Universidad Tecnológica de la Mixteca

RESTAURACIÓN DE IMÁGENES CON EL MÉTODO DE GRADIENTE BARZILAI Y BORWEIN

TESIS

PARA OBTENER EL TÍTULO PROFESIONAL DE:

INGENIERO EN COMPUTACIÓN

PRESENTA:

GERARDO SOSA RAMÍREZ

ASESOR:

M.C. LUIS RENÉ MARCIAL CASTILLO

HUAJUAPAN DE LEÓN, OAXACA JUNIO DE 2001

Índice General

Capítulo 1

Introducción

El procesamiento digital de imágenes se divide en varias áreas entre las cuales se pueden mencionar: digitalización, compresión, realzado, restauración, reconstrucción, análisis, modelado y representación, entre otras. Este trabajo de tesis se enfoca en la restauración de imágenes.

La restauración de imágenes es necesaria debido a que algunas veces en el proceso de captura de la imagen, esta se obtiene con degradaciones o ruido; por lo tanto es necesario establecer un modelo para quitar ese ruido, considerando que aunque capturemos la imagen en repetidas ocasiones, con el mismo equipo, no siempre se va a obtener la misma imagen.

Este trabajo de tesis propone un modelo *a priori* para resturar una imagen con ruido gaussiano, que consiste en modular una pregunta a priori y producir una respuesta a posteriori. El caso de la restauración de una imagen implica ver qué probabilidad a priori hay de que un pixel tenga un valor de acuerdo a las observaciones o resultados a posteriori que anteriormente fueron también hipótesis a priori. Para ello se utiliza el teorema de Bayes, ya que permite involucrar un cierto tipo de probabilidad para hacer este modelado.

Una herramienta que permite modelar esta probabilidad con los pixeles son los campos aleatorios markovianos, puesto que proprorcionan una manera conveniente para modelar entidades dependientes de contexto como los pixeles, además de que están formulados dentro del campo de trabajo bayesiano. Estos campos aleatorios son sumamente útiles debido a que se pueden determinar las condicionantes de un pixel sólo con las condicionantes de sus vecinos. En consecuencia se tiene que instrumentar un estimador bayesiano que haga mínima una determinada función.

Al hablar de minimización de funciones, se debe tener en mente la programación no lineal sin restricciones y de ahí evidentemente las condiciones para probar que un punto es un mínimo o no, para ello se deben de conocer los tipos de mínimos, así como las condiciones necesarias y suficientes para este.

Hay varios métodos para la solución de sistemas de ecuaciones lineales entre los cuales se pueden destacar los métodos iterativos como el método de Jacobi, el método de Gauss-Seidel y el método de Sobre-relajaciones, sin embargo, se pueden notar algunas deficiencias de estos métodos. Así pues, de acuerdo al artículo que aparece en la revista SIAM J. OPTIMIZATION Febrero de 1997, se muestra un método para minimizar funciones no lineales sin restricciones a gran escala, llamado Método de Gradientes de Barzilai y Borwein, el cual se utiliza en este trabajo para minimizar la función de costo cuadrática que resulta al hacer la restauración de imágenes al utilizar el teorema de Bayes y los campos aleatorios markovianos. Cada iteración requiere solo $O(n)$ operaciones en punto flotante y una evaluación de gradiente.

Este trabajo de tesis se divide en diez capítulos; en el capítulo dos se describe lo concerniente a inferencia bayesiana, verosimilitud y distribuciones a priori, lo cual ayudará a formular la función de estimación, en el capítulo tres se describe cómo se integran los pixeles con la estimación de probabilidad y la forma en que se agrupan los mismos, el capítulo cuatro describe las condiciones necesarias y suficientes de un mínimo sin restricciones y algunos algoritmos estándar de minimización, en el capítulo cinco se

tratan los métodos iterativos, Jacobi, Gauss-Seidel y Sobre-relajaciones para encontrar la solución de sistemas de ecuaciones, en el capítulo seis se detalla el método de gradiente de Barzilai y Borwein en el cual se basa este trabajo para restaurar una imagen con ruido cuando se utilizan resortes cuadráticos, en el capítulo siete se describen los algoritmos instrumentados, en el capítulo ocho se describen las pruebas y se muestran los resultados obtenidos, en el capítulo nueve, se presentan las conclusiones, en el capítulo diez se presenta el manual de usuario del programa que se instrumentó para la restauración de imágenes, posteriormente se presenta la bibliografía utilizada en este trabajo de tesis y finalmente se presenta un Apéndice de conceptos Básicos.

Capítulo 2

Estimación Bayesiana

En este capítulo se describe la inferencia bayesiana como una alternativa para el análisis estadístico de datos, la cual se utiliza en la robótica, visión computacional y en métodos para reconstrucción y restauración de imágenes. Se presenta el teorema de Bayes y se describen los estimadores MAP y ML.

2.1 Inferencia bayesiana

La estimación bayesiana es un enfoque probabilístico a la inferencia, se basa en la suposición de que las cantidades que nos interesan son dadas por distribuciones de probabilidad y que, por lo tanto, se pueden tomar decisiones óptimas relacionando estas probabilidades y los datos que se obtengan. Este enfoque se está presentando como una alternativa para el análisis estadístico de datos.

La probabilidad bayesiana de un evento x representa el grado de convicción personal, mientras que la probabilidad clásica representa una propiedad física del mundo(por ejemplo: que en un volado, la moneda caiga águila), la probabilidad bayesiana es una propiedad de la persona que asigna la probabilidad(por ejemplo: el grado de convicción

de que en un volado la moneda caiga águila). La probabilidad clásica de un evento se refiere a una verdad o probabilidad física y la probabilidad bayesiana o personal de un evento se refiere al grado de convicción.

Una diferencia importante entre probabilidad clásica y probabilidad bayesiana es que para medir la probabilidad bayesiana no se necesita hacer repeticiones sobre un evento. Por ejemplo, repetir el lanzamiento de un cubo de azúcar en una superficie mojada. Cada vez que el cubo se lanza, sus dimensiones cambian ligeramente, de modo que la probabilidad clásica tendrá un costo alto para la probabilidad de que el cubo caiga en una cara particular, mientras que la probabilidad bayesiana simplemente restringe su atención al siguiente lanzamiento y asigna una probabilidad. Un ejemplo más simple sería considerar la siguiente pregunta: ¿Cuál es la probabilidad de que el equipo de futbol Pumas gane el campeonato del 2002? La probabilidad clásica no podría decir nada, mientras que la probabilidad bayesiana asigna una probabilidad.

Una crítica común a la definición de probabilidad bayesiana es que esas probabilidades parecen arbitrarias. ¿Por qué se debe de satisfacer un grado de convicción en las reglas de probabilidad ?, ¿ sobre qué escala se debe medir la probabilidad?. En particular tiene sentido asignar una probabilidad de uno o cero a un evento que ocurre o no, pero ¿qué probabilidad se asigna a la convicción de modo que no sea extremista? Con atención a la primera pregunta, muchos investigadores han sugerido diferentes conjuntos de propiedades que se satisfacen por los grados de convicción [1].

2.2 Teorema de Bayes

¿Por qué nos interesa la inferencia bayesiana en la restauración de imágenes?. Se puede contestar con el siguiente ejemplo. Si se desea tomar una fotografía con una cámara digital, denotando la imagen real por $f = \{X_1, X_2, ...\}$, el equipo rara vez mide

f directa o completamente; en lugar de esto produce datos $g = \{\hat{x}_1, \hat{x}_2, ...\}$, los cuales dependen de f de una cierta manera, que pudo depender de efectos complicados tales como imperfecciones de la lente. Incluso sería curioso mencionar que fotografías repetidas respecto al mismo objetivo darían diferentes datos debido al ruido, entonces, ¿cómo se puede obtener la imagen original?. Suponga que la imagen original se denota por f y se cuenta con la imagen con ruido g, un filtro H que es por donde se obtiene la imagen, y η, una distribución de ruido conocida. Se puede ver la relación de las entidades anteriores en la figura 2-1, donde $g = Hf + \eta$.

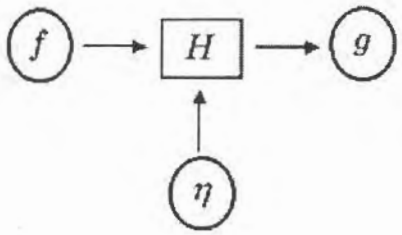


Figura 2-1: Relación entre una imagen, un filtro y la imagen de salida con una distribución de ruido asociada.

Si H es invertible, entonces:

$$f = H^{-1}(g - \eta), \tag{2.1}$$

sin embargo, aunque H sea invertible, no es muy conveniente optar por una solución de este tipo, dado que de η sólo se conoce su distribución y no puntualmente. En este trabajo se usa el modelo tomando $H = I$, de tal forma que:

$$g = f + \eta. \tag{2.2}$$

Por ello se necesita un método *a priori,* que consiste en modular una pregunta a priori y producir una respuesta a posteriori, el teorema de Bayes es ideal porque permite involucrar los conceptos a priori y a posteriori.

En la restauración de imágenes digitales, interesa saber cuál es el mejor valor que podría adoptar un pixel, es decir, cuál es la mejor hipótesis dados los datos; se denota con $P(D)$ a la probabilidad a priori de los datos, a $P(D|h)$ como la probabilidad de los datos dada la hipótesis h y se le denomina función de verosimilitud; lo que se desea estimar es: $P(h|D)$, la probabilidad a posteriori de h, dado los datos D. El Teorema de Bayes proporciona un método directo para calcular estas probabilidades, definido por:

$$P(h|D) = \frac{P(D|h)\,P(h)}{P(D)}. \tag{2.3}$$

En la mayoría de problemas donde se plantea la inferencia bayesiana, se parte de un conjunto de hipótesis H y se trata de encontrar la hipótesis más probable $h \in H$. A esta hipótesis más probable se le suele denominar *hipótesis máximo a posteriori* o *estimador MAP*. En base al teorema de Bayes, se define h_{MAP} como una hipótesis *MAP* de acuerdo a:

$$h_{MAP} = \arg\max_{h \in H} \left(P(D|h)\,P(h)\right). \tag{2.4}$$

Ahora, si se supone que todas las hipótesis tienen la misma probabilidad, entonces sólo queda la hipótesis de máxima verosimilitud o ML (maximum likelihood):

$$h_{ML} = \arg\max_{h \in H} \left(P(D|h)\right). \tag{2.5}$$

Para utilizar la estimación bayesiana se deben que seguir los siguientes pasos:

1. Hacer una pregunta para establecer un $P(D)$ a priori.
2. Describir la función de la verosimilitud $P(D|h)$.
3. Adquirir los datos determinados D.
4. Opcionalmente, calcular la evidencia $P(D)$ como medida de la calidad.
5. Evaluar la respuesta como la inferencia a posteriori $P(h|D)$.

La restauración de una imagen implicaría ver qué probabilidad a priori hay de que un pixel tenga un valor de acuerdo a las observaciones o resultados a posteriori que anteriormente fueron también hipótesis a priori y de acuerdo a esto, asignar el valor correspondiente al pixel.

Entre las características de la inferencia bayesiana se tienen:

1. Cada ejemplo nuevo puede aumentar o disminuir la estimación de una hipótesis (flexibilidad-incrementabilidad).

2. El conocimiento a priori se puede combinar con datos para determinar la probabilidad de la hipótesis.

3. Se obtienen resultados con probabilidades asociadas.

4. Puede clasificar combinando las predicciones de varias hipótesis.

5. Sirve de comparación con otros algoritmos.

Capítulo 3

Campos Aleatorios Markovianos

En este capítulo se descibe la teoría de campos aleatorios markovianos, que es una rama de la teoría de probabilidad para análisis espacial o dependencia contextual de fenómenos físicos, se usa para establecer distribuciones de probabilidad de clasificaciones interactivas; permite desarrollar algoritmos de visión óptima cuando se utilizan principios de optimización. También se describen los sistemas de vecindad y cliques, la distribución de Gibbs y se distingue un campo aleatorio markoviano de un campo aleatorio de Gibbs.

3.1 El problema del etiquetado

Para describir los campos aleatorios markovianos, se utiliza una representación natural denominada *Etiquetado* [13]. Un Problema de etiquetado se define en términos de un conjunto de estados y un conjunto de etiquetas. Sea S un conjunto discreto con m estados

$$S = \{1, 2, ..., m\}. \tag{3.1}$$

Un estado puede representar un punto o una región del espacio Euclidiano. El con-

junto de estados puede ser clasificado en términos de su homogeneidad. Se denota al conjunto de localizaciones de los pixels de una imagen 2D de tamaño $n \times n$ de la siguiente manera:

$$S = \{(i,j) \,|1 \leq i,j \leq n\}\,. \tag{3.2}$$

Sea L el conjunto de etiquetas, pudiendo ser un conjunto discreto o continuo. Por lo tanto el problema de etiquetado consiste en asignar una etiqueta(del conjunto de etiquetas L) a cada estado de S. De este modo se denomina etiquetado a :

$$f = \{f_1, ..., f_m\}\,,\ f_i \in L,\ i = 1, ..., m \tag{3.3}$$

y *mapeo* a toda funcion:

$$f : S \rightarrow L. \tag{3.4}$$

En terminología de campos aleatorios markovianos, se denomina *configuración* al etiquetado. Cuando todos los estados toman valores en el mismo conjunto de etiquetas L, el conjunto de todas las configuraciones posibles, es el producto cartesiano:

$$\Im = \underbrace{L \times L \times ... \times L}_{m\ veces} = L^m, \tag{3.5}$$

donde m es el tamaño de S.

Los campos aleatorios markovianos son una parte de la Teoría de Probabilidad, que proporciona una herramienta para analizar dependencias espaciales o contextuales de fenómenos físicos.

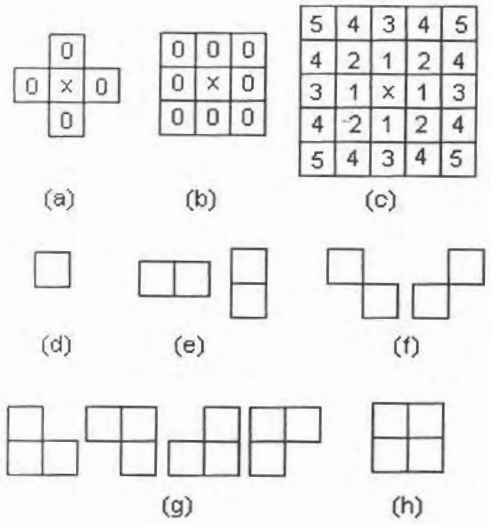


Figura 3-1: Vecinos en una lattice S

3.2 Sistemas de vecinos y cliques

Los estados de S están relacionados mediante un *Sistema de vecinos.* Este sistema se define para el conjunto de estados S como:

$$N = \{N_i | \ \forall i \in S\}, \tag{3.6}$$

donde N_i es el conjunto de los estados vecinos a i para los que se cumple lo siguiente:

(1) Un sitio no es vecindad de sí mismo: $i \notin N_i$.

(2) La relación entre las vecindades es mutua: $i \in N_{i'} \Leftrightarrow i' \in N_i$.

Para una rejilla (lattice) S, el conjunto de vecinos de i está definido como el conjunto de sitios próximos dentro de un radio r.

$$N_i = \left\{i' \in S | \ [dist\,(pixel_{i'}, pixel_i)]^2 \leq r, i' \neq i\right\}, \tag{3.7}$$

donde $dist(A, B)$ denota la distancia Euclidiana entre A y B y r toma un valor entero.

En el sistema de vecinos de primer orden, también llamado sistema 4-vecindad, cada

sitio(interior) tiene cuatro vecinos, como se muestra en la figura 3-1(a) donde x denota los sitios considerados y los 0's sus vecinos. En el sistema de vecinos de segundo orden, también llamado sistema 8-vecindad, hay 8 vecinos por cada sitio, como se muestra en la figura 3-1(b). Los números $n = 1, ..., 5$ mostrados en la figura 3-1(c) indican las vecindades más alejadas en el n-ésimo orden de sistemas de vecinos. La forma de un conjunto de vecinos puede describirse como la cáscara que encierra a todos los sitios en el conjunto.

Cuando el orden de los elementos en S es específico, el vecino puede ser determinado más explícitamente. Por ejemplo, cuando $S = \{1, ..., m\}$ está en un conjunto ordenado de sitios y sus elementos indexan los pixeles de una imagen 1D, un sitio interior $i \in \{2, ..., m-1\}$ tiene dos vecinos cercanos, $N_i = \{i-1, i+1\}$ y un sitio en los límites(cualquiera de los dos) tiene sólo un vecino $N_1 = \{2\}$ y $N_m = \{m-1\}$. Cuando los sitios en una rejilla rectangular $S = \{(i,j) \,|1 \leq i, j \leq n\}$ corresponden a los pixeles de una imagen $n \times n$ en el plano 2D, un sitio interno (i,j) tiene cuatro vecinos cercanos como $N_{i,j} = \{(i-1,j), (i+1,j), (i,j-1), (i,j+1)\}$, un sitio en los límites tiene tres y los sitios de las esquinas tienen dos.

Para una rejilla irregular S, el conjunto de vecinos N_i de i está definido de la misma manera como en la ecuación (3.7) que comprende los sitios adyacentes dentro de un radio r.

$$N_i = \left\{ i' \in S | \left[dist(rasgo_{i'}, rasgo_i) \right]^2 \leq r, i' \neq i \right\} \tag{3.8}$$

La función $dist(A, B)$ necesita estar definida apropiadamente para rasgos no puntuales. Alternativamete, la vecindad puede ser definida por la triangulación Delaunary o su dual, los polígonos Voroni de los sitios [13]. En general, el conjunto de vecinos N_i para una S irregular tienen varias formas y tamaños.

Un *clique* c para $\{S, N\}$ es un subconjunto de S tal que c consiste de un único estado

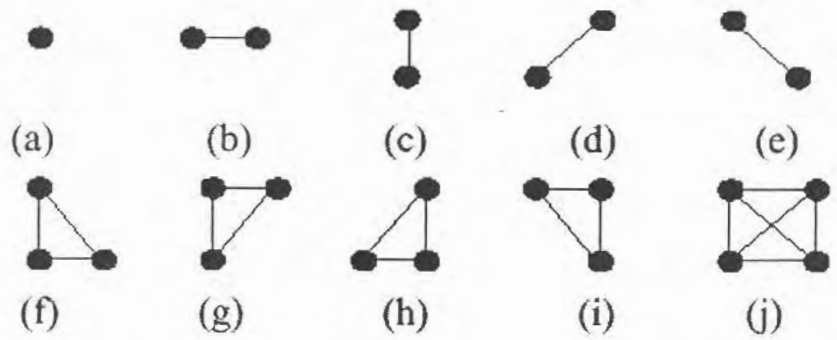


Figura 3-2: Cliques en una regilla regular S

$c = \{i\}$ o un par de estados vecinos $c = \{i, j\}$ o tres estados vecinos $c = \{i, j, k\}$ y así sucesivamente, es decir:

C_1 es el conjunto de cliques de un sólo estado.

