Métodos de búsqueda por línea

Octavio Alberto Agustín Aquino

4 de junio de 2005

Índice

Índice de figuras

1. Búsqueda por línea

La iteración en una búsqueda por línea calcula una dirección de descenso en cada paso. La iteración está dada por

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

donde el escalar positivo α_k es el *tamaño de paso*. Una forma de garantizar que la elección del tamaño de paso es adecuada, se utilizan las condiciones de Wolfe:

$$\begin{aligned} f(x_k + \alpha_k p_k) &\leq f(x_k) + c_1 \alpha_k \nabla f_k^T p_k & (2)\\ \nabla f\,(x_k + \alpha_k p_k)^T p_k &\geq c_2 \nabla f_k^T p_k. & (3) \end{aligned}$$

Un tamaño de paso puede satisfacer las condiciones de Wolfe y aún estar lejos de ser óptimo. Esto puede subsanarse utilizando la siguiente condición:

$$|\nabla (f_k + \alpha_k p_k)^T p_k| \leq c_k |\nabla f_k^T p_k|, \tag{4}$$

donde $0 < c_1 < c_2 < 1$.

1.1. Convergencia

Teorema 1 (Zoutendijk). *Considérese cualquier iteración de la forma* (1)*, donde* p_k *es una dirección de descenso y* α_k *satisface las condiciones de Wolfe* (2) *y* (3)*. Supóngase que* f *está acotada por debajo en* $\mathbb{R}^n$ *y que* f *es continuamente diferenciable es un conjunto abierto* $\mathcal{N}$ *que contiene el conjunto*

$$\mathcal{L} := \{x : f(x) \leq f(x_0)\},$$

donde x_0 *es el punto de partida de la iteración. Supongamos, además, que el gradiente* ∇f *es continua según Lipschitz en* $\mathcal{N}$*, esto es, existe una constante* $L > 0$ *tal que*

$$\|\nabla f(x) - \nabla f(\tilde{x})\| \leq L\|x - \tilde{x}\|, \qquad \textit{para todo } x, \tilde{x} \in \mathcal{N}. \tag{5}$$

Entonces

$$\sum_{k \geq 0} \cos^2 \theta_k \|\nabla f_k\|^2 < \infty. \tag{6}$$

Demostración. De (4) y de (1) tenemos que

$$\nabla f_{k+1} p^T - \nabla f_k p^T \geq c_2 \nabla f_k p^T - \nabla f_k p^T$$
$$(\nabla f_{k+1} - \nabla f_k)^T p_k \geq (c_2 - 1) \nabla f_k^T p_k.$$

La condición de Lipschitz (5),

$$\|\nabla f_{k+1} - \nabla f_k\| \leq \alpha_k L \|p_k\|,$$

aunado a la desigualdad de Cauchy-Schwarz,

$$(\nabla f_{k+1} - \nabla f_k)^T p_k \leq \|\nabla f_{k+1} - \nabla f_k\| \|p_k\|,$$

arroja la siguiente desigualdad:

$$(\nabla f_{k+1} - \nabla f_k)^T p_k \leq \alpha_k L \|p_k\|^2.$$

Combinando ambas relaciones tenemos

$$\alpha_k \geq \frac{c_2 - 1}{L} \frac{\nabla f_k^T p_k}{\|p_k\|^2},$$

y sustituyendo esta desigualdad en la primera condición de Wolfe (2), obtenemos

$$f_{k+1} \leq f_k - c \cos^2 \theta_k \|\nabla f_k\|^2,$$

donde $c = c_1(1 - c_2)/L$. Sumando esta expresión sobre todos los índices menores o iguales que k, obtenemos

$$f_{k+1} \leq f_0 - c \sum_{j=0}^{k} \cos^2 \theta_j \|\nabla f_k\|^2. \tag{7}$$

Como f está acotada por debajo, tenemos que $f_0 - f_{k+1}$ es menor que alguna constante positiva, para toda k. Por lo tanto, tomando límites en (7), obtenemos

$$\sum_{j=0}^{\infty} \cos^2 \theta_j \|\nabla f_k\|^2 < \infty,$$

lo que concluye la prueba. ■

Este teorema muestra que el método del descenso más pronunciado, donde p_k se elige como

$$p_k = \frac{\nabla f_k}{\|f_k\|},$$

converge globalmente.

