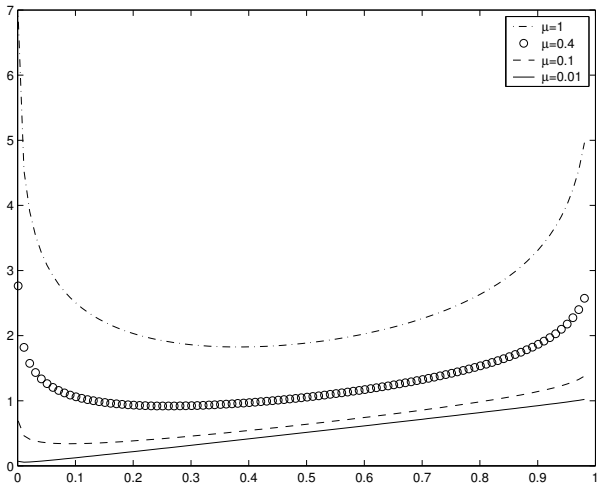


Figura 5.36: Función lineal con diferentes valores de μ

En la tabla 5.14, se presentan algunas iteraciones para el método de Barrera aplicada a la solución del problema. Los datos iniciales para el método de Barrera son $\mu^{(0)} = 100$ y $x^{(0)} = 0.9$

Ver Restricciones_ej01.java

5.9.3. Ejemplo 2

Calcular el mínimo de la función

$$\begin{aligned} f(x) &= (x_1-2)^2 + (x_2-2)^2 \\ &\quad s.a \\ x_1 + x_2 &\geq 4 \\ 2x_1 - x_2 &\geq 0 \end{aligned}$$

k	μ	x
1	100.0	0.8121
2	100.0	0.6747
3	100.0	0.5371
4	100.0	0.4992
5	100.0	0.4988
6	100.0	0.4988
7	66.66	0.4981
8	66.66	0.4981
9	44.44	0.4972
10	44.44	0.4972
11	29.62	0.4958
12	29.62	0.4958
13	19.75	0.4937
14	19.75	0.4937
15	13.16	0.4905
16	13.16	0.4905
17	8.77	0.4858
18	8.77	0.4858
19	5.85	0.4787
20	5.85	0.4787
21	3.90	0.4681
22	3.90	0.4681
23	2.60	0.4523
24	2.60	0.4524
⋮	⋮	⋮
56	0.01	0.0101
57	0.01	0.0125
58	0.01	0.0131
⋮	⋮	⋮
98	4.57E-5	0.0

Cuadro 5.14: Solución utilizando el método de Barrera para el ejemplo 1

k	μ	x_1	x_2
1	100.0	2.5643	-2.3475
2	100.0	3.1638	-5.4244
3	100.0	3.2653	-7.2717
4	100.0	3.1971	-7.5549
5	100.0	3.1937	-7.5606
6	66.66	2.8886	-5.6015
7	66.66	2.9024	-5.75
8	66.66	2.9023	-5.7514
9	44.44	2.6565	-4.1549
10	44.44	2.6655	-4.2739
⋮	⋮	⋮	⋮
65	1.54E-4	1.994	1.9938
66	1.03E-4	1.9951	1.995
67	6.86E-5	1.996	1.9959
68	4.57E-5	1.9967	1.9967
69	3.05E-5	1.9973	1.9973
70	2.03E-5	1.9978	1.9978
71	1.35E-5	1.9982	1.9982
72	9.04E-6	1.9985	1.9985
⋮	⋮	⋮	⋮
89	9.17E-9	2.0	2.0
90	6.11E-9	2.0	2.0

Cuadro 5.15: Resultados del ejemplo 2

considerando como punto inicial $x^{(0)} = [2, 0]^T$ y $\mu = 100$.

La función de Barrera en este ejemplo esta dada como

$$P(x; \mu) = (x_1 - 2)^2 + (x_2 - 2)^2 - \mu \log (x_1 + x_2 - 4) - \mu \log(2x_1 - x_2)$$

y la solución para este problema se da en la tabla 5.15. Note como el valor de μ, se reduce cada vez que el algoritmo alcanza un estado de sub-óptimo.

Ver Restricciones_ej02.java

5.9.4. Ejemplo 3

Dada la matriz

$$A = \begin{pmatrix} 16 & 8 & 4 & 12 \\ 8 & 29 & -3 & 16 \\ 4 & -3 & 102 & 21 \\ 12 & 16 & 21 & 53 \end{pmatrix}$$

y el vector $b = [5, -2, 10, 6]^T$, calcular el mínimo de la función cuadrática

$$\begin{aligned} f(x) = 0.5x^T Ax – b^T x \\ s.a \\ 0 \leq x_i \leq 1 \\ \forall i = 1 \cdots 4 \end{aligned}$$

Considere como valor inicial $x_i = 0.9$ y $\mu = 100$

En el caso de este problema los resultados se presentan en la tabla 5.16

k	μ	x
1	100.0	1.0, 1.0, 1.0, 1.0
2	100.0	1.0, 1.0, 1.0, 1.0
3	100.0	1.0, 1.0, 1.0, 1.0
4	100.0	1.0, 1.0, 1.0, 1.0
5	100.0	1.0, 1.0, 1.0, 1.0
6	100.0	0.9999, 0.9999, 0.9999, 0.9999
7	100.0	0.9999, 0.9999, 0.9999, 0.9999
8	100.0	0.9997, 0.9997, 0.9997, 0.9997
9	100.0	0.9995, 0.9995, 0.9995, 0.9995
⋮	⋮	⋮
23	66.66	0.4755, 0.4542, 0.4218, 0.4289
24	66.66	0.4755, 0.4543, 0.4221, 0.4291
25	66.66	0.4755, 0.4543, 0.4221, 0.4291
⋮	⋮	⋮
500	1.46E-20	0.2789, 0.0, 0.0836, 0.0169
501	1.46E-20	0.2789, 0.0, 0.0836, 0.0169
502	1.46E-20	0.2789, 0.0, 0.0836, 0.0169

Cuadro 5.16: Resultados del ejemplo 3

Ver Restricciones_ej03.java

5.10. Métodos de Punto Interior.

Por simplicidad, únicamente utilizaremos restricciones de desigualdad

$$\begin{aligned} \text{mín}\, q(x) &= \frac{1}{2}x^T H(x)x + x^T d(x) \\ &\underset{s.a}{} \\ Ax &\geq b \end{aligned}$$

donde $H(x)$ es una matriz definida semi positiva, A es una matriz de $m \times n$ definida como $A = [a_i]_{i\in\mathcal{I}}$ y $b = [b_i]_{i\in\mathcal{I}}$ con $\mathcal{I} = \{1, 2, 3, ..., m\}$.

Podemos especializar las condiciones KKT para obtener el conjunto necesario de condiciones: si x^* es solución, hay un vector de multiplicadores de lagrange λ^*, tales que las siguientes condiciones se satisfacen para $(x, \lambda) = (x^*, \lambda^*)$

$$\begin{aligned} H(x)x - A^T\lambda + d(x) &= 0 \\ Ax - b &\geq 0 \\ (Ax - b)\,\lambda_i &= 0 \\ \lambda_i &\geq 0 \\ i &= 1, 2, .., m \end{aligned} \tag{5.60}$$

Introduciendo el vector laxo $y = Ax - b$, el cual mide la brecha entre la desigualdad y la igualdad, podemos escribir las ecuaciones (5.60) como:

$$\begin{aligned} H(x)x - A^T\lambda + d(x) &= 0 \\ Ax - y - b &= 0 \\ y_i\lambda_i &= 0 \\ (y, \lambda) &\geq 0 \end{aligned} \tag{5.61}$$

Este sistema de ecuaciones (5.61), lo podemos escribir como

$$\begin{aligned} F(x, y, \lambda) &= \begin{bmatrix} H(x)x - A^T\lambda + d(x) \\ Ax - y - b \\ Y\Lambda e \end{bmatrix} \\ (y, \lambda) &\geq 0 \end{aligned}$$

donde $Y = diag[y_1, y_2, ..., y_m]$, $\Lambda = diag[\lambda_1, \lambda_2, ..., \lambda_m]$ y $e = [1, 1, ...1]^T$.