C_2 es el conjunto de cliques de dos estados.

C_3 es el conjunto de cliques de tres estados.

$\vdots$

por lo tanto, el conjunto de todas las cliques para $\{S, N\}$ es: $C = C_1 \cup C_2 \cup C_3 \cup \cdots$.

El tipo de clique para $\{S, N\}$ en una rejilla regular, está dado por su tamaño, dimensión y orientación. La figura 3-2 muestra distintos tipos de cliques para sistemas de vecinos de primer y segundo orden. A medida que aumenta el orden del sistema de vecinos, tambien crece el número de cliques y su costo computacional asociado.

Sea $F = \{F_1, ..., F_m\}$, una familia de variables aleatorias definidas en S, donde cada variable aleatoria F_i toma valores f_i en L. Se llama *campo aleatorio* a la familia de variables aleatorias F. Se utilizará $F_i = f_i$ para denotar que el suceso F_i toma el valor f_i.

Ahora, un conjunto $\{F_1 = f_1, ..., F_m = f_m\}$ $(F = f)$ es una realización de F; se le

llama *configuración* del suceso conjunto F a $f = \{f_1, ...f_m\}$. Una familia de variables aleatorias F se dice que es un *campo aleatorio markoviano* sobre S con respecto a N sí y solo sí:

1. $P(F = f) > 0$.

2. $P\left(F_i = f_i | F_j = f_j,\ j \in S,\ j \neq i\right) = P\left(F_i = f_i | F_j = f_j, j \in N_i\right)$.

Esto quiere decir que la probabilidad de un suceso i condicionado a todos los sucesos restantes es igual a aquella condicionada a los sucesos vecinos de i. Puede demostrarse que la probabilidad conjunta $P(F = f)$ de cualquier campo aleatorio se determina únicamente por estas probabilidades condicionales locales [13]. Cualquier F que cumple estas condiciones es un *campo aleatorio markoviano* con respecto a tal sistema de vecinos.

Un campo aleatorio markoviano puede tener otras características tales como homogeneidad e isotropía.

- La homogeneidad consiste en decir que $P(f_i|f_{N_i})$ se calcula sin importar la posición relativa del estado i en S.

- La isotropía se refiere a las funciones *clique potenciales* (mencionadas posteriormente).

Los campos aleatorios markovianos son una generalización de los procesos markovianos que han sido extensamente utilizados en análisis de secuencias. Los procesos markovianos se definen normalmente en dominios de tiempo, más que en dominios de espacio, son una secuencia de variables aleatorias ..., F_1, ..., F_m, ... definidas sobre un conjunto de índices de tiempo $\{..., 1, ..., m, ...\}$. Un proceso markoviano unilateral de orden n-ésimo satisface:

$$P\left(f_i | ..., f_{i-2}, f_{i-1}\right) = P\left(f_i | f_{i-1}, ..., f_{i-n}\right). \tag{3.9}$$

Un proceso markoviano bilateral no causal no depende solamente del pasado, sino también del futuro. Un proceso markoviano bilateral de orden n-ésimo satisface:

$$P\left(f_i|..., f_{i-2}, f_{i-1}, f_{i+2}, ...\right) = P\left(f_i|f_{i+n}, ...f_{i+1}, f_{i-1}, ..., f_{i-n}\right). \tag{3.10}$$

Hay dos aproximaciones para especificar un campo aleatorio markoviano, en términos de probabilidades condicionales y en términos de probabilidad conjunta. En primer lugar, no hay método obvio disponible para deducir la probabilidad conjunta de las probabilidades condicionales asociadas. En segundo lugar, las probabilidades condicionales están sujetas a algunas condiciones de consistencia no obvias y altamente restrictivas. En tercer lugar, la especificación natural de equilibrio en un proceso estadístico, es en términos de probabilidad conjunta, en vez de términos de distribución condicional de las variables. Afortunadamente, un resultado teórico sobre la equivalencia entre los campos aleatorios de markov y la distribución de probabilidad de Gibbs [13] proporciona medios de especificar la probabilidad conjunta de un campo aleatorio de markov matemáticamente manejable.

3.3 Distribución de Gibbs y equivalencia Markov-Gibbs

Un conjunto de variables aleatorias F es un campo aleatorio de Gibbs en S con respecto a N si y sólo si sus configuraciones siguen una distribución de Gibbs. Una distribución de Gibbs tiene la siguiente función de densidad:

$$P(f) = \frac{1}{Z} e^{-\frac{1}{T}U(f)}, \tag{3.11}$$

donde:

$U(f)$, denota la función de energía.

Z, es una constante de normalización.

T, es la temperatura.

La función de energía $U(f)$ viene definida por:

$$U(f) = \sum_{c} V_c(f), \tag{3.12}$$

es decir, una sumatoria de las funciones "clique potenciales" $V_c(f)$ para todos los cliques posibles. El valor de $V_c(f)$ depende de la configuración local del clique c. Sin embargo, es conveniente expresar la energía de una distribucion de Gibbs como la suma de varios términos, cada uno correspondiente a cliques de tamaño distinto:

$$U(f) = \sum_{c \in C} V_c(f) = \sum_{\{i\} \in C_1} V_1(f_i) + \sum_{\{i,j\} \in C_2} V_2(f_i, f_j) + \dots . \tag{3.13}$$

Un campo aleatorio markoviano se caracteriza por sus propiedades locales (propiedades markovianas) mientras que un campo aleatorio de Gibbs se caracteriza por sus propiedades globales (la distribución de probabilidad de Gibbs). El teorema de Hammersley-Clifford [13] establece la equivalencia entre estos dos tipos de características.

Teorema 3.1 F es un Campo Aleatorio Markoviano en S con respecto a N si y sólo si F es un Campo Aleatorio de Gibbs en S con respecto a N.

La demostración de que campo aleatorio de markov es un campo aleatorio de Gibbs es complicada, en [13] se da una referencia de la demostración. A continuación se desmuestra que un campo aleatorio de Gibbs es un campo aleatorio de Markov, lo cual no es tan complicado.

Demostración:

Sea $P(f)$ una distribución de Gibbs en S con respecto al sistema de vecinos N.

Considere la probabilidad condicional

$$P(f_i|f_{j,\ j\in S,i\neq j}) = \frac{P\left(f_i, f_{j,\ j\in S,i\neq j}\right)}{P\left(f_{j,\ j\in S,i\neq j}\right)} = \frac{P\left(f\right)}{\sum\limits_{f_i'\in L} P\left(f'\right)}, \tag{3.14}$$

donde $f' = \{f_1, ..., f_{i-1}, f_i', ...f_m\}$.

Sustituyendo $P(f) = \frac{1}{Z} e^{-\sum\limits_{c\in C} V_c(f)}$ se tiene que:

$$P(f_i|f_{j,\ j\in S,i\neq j}) = \frac{e^{-\sum\limits_{c\in C} V_c(f)}}{\sum\limits_{f_i'\in L} e^{-\sum\limits_{c\in C} V_c(f')}}. \tag{3.15}$$

Se divide L en dos conjuntos A, B de modo que A es el conjunto de cliques que contienen a i y B el que no contiene a i, entonces, se puede escribir

$$P(f_i|f_{j,\ j\in S,i\neq j}) = \frac{\left[e^{-\sum\limits_{c\in A} V_c(f)}\right]\left[e^{-\sum\limits_{c\in B} V_c(f)}\right]}{\sum\limits_{f_i'\in L}\left\{\left[e^{-\sum\limits_{c\in A} V_c(f')}\right]\left[e^{-\sum\limits_{c\in B} V_c(f')}\right]\right\}}, \tag{3.16}$$

como $Vc\left(f\right) = Vc\left(f'\right)$ para cualquier clique c que no contenga a i, $e^{-\sum\limits_{c\in B} V_c(f)}$ se simplifica en el numerador y en el denominador y se obtiene:

$$P(f_i|f_{j,\ j\in S,i\neq j}) = \frac{e^{-\sum\limits_{c\in A} V_c(f)}}{\sum\limits_{f_i'\in L} e^{-\sum\limits_{c\in A} V_c(f')}}, \tag{3.17}$$

es decir, sólo depende de la etiqueta de los vecinos de i. Esto prueba que un campo aleatorio de Gibbs es un campo aleatorio markoviano. ■

Capítulo 4

Programación no lineal sin restricciones

En este capítulo se caracterizan los diversos tipos de mínimos, se presentan las condiciones necesarias y suficientes para un punto mínimo, la convergencia de los algoritmos de minimización, el esquema general de búsqueda lineal y por último se muestran las direcciones de búsqueda de algunos métodos muy importantes en la programación no lineal sin restricciones[1].

4.1 Tipos de mínimos

Los tipos de mínimos se caracterizan como: mínimo global, mínimo local y mínimo local estricto (mínimo local fuerte).

Definición 4.1: Dada $f(x) : \Re^n \rightarrow \Re$, un punto $x^* \in \Re^n$ es un mínimo global de

[1] La programación matemática se divide en programación lineal, programación entera y programación no lineal. La programación no lineal se divide a su vez en: con restricciones y sin restriccciones.

$f(x)$, si para todo x se cumple que:

$$f(x^*) \leq f(x). \tag{4.1}$$

Definición 4.2: Dada $f(x) : \Re^n \rightarrow \Re$, un punto $x^* \in \Re^n$ es un mínimo local de $f(x)$, si existe una vecindad[2] N de x^* para $x \in N$. tal que:

$$f(x^*) \leq f(x). \tag{4.2}$$

Definición 4.3: Dada $f(x) : \Re^n \rightarrow \Re$, un punto $x^* \in \Re^n$ es un mínimo local estricto (fuerte) de $f(x)$, si existe una vecindad N de x^* tal que para todo $x \in N$. con $x \neq x^*$ se cumple que:

$$f(x^*) < f(x) \ \forall x \in N \text{ con } x \neq x^*. \tag{4.3}$$

En las figuras 4-1 y 4-2 se pueden ver representaciones gráficas de los diversos tipos de mínimos de la función $(x_2 - x_1)^4 + 8x_1x_2 - x_1 + x_2 + 3$.

4.2 Condiciones necesarias y suficientes para mínimos sin restricciones

Es de suma importancia tener condiciones para probar que un punto es un mínimo o no. A continuación se muestran las condiciones para mínimos sin restricciones.

Condición suficiente de primer orden

Si $\nabla f(x^*) = 0$, entonces x^* es un punto estacionario.

Condición necesaria de primer orden

[2]Una vecindad de x^* es un conjunto abierto que contiene a x^*.

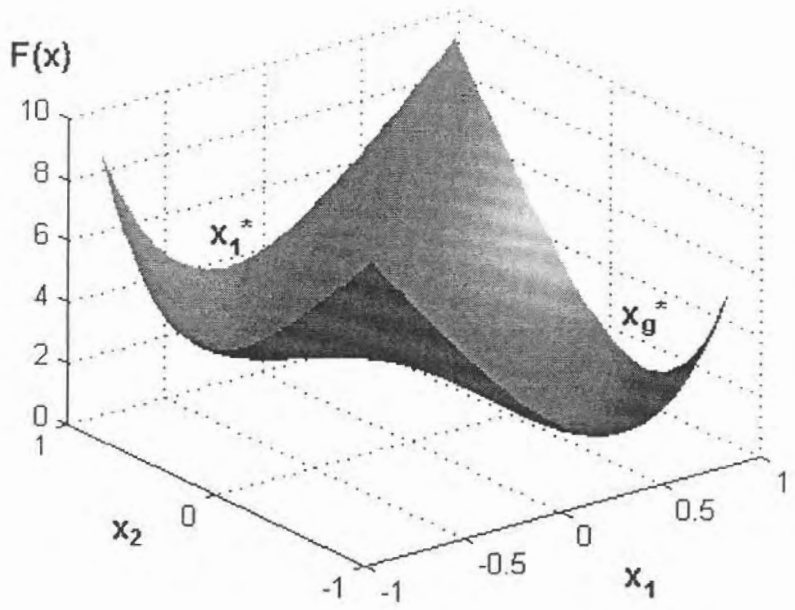


Figura 4-1: Función de dos variables que muestra dos mínimos locales, de los cuales uno es un mínimo global.

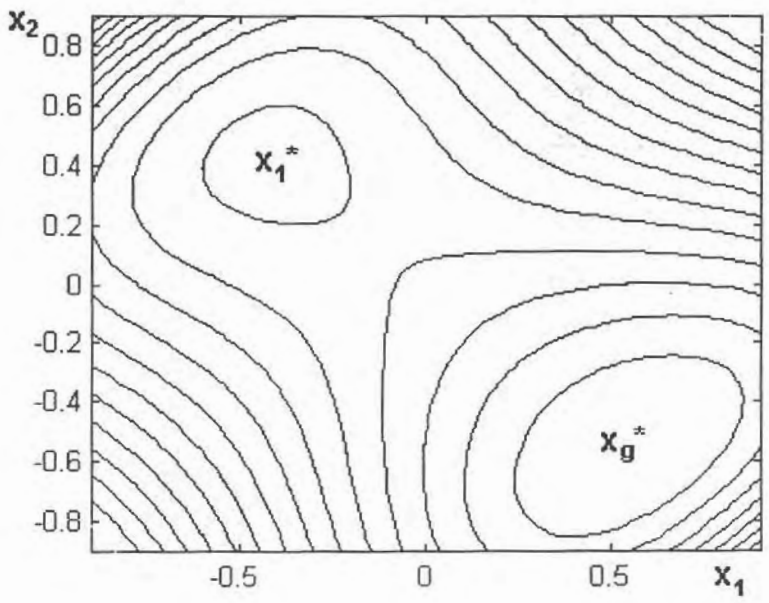


Figura 4-2: Mapa de contornos de la función mostrada en la figura 4-1 y que muestra el mínimo global x_g^* y el otro mínimo local x_1^*.

Si x^* es un mínimo local y f es continuamente diferenciable en una vecindad abierta de x^*, entonces $\nabla f(x^*) = 0$.

Condición necesaria de segundo orden

Si x^* es un mínimo local de $f(x)$, entonces $\nabla^2 f(x^*)$ es semidefinida positiva[3].

Condición suficiente de segundo orden

Si $\nabla^2 f(x^*)$ esdefinida positiva[4], entonces x^* es un mínimo local de $f(x)$.

4.3 Convergencia de los algoritmos de minimización

Hay dos factores de suma importancia para el diseño de algoritmos de minimización, la *estabilidad* y la *tasa de convergencia.*

Se dice que un algoritmo es *estable* o que muestra *convergencia global* si se puede probar que converge a un mínimo, independientemente del punto de inicio. En optimización no lineal la estabilidad se asocia siempre con esquemas iterativos, lo cual garantiza una reducción suficiente en los valores de la función en cada iteración a menos que el mínimo haya sido encontrado y se establezca un grado de exactitud. Frecuentemente, se encuentran algoritmos con algún otro tipo de convergencia, por ejemplo aquellos que convergen sólo cuando inician cerca del mínimo. Este tipo de algoritmos se conoce como de convergencia local. La convergencia global o local no debe confundirse con mínimo local y global.

La *tasa de convergencia* es importante debido a que un algoritmo debe llegar al menos muy cerca del mínimo en muy pocas iteraciones. Se está interesado en ver si el algoritmo tiene un *p-ésimo orden de convergencia* que consiste en encontrar el mayor valor de p si es que existe, para el cual su límite:

[3]Una matriz cuadrada A es semidefinida positiva, si para cada vector X, se satisface que $X^T AX \geq 0$.
[4]Una matriz cuadrada A es definida positiva, si para cada vector X, se satisface que $X^T AX > 0$.

$$K = \lim_{k \to \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^p}, \tag{4.4}$$

existe; x_k es el punto que alcanza en la k-ésima iteración y x^* es el mínimo. Se denomina $K\ (0 \leq K < 1)$ como *convergencia radial.* La rapidez de la tasa de convergencia es asociada con valores grandes de p y valores pequeños de K; cuando $p = 1$, es llamado tasa de convergencia lineal y es la tasa de convergencia más rapida; cuando $K = 0$, es llamado tasa de *convergencia Superlineal.* Generalmente p depende del algoritmo, mientras que K depende de la función que se minimiza. Se define el número de condición γ como un índice de la curvatura radial del máximo al mínimo en dos direcciones hacia el mínimo, esta condición está dada por:

$$\gamma = \frac{\lambda_{\max}}{\lambda_{\min}}, \tag{4.5}$$

donde $\lambda_{\max}$ y $\lambda_{\min}$ son los eigenvalores máximo y mínimo de la matriz Hessiana en el mínimo. Una función con un valor suficientemente grande de γ puede significar grandes problemas, es decir, se dice que el problema está mal condicionado mientras que una función con un valor relativamente pequeño se dice que está bien condicionada. En problemas mal condicionados, los contornos alrededor del mínimo son elipses altamente aplanadas y en problemas bien condicionados son más circulares, esto se puede observar en las figura 4-3, cuya función $f(x) = x_1^2 + x_2^2$ está bien condicionada y en la figura 4-4 cuya función $f(x) = 10x_1^2 + x_2^2$ es mal condicionda.

4.4 Búsqueda Lineal

El proceso de búsqueda lineal puede describirse de la siguiente manera:

Al inicio de la k-ésima iteración el estimado actual del mínimo es x_k y se hace una

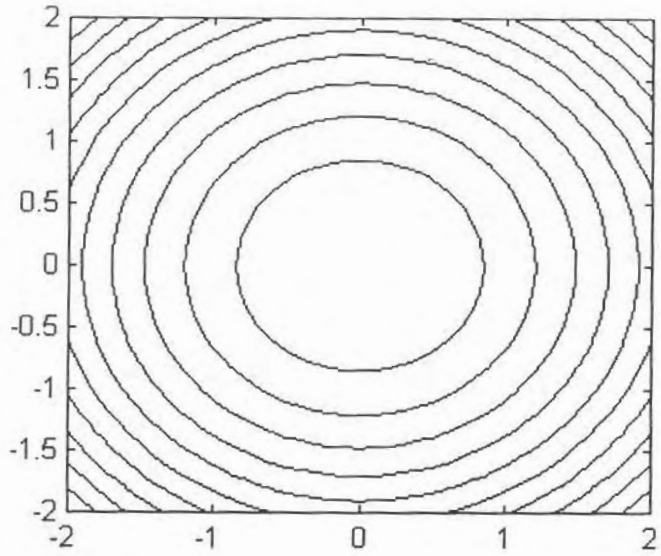


Figura 4-3: Contornos de la función bien condicionada $f(x) = x_1^2 + x_2^2$.