1.2. Tasa de convergencia

Puede aprenderse mucho del método del descenso más pronunciado considerando el caso ideal, en el cual la función es cuadrática y las búsquedas por línea son exactas. Supongamos que

$$f(x) = \frac{1}{2}x^T Q x - b^T x, \tag{8}$$

donde Q es una matriz simétrica definida positiva. El gradiente está dado por

$$\nabla f(x) = Qx - b,$$

y el minimizador x^* es la única solución del sistema lineal

$$Qx = b.$$

Calculemos el tamaño de paso α_k que minimice $f(x_k - \alpha \nabla f_k)$. Diferenciando

$$f(x_k - \alpha g_k) = \frac{1}{2}(x_k - \alpha g_k)^T Q (x_k - \alpha g_k) - b^T (x_k - \alpha g_k)$$

respecto a α obtenemos

$$\alpha_k = \frac{\nabla f_k^T \nabla f_k}{\nabla f_k^T Q \nabla f_k}. \tag{9}$$

Usando esta dirección la ecuación (1) nos queda

$$x_{k+1} = x_k - \left(\frac{\nabla f_k^T \nabla f_k}{\nabla f_k^T Q \nabla f_k} \right) \nabla f_k, \tag{10}$$

y considerando que $\nabla f_k = Q x_k - b$, esta ecuación nos da una expresión en forma cerrada de x_{k+1} en términos de x_k.

Para cuantificar la tasa de convergencia introducimos la norma pesada

$$\|x\|_Q^2 = x^T Q x,$$

y observamos que

$$\begin{aligned} \|x - x^*\|_Q^2 &= (x - x^*)^T Q (x - x^*) = (x - x^*)^T (Qx - b) \\ &= (x - x^*)^T Q x - (x - x^*)^T b = x^T Q x - x^T b - (x^{*T} Q x^* - x^{*T} b) \\ &= 2(f(x) - f(x^*)), \end{aligned}$$

así que esta norma mide la diferencia entre el valor objetivo actual y el valor óptimo.

Lema 1. *La iteración* (10) *satisface*

$$\|x_{k+1} - x^*\|_Q^2 = \left[1 - \frac{(\nabla f_k^T \nabla f_k)^2}{(\nabla f_k^T Q \nabla f_k)(\nabla f_k^T Q^{-1} \nabla f_k)}\right] \|x_k - x^*\|_Q^2.$$

Demostración. Primeramente

$$\|x_k - x^*\|_Q^2 = (x_k - x^*)^T Q(x_k - x^*) = x_k^T Q x_k - 2x^{*T} Q x^k + x^{*T} Q x^*,$$

y, según (10) y (9), tenemos $x_{k+1} = x_k - \alpha_k \nabla f_k$. Por lo tanto

$$\begin{aligned}\|x_{k+1} - x^*\|_Q^2 &= (x_k - \alpha \nabla f_k - x^*)^T Q(x_k - \alpha \nabla f_k - x^*) \\ &= x_k^T Q x_k - 2x^{*T} Q x^k + x^{*T} Q x^* - 2\alpha \nabla f_k^T Q(x_k - x^*) + \alpha^2 \nabla f_k^T Q \nabla f_k,\end{aligned}$$

de donde se sigue que

$$\|x_k - x^*\|_Q^2 - \|x_{k+1} - x^*\|_Q^2 = 2\alpha \nabla f_k^T Q(x_k - x^*) - \alpha^2 \nabla f_k^T Q \nabla f_k.$$

