Tarea 10: Gradiente Conjugado No Lineal

Optimización I

Maestría en Computación

Centro de Investigación en Matemáticas

Esteban Reyes Saldaña
esteban.reyes@cimat.mx

4 de mayo de 2021

Resumen

En esta tarea se implementaron cuatro variantes del método de Gradiente conjugado no lineal. El método de Gradiente Conjugado resuelve el problema de minimización de una función cuadrática mediante la diagonalización de su expresión algebraica. Este método se puede extender para el caso no lineal considerando la búsqueda en línea y encontrando el tamaño de paso β adecuado. Se presenta a continuación una descripción general, así como el pseudocódigo de los métodos implementados. Finalmente se incluyen conclusiones observadas a partir de la experimentación.

1 Introducción

1.1 Gradiente Conjugado

El Algoritmo GC se usa para resolver el problema de optimización

$$\min_{x\in\mathbb{R}^n} f(x) = x^T Q x - b^T x. \tag{1}$$

donde Q es una matriz simétrica y definida positiva. Notemos que esto es equivalente a resolver el sistema de ecuaciones

$$\begin{aligned} Qx &= b \\ x^* &= Q^{-1}b \end{aligned}$$

¿Será posible usarlo cuando $f(\cdot)$ es una función convexa cualquiera?
Fletcher y Reeves mostraron que se podía extender el método de Gradiente Conjugado a funciones no lineales realizando una iteración entre métodos sobre el algoritmo. El primero de ellos es realizar el cálculo del tamaño de paso α_k a través de búsqueda en lineapara aproximar el mínimo de la función no lineal sobre la dirección d_k, el cálculo del gradiente g_k (o el residuo) del algoritmo de Gradiente Conjugado debe ser remplazado por el valor del gradiente de la función no lineal.

2 Método

2.1 Fletcher-Reeves (FR)

El método de Fletcher-Reeves (FR) converge si el punto inicial está suficientemente cerca del óptimo. Para ello se requiere que

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

Y luego,

$$d_{k+1} = -\nabla f(x_{k+1}) + \beta_{k+1}^{FR} d_k \tag{3}$$

Existen variantes del método FR. La diferencia fundamental es la forma de calcular el parámetro β_k.

2.2 Polak-Ribiere(PR)

Para este método, b_k se calcula como

$$\beta_{k+1}^{PR} = \frac{\nabla f(x_{k+1})(\nabla f(x_{k+1}) - \nabla f(x_k))}{\nabla f(x_k)\nabla f(x_k)}$$

En algunos casos (raros) el algoritmo de PR se puede ciclar infinitamente. Sin embargo, se puede garantizar la convergencia tomando

$$\beta_{k+1}^{+} = \text{máx}(0, \beta_{k+1}^{PR})$$

2.3 Hestenes-Stiefel (HS)

El método de Fletcher-Reeves (FR) converge si el punto inicial está suficientemente cerca del óptimo. Existen variantes del método FR. La diferencia fundamental es la forma de calcular el parámetro β_k. Para este método se usa

$$\beta_{k+1}^{HS} = \frac{\nabla f(x_{k+1})(\nabla f(x_{k+1}) - \nabla f(x_k))}{(\nabla f(x_{k+1}) - \nabla f(x_k))^T d_k}$$

2.4 Fletcher-Reeves Polak-Ribiere

Es posible garantizar la convergencia si

$$|\beta_k| \leq \beta_k^{FR}.$$

Lo anterior nos da una idea para dar una variante adecuada para el método de FR. Si $k \geq 2$, definimos

$$\beta_k = \begin{cases} -\beta_k^{FR}, & \beta_k^{PR} < -\beta_k^{FR} \\ \beta_k^{PR}, & |\beta_k^{PR}| \leq \beta_k^{FR} \\ \beta_k^{FR}, & \beta_k^{PR} > \beta_k^{FR} \end{cases} \tag{4}$$

Nota. La idea anterior se basa en que para cualquier $|\beta_k| \leq \beta_k^{FR}$ se puede demostrar convergencia global para $k \geq 2$.