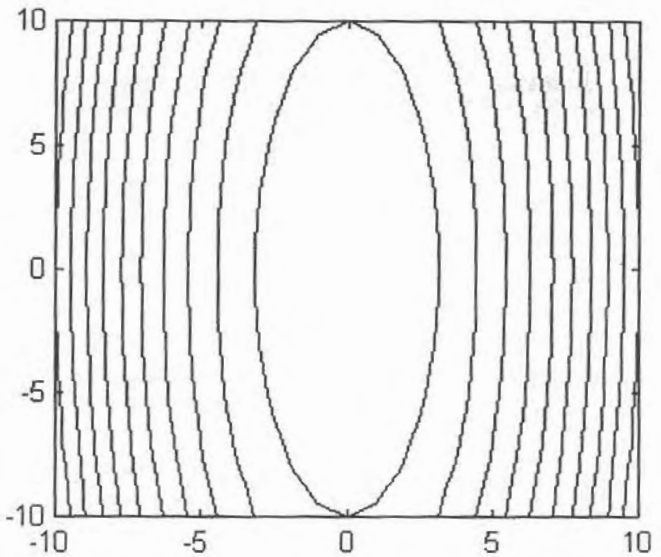


Figura 4-4: Contornos de la función mal condicionada $f(x) = 10x_1^2 + x_2^2$.

búsqueda en $\Re^n$ de x_k, a lo largo de un vector de búsqueda p_k como un intento para encontrar un nuevo punto x_{k+1} que sea un mínimo en esa dirección o que de una reducción suficiente en el valor de la función.

El nuevo punto x_{k+1} se puede expresar como $x_k + \alpha_k p_k$ para algún escalar α_k, $-\infty < \alpha_k < \infty$; el problema fundamental es por lo tanto la minimización invariante de $F(x_k + \alpha p_k)$ con respecto a α.

En la Figura 4-5 se puede observar la búsqueda lineal de una función de dos variables, el punto inicial es x_0, el vector de búqueda es p_0 y el mínimo lineal llega a ser el nuevo punto x_1. El mínimo global es x^*.

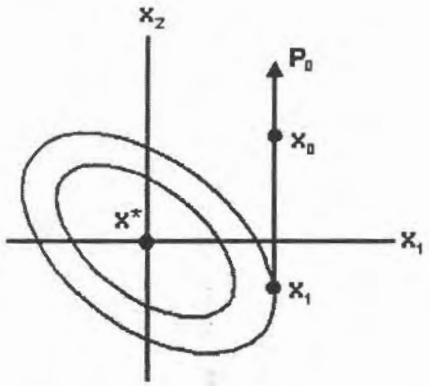


Figura 4-5. Búsqueda lineal de dos variables.

Los métodos de búsqueda lineal se resumen en tres pasos:

1. Calcular una dirección de búsqueda, p_k.

2. Calcular el tamaño de paso, $\alpha_k \approx \min f(x_k + \alpha p_k)$.

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

Hay diversas direcciones de descenso, de acuerdo a los algoritmos, a continuación se muestran algunos:

- $p_k = -\nabla f(x_k)$, Máximo descenso.

- $p_k = -\left(\nabla^{-2} f_k\right)^T \nabla f_k$, Newton.
- $p_k = B_k^{-1} \nabla f_k$, Cuasi-Newton.
- $p_k = -\nabla f\left(x_k\right)^T + \beta_k p_{k-1}$, Gradientes Conjugados.

El método de máximo descenso es el método fundamental de gradiente y de descenso. Históricamente se ha catalogado a este método como no muy confiable para propósitos generales de minimización de funciones no lineales debido a su deficiente tasa de convergencia.

Los métodos de Newton son diseñados por ser de convergencia rápida cuando se tiene la matriz Hessiana de la función objetivo. Sin embargo, no se garantiza que converja al mínimo desde un punto cualquiera.

Los métodos Cuasi-Newton son diseñados por tener mejores tasas de convergencia que los de gradiente, pero requieren un almacenamiento similar a los de Newton.

Los métodos de gradientes son aquellos que usan el vector gradiente de la función objetivo para calcular el vector de búsqueda independientemente de que el gradiente sea calculado analíticamente o aproximado por técnicas de diferencias finitas.

Capítulo 5

Métodos Iterativos

En este capítulo se muestran los métodos iterativos de Jacobi, Gauss-Seidel y Sobre-relajaciones para resolver sistemas de ecuaciones lineales. Así como las iteraciones cuando se resuelve un problema pequeño.

5.1 Métodos de aproximaciones sucesivas

Los *métodos iterativos* o de *aproximaciones sucesivas* requieren de mayores cálculos que los requeridos por el método de Gauss-Jordan para la mayoría de las matrices **A**, para llegar a un grado preestablecido de aproximación. Sin embargo, cuando los elementos no nulos de la matriz se acumulan a lo largo de su diagonal principal, siendo los elementos de la misma los mayores en valor absoluto, como ocurre al resolver ecuaciones diferenciales por medio de diferencias finitas, las técnicas iterativas pueden compararse favorablemente con los métodos directos, por lo que se refiere a la cantidad total de cálculos. Además, debido a que los métodos iterativos usan relativamente poca parte de la memoria de una computadora, son particularmente ventajosos para resolver sistemas de gran número de ecuaciones.

5.1.1 Método de Jacobi.

Supóngase que en el sistema

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{5.1}$$

$\mathbf{A} = \mathbf{D} + \mathbf{R}$, donde $\mathbf{D}$ es una matriz diagonal[1], entonces:

$$\begin{aligned} (\mathbf{D}+\mathbf{R})\,\mathbf{x} &= \mathbf{b} \\ \mathbf{D}\mathbf{x} &= \mathbf{b} - \mathbf{R}\mathbf{x} \\ \mathbf{x} &= \mathbf{D}^{-1}\mathbf{b} - \mathbf{D}^{-1}\mathbf{R}\mathbf{x}. \end{aligned} \tag{5.2}$$

Los elementos de la diagonal de la matriz $\mathbf{A}$ no deben contener elementos nulos para que exista la inversa $\mathbf{D}^{-1}$; si los tuviera, una simple reordenación de los renglones o columnas de la matriz pueden resolver este problema. La ecuación (5.2) sugiere el método iterativo:

$$\mathbf{x}_{k+1} = \mathbf{D}^{-1}\mathbf{b} - \mathbf{D}^{-1}\mathbf{R}\mathbf{x}_k, \tag{5.3}$$

llamado de *Jacobi*, de *iteraciones totales* o *desplazamientos simultáneos*. Para que éste converja es necesario que los elementos diagonales de $\mathbf{A}$ sean lo más grande posibles, por lo que siempre convendrá reordenar sus renglones y columnas en este sentido. Se puede demostrar que el método converge siempre que cada elemento diagonal sea, en valor absoluto, mayor que la suma del resto de los elementos del mismo renglón, también en valor absoluto.

El método de Jacobi, definido por la ecuación de recurrencia (5.3), significa que, del sistema (A.4) se despeja x_1 de la primer ecuación, x_2 de la segunda, x_3 de la tercera, etc.,

[1]Una matriz cuadrada cuyos elementos sobre la diagonal principal son los únicos diferentes de cero.

obteniéndose lo siguiente:

$$\begin{aligned} x_1 &= \frac{1}{a_{11}}\left(b_1 - a_{12}x_2 - a_{13}x_3 - \ldots - a_{1n}x_n\right) \\ x_2 &= \frac{1}{a_{22}}\left(b_2 - a_{21}x_1 - a_{23}x_3 - \ldots - a_{2n}x_n\right) \\ x_3 &= \frac{1}{a_{33}}\left(b_3 - a_{31}x_1 - a_{32}x_2 - \ldots - a_{3n}x_n\right) \\ &\ldots \\ &\ldots \\ x_n &= \frac{1}{a_{nn}}\left(b_n - a_{n1}x_1 - a_{n2}x_2 - \ldots - a_{nn-1}x_{n-1}\right). \end{aligned} \tag{5.4}$$

Se considera la primera aproximación

$$x_0 = \left\{x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, ..., x_n^{(0)}\right\}^T, \tag{5.5}$$

a la solución; se sustituye en los segundos miembros de las ecuaciones y se obtiene la nueva aproximación:

$$x_1 = \left\{x_1^{(1)}, x_2^{(1)}, x_3^{(1)}, ..., x_n^{(1)}\right\}^T. \tag{5.6}$$

El superíndice indica el número de la sustitución efectuada. La última aproximación se sustituye en los segundos miembros y se obtiene la nueva aproximación:

$$x_2 = \left\{x_1^{(2)}, x_2^{(2)}, x_3^{(2)}, ..., x_n^{(2)}\right\}^T, \text{ etc.} \tag{5.7}$$

Repitiendo este proceso $k+1$ veces, se llega a la aproximación:

$$\begin{aligned}
x_1^{(k+1)} &= \frac{1}{a_{11}}\left(b_1 - a_{12}x_2^{(k)} - a_{13}x_3^{(k)} - \ldots - a_{1n}x_n^{(k)}\right) \\
x_2^{(k+1)} &= \frac{1}{a_{22}}\left(b_2 - a_{21}x_1^{(k)} - a_{23}x_3^{(k)} - \ldots - a_{2n}x_n^{(k)}\right) \\
x_3^{(k+1)} &= \frac{1}{a_{33}}\left(b_3 - a_{31}x_1^{(k)} - a_{32}x_2^{(k)} - \ldots - a_{3n}x_n^{(k)}\right) \\
&\cdots \\
x_n^{(k+1)} &= \frac{1}{a_{nn}}\left(b_n - a_{n1}x_1^{(k)} - a_{n2}x_2^{(k)} - \ldots - a_{nn-1}x_{n-1}^{(k)}\right).
\end{aligned} \tag{5.8}$$

Por ejemplo, se resolverá el siguiente sistema aplicando el método de Jacobi

$$\begin{aligned}
4x_1 - x_2 &= 2 \\
-x_1 + 4x_2 - x_3 &= 6 \\
-x_2 + 4x_3 &= 2.
\end{aligned} \tag{5.9}$$

Despejando x_1 de la primera ecuación, x_2 de la segunda y x_3 de la tercera, se tiene:

$$\begin{aligned}
x_1 &= 0.50 + 0.25x_2 \\
x_2 &= 1.50 + 0.25x_1 + 0.25x_3 \\
x_3 &= 0.50 + 0.25x_2.
\end{aligned} \tag{5.10}$$

o sea que,

$$\begin{aligned}
x_1^{(k+1)} &= 0.50 + 0.25x_2^{(k)} \\
x_2^{(k+1)} &= 1.50 + 0.25x_1^{(k)} + 0.25x_3^{(k)} \\
x_3^{(k+1)} &= 0.50 + 0.25x_2^{(k)}.
\end{aligned} \tag{5.11}$$

Considérese

$$x_0 = \{0, 0, 0, ..., 0\}^T, \tag{5.12}$$

es decir,

$$x_1^{(0)} = 0, x_2^{(0)} = 0, x_3^{(0)} = 0. \tag{5.13}$$

Este primer vector de solución puede tener cualquier valor, y entre más cercano sea el valor supuesto con respecto al valor final, la convergencia es más rápida.

En general no se conocen los signos de los resultados y por esta razón se escoge el vector inicial igual a cero. Sustituyendo en el sistema (5.11), haciendo $k = 0$, se obtiene:

$$\begin{aligned} x_1^{(1)} &= 0.50 + 0.25\,(0) = 0.50 \\ x_2^{(1)} &= 1.50 + 0.25\,(0) + 0.25\,(0) = 1.50 \\ x_3^{(1)} &= 0.50 + 0.25\,(0) = 0.50 \end{aligned} \qquad x^{(1)} = \begin{bmatrix} 0.50 \\ 1.50 \\ 0.50 \end{bmatrix}. \tag{5.14}$$

Siguiendo en igual forma las iteraciones, resultan:

$$\begin{aligned} x^{(2)} &= \begin{bmatrix} 0.875 \\ 1.75 \\ 0.875 \end{bmatrix} \quad x^{(3)} = \begin{bmatrix} 0.938 \\ 1.94 \\ 0.938 \end{bmatrix} \quad x^{(4)} = \begin{bmatrix} 0.985 \\ 1.97 \\ 0.985 \end{bmatrix} \quad x^{(5)} = \begin{bmatrix} 0.995 \\ 1.99 \\ 0.995 \end{bmatrix} \\ x^{(6)} &= \begin{bmatrix} 0.998 \\ 2.00 \\ 0.998 \end{bmatrix} \quad x^{(7)} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \quad x^{(8)} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}. \end{aligned} \tag{5.15}$$

y la solución del sistema es $x_1 = 1$, $x_2 = 2$, $x_3 = 1$.

Nota: El método de Jacobi se usa muy poco en la práctica.

5.1.2 Método de Gauss-Seidel

El método de *Gauss-Seidel* es en general muy parecido al de Jacobi, la diferencia consiste en que una vez que se calcula la componente $x_i^{(k+1)}$, ésta se usa inmediatamente en la misma iteración. Por esta razón también es llamado método de *iteraciones parciales* o *desplazamientos sucesivos.*

Las ecuaciones que determinan la iteración $k+1$ son:

$$\begin{aligned}
x_1^{(k+1)} &= \frac{1}{a_{11}}\left(b_1 - a_{12}x_2^{(k)} - a_{13}x_3^{(k)} - ... - a_{1n}x_n^{(k)}\right) \\
x_2^{(k+1)} &= \frac{1}{a_{22}}\left(b_2 - a_{21}x_1^{(k+1)} - a_{23}x_3^{(k)} - ... - a_{2n}x_n^{(k)}\right) \\
x_3^{(k+1)} &= \frac{1}{a_{33}}\left(b_3 - a_{31}x_1^{(k+1)} - a_{32}x_2^{(k+1)} - ... - a_{3n}x_n^{(k)}\right) \\
&\cdots \\
&\cdots \\
x_n^{(k+1)} &= \frac{1}{a_{nn}}\left(b_n - a_{n1}x_1^{(k+1)} - a_{n2}x_2^{(k+1)} - ... - a_{nn-1}x_{n-1}^{(k+1)}\right).
\end{aligned} \tag{5.16}$$

Retomando el sistema ejemplificado en (5.9), y resolviéndolo por este método, se obtiene:

$$\begin{aligned}
x_1^{(k+1)} &= 0.50 + 0.25x_2^{(k)} \\
x_2^{(k+1)} &= 1.50 + 0.25x_1^{(k+1)} + 0.25x_3^{(k)} \\
x_3^{(k+1)} &= 0.50 + 0.25x_2^{(k+1)}.
\end{aligned} \tag{5.17}$$

Considerando $x_0 = \{0,0,0\}^T$ como primera aproximación, se obtiene de (5.17) la

primera iteración

$$\begin{aligned} x_1^{(1)} &= 0.50 + 0.25\,(0) = 0.50 \\ x_2^{(1)} &= 1.50 + 0.25\,(0.50) + 0.25\,(0) = 1.63 \\ x_3^{(1)} &= 0.50 + 0.25\,(1.63) = 0.91. \end{aligned} \tag{5.18}$$

La segunda iteración es

$$\begin{aligned} x_1^{(2)} &= 0.50 + 0.25\,(1.63) = 0.91 \\ x_2^{(2)} &= 1.50 + 0.25\,(0.91) + 0.25\,(0.91) = 1.96 \\ x_3^{(2)} &= 0.50 + 0.25\,(1.96) = 0.99. \end{aligned} \tag{5.19}$$

Siguiendo en igual forma, se obtiene

$$x^{(3)} = \begin{bmatrix} 0.99 \\ 2.00 \\ 1.00 \end{bmatrix} \quad x^{(4)} = \begin{bmatrix} 1.00 \\ 2.00 \\ 1.00 \end{bmatrix} \quad x^{(5)} = \begin{bmatrix} 1.00 \\ 2.00 \\ 1.00 \end{bmatrix}. \tag{5.20}$$

Obsérvese que con este método se ha llegado a la solución del sistema en cinco iteraciones, mientras que con el de Jacobi se llegaba a la solución hasta la octava iteración.

A diferencia del método de Jacobi, el método de Gauss-Seidel siempre converge cuando el de Jacobi lo hace, puede converger cuando el de Jacobi no lo hace, y en general converge más rápidamente que el método de Jacobi.

5.1.3 Método de Sobre-Relajaciones

El *método de Sobre-Relajaciones* es un método para resolver sistemas de ecuaciones algebraicas lineales, con la finalidad de converger hacia la solución más rápido, utilizando

la solución que proporciona el método Gauss-Seidel y una constante ω, definiéndose de la siguiente manera:

$$x^{(k+1)} = (1-\omega)\, x^{(k)} + \omega \hat{x}^{(k+1)}, \tag{5.21}$$

donde $\hat{x}$ sería la solución que da el método Gauss-Seidel, por lo tanto, las ecuaciones que determinan la iteración $k+1$ son:

$$\begin{aligned}
x_1^{(k+1)} &= (1-\omega)\, x_1^{(k)} + \frac{\omega}{a_{11}} \left(b_1 - a_{12}x_2^{(k)} - a_{13}x_3^{(k)} - \ldots - a_{1n}x_n^{(k)} \right) \\
x_2^{(k+1)} &= (1-\omega)\, x_2^{(k)} + \frac{\omega}{a_{22}} \left(b_2 - a_{21}x_1^{(k+1)} - a_{23}x_3^{(k)} - \ldots - a_{2n}x_n^{(k)} \right) \\
x_3^{(k+1)} &= (1-\omega)\, x_3^{(k)} + \frac{\omega}{a_{33}} \left(b_3 - a_{31}x_1^{(k+1)} - a_{32}x_2^{(k+1)} - \ldots - a_{3n}x_n^{(k)} \right) \\
&\ldots \\
&\ldots \\
x_n^{(k+1)} &= (1-\omega)\, x_4^{(k)} + \frac{\omega}{a_{nn}} \left(b_n - a_{n1}x_1^{(k+1)} - a_{n2}x_2^{(k+1)} - \ldots - a_{nn-1}x_{n-1}^{(k+1)} \right).
\end{aligned} \tag{5.22}$$

Notas sobre ω:

- Si $\omega = 1$ el método se reduce al de Gauss-Seidel.
- Para matrices definidas positivas el método converge para $0 < \omega < 2$.
- Se obtienen buenas velocidades de convergencia para $\omega \approx 1.6 - 1.8$.
- La velocidad de convergencia puede llegar a ser uno o dos órdenes de magnitud superior a la del método de Gauss-Seidel estándar.