Dada una iteración (x, y, λ), que satisface $(y, \lambda) \geq 0$, podemos definir una medida de dualidad μ como

$$\mu = \frac{1}{m} \sum_{i=1}^{m} y_i \lambda_i = \frac{y^T \lambda}{m}$$

donde esta medida de dualidad μ, representa el promedio del producto $y^T \lambda$. La ruta central $\mathcal{C}$, es el conjunto de puntos $(x_\tau, y_\tau, \lambda_\tau)$, con $\tau > 0$, tal que

$$\begin{aligned} F(x_\tau, y_\tau, \lambda_\tau) &= \begin{bmatrix} 0 \\ 0 \\ \tau e \end{bmatrix} \\ (y_\tau, \lambda_\tau) &> 0 \end{aligned} \tag{5.62}$$

donde τ se escoge tal que $\tau = \sigma\mu$

El paso genérico es un paso dado por el algoritmo de Newton-Raphson, para el punto actual (x, y, λ) hacia el punto en la ruta central $(x_{\sigma\mu}, y_{\sigma\mu}, \lambda_{\sigma\mu})$ donde σ es un parámetro, en el intervalo $[0, 1]$, seleccionado de acuerdo con las características de la función y elegido de manera manual.

Utilizando la serie de Taylor, podemos hacer una aproximación lineal para cada una de las funciones

$$F_j\left(x^{(k)} + \Delta x^{(k)}, y^{(k)} + \Delta y^{(k)}, \lambda^{(k+1)} + \Delta\lambda^{(k)}\right) \equiv F_j(x_\tau, y_\tau, \lambda_\tau)$$

dado como

$$F_j(x_\tau, y_\tau, \lambda_\tau) = F_j(x^{(k)}, y^{(k)}, \lambda^{(k)}) + \frac{\partial F_j}{\partial x}\Delta x + \frac{\partial F_j}{\partial y}\Delta y + \frac{\partial F_j}{\partial \lambda}\Delta\lambda$$

En forma matricial podemos expresar la ecuación anterior como

$$\begin{bmatrix} \frac{\partial F_1}{\partial x} & \frac{\partial F_1}{\partial y} & \frac{\partial F_1}{\partial \lambda} \\ \frac{\partial F_2}{\partial x} & \frac{\partial F_2}{\partial y} & \frac{\partial F_2}{\partial \lambda} \\ \frac{\partial F_3}{\partial x} & \frac{\partial F_3}{\partial y} & \frac{\partial F_3}{\partial \lambda} \end{bmatrix} \begin{bmatrix} \Delta x^{(k)} \\ \Delta y^{(k)} \\ \Delta\lambda^{(k)} \end{bmatrix} + \begin{bmatrix} F_1(x^{(k)}, y^{(k)}, \lambda^{(k)}) \\ F_2(x^{(k)}, y^{(k)}, \lambda^{(k)}) \\ F_3(x^{(k)}, y^{(k)}, \lambda^{(k)}) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \tau e \end{bmatrix}$$

o también como

$$\begin{bmatrix} \frac{\partial F_1}{\partial x} & \frac{\partial F_1}{\partial y} & \frac{\partial F_1}{\partial \lambda} \\ \frac{\partial F_2}{\partial x} & \frac{\partial F_2}{\partial y} & \frac{\partial F_2}{\partial \lambda} \\ \frac{\partial F_3}{\partial x} & \frac{\partial F_3}{\partial y} & \frac{\partial F_3}{\partial \lambda} \end{bmatrix} \begin{bmatrix} \Delta x^{(k)} \\ \Delta y^{(k)} \\ \Delta \lambda^{(k)} \end{bmatrix} = \begin{bmatrix} -F_1(x^{(k)}, y^{(k)}, \lambda^{(k)}) \\ -F_2(x^{(k)}, y^{(k)}, \lambda^{(k)}) \\ -F_3(x^{(k)}, y^{(k)}, \lambda^{(k)}) + \tau e \end{bmatrix}$$

De acuerdo a esto, para las funciones dadas por la ecuación (5.61), tenemos que resolver el sistema de ecuaciones dado por (5.63)

$$\begin{bmatrix} H(x^{(k)}) & -A^T & 0 \\ A & 0 & -I \\ 0 & Y^{(k)} & \Lambda^{(k)} \end{bmatrix} \begin{bmatrix} \Delta x^{(k)} \\ \Delta \lambda^{(k)} \\ \Delta y^{(k)} \end{bmatrix} = \begin{bmatrix} -H(x^{(k)})x^{(k)} - d(x^{(k)}) + A^T\lambda^{(k)} \\ -Ax^{(k)} + y^{(k)} + b \\ -\Lambda_k Y^{(k)} e + \sigma\mu^{(k)} e \end{bmatrix} \tag{5.63}$$

Las actualizaciones se calculan con

$$\begin{aligned} x^{(k+1)} &= x^{(k)} + \alpha \Delta x^{(k)} \\ \lambda^{(k+1)} &= \lambda^{(k)} + \alpha \Delta \lambda^{(k)} \\ y^{(k+1)} &= y^{(k)} + \alpha \Delta y^{(k)} \end{aligned}$$

y α es seleccionado para mantener $\left(y^{(k+1)}, \lambda^{(k+1)}\right) > 0$

Con el propósito de mejorar el desempeño numérico del método de punto interior, del sistema de ecuaciones (5.63) eliminaremos el termino correspondiente a Δy. Para ello tomamos la última ecuación y la multiplicamos por $[\Lambda^{(k)}]^{-1}$

$$\begin{aligned} \left[\Lambda^{(k)}\right]^{-1} (Y^{(k)}\Delta\lambda^{(k)} + \Lambda^{(k)}\Delta y^{(k)} &= -\Lambda^{(k)}Y^{(k)}e + \sigma\mu_k e) \\ \left[\Lambda^{(k)}\right]^{-1} Y^{(k)}\Delta\lambda^{(k)} + \Delta y^{(k)} &= -Y^{(k)}e + \sigma\mu^{(k)} \left[\Lambda^{(k)}\right]^{-1} e \end{aligned}$$

si sumamos a la segunda ecuación tenemos

$$\begin{aligned} A\Delta x^{(k)} - \Delta y^{(k)} &= -Ax^{(k)} + y^{(k)} + b \\ &+ \\ \left[\Lambda^{(k)}\right]^{-1} Y^{(k)}\Delta\lambda^{(k)} + \Delta y^{(k)} &= -y^{(k)} + \sigma\mu^{(k)} \left[\Lambda^{(k)}\right]^{-1} e \end{aligned}$$

con lo cual obtenemos una ecuación equivalente

$$A\Delta x^{(k)} + \left[\Lambda^{(k)}\right]^{-1} Y^{(k)} \Delta\lambda_k = -Ax^{(k)} + b + \sigma\mu^{(k)} \left[\Lambda^{(k)}\right]^{-1} e$$

Con todo esto las condiciones de KKT reducidas para la ecuación (5.63) quedan como (5.64)

$$\begin{bmatrix} H(x^{(k)}) & -A^T \\ A & \left[\Lambda^{(k)}\right]^{-1} Y^{(k)} \end{bmatrix} \begin{bmatrix} \Delta x^{(k)} \\ \Delta\lambda^{(k)} \end{bmatrix} = \begin{bmatrix} -r_d(x^{(k)}) \\ -r_b(x^{(k)}) \end{bmatrix} \tag{5.64}$$

donde $r_d(x^{(k)}) = \nabla f(x^{(k)}) - A^T\lambda^{(k)}$ y $r_b(x^{(k)}) = y^{(k)} - \sigma\mu^{(k)} \left[\Lambda^{(k)}\right]^{-1} e$. Las actualizaciones las haremos

$$\begin{aligned} x^{(k+1)} &= x^{(k)} + \Delta x^{(k)} \\ \lambda^{(k+1)} &= \lambda^{(k)} + \Delta\lambda^{(k)} \end{aligned}$$

El algoritmo 22, muestra los pasos a seguir para calcular el optimo utilizando el método del punto interior