2.5 Descenso de Flercher-Reeves

$$d_k = -g_{k+1} + \beta_{k+1}^{FR} d_k$$

Si α_k es un tamaño de paso exacto entonces $g_{k+1}d_k = 0$, luego

$$\begin{aligned} g_{k+1}^T d_{k+1} &= -||g_{k+1}||^2 + \beta_{k+1}^{FR} g_{k+1}^T d_k \\ &= -||g_{k+1}||^2 \\ &< 0. \end{aligned}$$

es decir, d_{k+1} es una dirección de descenso. Si α_k no es un tamaño de paso exacto, el término $\beta_{k+1}^{FR} g_{k+1}^T d_k$ podría dorminar al primero y no se garantiza descenso.

2.6 Función para Optimizar

Se requiere resolver el siguiente problema de optimización no lineal

$$\min_X \sum_{i,j} \left[(x_{ij} - g_{ij})^2 + \lambda \sum_{(i,m)\in\Omega_{ij}} \sqrt{(x_{i,j} - x_{l,m})^2 + \mu} \right] \tag{5}$$

donde $\mu = 0{,}01$, $\lambda > 0$ es un es un parámetro de regularización dado; g es la función a la cual se le desea filtrar el ruido, por ejemplo una imagen; Ω_{ij} es el conjunto de índices formados por los vecinos del pixel (i, j), es decir

$$\Omega_{ij} = \{(i+1, j), (i-1, j), (i, j+1), (i, j-1)\}$$

Observe que si la imagen original es de tamaño $m \times n$ entonces la imagen con el ruido filtrado será

$$X^* = [x_{i,j}]_{1\leq i\leq m, 1\leq i\leq n}$$

donde X^* es el minimizador del problema.

2.7 Gradiente de la Función

Dado un x_{ij} fijo, tenemos que

$$
\begin{aligned}
\frac{\partial f}{\partial x_{ij}} &= 2(x_{ij} - g_{ij}) \\
&\quad +2\lambda \sum_{(i,m)\in\Omega_{ij}} \frac{\partial}{\partial x_{ij}} \sqrt{(x_{ij} - x_{lm})^2 + \mu} \\
&= 2(x_{ij} - g_{ij}) \\
&\quad +2\lambda \sum_{(i,m)\in\Omega_{ij}} \frac{1}{2} \frac{2(x_{ij} - x_{lm})}{\sqrt{(x_{ij} - x_{lm})^2 + \mu}} \\
&= 2(x_{ij} - g_{ij}) \\
&\quad +2\lambda \sum_{(i,m)\in\Omega_{ij}} \frac{(x_{ij} - x_{lm})}{\sqrt{(x_{ij} - x_{lm})^2 + \mu}}
\end{aligned}
$$

2.8 Pseudocódigo

2.8.1 Calcular β_k

Input: g_{k+1}, g_k, d_k, *variant*
Output: $beta_k$

if $variant = FR$ **then**
  $\beta_{k+1}^{FR} = \dfrac{\nabla f(x_{k+1})\nabla f(x_{k+1})}{\nabla f(x_k)\nabla f(x_k)}$
else if $variant = PR$ **then**
  $\beta_{k+1}^{PR} = \dfrac{\nabla f(x_{k+1})(\nabla f(x_{k+1}) - \nabla f(x_k))}{\nabla f(x_k)\nabla f(x_k)}$
else if $variant = HS$ **then**
  $\beta_{k+1}^{HS} = \dfrac{\nabla f(x_{k+1})(\nabla f(x_{k+1}) - \nabla f(x_k))}{(\nabla f(x_{k+1}) - \nabla f(x_k))^T d_k}$
else if $variant = FR - PR$ **then**
  $\beta_k = \begin{cases} -\beta_k^{FR}, & \beta_k^{PR} < -\beta_k^{FR} \\ \beta_k^{PR}, & |\beta_k^{PR}| \leq \beta_k^{FR} \\ \beta_k^{FR}, & \beta_k^{PR} > \beta_k^{FR} \end{cases}$
end if

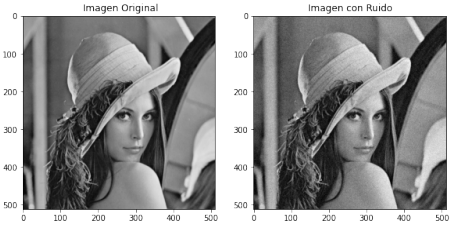
2.8.2 Gradiente Conjugado No Lineal

Input: x_0
Output: x^*

Haga $d_0 = -\nabla f(x_0)$, $k = 0$
while $||g_k|| \neq 0$ **do**
  Calcular α_k usando un método de búsqueda en línea
  $x_{k+1} = x_k + \alpha_k d_k$
  Calcular $\nabla f(x_{k+1})$
  Calcular β_{k+1} usando algún método del algoritmo anterior
  Hacer $d_{k+1} = -\nabla f(x_{k+1}) + \beta_k d_k$
  $k = k + 1$
end while

3 Resultados

Los algoritmos descritos en la sección pasada se usaron para eliminar el ruido de una imagen popular de interior llamada *lena.png* y una imagen con ruido llamada *lenanoise.png*.