Ahora se resolverá el sistema que se mostró en (5.9) por este método.

$$\begin{aligned}
x_1^{(k+1)} &= (1-\omega)x_1^{(k)} + \omega \left(0.50 + 0.25x_2^{(k)} \right) \\
x_2^{(k+1)} &= (1-\omega)x_2^{(k)} + \omega \left(1.50 + 0.25x_1^{(k+1)} + 0.25x_3^{(k)} \right) \\
x_3^{(k+1)} &= (1-\omega)x_3^{(k)} + \omega \left(0.50 + 0.25x_2^{(k+1)} \right).
\end{aligned} \tag{5.23}$$

Considerando $x_0 = \{0, 0, 0\}^T$ y $\omega = 1.1$ como primera aproximación, se obtiene de (5.23) la primera iteración

$$\begin{aligned} x_1^{(1)} &= (1 - 1.1)(0) + 1.1(0.50 + 0.25(0)) = 0.55 \\ x_2^{(1)} &= (1 - 1.1)(0) + 1.1(1.50 + 0.25(0.55) + 0.25(0)) = 1.80 \\ x_3^{(1)} &= (1 - 1.1)(0) + 1.1(0.50 + 0.25(1.80)) = 1.05. \end{aligned} \tag{5.24}$$

La segunda iteración es

$$\begin{aligned} x_1^{(2)} &= (1 - 1.1)(0.55) + (1.1)(0.50 + 0.25(1.80)) = 0.99 \\ x_2^{(2)} &= (1 - 1.1)(1.80) + (1.1)(1.50 + 0.25(0.99) + 0.25(1.05)) = 2.03 \\ x_3^{(2)} &= (1 - 1.1)(1.05) + (1.1)(0.50 + 0.25(2.03)) = 1. \end{aligned} \tag{5.25}$$

Siguiendo en igual forma, se obtiene

$$x^{(3)} = \begin{bmatrix} 1.00 \\ 2.00 \\ 1.00 \end{bmatrix} \quad x^{(4)} = \begin{bmatrix} 1.00 \\ 2.00 \\ 1.00 \end{bmatrix}. \tag{5.26}$$

Obsérvese que con este método se ha llegado a la solución del sistema en cuatro iteraciones, sólo que esto dependerá de la ω que se considere.

Capítulo 6

El método de gradiente Barzilai y Borwein

En este capítulo se describe el método de gradiente de Barzilai y Borwein para minimización de problemas sin restricciones a gran escala, el cual se utiliza para minimizar la función de costo cuadrática que se utiliza para restaurar imágenes digitales.

6.1 Introducción

El método de gradientes Barzilai y Borwein se utiliza para la minimización de problemas sin restricciones de gran escala:

$$\min_{x \in \Re^n} f(x), \tag{6.1}$$

donde $f : \Re^n \to \Re$. El método se define por:

$$x_{k+1} = x_k - \frac{1}{\alpha_k} g_k, \tag{6.2}$$

donde g_k es el vector gradiente de f en x_k y el escalar α_k está dado por:

$$\alpha_k = \frac{s_{k-1}^t y_{k-1}}{s_{k-1}^t s_{k-1}}, \tag{6.3}$$

donde $s_{k-1} = x_k - x_{k-1}$ y $y_{k-1} = g_k - g_{k-1}$.

Cada iteración del método Barzilai y Borwein requiere solamente operaciones de punto flotante y la evaluación de un gradiente. No hay cálculos de matriz y no se requiere búsqueda lineal durante el proceso. La dirección de la búsqueda es siempre el negativo de la dirección del gradiente, pero la elección del tamaño de paso no es la elección clásica del método de máximo descenso. De hecho Barzilai y Borwein[2] observaron que esta nueva elección de tamaño de paso requirió menos trabajo computacional y mostró velocidad de convergencia superior a la del método de gradientes para casos cuadráticos.

El objetivo de este capítulo es incrustar el método de gradientes de Barzilai y Borwein en una estrategia de globalización que acepten el tamaño de paso dado por (6.3) tan frecuentemente como sea posible.

En las siguientes subsecciones se presenta una estrategia de globalización adecuada para el método Barzilai y Borwein. Esta estrategia está basada en la técnica de búsqueda lineal no monótona de Grippo, Lampariello y Lucidi [3], para los métodos de Newton. Se discuten las propiedades de este nuevo algoritmo y se establece la convergencia global bajo suposiciones moderadas.

6.2 Estrategia de globalización

Los métodos estándares para la solución de (6.1), usualmente generan una secuencia de iteraciones para las cuales un decremento suficiente en la función objetivo es aplicado en cada iteración. En muchos casos, la estrategia global consiste en aceptar el tamaño de

paso hacia la línea de dirección, si se satisfacen las bien conocidas condiciones Armijo-Goldstein-Wolfe. Esquemas prácticos de búsqueda lineal han sido desarrollados para ejecutar estas condiciones cuando se combina con Newton, quasi-newton y métodos de gradientes conjugados[1].

Hay algunas desventajas para forzar las condiciones Armijo-Goldstein-Wolfe cuando se combina con el método de gradientes Barzilai y Borwein. Una de las desventajas es que ejecutando decrementos en cada iteración destruirá algunas de las propiedades locales del método. Así como se argumentó en Fletcher[6] y Glunt, Hayden y Raydan[8] la elección del tamaño de paso (6.3) es asociada a los eigenvalores de la Hessiana en el minimizador y no a los valores de la función. Además, dado que la dirección de búsqueda es siempre el negativo de la dirección del gradiente, obligar a decrementar en cada iteración, reduciéndose al método de máximo descenso, el cual es conocido por ser lento.

Por consiguiente se ejecuta una condición mas débil de la forma:

$$f(x_{k+1}) \leq \max_{0 \leq j \leq M} f(x_{k-j}) + \gamma g_k^t (x_{k+1} - x_k), \tag{6.4}$$

donde M es un entero no negativo y γ es un número positivo pequeño. Este tipo de condición (6.4) fue introducida por Grippo, Lampariello y Lucidi[3] y aplicado exitósamente a los métodos de Newton para un conjunto de funciones de prueba. Reciéntemente, el mismo tipo de técnica de búsqueda lineal no monótona ha sido incorporado en una variedad de algoritmos de optimización[2]. La condición (6.4) permite a la función incrementarse en algunas iteraciones y a pesar de eso garantiza la convergencia global. Esta característica se adecua bastante bien con el comportamiento no monótono del método de gradiente Barzilai y Borwein.

[1] Para mayor información sobre este tópico vea [4], [5] y [10].
[2] Ver por ejemplo [9], [11] y [12].

6.3 Algoritmo Global Barzilai y Borwein

Dados:

x_0, α_0,

los enteros $M \geq 0$, $\gamma \in (0,1)$, $\delta > 0$,

$0 < \sigma_1 < \sigma_2 < 1$, $0 < \epsilon < 1$,

Hacer $k = 0$.

Paso 1: Si $\|g_k\| = 0$ detenerse.

Paso 2: Si $\alpha_k \leq \epsilon$ o $\alpha_k \geq \dfrac{1}{\epsilon}$ entonces hacer $\alpha_k = \delta$.

Paso 3: Hacer $\lambda = \dfrac{1}{\alpha_k}$

Paso 4: (búsqueda lineal no monótona)

Si $f(x_k - \lambda g_k) \leq \max_{o \leq j \leq \min(k,M)} (f_{k-j}) - \gamma \lambda g_k^t g_k$

entonces hacer $\lambda_k = \lambda$, $x_{k+1} = x_k - \lambda_k g_k$ e ir al **Paso 6**.

Si no ir al **Paso 5.**

Paso 5: Escoger $\sigma \in [\sigma_1, \sigma_2]$, $\lambda = \sigma\lambda$ e ir al **Paso 4**.

Paso 6: Hacer $\alpha_{k+1} = -\dfrac{g_k^t y_k}{\lambda_k g_k^t g_k}$, $k = k+1$ e ir **Paso 1**.

Observaciones:

(1) El objetivo del paso 2 es para evitar direcciones ascendentes y para mantener la secuencia $\{\lambda_k\}$ uniformemente limitada. De hecho para toda k

$$0 < \min\left(\epsilon, \frac{1}{\delta}\right) \leq \lambda_k \leq \max\left(\frac{1}{\epsilon}, \frac{1}{\delta}\right),$$

(2) El algoritmo global Barzilai y Borwein no puede ciclarse indefinidamente entre los pasos 4 y 5. Desde luego, que dado $\lambda g_k^t g_k > 0$ y $\gamma < 1$, para valores suficientemente pequeños de λ, la condición en el paso 4 se satisface.

(3) Dado que $s_k = -\lambda_k g_k$, entonces la definición de α_{k+1} dada en el paso 6 es equivalente a la dada en (6.3). La ventaja de la definición usada en el algoritmo es que evita

el almacenamiento del vector s_k y reduce a 3n localidades de almacenamiento del algoritmo GBB.

(4) Para $k = 0$, la condición en el paso 4 se reduce a la condición de Armijo. Para $k > 0$, la función objetivo quizá incremente a algunas iteraciones. Sin embargo, $f(x_k) \leq f(x_0)$ para toda k y para el conjunto de nivel $\{x : f(x) \leq f(x_0)\}$ contiene la secuencia entera de iteraciones $\{x_k\}$.

Las propiedades de convergencia del algoritmo global Barzilai y Borwein son indicadas en el siguiente teorema.

Teorema 7.1. Asuma que $\Omega_0 = \{x : f(x) \leq f(x_0)\}$ es un conjunto limitado. Sea $f : \Re^n \to \Re$ continuamente diferenciable en alguna vecindad N de Ω_0. Sea $\{x_k\}$ la secuencia generada por el algoritmo global Barzilai y Borwein, entonces $g(x_j) = 0$ para algún j finito o se cumplen las siguientes propiedades:

(i) $\lim_{k\to\infty} \|g_k\| = 0$.

(ii) Un punto no límite de $\{x_k\}$ es un máximo local de f.

(iii) Si el número de puntos estacionarios de f en Ω_0 es finito, entonces la secuencia $\{x_k\}$ converge.

Comprobación [14]. Para establecer (i), se hizo la primera parte de la prueba del teorema de convergencia en [3].

Se define $m(k) = \min(k, M)$. Evidentemente, $m(0) = 0$ y

$$0 \leq m(k) \leq \min(m(k-1) + 1, M) \quad \text{para } k \geq 1.$$

Por ende, existe una constante a, tal que $0 < \lambda_k \leq a$ para toda k. Por supuesto, en el algoritmo global Barzilai y Borwein $a = \max(\epsilon^{-1}, \delta^{-1})$. Finalmente, en consecuencia existen números positivos c_1 y c_2, tales que la dirección de búsqueda d_k satisface

$g_k^t d_k \leq -c_1 \|g_k\|_2^2$ y $\|d_k\|_2 \leq c_2 \|g_k\|_2$. De hecho, en el algoritmo global Barzilai y Borwein, la dirección de búsqueda d_k es $-g_k$ para toda k, y así, $c_1 = c_2 = 1$. Por consiguiente, se obtiene la ecuación (14) en [3] que en este caso se reduce a:

$$\lim_{k\to\infty} \lambda_k \|g_k\|_2 = 0.$$

Dado $\lambda_k \geq \min\left(\epsilon, \frac{1}{\delta}\right)$ para toda k, entonces parte de (i) se mantiene. Las afirmaciones (ii) y (iii) se siguen directamente de la convergencia del teorema en [3].

Nótese que, forzando la condición débil (6.4), la secuencia generada por el algoritmo global Barzilai y Borwein tiene la siguiente propiedad:

$$\lim_{k\to\infty} \|g_k\| = 0.$$

Este es un contraste brusco para los métodos de gradientes conjugados (Fletcher-Reeves, Polak-Ribière, etc.), para los cuales muchas de las condiciones fuertes han sido impuestas para obtener el resultado más débil:

$$\liminf_{k\to\infty} \|g_k\| = 0.$$

Para una discusión más amplia sobre las propiedades de convergencia de los métodos de gradientes conjugados vea Nocedal [10] y Gilbert y Nocedal [7].

En la instrumentación del algoritmo global Barzilai y Borwein en [14] se usó $\gamma = 10^{-4}$, $\epsilon = 10^{-10}$, $\sigma_1 = 0.1$, $\sigma_2 = 0.5$, $\alpha_0 = 1$ y $M = 10$. Se escogió el parámetro ϵ por ser un número muy pequeño, para que tomara el paso de Barzilai y Borwein tantas veces como fuera posible. No obstante, si la condición en el paso 2 se cumplió en la iteración

k, entonces el parámetro δ fue escogido de la siguiente manera:

$$\delta = \begin{cases} 1 & \text{si} \quad \|g_k\|_2 > 1 \\ \|g_k\|_2^{-1} & \text{si} \quad 10^{-5} < \|g_k\|_2 \leq 1 \\ 10^5 & \text{si} \quad \|g_k\|_2 < 10^{-5} \end{cases}$$

Con esta elección de δ, la secuencia permanece uniformemente limitada. En el paso 5, σ es escogido por medio de una interpolación cuadrática descrita en [4].

6.4 Observaciones sobre el algoritmo

El método de Barzilai y Borwein puede incorporarse en una estrategia global de modo que conserve las buenas características del método y solamente requiere 3n localidades de almacenamiento. Dado que la dirección de búsqueda es siempre el negativo de la dirección del gradiente, es trivial asegurar que las direcciones de descenso son generadas en cada iteración. Esto es un contraste para los métodos de gradientes conjugados, para los cuales, una búsqueda lineal muy precisa ha sido desarrollada en cada iteración para generar direcciones de descenso.

La elección del parámetro M es un asunto delicado y amerita más investigación.

Capítulo 7

Algoritmo Propuesto

Este capítulo comienza describiendo cuál va a ser la función a minimizar, incluyendo el concepto de modelo de resortes junto con la distribución de Gibbs. También se describen cómo quedan las funciones de cada algoritmo y un diagrama a bloques para los métodos iterativos. Finalmente se describe lo que significan las variables en el método de Gradiente de Barzilai y Borwein.

7.1 Obtención de la función de restauración

Para obtener la función que hará la restauración de la imagen, es necesario introducir un último concepto de suma importancia, el *modelo de resortes*. Este modelo es utilizado para aplanar una imagen, por lo que un resorte ideal será aquél cuya longitud de reposo sea igual a cero, entonces teniendo en cuenta la ecuación (3.13), se tiene que:

$$U(f) = \lambda \sum_{\{i,j\}} (f_i - f_j)^2, \tag{7.1}$$

donde λ, que se elige de manera a priori, es la constante que trata de hacer la longitud de reposo igual a cero. En la distribución de Gibbs, significa que si tiene una probabilidad

máxima, su energía potencial será cero, y si tiene una probabilidad muy baja, su energía potencial será muy alta. Lo que se busca es encontrar una probabilidad máxima.

Completando la función, se obtiene:

$$U(f) = \frac{1}{2}\left[\sum_i (f_i - g_i)^2 + \lambda \sum_{i,j} (f_i - f_j)^2\right], \tag{7.2}$$

y lo que interesa es minimizar esta función, es decir:

$$\min U(f) = \frac{1}{2}\left[\sum_i (f_i - g_i)^2 + \lambda \sum_{i,j} (f_i - f_j)^2\right]. \tag{7.3}$$

La solución de la ecuación (7.3) es:

$$f_i(1 + \lambda |N_i|) - \lambda \sum_{f_j \in N_i} f_j = g_i, \tag{7.4}$$

donde:

$|N_i|$ es el número de vecinos de i.

N_i son los vecinos de i.

g_i es la imagen original(observaciones).

7.2 Funciones de los algoritmos

7.2.1 Método de Jacobi

Las iteraciones del método de Jacobi aplicado a la ecuación (7.4) tienen la forma:

$$f_i^{(k+1)} = \frac{g_i + \lambda \sum_{f_j \in N_i} f_j^{(k)}}{(1 + \lambda |N_i|)}. \tag{7.5}$$

En la figura 7-1 se puede ver un diagrama a bloques de cómo se puede instrumentar el algoritmo.

7.2.2 Método de Gauss-Seidel

Las iteraciones del método de Gauss-Seidel aplicado a la ecuación (7.4) tienen la forma:

$$f_i^{(k+1)} = \frac{g_i + \lambda \sum\limits_{f_j \in N_i} f_j^{(k+1)}}{(1 + \lambda \left| N_i \right|)}, \tag{7.6}$$

debido a que, el método de Gauss-Seidel obtiene los resultados de sus iteraciones de los resultados de la iteración anterior, y los que ya han obtenido en la iteración actual.

En la figura 7-2 se puede ver un diagrama a bloques de cómo se puede instrumentar el algoritmo.

7.2.3 Método de Sobre-Relajaciones

Las iteraciones del método de Sobre-relajaciones aplicado a la ecuación (7.4) tienen la forma:

$$f_i^{(k+1)} = (1 - \omega) f_j^{(k)} + \omega \left(\frac{g_i + \lambda \sum\limits_{f_j \in N_i} f_j^{(k+1)}}{(1 + \lambda \left| N_i \right|)} \right), \tag{7.7}$$

debido a que el método de Sobre-relajaciones obtiene los resultados de sus iteraciones, a partir de los resultados que da el Gauss-Seidel y el valor de ω.

En la figura 7-3 se puede ver un diagrama a bloques de cómo se puede instrumentar el algoritmo.

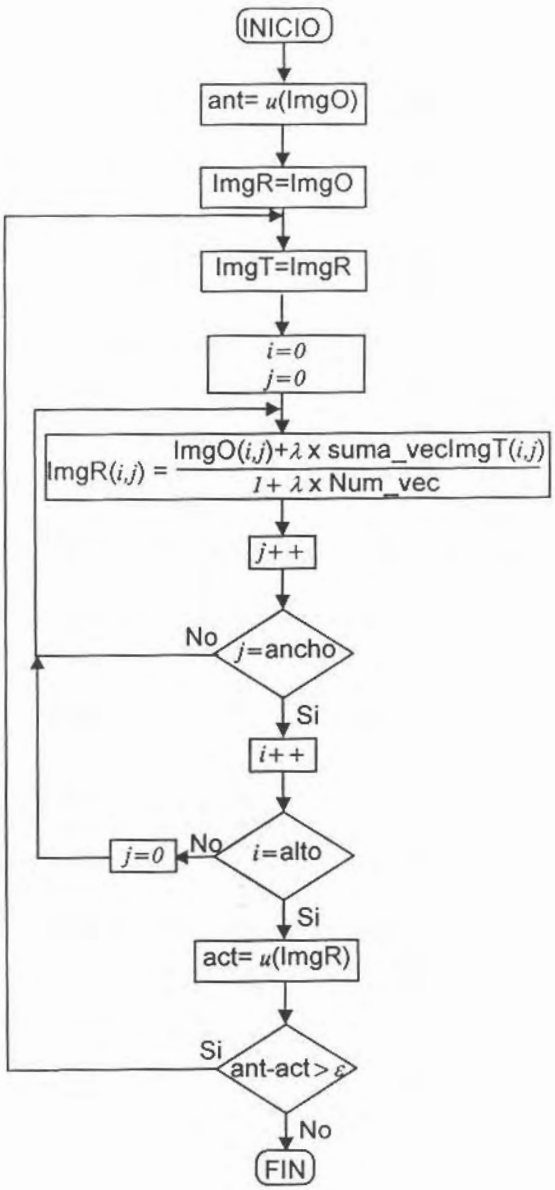


Figura 7-1: Diagrama a bloques del método de Jacobi para la restauración de una imagen, donde ImgO es la imagen original y ImgR es la imagen restaurada.