Puesto que $\nabla f_k = Q x_k - b$ y $b = Q x^*$, entonces

$$\nabla f_k = Q(x_k - x^*). \tag{11}$$

Usando esto y la ecuación (9), resulta

$$\begin{aligned}\|x_k - x^*\|_Q^2 - \|x_{k+1} - x^*\|_Q^2 &= 2\alpha \nabla f_k^T \nabla f_k - \alpha^2 \nabla f_k^T Q \nabla f_k \qquad (12) \\ &= 2\left(\frac{\nabla f_k^T \nabla f_k}{\nabla f_k^T Q \nabla f_k}\right) \nabla f_k^T \nabla f_k - \left(\frac{\nabla f_k^T \nabla f_k}{\nabla f_k^T Q \nabla f_k}\right)^2 \nabla f_k^T Q \nabla f_k \\ &= \frac{2(\nabla f_k^T \nabla f_k)^2}{\nabla f_k^T Q \nabla f_k} - \frac{(\nabla f_k^T \nabla f_k)^2}{\nabla f_k^T Q \nabla f_k} \\ &= \frac{(\nabla f_k^T \nabla f_k)^2}{\nabla f_k^T Q \nabla f_k}.\end{aligned}$$

Ahora, usando (11)

$$\begin{aligned}\|x_k - x^*\|_Q^2 &= (x_k - x^*)^T Q(x_k - x^*) \\ &= (x_k - x^*)^T Q Q^{-1} Q(x_k - x^*) \\ &= \nabla f_k Q^{-1} \nabla f_k.\end{aligned}$$

Transponiendo términos en (12) llegamos a

$$\|x_{k+1} - x^*\|_Q^2 = \|x_k - x^*\|_Q^2 - \frac{(\nabla f_k^T \nabla f_k)^2}{\nabla f_k^T Q \nabla f_k},$$

lo que se factoriza para obtener el resultado deseado. ■

Teorema 2 (Desigualdad de Kantorovich). *Sea* Q *una matriz simétrica definida positiva de* $n \times n$. *Para cualquier vector* x *se cumple*

$$\frac{(x^T x)^2}{(x^T Q x)(x^T Q^{-1} x)} \geq \frac{4aA}{(a+A)^2}.$$

Demostración. Sean $\lambda_1, \ldots, \lambda_n$ los valores propios de Q de tal modo que

$$0 < a = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n = A.$$

Sin pérdida de generalidad podemos suponger que $Q = \operatorname{diag} \lambda_1, \ldots, \lambda_n$, pues tal matriz es diagonalizable. Tenemos

$$\frac{(x^T x)^2}{(x^T Q x)(x^T Q^{-1} x)} = \frac{\left(\sum_{i=1}^n x_i^2\right)^2}{\left(\sum_{i=1}^n \lambda_i x_i^2\right)\left(\sum_{i=1}^n \lambda_i^{-1} x_i^2\right)},$$

lo cual puede escribirse como

$$\frac{(x^T x)^2}{(x^T Q x)(x^T Q^{-1} x)} = \frac{1/\sum_{i=1}^n \xi_i \lambda_i}{\sum_{i=1}^n \xi_i/\lambda_i} := \frac{\phi(\xi)}{\psi(\xi)},$$

donde $\xi_i = x_i^2 / \sum_{i=1}^n x_i$. Hemos escrito la expresión en términos de la razón de dos funciones que involucran combinaciones convexas. Considerando a la curva definida por $f(x) = 1/x$, tenemos que el valor de $\phi(\xi)$ es un punto sobre la curva y que $\psi(\xi)$ es una combinación convexa de puntos sobre la curva. Por lo tanto, el valor mínimo de la razón entre ellos se alcanza en algún $\lambda = \xi_1 \lambda_1 + \xi_n \lambda_n$, con $\xi_1 + \xi_n = 1$. Usando la relación

$$\frac{\xi_1}{\lambda_1} + \frac{\xi_n}{\lambda_n} = \frac{\lambda_1 + \lambda_n - \xi_1 \lambda_1 - \xi_n \lambda_n}{\lambda_1 \lambda_n},$$

vemos que una cota apropiada es

$$\frac{\phi(\xi)}{\psi(\xi)} \geq \min_{\lambda_1 \leq \lambda \leq \lambda_n} \frac{1/\lambda}{(\lambda_1 + \lambda_n - \lambda)/(\lambda_1 \lambda_n)}.$$