5.10.1. Cálculo del tamaño de paso óptimo

Para los valores de $x^{(k+1)}$ resolvemos el α tal que $a_i(x^{(k)} + \alpha p^{(k)}) = b_i$ y para los multiplicadores de Lagrange $\lambda^{(k+1)}$ resolvemos $\lambda_i^{(k)} + \alpha\delta\lambda_i^{(k)} \approx 0$

$$\alpha^{(k)} = \text{mín}\left(1, \min_{a_i^T\Delta x_i<0} \frac{b_i - a_i^T x^{(k)}}{a_i^T \Delta x^{(k)}}, \min_{\delta\lambda_i^{(k)}<0} \frac{10^{-6} - \lambda_i^{(k)}}{\delta\lambda_i^{(k)}}\right) \tag{5.65}$$

5.10.2. Ejemplo 1

Calcular el mínimo de la función utilizando el método de punto interior.

$$\begin{aligned} f(x) &= x \\ &\quad s.a \\ 0 \leq x &\leq 1 \end{aligned}$$

Algoritmo 22 Metodo de Punto Interior

Entrada: $f(x), x^{(0)}, \lambda^{(0)}$
Salida: $x^{(k)}$
mientras $k = 0 \dots Max$ **hacer**
Determinar el Hessiano $H(x^{(k)}) = \nabla^2 f(x^{(k)})$ y el gradiente $\nabla f(x^{(k)})$
Calcular:
$y^{(k)} = Ax^{(k)} - b$
$\mu^{(k)} = \frac{1}{m} \left[y^{(k)}\right]^T \lambda^{(k)}$
$\Lambda^{(k)} = diag(\lambda^{(k)})$
$Y^{(k)} = diag(y^{(k)})$
$r_d(x^{(k)}) = \nabla f(x^{(k)}) - A^T \lambda^{(k)}$
$r_b(x^{(k)}) = y^{(k)} - \sigma\mu^{(k)} \left[\Lambda^{(k)}\right]^{-1} e$
Resolver el sistema de ecuaciones dado por (5.64) para $\Delta x^{(k)}$, y $\Delta\lambda^{(k)}$
Actualizar:
Calcular un α tal que $x^{(k+1)}$ y $\lambda^{(k+1)}$ esten en la región de factibilidad (5.65)
$x^{(k+1)} = x^{(k)} + \alpha\Delta x^{(k)}$
$\lambda^{(k+1)} = \lambda^{(k)} + \alpha\Delta\lambda^{(k)}$
si $(\|\Delta x^{(k)}\| + \|\Delta\lambda^{(k)}\| < \epsilon)$ **entonces**
devolver $x^{(k)}$
fin si
fin mientras

Considere como punto inicial $x^{(0)} = 0.8$, $\lambda^{(0)} = [0.1, 0.1]^T$, α =0.2 y $\sigma = 0.2$.

La matriz de restricciones $Ax \geq b$ queda como

$$\begin{bmatrix} 1.0 \\ -1.0 \end{bmatrix} x \geq \begin{bmatrix} 0.0 \\ -1.0 \end{bmatrix}$$

La variable Dual y para la primer iteración es:

$$\begin{aligned} y^{(0)} &= Ax^{(0)} - b \\ y^{(0)} &= \begin{bmatrix} 1.0 \\ -1.0 \end{bmatrix} 0.8 - \begin{bmatrix} 0.0 \\ -1.0 \end{bmatrix} = \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix} \end{aligned}$$

El promedio μ es:

$$\begin{aligned} \mu^{(0)} &= \frac{1}{m} \left[y^{(0)} \right]^T \lambda^{(0)} \\ \mu^{(0)} &= \frac{1}{2} [0.8, 0.2] \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix} \\ \mu^{(0)} &= 0.05 \end{aligned}$$

$$Y^{(0)} = \begin{bmatrix} 0.8 & 0.0 \\ 0.0 & 0.2 \end{bmatrix}$$

$$\left[\Lambda^{(0)} \right]^{-1} = \begin{bmatrix} 10.0 & 0.0 \\ 0.0 & 10.0 \end{bmatrix}$$

Calculamos los valores de r_d y r_b

$$\begin{aligned} r_d(x^{(0)}) &= \nabla f(x^{(0)}) - A^T \lambda^{(0)} \\ r_d(x^{(0)}) &= 1.0 - [1.0, -1.0] \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix} \\ r_d(x^{(0)}) &= 1.0 \end{aligned}$$

$$\begin{aligned} r_b(x^{(0)}) &= y^{(0)} - \sigma\mu^{(0)} \left[\Lambda^{(0)}\right]^{-1} e \\ r_b(x^{(0)}) &= \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix} - 0.2 \times 0.05 \times \begin{bmatrix} 10 \\ 10 \end{bmatrix} \\ r_b(x^{(0)}) &= \begin{bmatrix} 0.70 \\ 0.10 \end{bmatrix} \end{aligned}$$

Con los datos anteriores podemos armar el sistema de ecuaciones a resolver el cual queda como

$$\begin{bmatrix} 0.0 & -1.0 & 1.0 \\ 1.0 & 8.0 & 0.0 \\ -1.0 & 0.0 & 2.0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda_1 \\ \Delta\lambda_2 \end{bmatrix} = \begin{bmatrix} -1.00 \\ -0.70 \\ -0.10 \end{bmatrix}$$

Los resultados de la primer iteración son $\Delta = [-1.6653, 0.1125, -0.8875]^T$ con lo que las actualizaciones son:

$$\begin{aligned} x^{(1)} &= x^{(0)} + \alpha\Delta x^{(0)} \\ x^{(1)} &= 0.8 + 0.2(-1.6653) \\ x^{(1)} &= 0.4669 \end{aligned}$$

$$\begin{aligned} \lambda^{(1)} &= \lambda^{(0)} + \alpha\Delta\lambda^{(0)} \\ \lambda^{(1)} &= \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix} + 0.2 \begin{bmatrix} 0.1125 \\ -0.8875 \end{bmatrix} \\ \lambda^{(1)} &= \begin{bmatrix} 0.1225 \\ -0.0775 \end{bmatrix} \end{aligned}$$

La variable Dual $y^{(1)}$ es:

$$\begin{aligned} y^{(0)} &= Ax^{(0)} - b \\ y^{(0)} &= \begin{bmatrix} 1.0 \\ -1.0 \end{bmatrix} 0.4669 - \begin{bmatrix} 0.0 \\ -1.0 \end{bmatrix} = \begin{bmatrix} 0.4669 \\ 0.5331 \end{bmatrix} \end{aligned}$$

Los resultados del método para este ejemplo se presenta en la tabla 5.17

k	$\|\Delta x\| + \|\Delta\lambda\|$	x
1	1.8904	0.4669
2	8.5709	-1.1762
3	2.5281	-0.6877
4	1.1979	-0.4657
5	0.7413	-0.3324
6	0.4590	-0.2547
7	0.3361	-0.1992
8	0.2678	-0.1548
9	0.2047	-0.1215
⋮	⋮	⋮
63	8.73E-7	0.0

Cuadro 5.17: Resultados del ejemplo 1

5.10.3. Ejemplo 2

Calcular el mínimo de la función utilizando el método de punto interior

$$\begin{aligned} f(x) &= (x_1-2)^2 + (x_2-2)^2 \\ & \quad s.a \\ x_1 + x_2 &\geq 4 \\ 2x_1 - x_2 &\geq 0 \end{aligned}$$

Considere como punto inicial $x_0 = [2,4]^T$, $\lambda_0 = [0.1, 0.1]^T$, α =0.1 y σ =0.2.