Los parámetros usados para la función objetivo fueron

Parámetro	Valor
n	512
μ	0,01
λ	$\{0{,}1, 1, 2\}$

Para búsqueda en línea se utilizó

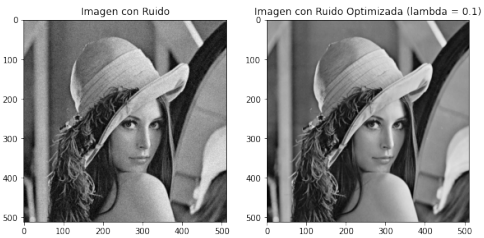
Parámetro	max_{iter}	α	c_1	ρ	τ_{grad}
Valor	500	0,9	10^{-4}	0,5	10^{-8}

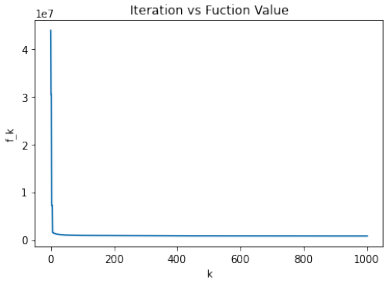
Dado que siempre se comienza del mismo punto inicial, el experimento sólo se repitió una vez para diferentes valores de λ.

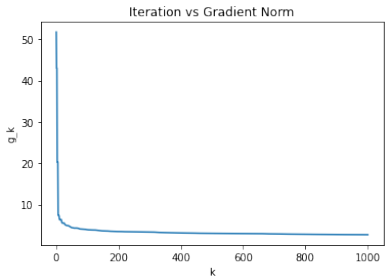
3.1 Fletcher-Reeves (FR)

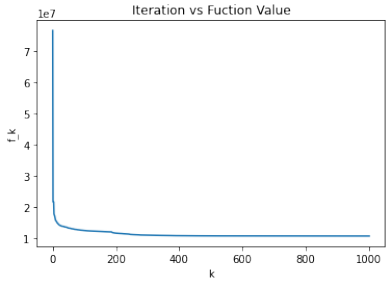
Algoritmo	Tiempo	Iteraciones
$\lambda = 0{,}1$	293.75 seg	1000
$\lambda = 1{,}0$	301.64 seg	1000
$\lambda = 2{,}0$	322.23 seg	1000