El mínimo se alcanza cuando $\lambda = \frac{1}{2}(\lambda_1 + \lambda_2)$, y entonces

$$\frac{\xi_1}{\lambda_1} + \frac{\xi_n}{\lambda_n} = \frac{4\lambda_1\lambda_n}{(\lambda_1 + \lambda_n)^2}.$$

como se requería. ■

Teorema 3. *En el método del descenso más pronunciado la iteración* (10) *con la función* (8)*, satisface*

$$\|x_{k+1} - x^*\|_Q^2 \leq \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right) \|x_k - x^*\|_Q^2.$$

Demostración. Por el Lema 1 y el Teorema 2, tenemos

$$\begin{aligned}
\|x_{k+1} - x^*\|_Q^2 &= \left[1 - \frac{(\nabla f_k^T \nabla f_k)^2}{(\nabla f_k^T Q \nabla f_k)(\nabla f_k^T Q^{-1} \nabla f_k)}\right] \|x_k - x^*\|_Q^2 \\
&\leq \left[1 - \frac{4aA}{(A+a)^2}\right] \|x_k - x^*\|_Q^2 \\
&\leq \left[\frac{(A-a)^2}{(A+a)^2}\right] \|x_k - x^*\|_Q^2,
\end{aligned}$$

una sustitución nos lleva al resultado. ■

El siguiente teorema es consecuencia inmediata del anterior.

Teorema 4. *Supóngase que* $f : \mathbb{R}^n \to \mathbb{R}$ *es dos veces continuamente diferenciable, y que las iteraciones generadas por el método del descenso más pronunciado con búsquedas por línea exactas convergen al punto* x^* *donde la matriz hessiana* $\nabla^2 f(x^*)$ *es definida positiva. Entonces*

$$f(x_{k+1}) - f(x^*) \leq \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}\right) [f(x_k) - f(x^*)],$$

donde $\lambda_1 \leq \ldots \leq \lambda_n$ *son los autovalores de* $\nabla^2 f(x^*)$.

1.3. Método de Newton

La elección de la dirección opuesta a la del gradiente no es la única posibilidad. Consideremos la aproximación a la función objetivo f en el punto $x_k + p$ dada por

$$f(x_k + p) =\approx f_k + p^T \nabla f_k + \frac{1}{2} p^T \nabla^2 f_k p.$$

Siempre que $\nabla^2 f_k$ sea definida positiva, podemos encontrar la *dirección de Newton* que minimiza la expresión anterior. Tomando derivadas respecto a p

$$\frac{\partial f(x_k + p)}{\partial p} = \nabla f_k + \nabla^2 f_k p,$$

e igualando a 0 y transponiendo términos:

$$p_k^N = -(\nabla^2 f_k)^{-1} \nabla f_k.$$

Este esquema funciona bien para funciones que están cercanas a su aproximación cuadrática, y en tal caso la convergencia al óptimo es cuadrática.

A modo de ejemplo, intentamos calcular el mínimo de la función de Rosenbrock

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

implementando en Octave la búsqueda por línea con la dirección de Newton, calculando el tamaño de paso con búsqueda hacía atrás.

```
function tabla=DescensoNewton(p,alpha,rho,c,maxit)
tabla = [];
for i=1:maxit
 tabla = [tabla; p];
 hessiano=[-400*p(2)^2+1200*p(1)^2 -400*p(1); -400*p(1) 200];
 gradiente=[-400*(p(2)-p(1)^2)*p(1)-2*(1-p(1)); 200*(p(2)-p(1)^2)];
 direccion=inv(hessiano)*gradiente;
 direccion=-direccion/norm(direccion);
 f0=100*(p(2)-p(1)^2)^2+(1-p(1))^2;
 u=p+(alpha*direccion)';
 f1=100*(u(2)-u(1)^2)^2+(1-u(1))^2;
 while (f1 > (f0+c*alpha*gradiente'*direccion))
  alpha=alpha*rho;
  u=p+alpha*direccion';
  f1=100*(u(2)-u(1)^2)^2+(1-u(1))^2;
 endwhile
 alpha
```

```
 p=p+alpha*direccion';
endfor
endfunction
```

La función `DescensoNewton` toma como parámetros la aproximación inicial al óptimo `p`, el valor del tamaño de paso `alpha`, una constante de contracción `rho` para el rastreo hacia atrás, la constante de la condición de Wolfe `c` y el número máximo de iteraciones `maxit`. La figura 1.3 muestra una gráfica de las aproximaciones de salida para `tabla=DescensoNewton([-1.2 1],2,0.9,0.01,200)`.