Para la primer iteración la matriz característica que debemos resolver es

$$\begin{bmatrix} 2.0 & 0.0 & -1.0 & -2.0 \\ 0.0 & 2.0 & -1.0 & 1.0 \\ 1.0 & 1.0 & 20.0 & 0.0 \\ 2.0 & -1.0 & 0.0 & 0.0 \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta\lambda_1 \\ \Delta\lambda_2 \end{bmatrix} = \begin{bmatrix} 0.3 \\ -4.0 \\ -1.8 \\ 0.2 \end{bmatrix}$$

la solución en la primer iteración es $x = [1.9317, 3.8433]^T$, $\lambda = [0.1022, 0.0156]^T$ y $y = [1.775, 0.02]^T$. Después de Iteración 292 (con $\epsilon = 10^{-6}$), los resultados son $x = [2.0, 2.0]^T$, $\lambda = [0,0]^T$ y $y = [0.0, 2.0]^T$. Analizando el vector y podemos ver que la solución se encuentra sobre la primer restricción y a dos unidades de distancia de la segunda restricción. El sistema a resolver en la última iteración es el siguiente, note que el vector del lado derecho

es cero, el la figura 5.37 se muestra la ruta de convergencia del método.

$$\begin{bmatrix} 2.0 & 0.0 & -1.0 & -2.0 \\ 0.0 & 2.0 & -1.0 & 1.0 \\ 1.0 & 1.0 & 1.0 & 0.0 \\ 2.0 & -1.0 & 0.0 & -1.09E12 \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta \lambda_1 \\ \Delta \lambda_2 \end{bmatrix} = \begin{bmatrix} 0.0 \\ 0.0 \\ 0.0 \\ 0.0 \end{bmatrix}$$

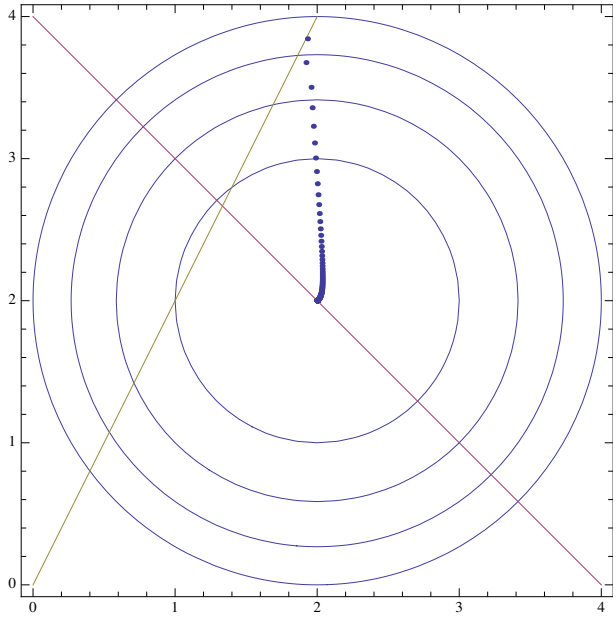


Figura 5.37: Solución del ejemplo 2

Ver Restricciones_ej02.java

5.10.4. Ejemplo 3

Dada la matriz

$$G = \begin{pmatrix} 16 & 8 & 4 & 12 \\ 8 & 29 & -3 & 16 \\ 4 & -3 & 102 & 21 \\ 12 & 16 & 21 & 53 \end{pmatrix}$$

y el vector $d = [5, -2, 10, 6]^T$, calcular el mínimo de la función cuadrática utilizando el método de punto interior.

$$\begin{aligned} f(x) = 0.5x^T Gx – d^T x \\ s.a \\ 0 \leq x_i \leq 1 \\ \forall i = 1 \cdots 4 \end{aligned}$$

Considere que:

$$\begin{aligned} x^{(0)} &= [0.9, 0.9, 0.9, 0.9]^T \\ \lambda^{(0)} &= [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]^T \\ \sigma &= 0.2 \\ \alpha &= 0.2 \end{aligned}$$

Comenzaremos por poner las restricciones en la forma canónica $Ax \geq b$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix}$$

En la tabla 5.18 se presenta el resultado de aplicar el método. Una vez que converge el método podemos calcular las variables de holgura y

$$y^* = [0.2789, 0.0000, 0.0836, 0.0169, 0.7211, 1.0, 0.9164, 0.9831]^T$$

Recordemos que esta variable mide la proximidad a una restricción. Note que el valor de $y_2^* = 0$ lo cual significa que nuestra solución viola la restricción 2 es decir en la restricción $x_2 = 0$

Ver Restricciones_ej03.java

k	$\|\Delta x\| + \|\Delta\lambda\|$	x
1	2.3643	0.7911, 0.6882, 0.7347, 0.7335
2	1.4893	0.7023, 0.5100, 0.6005, 0.6006
3	1.0965	0.6303, 0.3700, 0.4935, 0.4941
4	0.8758	0.5724, 0.2585, 0.4080, 0.4089
5	0.7062	0.5258, 0.1699, 0.3396, 0.3406
6	0.5877	0.4882, 0.1000, 0.2851, 0.2857
7	0.5738	0.4574, 0.0461, 0.2416, 0.2413
⋮	⋮	⋮
84	9.01E-7	0.2789, 0.0, 0.0836, 0.0169

Cuadro 5.18: Resultados de ejemplo 3

5.10.5. Ejemplo 4

Calcular el mínimo de la función

$$\begin{aligned} f(x) &= 2x_1^2 + 3x_2^2 \\ s.a & \\ 3x_1 &+ x_2 \geq 3 \end{aligned}$$

Considere como punto inicial $x^{(0)} = [1, 1]^T$, $\lambda^{(0)} = 0.1$, α =0.1 y σ =0.1.

Para la primer iteración la matriz característica que debemos resolver es

$$\begin{bmatrix} 2.0 & 0.0 & -3.0 \\ 0.0 & 3.0 & -1.0 \\ 3.0 & 1.0 & 10.0 \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta\lambda_1 \end{bmatrix} = \begin{bmatrix} -3.7 \\ -5.9 \\ -0.9 \end{bmatrix}$$

la solución en la primer iteración es $x^{(1)} = [0.8819, 0.8182]^T$, $\lambda_1^{(1)} = 0.1446$ y $y_1^{(1)} = 0.4639$. Después de Iteración 134 (con $\epsilon = 10^{-6}$), los resultados son $x = [0.9310, 0.2069]^T$, $\lambda_1 = 1.2414$ y $y_1 = 0.0$. Analizando el vector y podemos ver que la solución se encuentra sobre la restricción. El sistema a resolver en la última iteración es el siguiente, note que el vector del lado derecho es cero, el la figura 5.38 se muestra la ruta de convergencia del método.

$$\begin{bmatrix} 2.0 & 0.0 & -3.0 \\ 0.0 & 3.0 & -1.0 \\ 3.0 & 1.0 & 0.0 \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \Delta\lambda_1 \end{bmatrix} = \begin{bmatrix} 0.0 \\ 0.0 \\ 0.0 \end{bmatrix}$$

Ver Restricciones_ej04.java

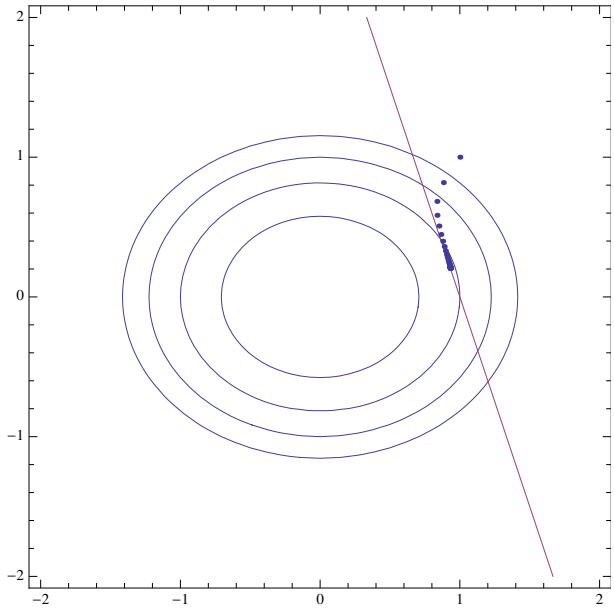


Figura 5.38: Solución del ejemplo 4

Optimización Lineal

6.1. Introducción

Programación lineal se enfoca en el análisis del siguiente problema en forma estándar

$$min \quad c^T x$$

sujeto a

$$Ax = b$$
$$x >= 0$$

donde c es un vector en $\mathbb{R}^N$, b es un vector $\mathbb{R}^M$ y A es una matriz de tamaño $M \times N$. Simples cambios pueden ser usados par transformar cualquier problema lineal a esta forma. Por ejemplo, dado el problema

$$min \quad c^T x \tag{6.66}$$

sujeto a

$$Ax >= b$$

podemos convertir las desigualdades a igualdades introduciendo un vector de variables de holgura z y escribiendo

$$min \quad c^T x$$

sujeto a

$$\begin{aligned} Ax - z &= b \\ z &>= 0 \end{aligned}$$

Podemos particionar los multiplicadores de Lagrange para el problema dado por (6.66) en dos vectores λ y s, donde $\lambda \in \mathbb{R}^M$ es el vector de multiplicador de lagrange para el sistema de igualdades $Ax = b$ mientras que $s \in \mathbb{R}^N$ es el vector de multiplicadores de Lagrange para la restricción $x >= 0$. Utilizando estas condiciones podemos escribir

$$L(x, \lambda, s) = c^T x - \lambda^T (Ax - b) - s^T x$$

Las condiciones necesarias de primer orden para x^* son

$$\begin{aligned} A^T\lambda + s &= c \\ Ax &= b \\ x &> 0 \\ s &> 0 \\ x_i s_i &= 0 \quad \forall i \in [1, .., N] \end{aligned}$$