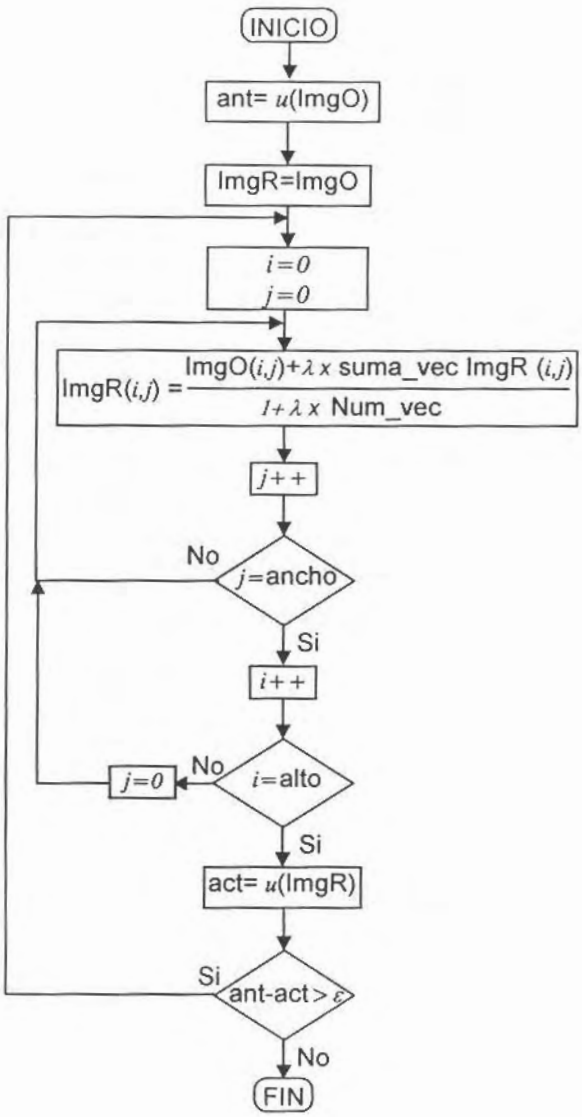


Figura 7-2: Diagrama a bloques del método de Gauss-Seidel para la restauración de una imagen, donde ImgO es la imagen original y ImgR es la imagen restaurada.

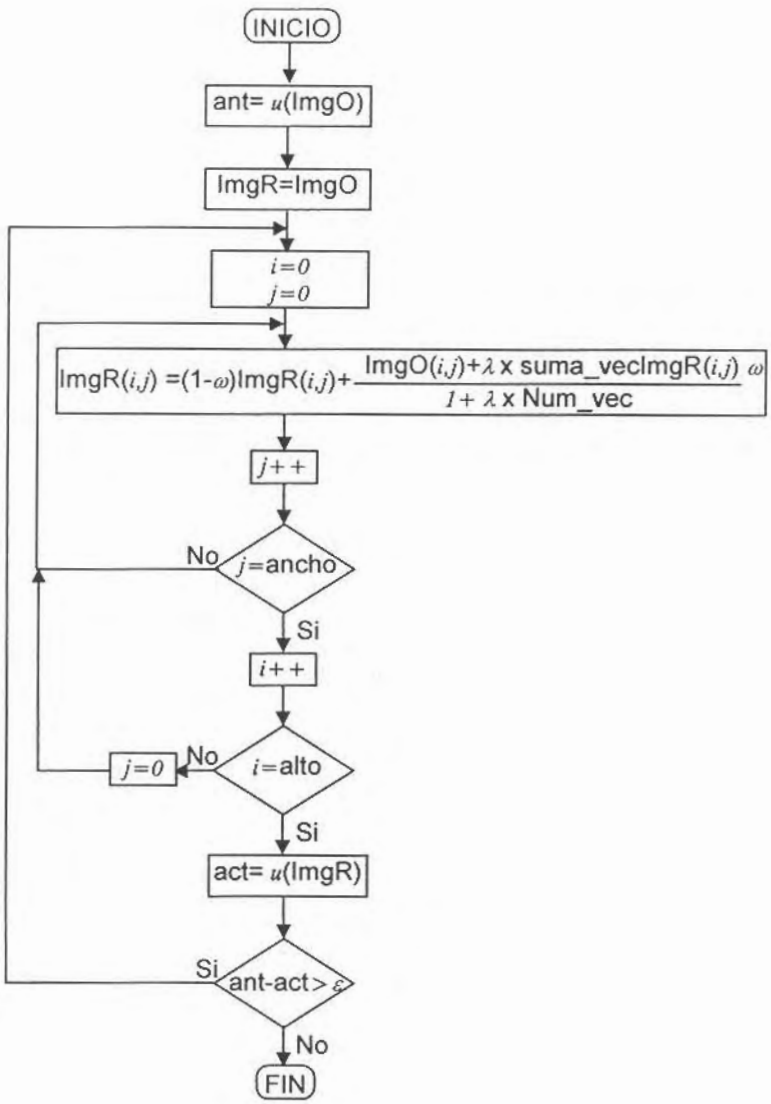


Figura 7-3: Diagrama a bloques del método de Sobre-Relajaciones para la restauración de una imagen, donde ImgO es la imagen original y ImgR es la imagen restaurada.

7.2.4 Método de Gradiente de Barzilai y Borwein

Para este método se utilizan las siguientes definiciones:

$$\hat{f}_k = \frac{1}{2}\left[\sum_i (f_i - g_i)^2 + \lambda \sum_{i,j} (f_i - f_j)^2\right] \tag{7.8}$$

$$\hat{g}_k = f_i\,(1 + \lambda\,|N_i|) - g_i - \lambda \sum_{f_j \in N_i} f_j \tag{7.9}$$

$$\hat{x}_k = f_i \tag{7.10}$$

donde $\hat{f}_k, \hat{g}_k$ y $\hat{x}_k$, son las variables f_k, g_k y x_k que maneja el algoritmo global Barzilai y Borwein.

Los parámetros utilizados fueron $\gamma = 10^{-4}$, $\epsilon = 10^{-10}$, $\sigma_1 = 0.1$, $\sigma_2 = 0.5$, $\alpha_0 = 1$ y $M = 10$.

Capítulo 8

Pruebas

En este capítulo se muestran las pruebas realizadas, las cuales consisten en la restauración de cinco imágenes, tres de ellas, son imágenes comerciales a las cuales se les sumó ruido Gaussiano, y las otras dos son imágenes sintéticas. Todas las imágenes que se utilizaron en estas pruebas están en escala de grises y de dimensión 128x128 pixeles.

Las imágenes con el ruido Gaussiano se generaron en Caliman[15], programa auxiliar para el entendimiento del procesamiento de señales y de imágenes, el cual realiza operaciones aritméticas, derivadas, convolución, transformadas y generación de imágenes.

El programa se implementó en Borland C++ Builder 3 Standard, y las pruebas se realizaron en una computadora Pentium III a 450 Mhz con 128 MB de RAM.

Para comparar las imágenes restauradas por los métodos iterativos (Jacobi, Gauss-Seidel y Sobre-relajaciones) y por el método de Barzilai y Borwein, además de los detalles que se pueden observar en las imágenes obtenidas, se midió el tiempo de restauración y el número de iteraciones empleadas.

Se agregaron tres pruebas extras con filtros de operaciones puntuales, en los que se utiliza 4-vecindad, con la intención de mostrar la diferencia que producen los filtros de operaciones puntuales sólo a nivel de observación, aunque de éstos no se menciona una descripción en el presente trabajo de tesis.

8.1 Pruebas sobre la imagen 1

(a) Imagen Original

(b) Imagen con ruido

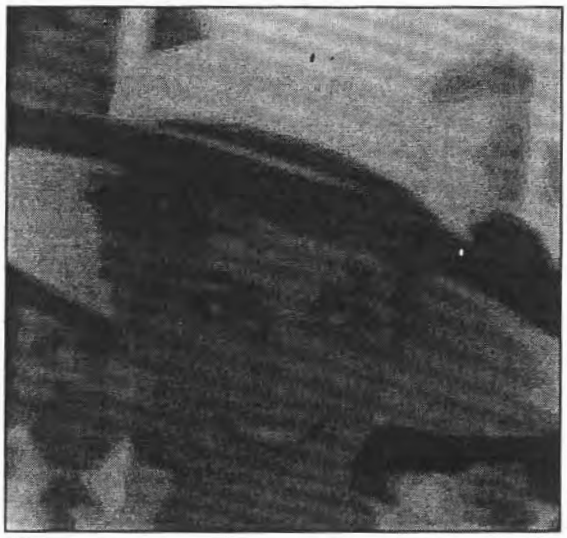
(c) Imagen restaurada con el método de B. y B.

Barzilai y Borwein: Tiempo: 0:0:06:920 No iter: 10

En la imagen (c) se puede apreciar la restauración con respecto a (b), se pueden observar claramente detalles, como en los dientes, ojos y el sombrero.

(d) Imagen restaurada con el método de Jacobi

(e) Imagen restaurada con el método de Gauss-Seidel



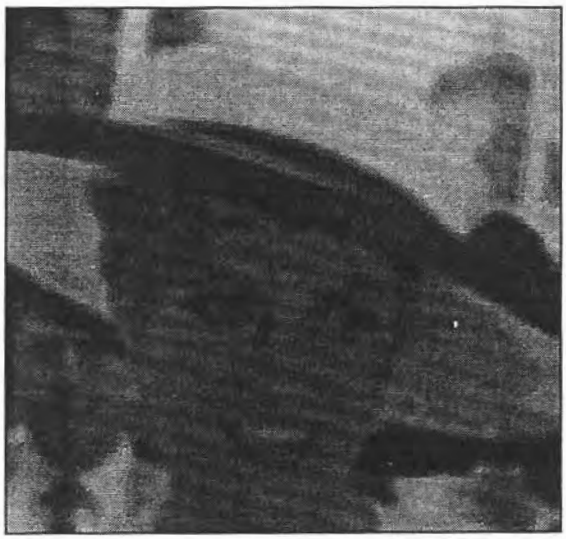
(f) Imagen restaurada con el método de Sobre-relajaciones

Jacobi	Tiempo: 0:0:35:480	No iter: 51
Gauss-Seidel	Tiempo: 0:0:09:390	No iter: 14
Sobre-Relajaciones	Tiempo: 0:0:34:710	No iter: 51

Se puede apreciar la restauración de (d), (e) y (f), con respecto a (b) al igual que (c). Hay que observar detalles en el mentón, frente y orillas del sombrero.

(g) Imagen restaurada con el filtro de convolución

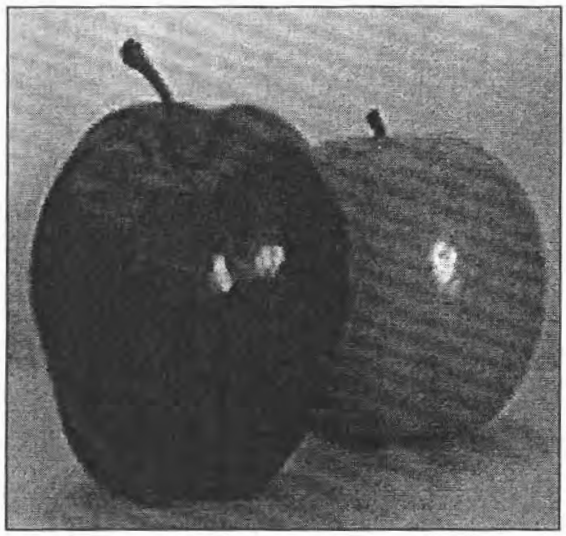
(h) Imagen restaurada con el filtro de mediana



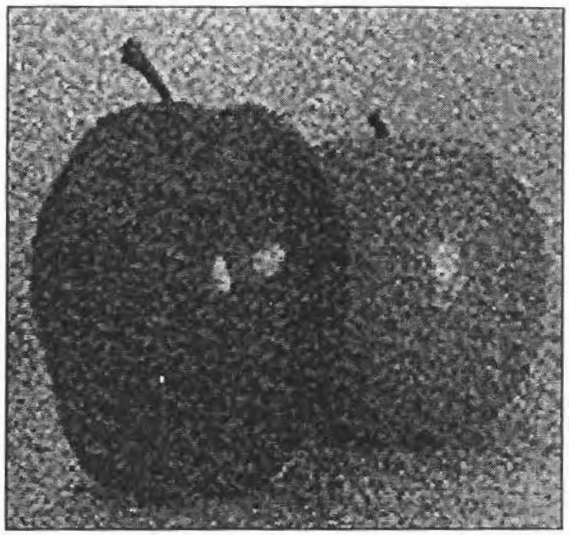
(i) Imagen restaurada con el filtro de media

(g) alcanza un grado de restauración similar a las anteriores, (h) e (i) definitivamente lucen mal.

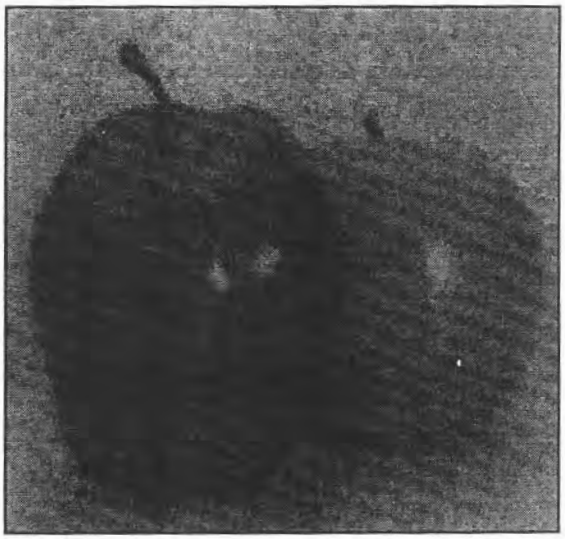
8.2 Pruebas sobre la imagen 2



(a) Imagen Original



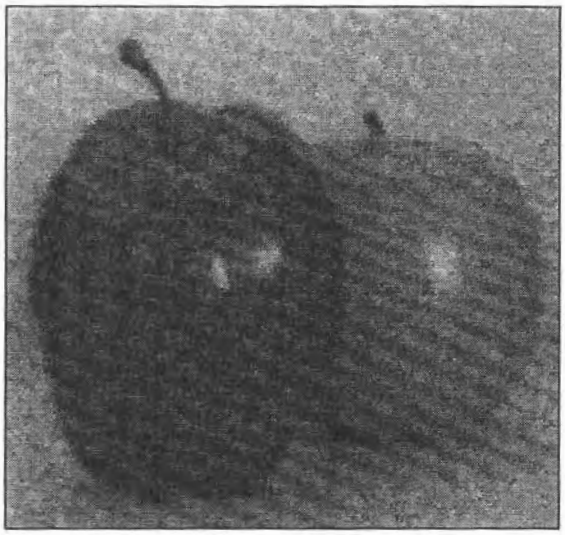
(b) Imagen con ruido



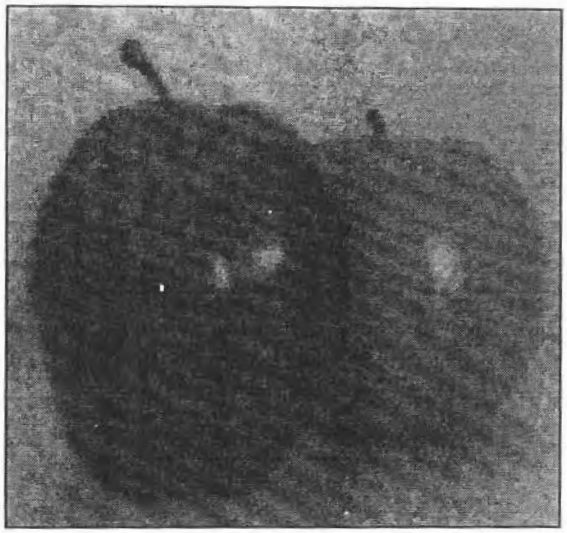
(c) Imagen restaurada con el método de B. y B.

Barzilai y Borwein: Tiempo: 0:0:04:620 No iter: 7

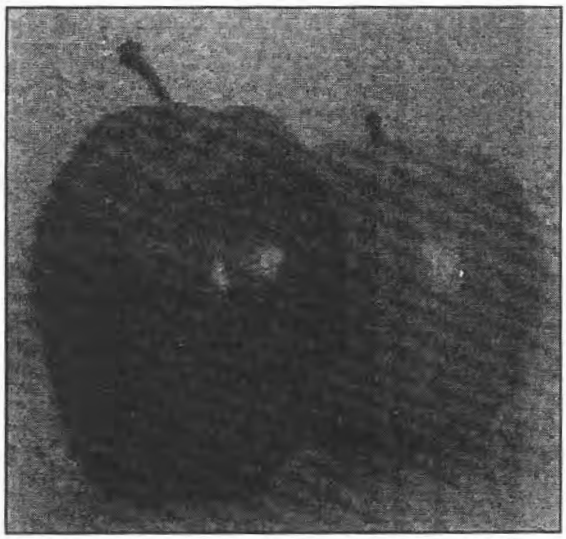
En la imagen (c) se puede apreciar la restauración con respecto a (b), se pueden observar las orillas y el brillo de las manzanas.