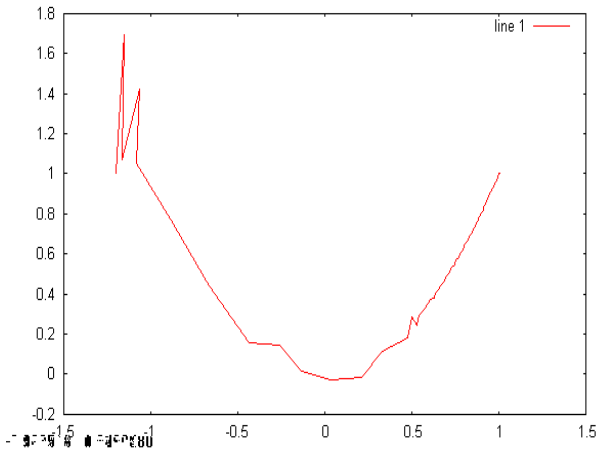

Figura 1: Método de Newton para la función de Rosenbrock con punto inicial $(-1{,}2, 1)$.

Observamos que el comportamiento en este caso es bastante errático, pues la matriz hessiana de la función de Rosenbrock varía bruscamente de un punto a otro. Sin embargo, en las cercanías del mínimo es definida positiva, y es por ello que a partir de cierto punto converge rápidamente a $(1, 1)$.

1.4. Métodos de quasi-Newton

La desventaja del método de Newton es se requiere el cálculo del hessiano, lo cual es computacionalmente costoso. Los métodos de quasi-Newton

en cada iteración usan una aproximación al hessiano B_k, la cual actualizan de acuerdo a dos esquemas, el *simétrico de rango uno*, SR1, que define como

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k},$$

y el de Broyden, Fletcher, Goldfarb y Shanno, BFGS, definido a través de

$$B_{k+1} = B_k + \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k},$$

donde

$$s_k = x_{k+1} - x_k, \quad y_k = \nabla f_{k+1} - \nabla f_k.$$

A continuación tenemos la implementación de ambos métodos en Octave para la función de Rosenbrock, usando la matriz identidad como aproximación inicial.

```
function matriz=SR1(matriz,x1,x0)
 s=x1-x0;
 y=[-400*(x1(2)-x1(1)^2)*x1(1)-2*(1-x1(1)); 200*(x1(2)-x1(1)^2)];
 y=y-[-400*(x0(2)-x0(1)^2)*x0(1)-2*(1-x0(1)); 200*(x0(2)-x0(1)^2)];
 matriz=matriz + ((y-matriz*s')*(y-matriz*s')')/((y-matriz*s')*s);
endfunction

function tabla=DescensoNewton2(p,alpha,rho,c,maxit)
hessiano = SR1([1 0; 0 1],p,p+[0.1 0.1])
// Cambiamos esta linea por
//  hessiano = BFGS([1 0; 0 1],p,p+[0.1 0.1])
// para el otro esquema.
tabla = [];
for i=1:maxit
 tabla = [tabla; p];
 gradiente=[-400*(p(2)-p(1)^2)*p(1)-2*(1-p(1)); 200*(p(2)-p(1)^2)];
 direccion=inv(hessiano)*gradiente;
 direccion=-direccion/norm(direccion);
 f0=100*(p(2)-p(1)^2)^2+(1-p(1))^2;
```

```
 u=p+(alpha*direccion)';
 f1=100*(u(2)-u(1)^2)^2+(1-u(1))^2;
 while (f1 > (f0+c*alpha*gradiente'*direccion))
  alpha=alpha*rho;
  u=p+alpha*direccion';
  f1=100*(u(2)-u(1)^2)^2+(1-u(1))^2;
 endwhile
 alpha
 hessiano = SR1(hessiano,p,u);
// Cambiamos esta linea por
//  hessiano = BFGS([1 0; 0 1],p,p+[0.1 0.1])
// para el otro esquema.
 p=p+alpha*direccion';
endfor
endfunction
```

Las corridas para `tabla=DescensoNewton2([-1.2 1],2,0.9,0.01,200)` (con SR1) y `tabla=DescensoNewton3([-1.2 1],2,0.9,0.01,200)` (con BFGS) se muestran en las figuras 1.4 y 1.4.