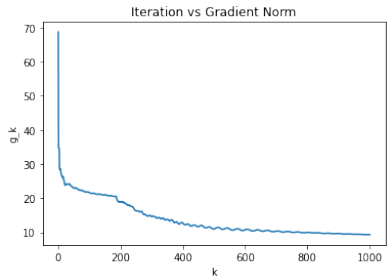
3.1.1 $\lambda = 0{,}1$



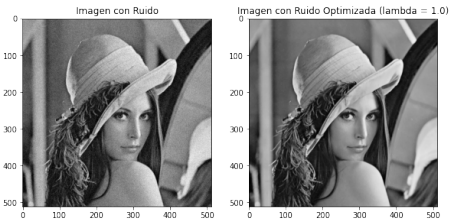


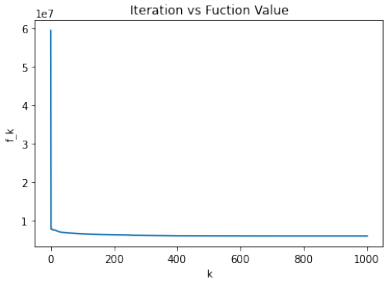


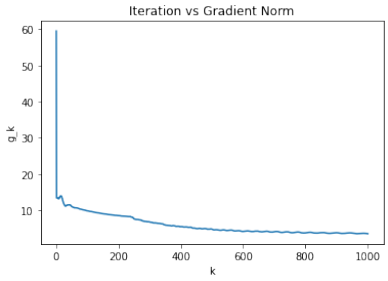




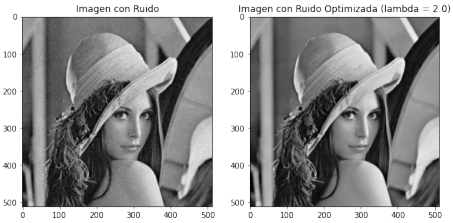
3.1.2 $\lambda = 1$







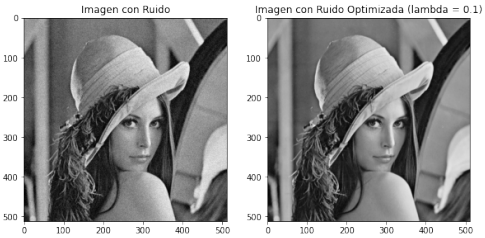
3.1.3 $\lambda = 2$

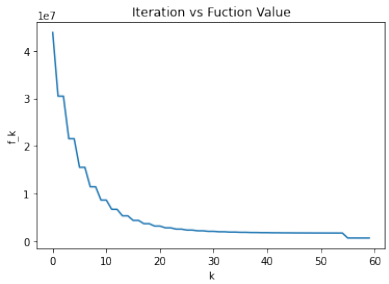


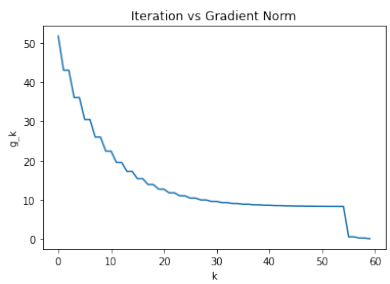
3.2 Polak-Ribiere(PR)

Algoritmo	Tiempo	Iteraciones
$\lambda = 0{,}1$	19.10 seg	60
$\lambda = 1{,}0$	26.30 seg	92
$\lambda = 2{,}0$	73.80 seg	256