(d) Imagen restaurada con el método de Jacobi



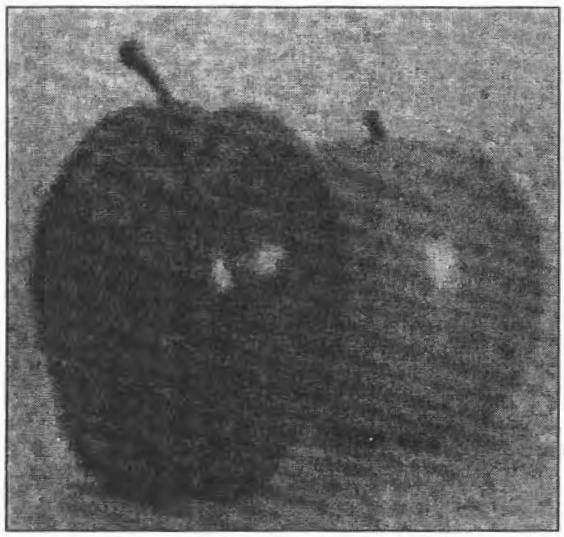
(e) Imagen restaurada con el método de Gauss-Seidel



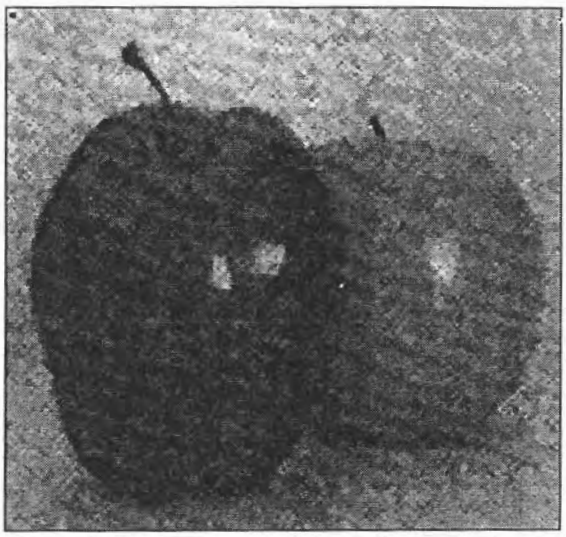
(f) Imagen restaurada con el método de Sobre-relajaciones

Jacobi	Tiempo: 0:0:33:450	No iter: 51
Gauss-Seidel	Tiempo: 0:0:09:170	No iter: 14
Sobre-Relajaciones	Tiempo: 0:0:34:450	No iter: 51

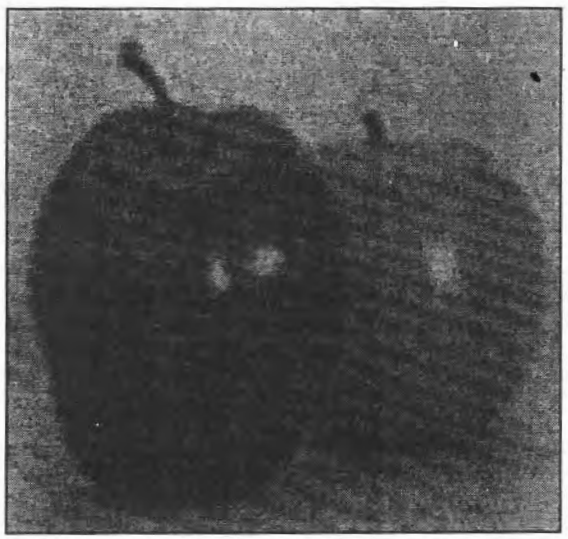
Para estos tres métodos se aprecia una restauración similar a (c). Hay que observar detalles del fondo.



(g) Imagen restaurada con el filtro de convolución



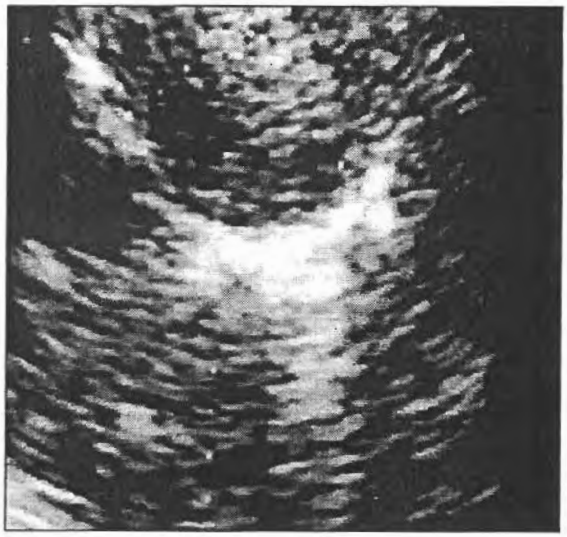
(h) Imagen restaurada con el filtro de mediana



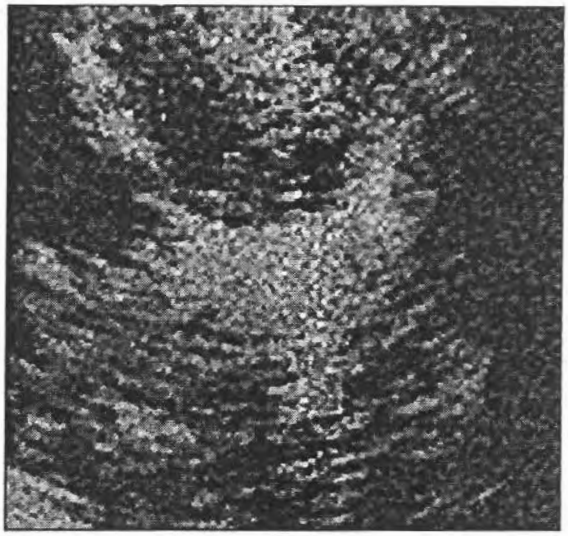
(i) Imagen restaurada con el filtro de media

Las tres restauraciones de (b) lucen mal con respecto a las anteriores.

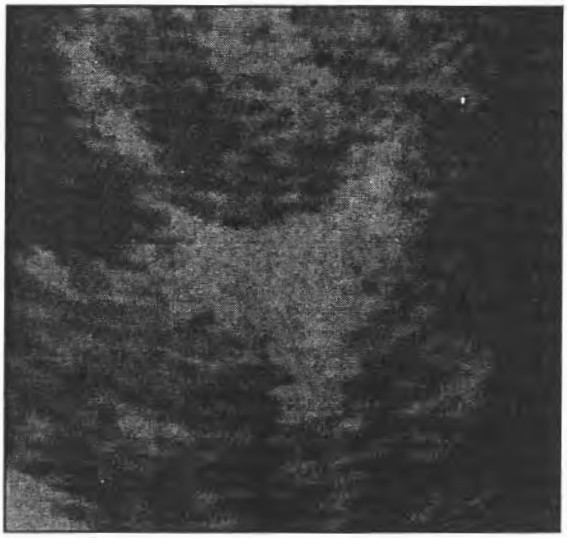
8.3 Pruebas sobre la imagen 3



(a) Imagen Original



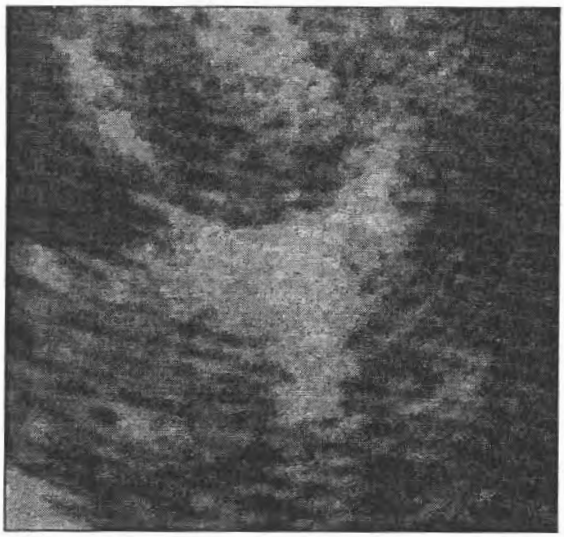
(b) Imagen con ruido



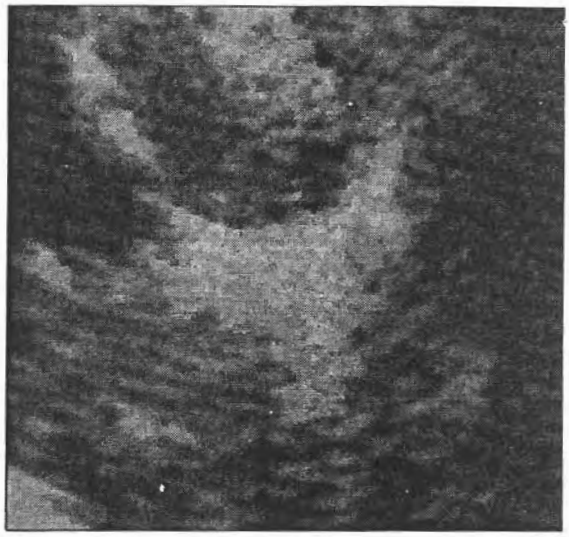
(c) Imagen restaurada con el método de B. y B.

Barzilai y Borwein: Tiempo: 0:0:04:620 No iter: 7

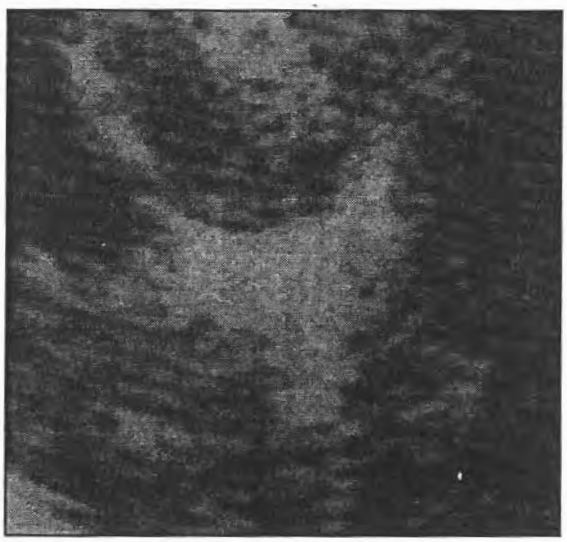
En la imagen (c) se puede apreciar la restauración con respecto a (b), se puede observar la uniformidad del tono obscuro como del tono claro.



(d) Imagen restaurada con el método de Jacobi



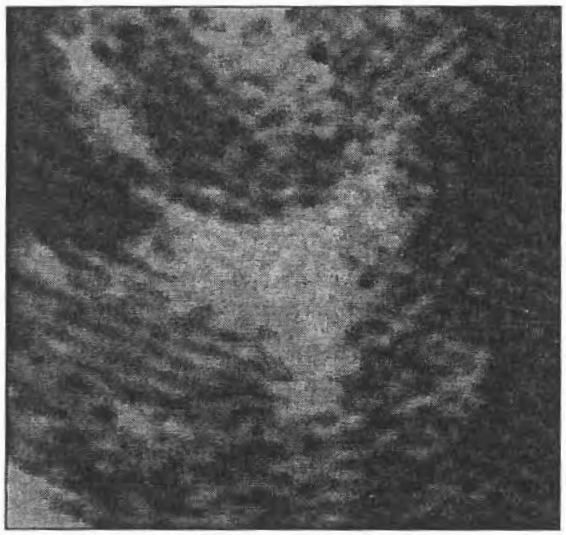
(e) Imagen restaurada con el método de Gauss-Seidel



(f) Imagen restaurada con el método de Sobre-relajaciones

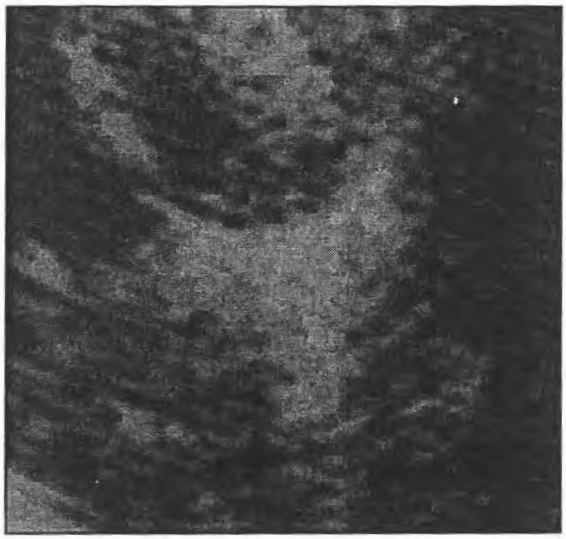
Jacobi	Tiempo: 0:0:33:450	No iter: 51
Gauss-Seidel	Tiempo: 0:0:09:170	No iter: 14
Sobre-Relajaciones	Tiempo: 0:0:33:450	No iter: 51

Para (d), (e) y (f) se aprecia una restauración muy similar a (c). Existe una mayor diferencia en el número de iteraciones.



(g) Imagen restaurada con el filtro de convolución

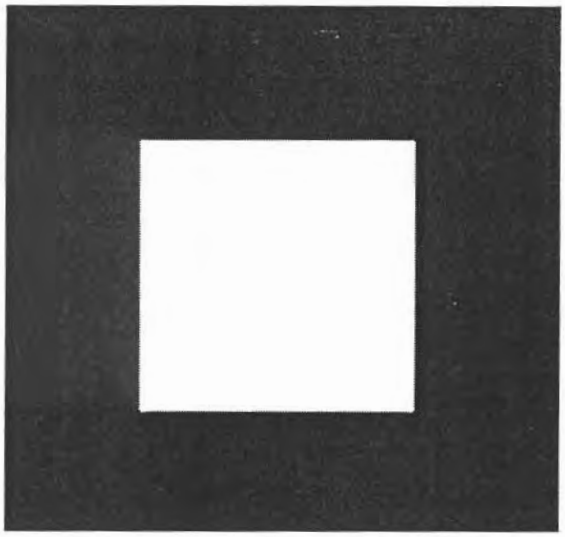
(h) Imagen restaurada con el filtro de mediana



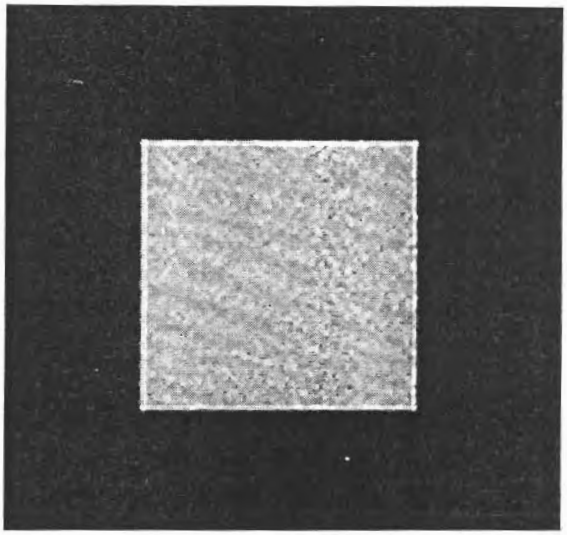
(i) Imagen restaurada con el filtro de media

En las tres restauraciones no se alcanza el mismo grado de restauración que en (c)

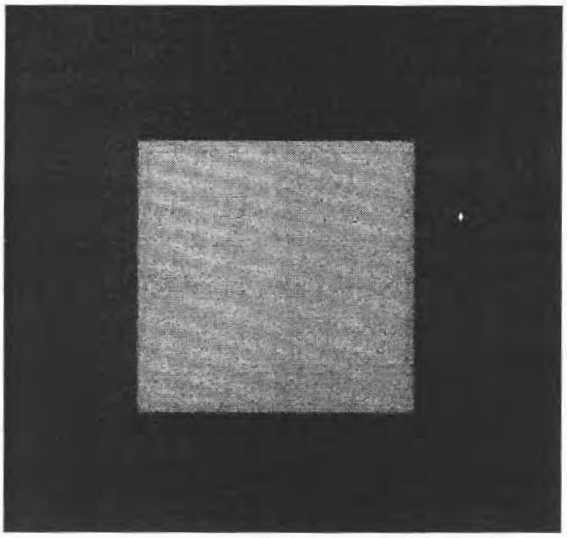
8.4 Pruebas sobre la imagen 4



(a) Imagen Original



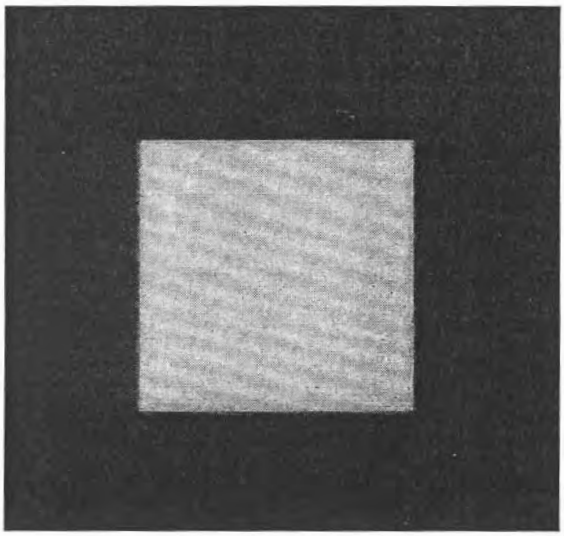
(b) Imagen con ruido



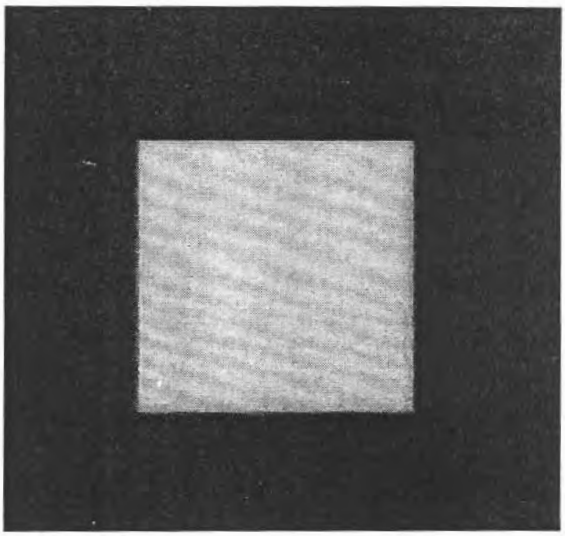
(c) Imagen restaurada con el método de B. y B.

Barzilai y Borwein: Tiempo: 0:0:05:440 No iter: 10

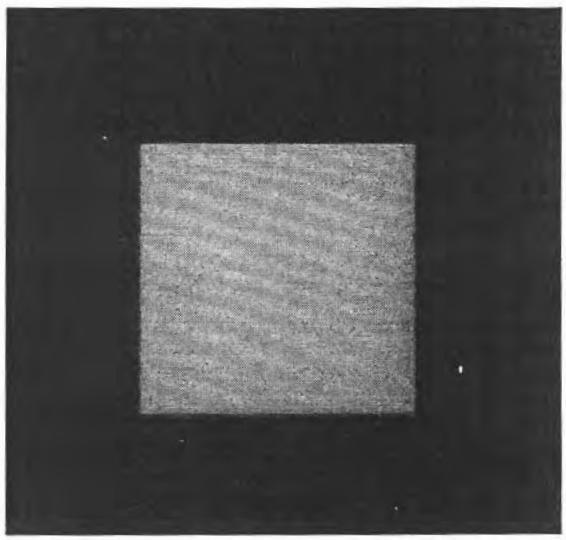
En la imagen (c) se puede apreciar la restauración con respecto a (b), se puede observar la uniformidad del tono exterior e interior.