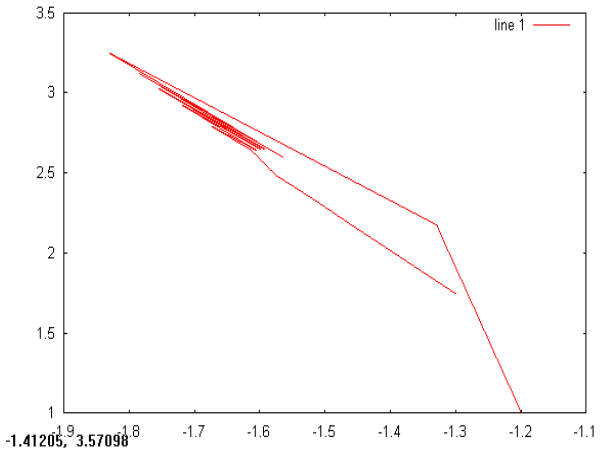

Figura 2: Método de Newton para la función de Rosenbrock con punto inicial $(-1{,}2, 1)$ (SR1).

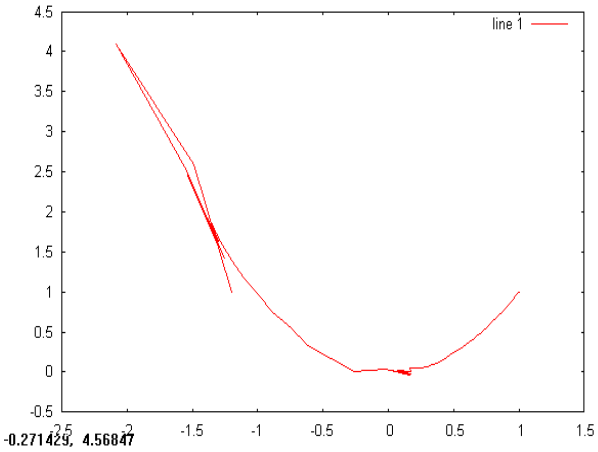

Figura 3: Método de Newton para la función de Rosenbrock con punto inicial $(-1{,}2, 1)$ (BFGS).

De las figuras podemos ver que el comportamiento de ambos esquemas es errático igual que el de Newton original, salvo por el BFGS, cuyo comportamiento fue excepcional convergiendo bien hacia el mínimo global $(1, 1)$, al contrario del SR1 cuyo desempeño fue el peor de todos.

2. Aplicación al registro automático de imágenes

Una imagen puede verse como una función

$$f : T \subset \mathbb{R} \times \mathbb{R} \to \mathbb{Z}$$

donde el conjunto T es la región donde aparece la imagen y el dominio de f está generalmente entre 0 y 255, representando el tono de gris de cada región de la imagen. En la computadora, sin embargo, tenemos la aproximación

$$f : T \subset \mathbb{N} \times \mathbb{N} \to \mathbb{Z}$$

y en este caso se representa por una matriz de $m \times n$, donde cada entrada de dicha matriz representa un pixel. En ocasiones, una imagen registrada

automáticamente sufre pequeñas transformaciones, tales como traslaciones y rotaciones. Una traslación seguida de una rotación puede escribirse como una transformación de $\Phi : \mathbb{R} \times \mathbb{R} \to \mathbb{R} \times \mathbb{R}$ de la siguiente manera:

$$\begin{pmatrix} \xi \\ \eta \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} u \\ v \end{pmatrix}.$$

En el problema de registro de imágenes se buscan los parámetros θ, u, v que minimicen la función objetivo

$$E(\Phi) = \int\int_T |g(x,y) - f(\Phi(x,y))|\, dxdy,$$

y que en el caso discretizado son

$$E(\Phi) = \sum_{i=1}^{m}\sum_{j=1}^{n} |g(i,j) - f(\Phi(i,j))|,$$

considerando que las funciones pueden extenderse fuera de su dominio valiendo 0, lo que corresponde al negro, y tomando el entero más cercano en el caso discreto para las coordenadas.