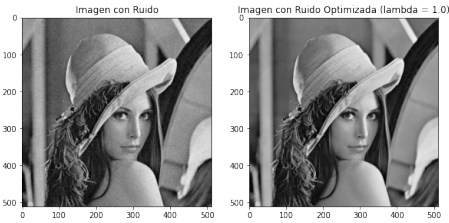
3.2.1 $\lambda = 0{,}1$

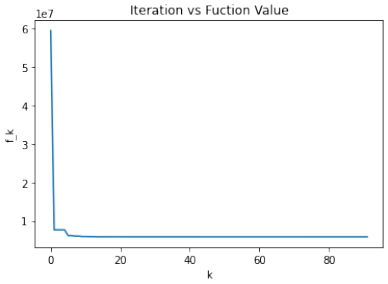


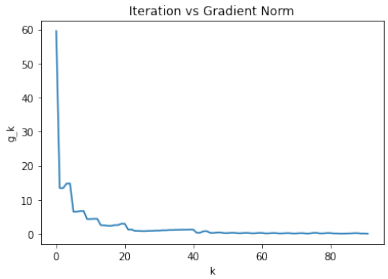




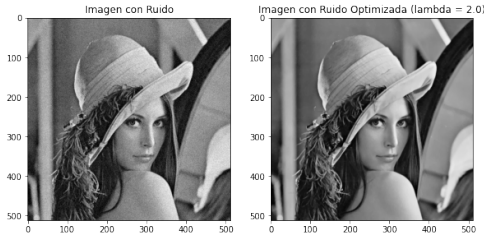
3.2.2 $\lambda = 1$

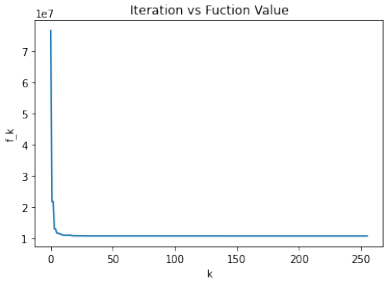


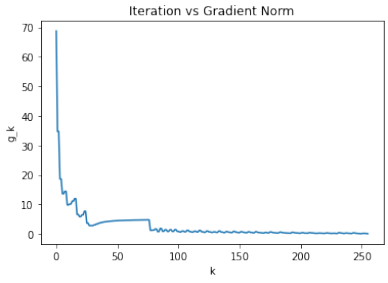




3.2.3 $\lambda = 2$



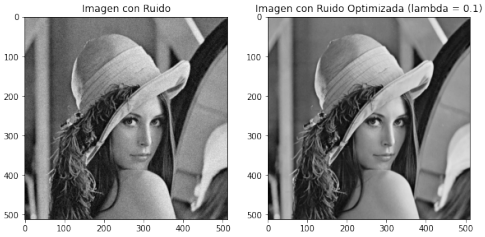


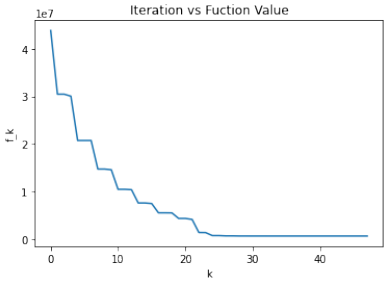


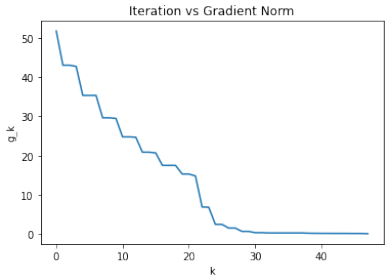
3.3 Hestenes-Stiefel (HS)

Algoritmo	Tiempo	Iteraciones
$\lambda = 0{,}1$	9.56 seg	48
$\lambda = 1{,}0$	20.52 seg	72
$\lambda = 2{,}0$	38.79 seg	138