(d) Imagen restaurada con el método de Jacobi



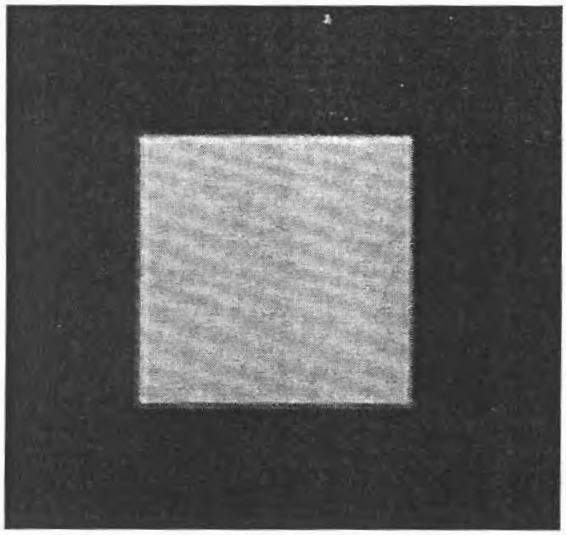
(e) Imagen restaurada con el método de Gauss-Seidel



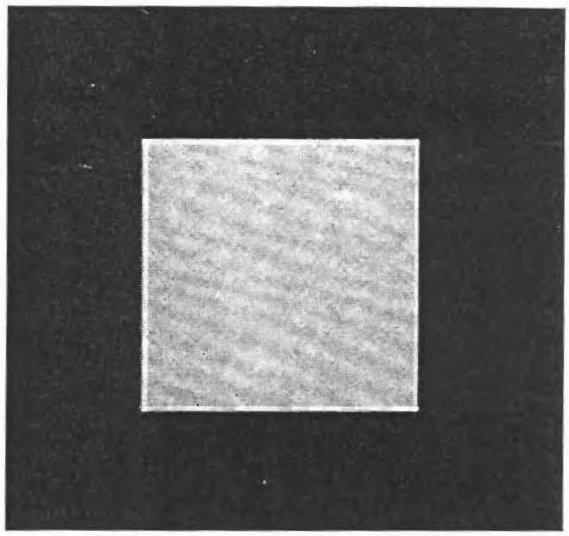
(f) Imagen restaurada con el método de Sobre-relajaciones

Jacobi	Tiempo: 0:0:30:700	No iter: 51
Gauss-Seidel	Tiempo: 0:0:07:690	No iter: 13
Sobre-Relajaciones	Tiempo: 0:0:15:490	No iter: 26

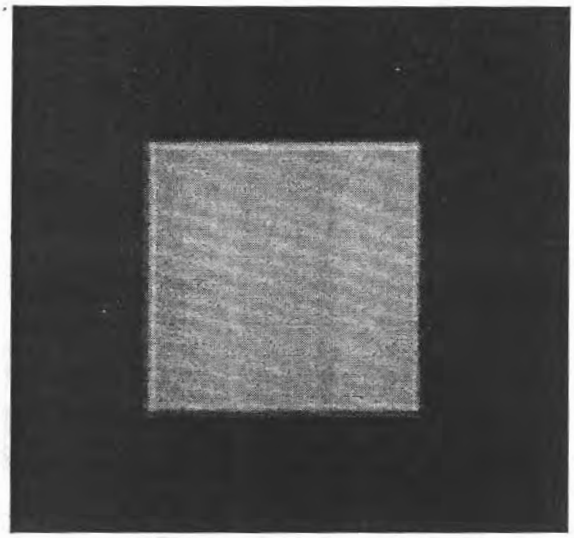
En estos tres métodos se aprecia una restauración muy similar a (c) debido a la falta de tonos de grises.



(g) Imagen restaurada con el filtro de convolución



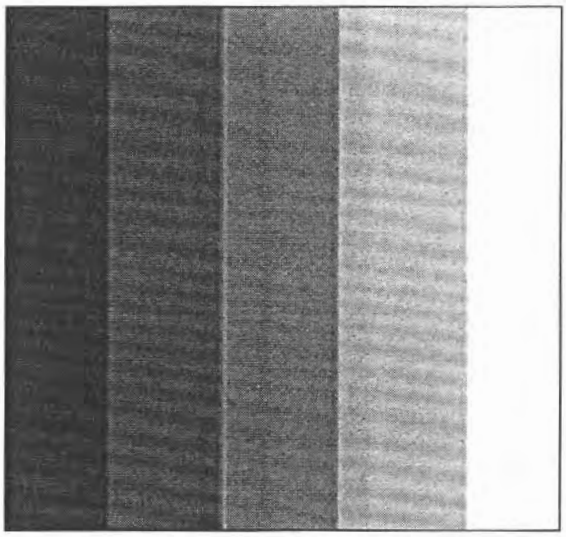
(h) Imagen restaurada con el filtro de mediana



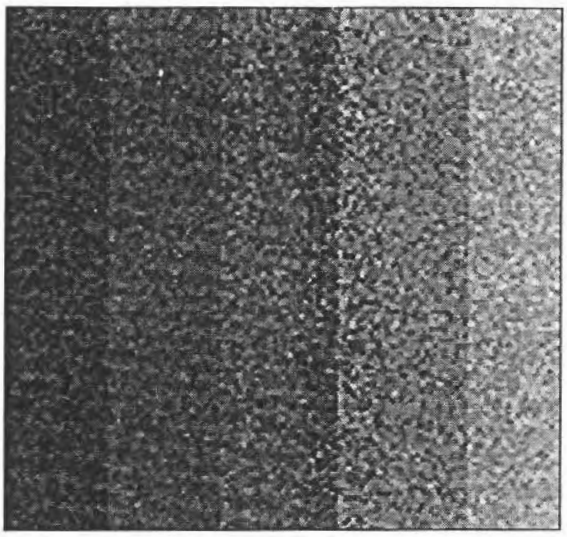
(i) Imagen restaurada con el filtro de media

En las tres restauraciones se puede observar la degradación en las orillas y montones de puntos menos obscuros en el área negra.

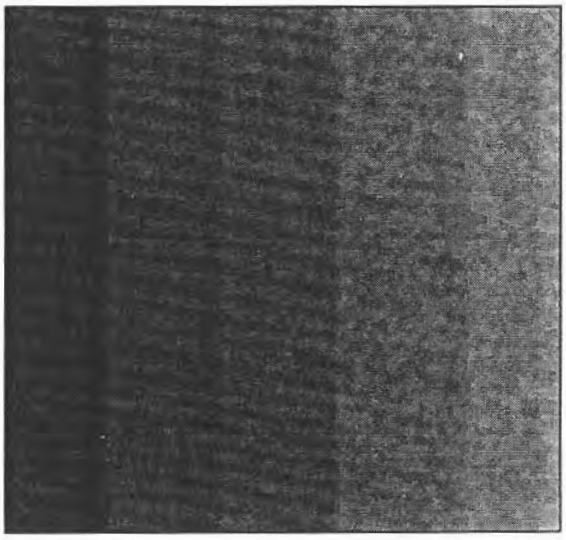
8.5 Pruebas sobre la imagen 5



(a) Imagen Original



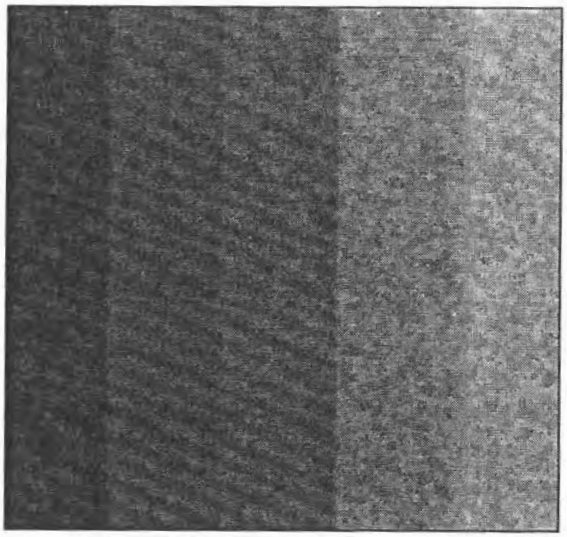
(b) Imagen con ruido



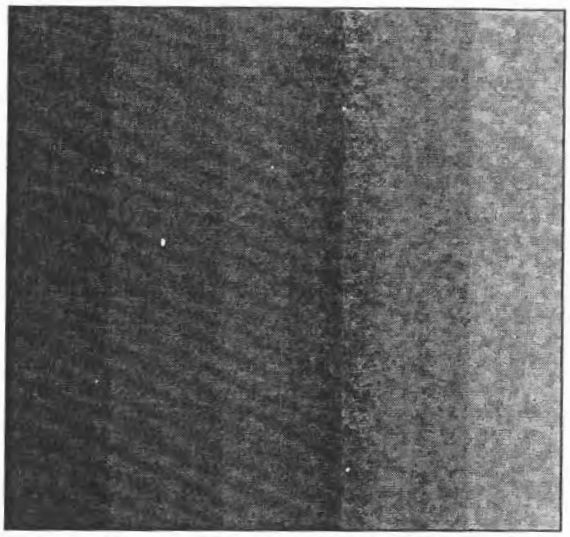
(c) Imagen restaurada con el método de B. y B.

Barzilai y Borwein: Tiempo: 0:0:05:220 No iter: 8

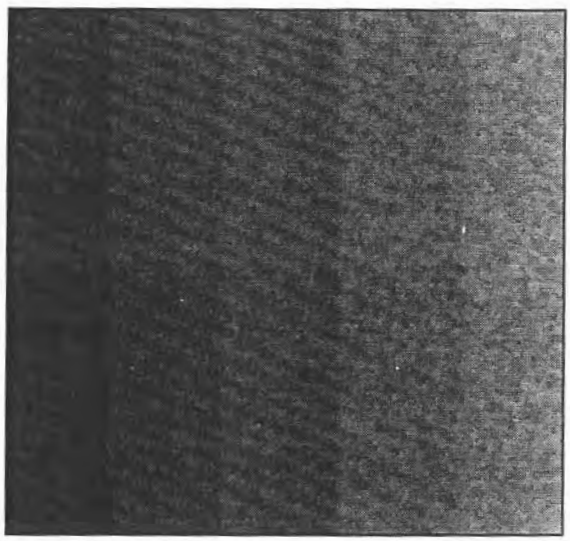
En la imagen (c) se puede apreciar la restauración con respecto a (b), se pueden observar las orillas.



(d) Imagen restaurada con el método de Jacobi



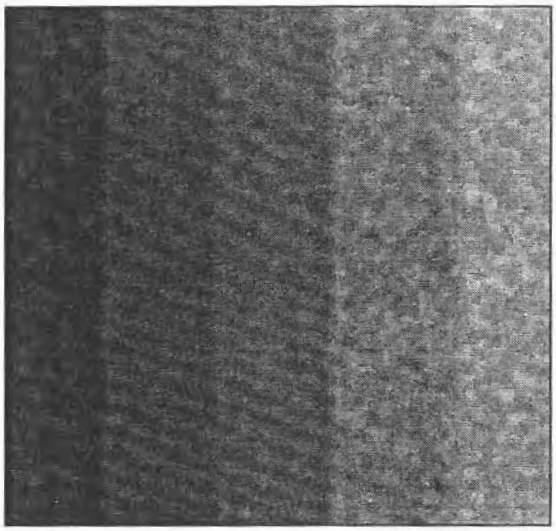
(e) Imagen restaurada con el método de Gauss-Seidel



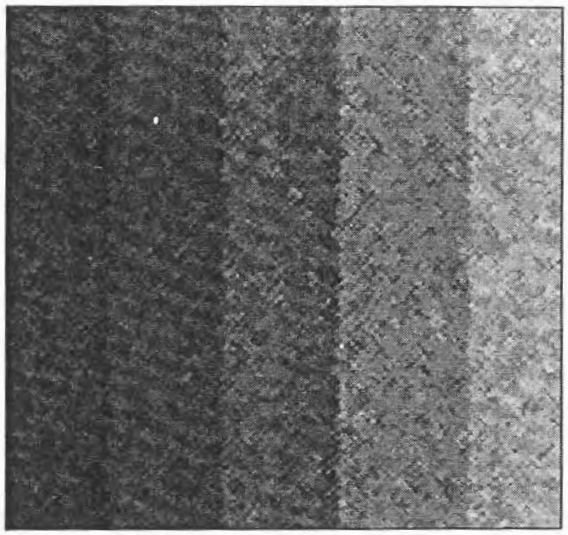
(f) Imagen restaurada con el método de Sobre-relajaciones

Jacobi	Tiempo: 0:0:36:250	No iter: 51
Gauss-Seidel	Tiempo: 0:0:07:630	No iter: 11
Sobre-Relajaciones	Tiempo: 0:0:35:980	No iter: 51

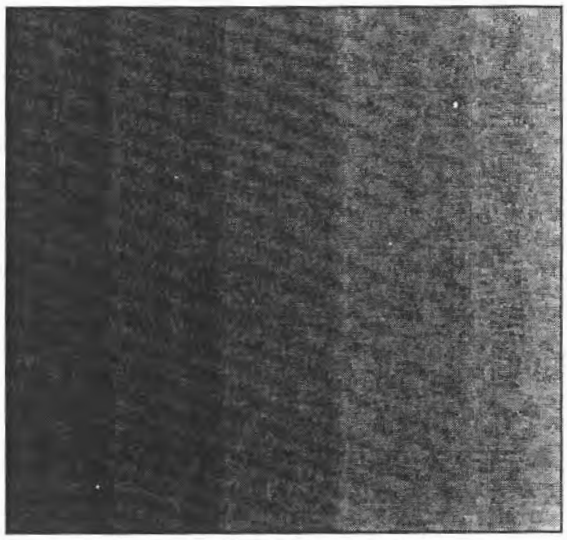
En estos métodos se aprecia una restauración muy similar a (c).



(g) Imagen restaurada con el filtro de convolución



(h) Imagen restaurada con el filtro de mediana



(i) Imagen restaurada con el filtro de media

En las tres restauraciones no se alcanza el mismo grado de restauración que en (c).

Capítulo 9

Conclusiones

A las conclusiones que se llegan al restaurar imágenes con el método de Gradiente de Barzilai y Borwein son las siguientes:

1. El método de Gradiente de Barzilai y Borwein demostró ser una mejor opción para la resturación de imágenes, debido a que los métodos iterativos de Jacobi, Gauss-Seidel y Sobre-relajaciones, emplearon más tiempo y número de iteraciones, y las imágenes restauradas fueron similares. De igual manera, la resturación de imágenes por el método de Gradiente de Barzilai y Borwein, fue mejor que la obtenida por los filtros de operaciones puntuales.

2. El método de Gradiente de Barzilai y Borwein superó a los métodos iterativos de Jacobi y Sobre-relajaciones, debido a que estos métodos al estar buscando el mínimo de la función de restauración, se pueden ciclar alrededor de él y por lo tanto gastan iteraciones y tiempo. El método de Gradiente de Barzilai y Borwein bajo ninguna alteración de parámetros se cicla alrededor del mínimo.

3. El parámetro λ, que se obtuvo de manera a priori, fue fundamental en la restauración de la imagen. Con valores de λ menores a 0.5, aún se observaban secuelas del ruido, y con valores mucho mayores a 1.5 lucía borrosa.

4. El resultado de la restauración de los tres métodos iterativos (Jacobi, Gauss-Seidel y Sobre-relajaciones) es muy parecido, esto se debe a que éstos, al igual que el de Barzilai y Borwein, utilizaban la misma función modelada por los campos aleatorios markovianos, esto con la intención de compararlos bajo las mismas condiciones y aún así, el método de Barzilai y Borwein obtuvo mejores resultados. Por esta razón los filtros de operaciones puntuales no alcanzaron a realizar una buena restauración.

5. Tener un conocimiento a priori sobre la restauración que se va a hacer en una iteración ayuda por mucho con respecto a la iteración anterior, esto incluso lo demostraron los métodos iterativos, los cuales al menos en esta aplicación tuvieron un buen comportamiento, aunque no mejor que el mostrado por el de Barzilai y Borwein.

6. El método de Barzilai y Borwein obtiene muy buenos resultados independientemente de que la imagen tenga mucho o poco ruido gaussiano.

7. A pesar de que los filtros de operaciones puntuales utilizaron una vecindad, el método de Gradiente de Barzilai y Borwein tomó su ventaja en cuanto al conocimiento a priori.

8. El tiempo de restauración es subjetivo, debido a que si se utiliza una computadora más veloz, la restauración se llevará a cabo en menos tiempo. Otro factor que afecta el tiempo de restauración son los procesos ocultos que efectúa Windows, aunque ésta perturbación es del orden de los milisegundos.

Capítulo 10

Manual de Usuario

En este capítulo se muestra la forma en que debe utilizarse el programa en el cual se hicieron las pruebas de la restauración de las imágenes.

10.1 Introducción

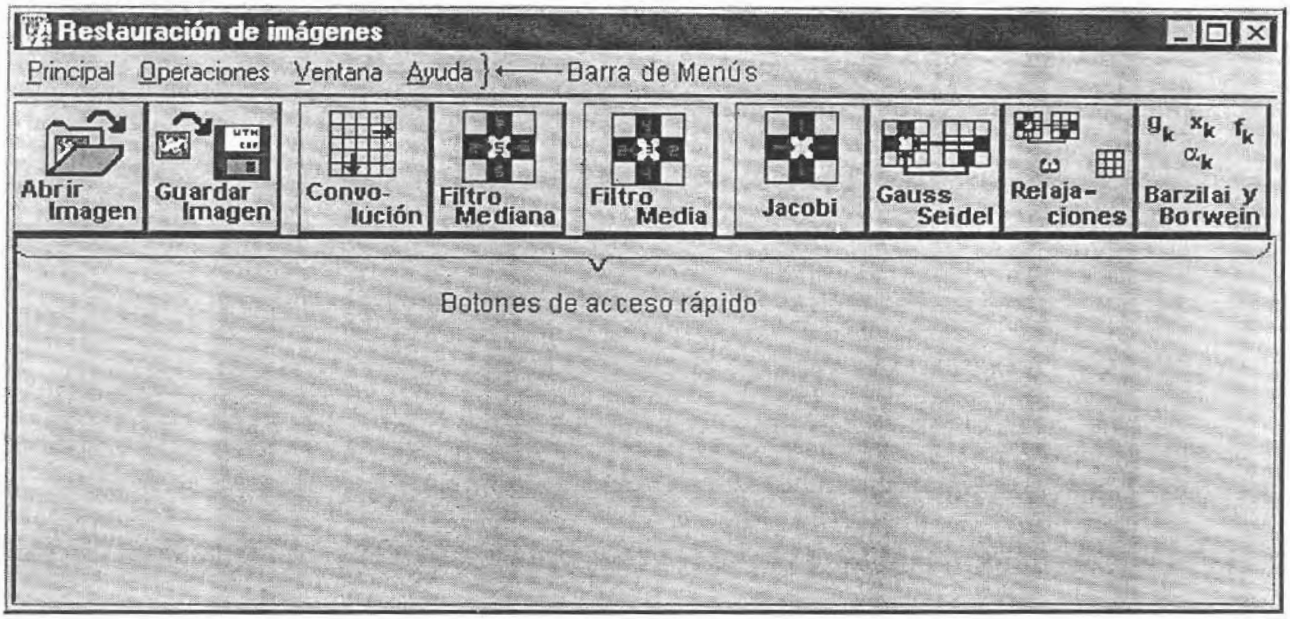
Tesisbyb es un programa que permite restaurar imágenes de maneras diferentes: Método de Barzilai y Borwein, Método de Sobre-relajaciones, Método de Gauss-Seidel, Método de Jacobi, Filtro de Media, Filtro de Mediana y Filtro de Covolución.