El primer problema que surge es que los métodos de optimización que se han visto dependen de la diferenciabilidad de la funcíon, propiedad que no necesariamente posee la función que representa a la imagen ni tampoco su discretización. Lo mejor que se puede hacerse es aproximar el gradiente. Considerando que Φ depende de θ, u, v, entonces E depende de θ, u, v, así que

$$\nabla E(\theta,u,v) \approx \sum_{i=1}^{3} \frac{E((\theta,u,v) + e_i\epsilon) - E(\theta,u,v)}{\epsilon} e_i.$$

donde e_i es el i-ésimo vector canónico de $\mathbb{R}^n$.

De aquí se tiene que al problema de registro de imágenes se pueden aplicar todos los métodos antes mencionados. Para ejemplificar, se utilizará el método de descenso más pronunciado con una imagen cuyo comportamiento de su gradiente no sea demasiado irregular. Se usó la imagen de 21×21 pixeles generada por

$$f(i,j) = \left\lfloor \frac{255}{1 + \frac{1}{10}(i-11)^2 + \frac{1}{10}(j-11)^2} \right\rfloor, \quad i = 1,\ldots,21, j = 1,\ldots,21.$$

a la cual se trasladó en un vector (-5,5). Las imágenes que se reproducen aquí están ampliadas 4 veces aproximadamente.

A continuación se reproduce el código fuente hecho en MSWLogo v. 6.5b. de Brian Harvey de la Universidad de California Berkeley, con un GUI de George Mills (*http://www.softronix.com/*).

```
to descenso :mat1 :mat2 :p0 :paso :maxit

;Iteracion de descenso mas pronunciado con punto inicial :p0
;y numero maximo de iteraciones :maxit, con el paso del
;gradiente igual a :paso. La matriz inicial es :mat1 y la
;matriz final es :mat2

repeat :maxit [~
make "direccion gradiente :img1 :img2 :p0 :paso
make "long norma :direccion
make "direccion prodesc -1/:long :direccion
make "p0 suma :p0 :direccion
]
op :p0
end

to dibujarimagen :img

;Dibuja la matriz de la imagen pixel por pixel.

localmake "n count :img
pu fd :n rt 90
for [i 1 :n 1] [~
 for [j 1 :n 1] [~
  make "cual item :i item :j :img
  setpencolor (list :cual :cual :cual) pd fd 1
 ]
 pu bk :n lt 90 bk 1 rt 90
]
```

```
end

to evalfunc :mat1 :mat2 :vect

;Evalua la funcion objetivo en :vect

make "n1 count :mat1
make "suma 0
for[i 1 :n1 1] [~
 for[j 1 :n1 1] [~
  make "aux 0
  make "aux :aux + item :i item :j :mat2
  make "u piso :i + item 1 :vect
  make "v piso :j + item 2 :vect
  ifelse (or (:u > :n1) (:v > :n1) not (:u > 0) not (:v > 0))
   [make "aux :aux - 1] [make "aux :aux - item :u item :v :mat1]
  make "aux :aux*:aux
  make "suma :suma + :aux
 ]
]
op :suma
end

to gradiente :mat1 :mat2 :p0 :paso

;Calcula el gradiente en :p0 con paso :paso
;de la funcion objetivo.

localmake "grad []
for [i 1 2 1] [~
 make "grad lput
  (evalfunc :mat1 :mat2 suma :p0 prodesc :paso unitario :i 2)-
  (evalfunc :mat1 :mat2 :p0) :grad
]
```

```
make "grad prodesc 1/:paso :grad
op :grad
end

to imagenerr :n :u :v

;Genera las imagenes prueba.

localmake "aux []
localmake "aux2 []
for[i 1 :n 1] [~
 make "aux []
 for[j 1 :n 1] [~
  make "aux lput
    round 255/(1+(:i-(round :n/2)+:u)*(:i-(round :n/2)+:u)/10
    +(:j-(round :n/2)+:v)*(:j-(round :n/2)+:v)/10) :aux
 ]
 make "aux2 lput :aux :aux2
]
op :aux2
end

to norma :vect
localmake "dim count :vect
localmake "ans 0
repeat :dim
  [make "ans :ans+(item repcount :vect)*(item repcount :vect)]
make "ans sqrt(:ans)
op :ans
end

to piso :x
ifelse (not (round :x) > :x) [op round :x] [op (round :x)-1]
end
```

```
to prodesc :alpha :vect
localmake "dim count :vect
localmake "ans []
repeat :dim [make "ans lput (item repcount :vect)*:alpha :ans]
op :ans
end

to suma :vect1 :vect2
localmake "ans []
localmake "dim count :vect1
repeat :dim
 [make "ans lput (item repcount :vect1) +
              (item repcount :vect2) :ans]
op :ans
end

to unitario :n :dim
localmake "ans [];
repeat :dim [ifelse repcount=:n [make "ans lput 1 :ans]
                                [make "ans lput 0 :ans]]
op :ans
end

to vertransformacion :mat1 :vect

;Tranforma :mat1 segun :vect.

make "n1 count :mat1
localmake "linea []
localmake "linea2 []
for[i 1 :n1 1] [~
 make "linea []
 for[j 1 :n1 1] [~
```