Para resolver por el método de punto interior tenemos que resolver el sistema de ecuaciones

$$F(x, \lambda, s) = \begin{bmatrix} A^T\lambda + s - c \\ Ax - b \\ XSe \end{bmatrix} = 0$$

Aplicando el método de Newton-Raphson podemos resolver de manera iterativa como

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^{(k)} & 0 & X^{(k)} \end{bmatrix} \begin{bmatrix} \Delta x^{(k)} \\ \Delta \lambda^{(k)} \\ \Delta s^{(k)} \end{bmatrix} = \begin{bmatrix} -rc^{(k)} \\ -rb^{(k)} \\ -X^{(k)} S^{(k)} e + \sigma \mu e \end{bmatrix}$$

donde $rc^{(k)} = A^T\lambda^{(k)} + s^{(k)} - c$, $rb^{(k)} = Ax^{(k)} - b$, $X^{(k)} = diag(x^{(k)})$, $S^{(k)} = diag(s^{(k)})$ y $\mu^{(k)} = [x^{(k)}]^T s/N$ y las actualizaciones las realizaremos como:

$$\begin{aligned} x^{(k+1)} &= x^{(k)} + \Delta x^{(k)} \\ \lambda^{(k+1)} &= \lambda^{(k)} + \Delta\lambda^{(k)} \\ s^{(k+1)} &= s^{(k)} + \Delta s^{(k)} \end{aligned}$$

!TEX root = optimizacion.tex

Tareas

A.1. Tarea 1

Dada la función

$$f(x) = 1 – e^{(-(x-3.1416)^2)}$$

Calcular el mínimo de la función utilizando

a) búsqueda secuencial con un $x_0 = 2.0$ y $h = 0.01$ b) razón dorada en el intervalo $2.0 < x < 4.0$ y c) Utilizando el método de Newton con un valor inicial $x_0 = 0.0$ y $x_0 = 2$ d) Concluir respecto al comportamiento de los métodos y el número de iteraciones realizadas.

A.2. Tarea 2

Dada la función en tres dimensiones

$$f(x_1, x_2, x_3) = (1 - x_1)^2 + (1 - x_2)^2 + 100(x_2 - x_1^2)^2 + 100(x_3 - x_2^2)^2$$

Calcular en el punto $x_0 = [0.8, 0.7, 0.9]^T$:

a) El vector Gradiente y la dirección de búsqueda de máximo descenso, b) La gráfica de la función phi(alpha) en la dirección de máximo descenso c) Cual es el mínimo de la función phi(alpha)

A.3. Tarea 3

Dadas la funciones

$$f_1(x, y) = xe^{(-x^2-y^2)}$$

con $X_0 = [0.5, 0.5]^T$

$$f_2(x, y) = xseno(y)$$

con $x_0 = [0.8, 4]^T$

Determinar para cada una de la funciones en los puntos iniciales :

a) La grafica de la función phi(alpha), b) El rango donde se cumplen las condiciones fuertes de Wolfe, c) El valor mínimo de alfa utilizando razón dorada, d) El valor mínimo de alfa utilizando máximo descenso, e) El valor mínimo de alfa utilizando interpolación y f) El valor mínimo de alfa utilizando el método de Newton para alpha de máximo descenso g) Concluir

A.4. Tarea 4

Dada la función de Rosenbrock en tres dimensiones

$$f(x_1, x_2, x_3) = (1 - x_1)^2 + (1 - x_2)^2 + 100(x_2 - x_1^2)^2 + 100(x3 - x_2^2)^2$$

y un punto inicial $x_0 = [0.8, 0.7, 0.9]^T$, para los métodos de Máximo descenso y Forsite con 2 pasos hacer :

a) llenar una tabla como la siguiente tabla

k x_k $f(x_k)$ p_k o d_k 0 1 2 3 4 ... k_{max}

b) Hacer una grafica de k vs $f(x_k)$ c) calcular el ángulo entre las direcciones k y $k+1$ para cada método.

A.5. Tarea 5

Dada la matriz

$$A = \begin{pmatrix} 16 & 8 & 4 & 12 \\ 8 & 29 & -3 & 16 \\ 4 & -3 & 102 & 21 \\ 12 & 16 & 21 & 53 \end{pmatrix}$$

y el vector $b = [5, -2, 10, 6]^T$, calcular el mínimo de la función cuadrática $f(x) = 0.5x^T Ax – b^T x$ utilizando :

a) La solución exacta y b) El método de Gradiente Conjugado

A.6. Tarea 6

Para una imagen dada en formato ppm aplicar el método de regularización y minimizar la función $E(f)$ utilizando

a) El método de Gauss-Seidel, b) El método de Descenso de Gradiente (verificar que las direcciones están a 90 grados) c) Solucionar utilizando el método de Gradiente conjugado d) Solucionar utilizando el método de Gradiente conjugado precondicionado e) Hacer una grafica donde se muestre la función de error E(f) contra el tiempo

A.7. Tarea 7

Para la función $xe^{(-x^2-y^2)}$ y dado un punto inicial $x_0 = -0.5$ y $y_0 = -0.5$ determinar el mínimo utilizando

a) El método de Newton, b) El método de descenso de Gradiente, c) El método de Gradiente Conjugado PR+, d) El método de Gradiente Conjugado FR y e) Hacer la mejora al método

A.8. Tarea 8

Escribir dos funciones para hacer la evaluación del Jacobiano y del Hessiano para el método de Gauss-Seidel aplicado al problema de registro de imágenes

A.9. Tarea 9

Hacer la implementación de los algoritmos de Gauss-Seidel y Levenberg Marquart aplicado al registro de imágenes.

A.10. Tarea 10

Para la función $f(x, y) = xe^{(-x^2-y^2)}$, sujeto a $x = y$ calcular de manera analitica el mínimo de la función utilizando multiplicadores de Lagrange.

A.11. Tarea 11

Dada la matriz

$$A = \begin{pmatrix} 16 & 8 & 4 & 12 \\ 8 & 29 & -3 & 16 \\ 4 & -3 & 102 & 21 \\ 12 & 16 & 21 & 53 \end{pmatrix}$$

y el vector $b = [5, -2, 10, 6]^T$, calcular el mínimo de la función cuadrática $f(x) = 0.5x^T Ax – b^T x$ s. a $0 \leq x_i \leq 1$, $\forall i = 1 \cdots 4$. Considere como valor inicial $x_i = 1.0$

!TEX root = optimizacion.tex

Algoritmo de factorización de Cholesky

B.1. Factorización de Cholesky

Dado un sistema de ecuaciones $Ax = b$, si sustituimos la matriz A por una matriz factorizada utilizando Choleski, tenemos que $A = L^T L$ y el sistema de ecuaciones a resolver es:

$$[L^T L]x = b \tag{B.67}$$

donde L es una matriz triangular superior, D es una matriz diagonal y L^T una matriz triangular inferior. En lugar de resolver el sistema de ecuaciones B.67, resolveremos dos sistemas de ecuaciones dados por:

$$L^T y = b$$

$$Lx = y$$

B.1.1. Ejemplo

Dada la factorización de una matriz A, determinar

- El producto de las $L^T L$ y
- la solución del sistema de ecuaciones

$$L = \begin{bmatrix} 4 & 2 & 3 & 1 \\ 0 & -5 & 3 & 2 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 2 \end{bmatrix}$$

$$L^T = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 2 & -5 & 0 & 0 \\ 3 & 3 & 2 & 0 \\ 1 & 2 & 1 & 2 \end{bmatrix}$$