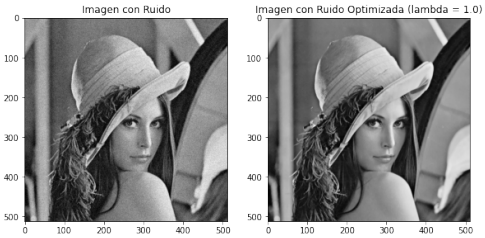
3.3.1 $\lambda = 0{,}1$

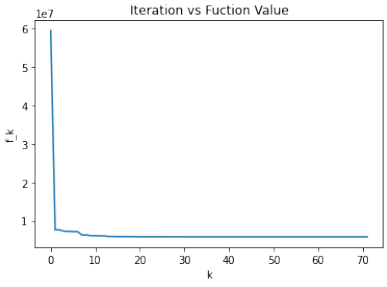


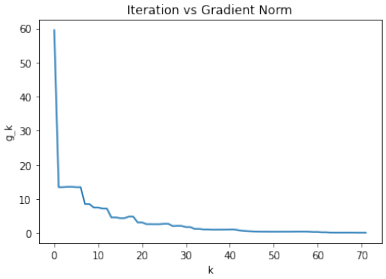




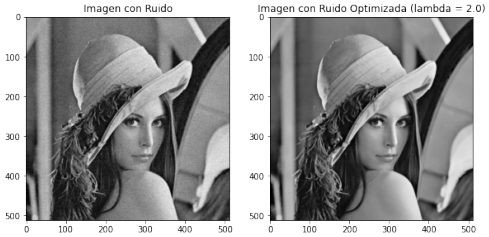
3.3.2 $\lambda = 1$

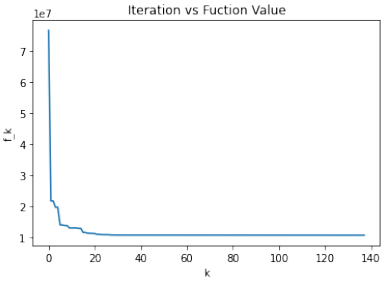


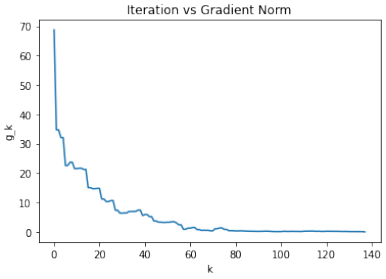




3.3.3 $\lambda = 2$



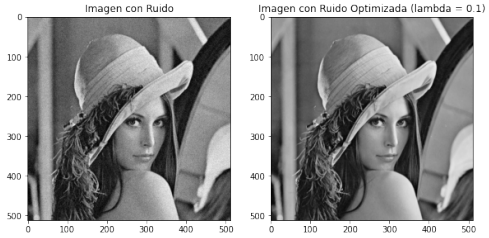


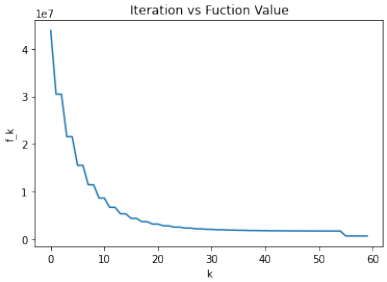


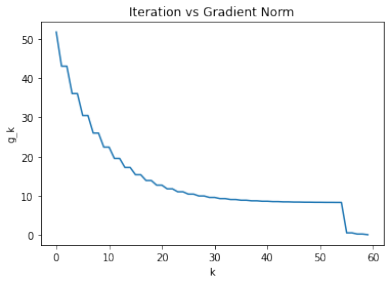
3.4 Fletcher-Reeves Polak-Ribiere

Algoritmo	Tiempo	Iteraciones
$\lambda = 0{,}1$	14.19 seg	60
$\lambda = 1{,}0$	21.85 seg	88
$\lambda = 2{,}0$	75.19 seg	277

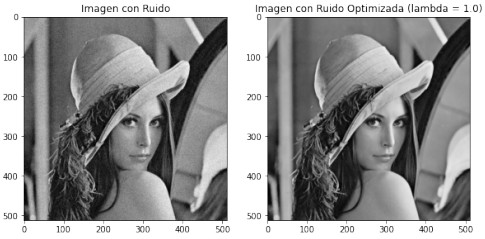
3.4.1 $\lambda = 0{,}1$

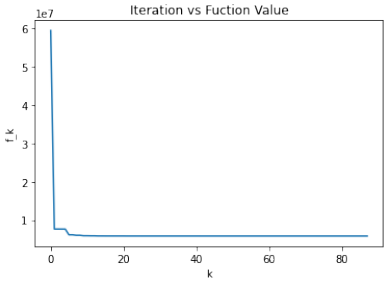


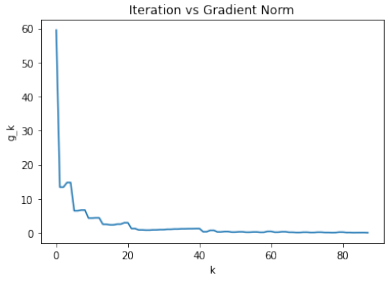




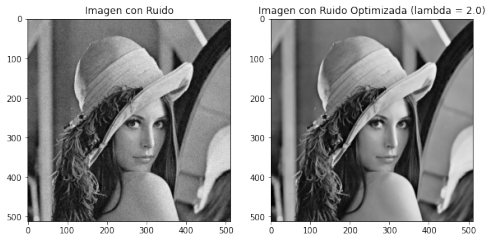
3.4.2 $\lambda = 1$

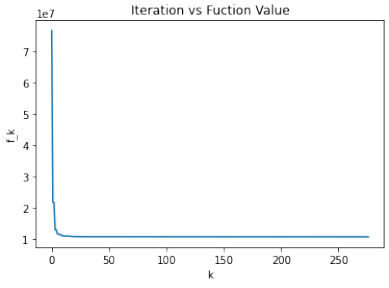


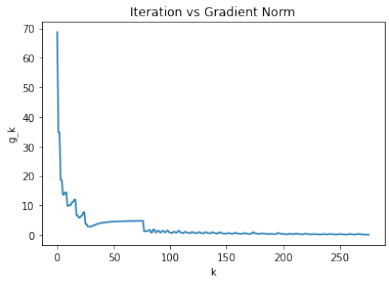




3.4.3 $\lambda = 2$







4 Conclusiones

En la experimentación se observó que el método de **Fletcher-Reeves** no convergió para ninguna elección de λ. Lo anterior se debe a que dicho algoritmo pide ortogonalidad entre direcciones consecutivas, lo cual no se cumple a priori. A pesar de lo anterior, las imágenes resultantes son parecidas a la imagen original.

Para el resto de los métodos se observó una rápida convergencia respecto al número de iteraciones. En primer lugar el método de **Hestenes-Stiefel** mostró menor número de iteraciones para todas las configuraciones. El método que itera entre **FR-PR** mejoró respecto al método original **FR**.

Respecto al valor de λ, en la experimentación se observó que el algoritmo de **gradiente conjugado no lineal** mostró un mejor desempeño para λ's pequeños, i.e, $\lambda \in (0, 1)$. Esto refleja la aportación de los vecinos de un pixel dado para eliminar el ruido. Si λ es grande, i.e, $\lambda \sim 100$, el algoritmo elimina más información de la requerida por la imagen original.