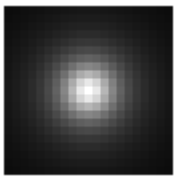

Figura 4: Imagen inicial del registro.

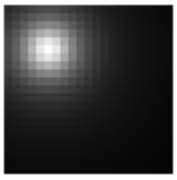

Figura 5: Imagen final del registro.

```
  make "u piso (:i + item 1 :vect)
  make "v piso (:j + item 2 :vect)
  ifelse (or (:u > :n1) (:v > :n1) (not (:u > 0)) (not (:v > 0)))
   [make "linea lput 255 :linea]
   [make "linea lput item :u item :v :mat1 :linea]
 ]
 make "linea2 lput :linea :linea2
]
op :linea2
end
```

Como puede verse en las imágenes, la convergencia es lenta al principio y se acelera hacía el final, además de que hay una pérdida significativa de información.

3. Conclusiones

Los métodos estudiados tienen todos en común que buscan el óptimo sobre una dirección la cual siguen de acuerdo a un cierto paso. La efectividad

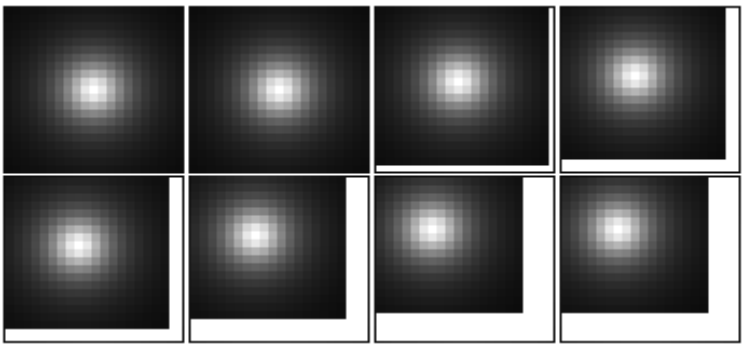

Figura 6: Ocho iteraciones del máximo descenso en el problema del registro.

para elegir dicho paso varía considerablemente de un método a otro. Para el caso de la función de Rosenbrock, el que resultó más rápido y preciso de todos fue el método de cuasi-Newton con aproximación al hessiano BFGS.

En el caso del registro de imágenes, puede verse que el método del gradiente resultó efectivo para una traslación sencilla, pues convergió en 8 iteraciones al óptimo.

Referencias

[1] Nocedal, Jorge *et al. Numerical optimization*, Springer-Verlag, Nueva York, 1999.