El producto $L^T L$ es :

$$\begin{bmatrix} 4 & 2 & 3 & 1 \\ 0 & -5 & 3 & 2 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} 4 & 0 & 0 & 0 \\ 2 & -5 & 0 & 0 \\ 3 & 3 & 2 & 0 \\ 1 & 2 & 1 & 2 \end{bmatrix}$$

lo cual da

$$A = \begin{bmatrix} 16 & 8 & 12 & 4 \\ 8 & 29 & -9 & -8 \\ 12 & -9 & 22 & 11 \\ 4 & -8 & 11 & 10 \end{bmatrix}$$

Comenzaremos por resolver el sistema de ecuaciones $L^T y = b$

$$\begin{bmatrix} 4 & 0 & 0 & 0 \\ 2 & -5 & 0 & 0 \\ 3 & 3 & 2 & 0 \\ 1 & 2 & 1 & 2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 20 \\ -4 \\ 30 \end{bmatrix}$$

Utilizando sustitución hacia adelante (ver ecuación B.68) tenemos

$$y = [0.2500, -3.9000, 3.4750, 17.0375]^T$$

$$y_k = \frac{b_k - \sum_{j=0}^{j<k} a_{k,j} * b_j}{a_{k,k}} \tag{B.68}$$

Como segundo paso resolvemos el sistema $Lx = y$ utilizando sustitución hacia atrás (ver ecuación B.69)

$$\begin{bmatrix} 4 & 2 & 3 & 1 \\ 0 & -5 & 3 & 2 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \begin{bmatrix} 0.2500 \\ -3.9000 \\ 3.4750 \\ 17.0375 \end{bmatrix}$$

con $x = [-1.5130, 2.6744, -2.5219, 8.5188]^T$

$$x_k = \frac{b_k - \sum_{j=k+1}^{j=N} a_{k,j} * y_j}{a_{k,k}} \tag{B.69}$$

B.2. Algoritmo

La factorización de Cholesky solamente es aplicable a sistemas de ecuaciones cuya matriz sea definida positiva y simétrica. El algoritmo lo podemos desarrollar a partir de una matriz A la cual escribimos como el producto de $L^T * L$.

Así, dada una matriz A

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix}$$

podemos escribir esta como

$$A = L^T L = \begin{bmatrix} l_{1,1} & 0 & 0 \\ l_{2,1} & l_{2,2} & 0 \\ l_{3,1} & l_{3,2} & l_{3,3} \end{bmatrix} \begin{bmatrix} l_{1,1} & l_{1,2} & l_{1,3} \\ 0 & l_{2,2} & l_{2,3} \\ 0 & 0 & l_{3,3} \end{bmatrix}$$

Para desarrollar el algoritmo hacemos elemento a elemento el producto de las matrices $L^T L$, así para los elementos del primer renglón se calculan como:

$$\begin{array}{rc} j = 1 & a_{1,1} = l_{1,1} l_{1,1} \\ & l_{1,1} = \sqrt{a_{1,1}} \\ \therefore & a_{1,1} = \sqrt{a_{1,1}} \\ j = 2 & a_{1,2} = l_{1,1} l_{1,2} \\ & l_{1,2} = a_{1,2} / l_{1,1} \\ \therefore & a_{1,2} = a_{1,2} / a_{1,1} \\ j = 3 & a_{1,3} = l_{1,1} l_{1,3} \\ & l_{1,3} = a_{1,3} / l_{1,1} \\ \therefore & a_{1,3} = a_{1,3} / a_{1,1} \end{array}$$

para el segundo renglón $i = 2$

$$\begin{array}{rc} j = 1 & a_{2,1} = l_{2,1} l_{1,1} \\ & l_{2,1} = a_{2,1} / l_{1,1} \\ \therefore & a_{2,1} = a_{2,1} / a_{1,1} \\ j = 2 & a_{2,2} = l_{2,1} l_{1,2} + l_{2,2} l_{2,2} \\ & l_{2,2} = \sqrt{a_{2,2} - l_{2,1} l_{1,2}} \\ \therefore & a_{2,2} = \sqrt{a_{2,2} - a_{2,1} a_{1,2}} \\ j = 3 & a_{2,3} = l_{2,1} l_{1,2} + l_{2,2} l_{2,3} \\ & l_{2,3} = [a_{2,3} - l_{2,1} l_{1,3}] / l_{2,2} \\ \therefore & a_{2,3} = [a_{2,3} - a_{2,1} a_{1,3}] / a_{2,2} \end{array}$$

Finalmente para $i = 3$

$$\begin{array}{cc} j = 1 & a_{3,1} = l_{3,1}l_{1,1} \\ & l_{3,1} = a_{3,1}/l_{1,1} \\ \therefore & a_{3,1} = a_{3,1}/a_{1,1} \\ j = 2 & a_{3,2} = l_{3,1}l_{1,2} + l_{3,2}l_{2,2} \\ & l_{3,2} = [a_{3,2} - l_{3,1}l_{1,2}]/l_{2,2} \\ \therefore & a_{3,2} = [a_{3,2} - a_{3,1}a_{1,2}]/a_{2,2} \\ j = 3 & a_{3,3} = l_{3,1}l_{1,3} + l_{3,2}l_{2,3} + l_{3,3}l_{2,3} \\ & l_{3,3} = \sqrt{a_{3,3} - l_{3,1}l_{1,3} - l_{3,2}l_{2,3}} \\ \therefore & a_{3,3} = \sqrt{a_{3,3} - a_{3,1}a_{1,3} - a_{3,2}a_{2,3}} \end{array}$$

Dado que la matriz es simétrica, solamente calcularemos los elementos de L^T, con :

$$a_{i,j} = \frac{a_{i,j} - \sum_{k=0}^{k<j} a_{i,k} * a_{j,k}}{a_{j,j}}$$

Para los elementos en la diagonal de L hacemos:

$$a_{i,i} = \sqrt{a_{i,i} - \sum_{k=0}^{k<i} a_{i,k}^2}$$

La implementación en Java es

```
static public void Cholesky(double A[][]) {
  int i, j, k, n, s;
  double fact, suma = 0;

  n = A.length;

  for (i = 0; i < n; i++) {
    for (j = 0; j <= i - 1; j++) {
     suma = 0;
      for (k = 0; k <= j - 1; k++)
         suma += A[i][k] * A[j][k];
      A[i][j] = (A[i][j] - suma) / A[j][j];
    }

    suma = 0;
    for (k = 0; k <= i - 1; k++)
      suma += A[i][k] * A[i][k];
    A[i][i] = Math.sqrt(A[i][i] - suma);
  }
}
```

Ejemplo

Realizar la factorización de Cholesky de la matriz

$$A = \begin{pmatrix} 4 & 2 & 0 & 2 \\ 2 & 5 & 2 & 1 \\ 0 & 2 & 17 & -4 \\ 2 & 1 & -4 & 11 \end{pmatrix}$$

Primer renglón

$a_{1,1} = \sqrt{a_{1,1}} = \sqrt{4} = 2$

Segundo renglón

$a_{2,1} = a_{2,1}/a_{1,1} = 2/2 = 1$

$a_{2,2} = \sqrt{a_{2,2} - a_{2,1}^2} = \sqrt{5 - 1^2} = 2$

Tercer renglón

$a_{3,1} = a_{3,1}/a_{1,1} = 0/2 = 0$

$a_{3,2} = [a_{3,2} - a_{3_1} * a_{2,1}]/a_{2,2} = [1 - 0 * 1]/1 = 1$

$a_{3,3} = \sqrt{a_{3,3} - a_{3,1}^2 - a_{3,2}^2} = \sqrt{17 - 0^2 - 1^2} = 4$

Cuarto renglón

$a_{4,1} = a_{4,1}/a_{1,1} = 2/2 = 1$

$a_{4,2} = [a_{4,2} - a_{4,1} * a_{2,1}]/a_{2,2} = [1 - 1 * 1]/1 = 0$

$a_{4,3} = [a_{4,3} - a_{4,1} * a_{3,1} - a_{4,2}a_{3,2}]/a_{3,3} = [-4 - 1 * 0 - 01]/4 = -1$

$a_{4,4} = \sqrt{a_{4,4} - a_{4,1}^2 - a_{4,2}^2 - a_{4,3}^2} = \sqrt{11 - 1^2 - 0^2 - 1^2} = \sqrt{9} = 3$

Finalmente la matriz L^T queda

$$L^T = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 0 & 1 & 4 & 0 \\ 1 & 0 & -1 & 3 \end{bmatrix}$$

B.3. Algoritmo de Cholesky incompleto

Para este algoritmo procedemos de manera similar que el Algoritmo complejo, la diferencia estriba que en aquellas localidades donde exista un cero en la matriz original, los elementos de la matriz factorizada no se calculan y en su lugar se coloca un cero

Ejemplo

Para la siguiente matriz

$$A = \begin{pmatrix} 3 & -1 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 3 & 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 & 3 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 3 \end{pmatrix}$$

con un vector de términos independientes

$$b = [1, 2, 3, 2, 1, 2, 3, 2]$$

Calcular:

a) La solución utilizando matriz inversa. La matriz inversa de A es :

$$A^{-1} = \begin{pmatrix} 0.4414 & 0.1466 & 0.0534 & 0.0253 & 0.1776 & 0.0915 & 0.0418 & 0.0224 \\ 0.1466 & 0.3482 & 0.1184 & 0.0534 & 0.0915 & 0.1280 & 0.0720 & 0.0418 \\ 0.0534 & 0.1184 & 0.3482 & 0.1466 & 0.0418 & 0.0720 & 0.1280 & 0.0915 \\ 0.0253 & 0.0534 & 0.1466 & 0.4414 & 0.0224 & 0.0418 & 0.0915 & 0.1776 \\ 0.1776 & 0.0915 & 0.0418 & 0.0224 & 0.4414 & 0.1466 & 0.0534 & 0.0253 \\ 0.0915 & 0.1280 & 0.0720 & 0.0418 & 0.1466 & 0.3482 & 0.1184 & 0.0534 \\ 0.0418 & 0.0720 & 0.1280 & 0.0915 & 0.0534 & 0.1184 & 0.3482 & 0.1466 \\ 0.0224 & 0.0418 & 0.0915 & 0.1776 & 0.0253 & 0.0534 & 0.1466 & 0.4414 \end{pmatrix}$$

y la solución del sistema de ecuaciones calculado con $A^{-1} * b$ es

$x = [1.4762, 1.9524, 2.3810, 2.1905, 1.4762, 1.9524, 2.3810, 2.1905]^T$

b) La solución aplicando factorización de Cholesky.

Aplicando la factorización de Cholesky tenemos $A = L^T L$ donde L^T es :

$$L^T = \begin{bmatrix} 1,7321 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0,5774 & 1,9149 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0,5222 & 1,9306 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0,518 & 1,6528 & 0 & 0 & 0 & 0 \\ -0,5774 & -0,1741 & -0,0471 & -0,0148 & 1,6229 & 0 & 0 & 0 \\ 0 & -0,5222 & -0,1413 & -0,0443 & -0,6767 & 1,8021 & 0 & 0 \\ 0 & 0 & -0,518 & -0,1623 & -0,0165 & -0,6057 & 1,8271 & 0 \\ 0 & 0 & 0 & -0,605 & -0,0055 & -0,0169 & -0,6067 & 1,5052 \end{bmatrix}$$

Dado que $L^T Ly = b$, usando sustitución hacia adelante, encontramos:

$$y = \begin{bmatrix} 0.5774 & 1.2185 & 1.8835 & 1.8004 & 1.02328 & 2.0391 & 3.0211 & 3.2970 \end{bmatrix}^T$$

Ahora aplicando sustitución hacia atrás, resolvemos el sistema Lx=y, cuya solución es:

$$x = \begin{bmatrix} 1.4762 & 1.9524 & 2.3810 & 2.1905 & 1.4762 & 1.9524 & 2.3810 & 2.1905 \end{bmatrix}^T$$

Note que la solución en similar a la del inciso (a)

c) La solución aplicando Cholesky incompleto.

La solución aplicando Cholesky incompleto es:

$$\hat{L} = \begin{bmatrix} 1,7321 & -1 & 0 & 0 & -1 & 0 & 0 & 0 \\ -0,5774 & 1,9149 & -1 & 0 & 0 & -1 & 0 & 0 \\ 0 & -0,5222 & 1,9306 & -1 & 0 & 0 & -1 & 0 \\ 0 & 0 & -0,518 & 1,6528 & 0 & 0 & 0 & -1 \\ -0,5774 & 0 & 0 & 0 & 1,633 & -1 & 0 & 0 \\ 0 & -0,5222 & 0 & 0 & -0,6124 & 1,8309 & -1 & 0 \\ 0 & 0 & -0,518 & 0 & 0 & -0,5462 & 1,8529 & -1 \\ 0 & 0 & 0 & -0,605 & 0 & 0 & -0,5397 & 1,5306 \end{bmatrix}$$

Dado que $\hat{L}^T \hat{L}x = b$, usando sustitución hacia adelante, encontramos:

$$\hat{y} = \begin{bmatrix} 0.5774 & 1.2185 & 1.8835 & 1.8004 & 0.8165 & 1.71300 & 2.6505 & 2.9529 \end{bmatrix}^T$$

Usando sustitución hacia atrás:

$$\hat{x} = \begin{bmatrix} 1.2235 & 1.5969 & 1.9919 & 1.7955 & 1.0737 & 1.5299 & 1.9924 & 1.9292 \end{bmatrix}$$

!TEX root = optimizacion.tex

Ejemplos

C.1. Ejemplo 1

A continuación se presentan los resultados obtenidos con la implementación para la siguiente función:

$$f(x, y, z) = \cos(z^2 - x^2 - y^2) \tag{C.70}$$

con un punto inicial $x = -2$, $y = -1$ y $z = -0.1$

El gradiente para esta función:

$$\nabla f(x, y, z) = \begin{pmatrix} 2x \sin(z^2 - x^2 - y^2) \\ 2y \sin(z^2 - x^2 - y^2) \\ -2z \sin(z^2 - x^2 - y^2) \end{pmatrix} \tag{C.71}$$

Su Hessiano:

$$\nabla^2 f(x, y, z) =$$

$$\begin{pmatrix} 2\sin(z^2 - x^2 - y^2) - 4x^2 \cos(z^2 - x^2 - y^2) & -4xy \cos(z^2 - x^2 - y^2) & 4xz \cos(z^2 - x^2 - y^2) \\ -4xy \cos(z^2 - x^2 - y^2) & 2\sin(z^2 - x^2 - y^2) - 4y^2 \cos(z^2 - x^2 - y^2) & 4yz \cos(z^2 - x^2 - y^2) \\ 4xz \cos(z^2 - x^2 - y^2) & 4yz \cos(z^2 - x^2 - y^2) & -2\sin(z^2 - x^2 - y^2) - 4z^2 \cos(z^2 - x^2 - y^2) \end{pmatrix} \tag{C.72}$$

Iteraciones

La siguiente tabla muestra los valores de $|\nabla f(x_k, y_k, z_k)|^2$ para los diferentes métodos, se asume que el algoritmo ha concluido cuando $|\nabla f(x_k, y_k, z_k)|$ es menor que 1×10^{-6}

k	GC FR	GC PR+	Newton	Secantes	BFGS	DG
0	18.534829	18.534829	18.534829	18.534829	18.534829	18.534829
1	5.9961×10^{-12}	18.534829	19.628691	19.628691	19.628691	4.9263×10^{-12}
2	1.4080×10^{-15}	5.9961×10^{-12}	1.758437	6137.0593	217310.73	1.4081×10^{-15}
3	0	5.9952×10^{-12}	6.9861×10^{-7}	144.96555	808444.91	0
4	0	1.4081×10^{-15}	3.4495×10^{-17}	351.09507	674857.35	0
5	0	0	0	107.18124	107.18124	0
6	0	0	0	284.91499	284.91495	0
7	0	0	0	297.44935	297.44960	0
8	0	0	0	3.679177	3.678912	0
9	0	0	0	396.48686	396.47662	0
10	0	0	0	0.505325	0.505207	0
11	0	0	0	0.053475	0.053454	0
12	0	0	0	5.5675×10^{-10}	5.5652×10^{-10}	0
13	0	0	0	5.0543×10^{-18}	5.0476×10^{-18}	0
$f(x,y,z)$	-0.99999999	-0.99999999	1.0	-1.0	-1.0	-0.99999999

Resultados

Para el método GC-FR la solución obtenida es:

$$x = -1.588993403399196$$
$$y = -0.794496701699598$$
$$z = -0.120550325305159$$

Para el método GC-PR+ la solución obtenida es

$$x = -1.588993403364383$$
$$y = -0.794496701682191$$
$$z = -0.120550324628847$$

Para el método de Newton la solución obtenida (*que no es mínimo*) es

$$x = -3.173837263856202$$
$$y = -1.586918631928101$$
$$z = -0.158691863192810$$

Para el método de secantes la solución obtenida es

$$x = -9.388337235773939$$
$$y = -4.694168617886971$$
$$z = -0.469416861788697$$

Para el método de secantes con actualización BFGS la solución obtenida es

$$x = -9.388337235773939$$
$$y = -4.694168617886971$$
$$z = -0.469416861788697$$

Para el método de descenso de gradiente la solución obtenida es

$$x = -1.588993398143894$$
$$y = -0.794496699071947$$
$$z = -0.120550326121605$$

Conclusiones

Las conclusiones aplican únicamente para esta función en particular, pues el comportamiento de los algoritmos depende en gran medida de la función y del punto inicial que se dé.

En cuestión de rendimiento el mejor desempeño lo tiene el método de Gradiente Conjugado de Flecher-Reeves junto con el Descenso de gradiente utilizando búsqueda lineal con muestreo hacia atrás y refinando con razón dorada. El peor desempeño se observó en el método de secantes y secantes con actualización BFGS, ambos con 13 iteraciones.

Cabe observar que ambos métodos de secantes llegan a un mínimo, aunque bastante lejano del punto inicial y de las soluciones dadas por los otros algoritmos.

El método de Newton, a pesar de tener un desempeño comparable al de Polak-Ribiere, no tiene un comportamiento deseable, pues la solución a la que llegó es un máximo.

C.2. Ejemplo 2

Dado $f(x, y) = x^2 + y^2 - y^3$ y $x_0 = [-0.1, -0.5]$ calcular x^* por los siguientes métodos:

a)El método descenso de gradiente

b)El método Flecher y Reaves

c)El método de Polak - Ribiere

d)El método de Newton

e)El método de Broyden's o secante

f)El método de Secante con actualización BFGS

El gradiente de la función es:

$$\nabla f(x) = \begin{bmatrix} 2x \\ 2y - 3y^2 \end{bmatrix}$$

El Hessiano de la función es:

$$\nabla^2 f(x) = \begin{bmatrix} 2 & 0 \\ 0 & 2-6y \end{bmatrix}$$

Resultados: en las tablas se muestra las iteraciones, el gradiente y en la parte inferior de las tablas se muestra el mínimo

Iteracion	**DG**	**FR**	**PR**
1	1.0529000000E+00	1.0261091560E+00	1.0529000000E+00
2	8.6524469501E-04	2.8641989200E-02	8.2036354532E-04
3	3.7906801746E-07	9.5247029376E-04	9.0815819313E-07
4	4.9409060070E-11	3.1628052834E-05	2.5771723344E-11
5	6.9412498475E-15	1.0500360359E-06	
x	-6.5710455395E-13	-6.0350436068E-09	-1.8607331472E-13
y	4.9395327839E-10	1.6352268545E-08	7.0174475477E-13

Iteracion	**Newton**	**Secante**	**BFGS**
1	1.0261091560E+00	1.0529000000E+00	1.0529000000E+00
2	2.3476331361E-02	5.5113813417E-04	5.5113813417E-04
3	3.8615695475E-04	2.4805908846E-05	2.4105600535E-05
4	1.1170848425E-07	7.0759602555E-09	6.8456011946E-09
5			
x	0.0000000000E+00	0.0000000000E+00	-6.0633893583E-11
y	-4.6795429763E-15	-9.8360197309E-12	1.0984631815E-11

En los resultados se observa que los mejores métodos para esta función son PR+, Newton, secante, BFGS ya que convergen en 4 iteraciones.

C.3. Ejemplo 3

A continuación se presentan los resultados obtenidos con la implementación para la siguiente función:

$$f(x,y) = 100(y-x^2)^2 + (1-x)^2 \tag{C.73}$$

desarrollando se tiene $f(x,y) = 100y^2 - 200x^2y + 100x^4 + x^2 - 2x + 1$. El punto inicial dado es: $x = 2$ e $y = 2$.

El gradiente para esta función:

$$\nabla f(x,y) = \begin{pmatrix} 400x^3 - 400xy + 2x - 2 \\ 200y - 200x^2 \end{pmatrix} \tag{C.74}$$

Su Hessiano:

$$\nabla^2 f(x,y) = \begin{pmatrix} 1200x^2 - 400y + 2 & -400x \\ -400x & 200 \end{pmatrix} \tag{C.75}$$

Iteraciones

La siguiente tabla muestra el valor de $|\nabla f(x_k, y_k)|^2$ en las primeras iteraciones para los diferentes métodos, se asume que el algoritmo ha concluido cuando $|\nabla f(x_k, y_k)|$ es menor que 1×10^{-6}

k	GC FR	GC PR+	Newton	Secantes	BFGS	DG
0	2726404.00	2726404.00	2726403.99	2726403.99	2726403.99	2726403.99
1	3.102538	2726404.00	3.999926	3.999926	3.999926	0.089682
2	9.092944	3.102538	197423.209	4.019346	3.979929	11.30171
3	1388.452	9.044424	6.0804×10^{-6}	66835.009	196440.73	0.087853
4	2250.645	1384.391	4.6213×10^{-7}	2.584125	4.649582	10.74220
5	2687.140	2195.172	1.3331×10^{-28}	8.053000	4.537298	0.086080
6	2897.356	113.0063	0	74.92009	7.120902	10.21182
7	2977.956	4.429209	0	34.93566	1.751424	0.084360
8	2980.577	111.1229	0	26.62128	3.199347	9.722758
9	2935.235	168.1488	0	39778.14	3.641651	0.082691
10	2860.440	17.50143	0	0.516530	4.272829	9.260353
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$f(x,y,z)$	9.0483×10^{-14}	0.0	0.0	0.0	0.0	3.0464×10^{-13}
Iteraciones	163	53	5	154	131	2037

Resultados

Para el método GC-FR la solución obtenida es:

$$x = 0.9999997237601747$$
$$y = 0.9999994446670981$$

Para el método GC-PR+ la solución obtenida es

$$x = 1.0000000002512976$$
$$y = 1.000000004827209$$

Para el método de Newton la solución obtenida es

$$x = 1.0$$
$$y = 1.0$$

Para el método de secantes la solución obtenida es

$$x = 0.9999999991748632$$
$$y = 0.9999999983510515$$

Para el método de secantes con actualización BFGS la solución obtenida es

$$x = 0.9999999999946342$$
$$y = 0.9999999999891986$$

Para el método de descenso de gradiente la solución obtenida es

$$x = 1.000000552035356$$
$$y = 1.000001103577654$$

Conclusiones

Las conclusiones aplican únicamente para esta función en particular, pues el comportamiento de los algoritmos depende en gran medida de la función y del punto inicial que se dé.

En cuestión de rendimiento el mejor desempeño lo tiene el método de Newton que además de ser el único en llegar a la solución exacta hace sólo 5 iteraciones. Seguido por Polak-Ribiere en 53 iteraciones. Tanto Gradiente Conjugado de Flecher-Reeves, secantes y secantes con actualización BFGS muestran un desempeño similar con 163, 154 y 131 iteraciones respectivamente. El método Descenso de gradiente utilizando búsqueda lineal con muestreo hacia atrás y refinando con razón dorada exhibe el mayor número de iteraciones, haciendo 2037 iteraciones.

Bibliografía

[Chapra and Canale, 2000] Chapra, S. C. and Canale, R. (2000). *Numerical Methods for Engineers: With Software and Programming Applications*. John Wiley and Song.

[Dennis and Schnabel, 1996] Dennis, J. E. and Schnabel, R. B. (1996). *Numerical Methods for Unconstrained Optimization and Nonlinear Equiations*. Society for Industrial and Applied Mathematics. siam, Philadelphia.

[Leithold, 1981] Leithold, L. (1981). *Calculus With Analytic Geometry*.

[Nocedal and Wright, 1999] Nocedal, J. and Wright, S. (1999). *Numerical Optimization,*. Springer Verlag.