El funcionamiento de Tesisbyb es muy simple, sólo se abre una imagen (mostrándose un cuadro de diálogo), se aplica la operación y se muestra la imagen resultante, además permite guardar las imágenes resultantes en un medio de almacenamiento secundario.

Los requerimientos mínimos para utilizar este programa son los siguientes:
Computadora Pentium MMX 200Mhz.
64 MB RAM.
9 MB de espacio disponible en disco duro.

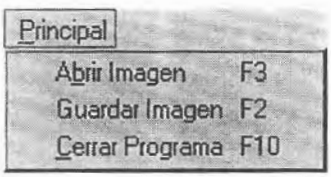
10.2 Partes del programa

La ventana principal del programa se encuentra de la siguiente manera:



En la imagen anterior se pueden observar las dos partes del programa, *Barra de Menús* y *Botones de acceso rápido*, las cuales se describen a continuación.

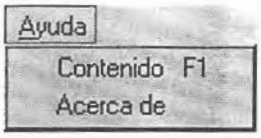
10.2.1 Barra de menús



El Menú Principal tiene tres submenús, *Abrir una imagen*, que muestra un cuadro de diálogo a través del cual se puede buscar la imagen a la que se desea aplicarle alguna operación, *Guardar imagen*, que muestra un cuadro de diálogo a través del cual se puede buscar la ruta en la que se desea guardar la imagen a la que se aplicó alguna operación y *Cerrar Programa*, que finaliza el programa. Los archivos de las imágenes que se deseen abrir o guardar deben ser de formato bmp.

El Menú Operaciones tiene siete submenús, los cuales restauran la imagen seleccionada de acuerdo a la leyenda del submenú.

El Menú Ventana tiene dos submenús, los cuales ordenan las imágenes que estén en la ventana principal, ya sea por título o en cascada.



El Menú Ayuda tiene dos submenús, *Contenido*, muestra la ayuda para la operación del programa y *Acerca de,* como dice su nombre muestra información del programa.

10.2.2 Botones de acceso rápido

A continuación se muestran la descripción de los botones que componen la ventana principal de Tesisbyb.

Al dar un click sobre este botón se muestra un cuadro de diálogo a través del cual se puede buscar la imagen .bmp a la que se desea aplicarle alguna operación.

Al dar un click sobre este botón se muestra un cuadro de diálogo a través del cual se puede buscar la ruta en la que se desea guardar la imagen a la que se le aplicó alguna operación. Esta imágen es guardada con formato bmp.

Al dar un click sobre este botón, se aplica el filtro de convolución a la imagen que se encuentre seleccionada.

Al dar un click sobre este botón, se aplica el filtro de mediana a la imagen que se encuentre seleccionada.

Al dar un click sobre este botón, se aplica el filtro de media a la imagen que se encuentre seleccionada.

Al dar un click sobre este botón, se restaura por el método de Jacobi la imagen que se encuentre seleccionada.

Al dar un click sobre este botón, se restaura por el método de Gauss-Seidel la imagen que se encuentre seleccionada.

Al dar un click sobre este botón, se restaura por el método de Sobre-relajaciones la imagen que se encuentre seleccionada.

Al dar un click sobre este botón, se restaura por el método de Gradiente de Barzilai y Borwein la imagen que se encuentre seleccionada.

10.3 Teclas Rápidas

El programa utiliza 4 teclas rápidas para accesar a las operaciones más comunes:

F1 Ayuda.

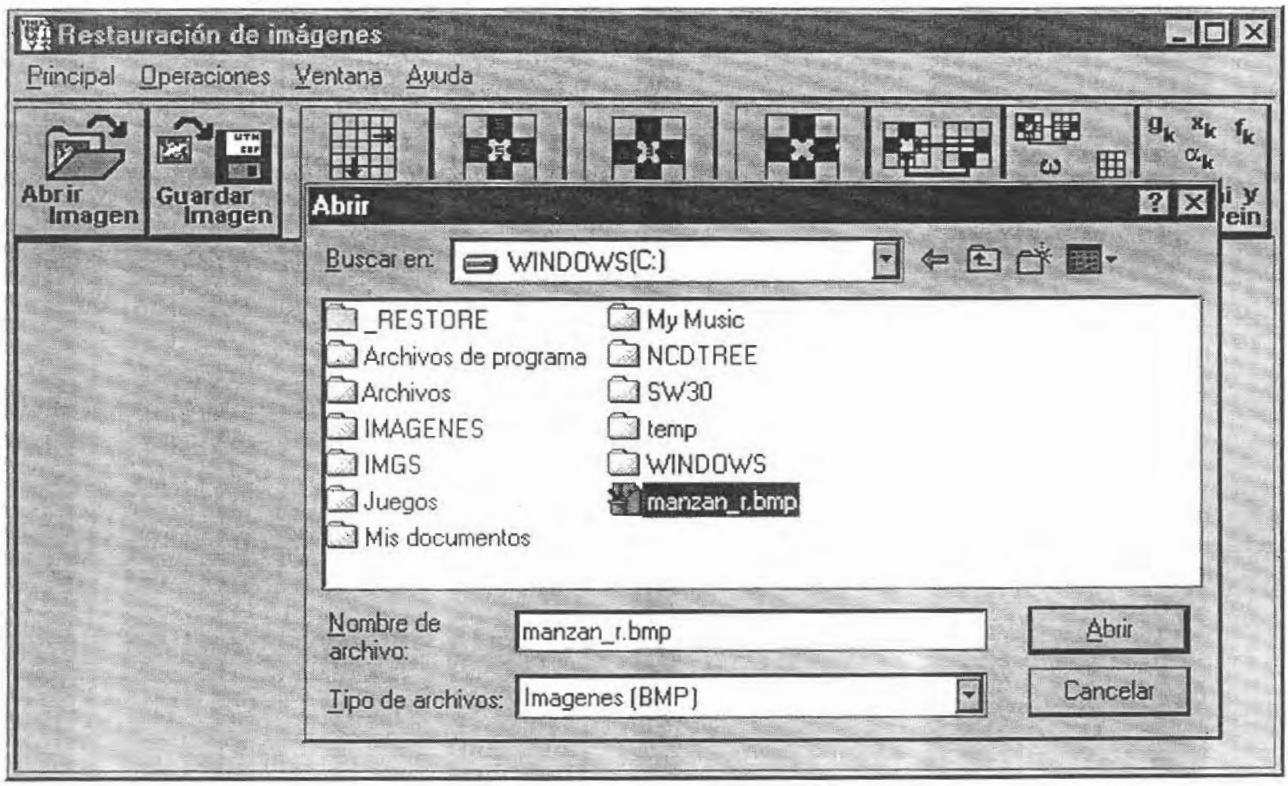
F2 Guardar Imagen.

F3 Abrir Imagen.

F10 Salir del Programa.

10.4 Ejemplo

Suponga que se va a restaurar la imagen c:\manzan_r.bmp, se presiona F3 se selecciona en la unidad C, la imagen manzan_r.bmp, como se ve en la siguiente figura:

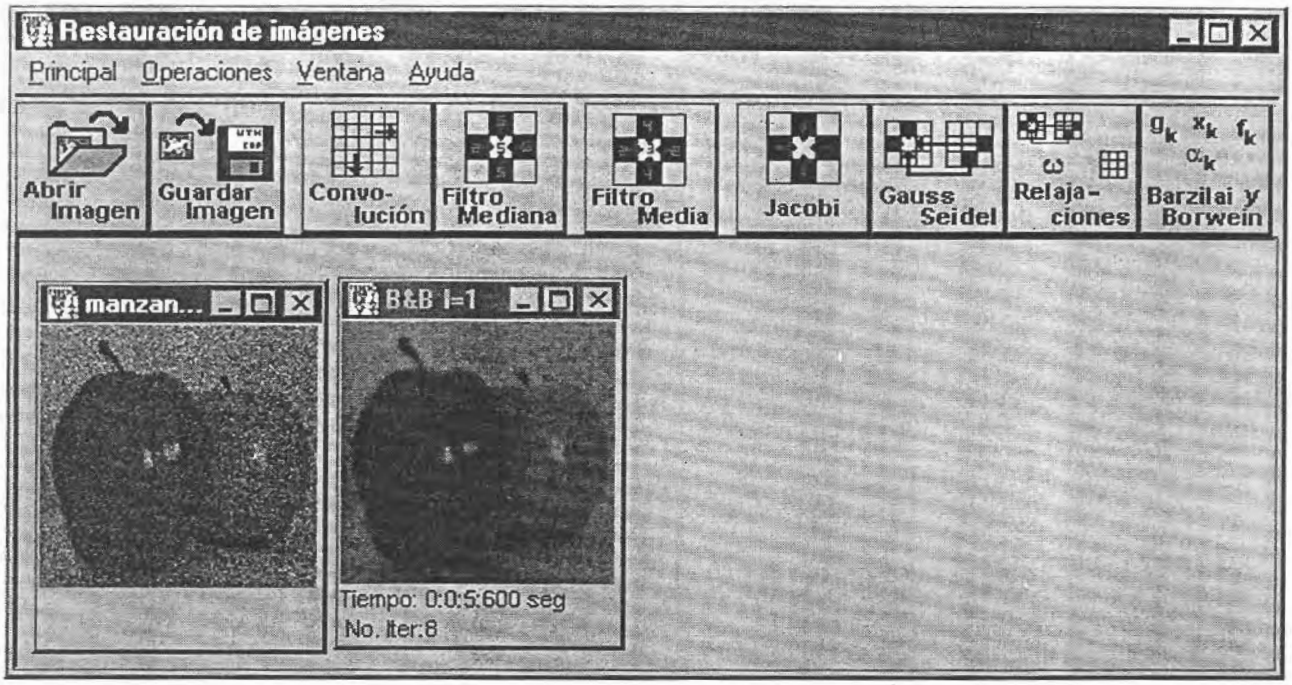


La imagen es mostrada en el programa, ahora se le desea aplicar el Método de Barzilai

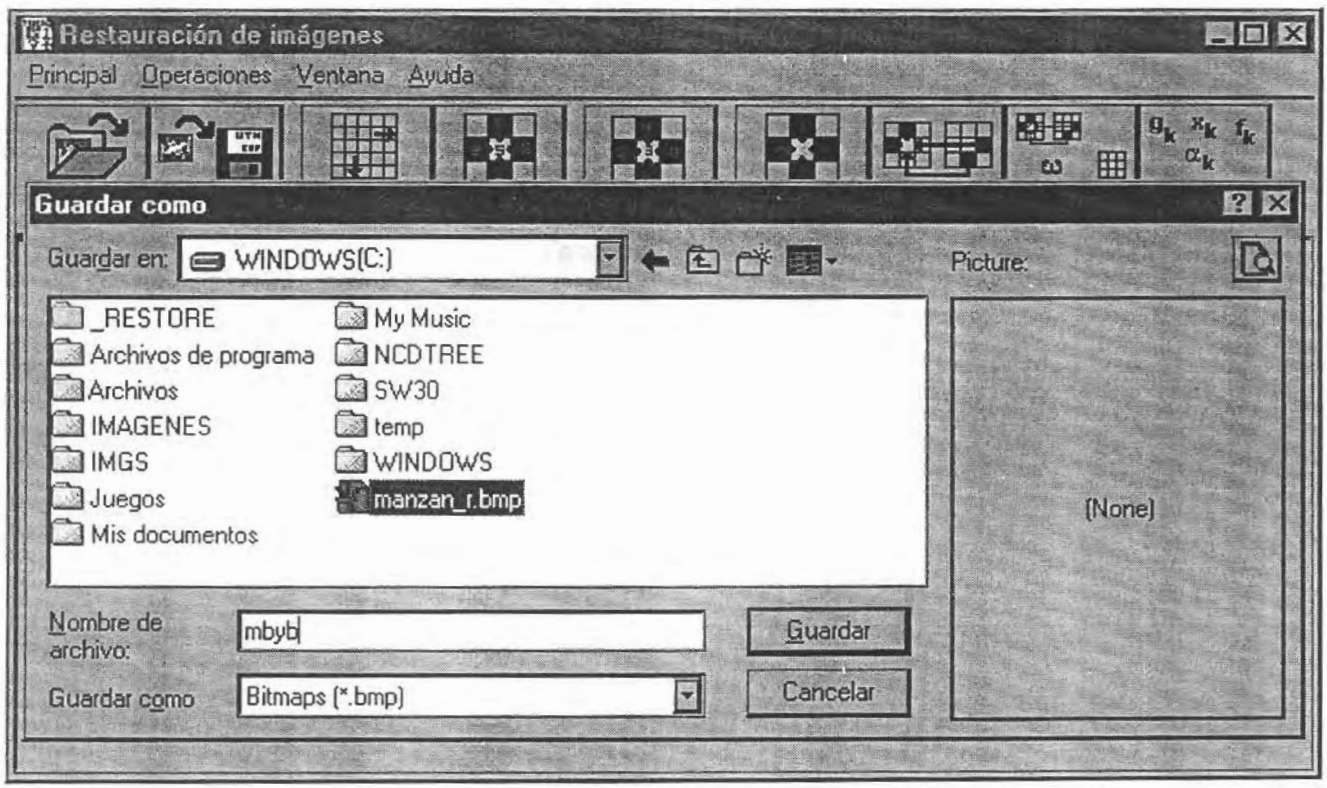
y Borwein, así que se presiona el botón que indica tal método. Lo primero que se va a observar es algo parecido a lo siguiente:

Cuando termine la restauración se verá:

Y uno puede arrastrar las imágenes para observar las dos al mismo tiempo.



Para guardar la imagen resultante en c:\ como mbyb se presiona F2 y se escribe el nombre, así como se ve en la siguiente figura:



Para hacer una restauración con cualquier otro método, basta con seleccionar la imagen y presionar el botón de la operación que se desee aplicar, es decir, tal como se explicó.

Bibliografía

[1] Heckerman David, Bayesian networks for data mining, www.cs.toronto.edu/~miller/csc2525/s01/papers/baysian.pdf.

[2] Barzilai J. and Borwein J. M., Two point step size gradient methods, IMA J. Numer. Anal., 8 (1988).

[3] Grippo L., Lampariello F. and Lucidi S., A nonmonotone line search technique for Newton's method, SIAM J. Numer. Anal., 23 (1986).

[4] Dennis J. E. Jr. and Schnabel R. B., Numerical methods for unconstrained optimization and Nonlinear Equations , Prentice-Hall, Englewood Cliffs, NJ, 1983.

[5] Fletcher R., Practical methods of optimization, John Wiley, New York, 1987.

[6] Fletcher R., Low storage methods for unconstrained optimization, in lectures in applied Mathematics, Vol. 26, American Mathematical Society, Providence, RI, 1990.

[7] Gilbert J. C. and Nocedal J., Global convergence propierties of conjugate gradient methods for optimization, SIAM J. Optim., 2 (1992).

[8] Glunt W, Hayden T. L. and Raydan M., Molecular conformations from distance matrices, J. Comput. Chem., 14 (1993).

[9] Grippo L., Lampariello F. and Lucidi S., A class of nonmonotone stabilization methods in unconstrained optimization, Numer. Math., 59 (1991).

[10] Nocedal J., Theory of algorithms for unconstrained optimization, Acta Numérica, 1 (1992).

[11] Pang J. S., Han S. P. and Rangaraj N., Minimization of locally lipschitzian functions, SIAM J. Optim., 1 (1991).

[12] Painer E.R. and Tits A. L., Avoiding the Maratos effect by means fo a nonmonotone line search I. General constrained problems, SIAM J. Numer. Anal., 28 (1991).

[13] S.Z. Li , Markov random field. Modeling in computer vision, Springer-Verlag 1995, http://www.research.microsoft.com/~szli/MRF_Book/MRF_Book.html.

[14] Raydan Marcos, The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem, SIAM J. Optimization, Vol. 7, No. 1, February 1997.

[15] Calculadora de imágenes Versión 0.1, CIMAT Departamento de Ciencias de la Computación, http://www.cimat.mx/proy_comp/caliman/caliman.html.

[16] Bouchaffra Djamel, Applied pattern recognition, Oakland University, www.oakland.edu/~bouchaff/PatternRecog/pdfslides/ch3part3.pdf.

[17] Morrell Darryl, EEE 598C: Statistical pattern recognition Lecture Note 5: Parameter Estimation September 24, 1996 www.eas.asu.edu/~morrell/598/lecture5.pdf.

[18] Skilling John, Probabilistic data analysis; An introductory guide January 1998, http://www.maxent.co.uk/documents/pda.pdf.

[19] Scales L.E., Introduction to non-linear optimization, 1984.

[20] Luthe, Olivera & Schutz, Métodos numéricos, Ed. Limusa, 1990.

Apéndice A

Conceptos Básicos

Vector gradiente.

El vector gradiente de una función $F(x)$ es:

$$g(x) = \begin{bmatrix} \dfrac{\partial F}{\partial x_1} \\ \vdots \\ \dfrac{\partial F}{\partial x_n} \end{bmatrix}. \tag{A.1}$$

Matriz Hessiana.

La matriz hessiana de una función $F(x)$ es:

$$G(x) = \begin{bmatrix} \dfrac{\partial^2 F}{\partial x_1^2} & \cdots & \dfrac{\partial^2 F}{\partial x_1 \partial x_n} \\ \vdots & & \vdots \\ \dfrac{\partial^2 F}{\partial x_n \partial x_1} & \cdots & \dfrac{\partial^2 F}{\partial x_n^2} \end{bmatrix}. \tag{A.2}$$

Ecuación algebraica lineal.

Es aquella en donde en cada término de la ecuación, aparece únicamente una variable

o incógnita elevada a la primera potencia. Por ejemplo,

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1, \tag{A.3}$$

es una ecuación algebraica lineal donde $x_1, x_2, ..., x_n$ son las variables y $a_{11}, a_{12}, ..., a_{1n}$ y b_1 constantes reales.

Sistema de ecuaciones.

Un sistema de ecuaciones es un conjunto de ecuaciones que deben resolverse simultáneamente. Se consideran sistemas de ecuaciones lineales de la siguiente forma:

$$\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + ... + a_{1n}x_n &= b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + ... + a_{2n}x_n &= b_2 \\
a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + ... + a_{3n}x_n &= b_3 \\
&..... \\
a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + ... + a_{nn}x_n &= b_n.
\end{aligned} \tag{A.4}$$

Por la definición de producto entre matrices, el sistema de n ecuaciones algebraicas lineales con n incógnitas puede escribirse en la forma matricial:

$$\begin{bmatrix} a_{11}\ a_{12}\ a_{13}a_{1n} \\ a_{21}\ a_{22}\ a_{23}a_{2n} \\ a_{31}\ a_{32}\ a_{33}a_{3n} \\ \\ \\ a_{n1}\ a_{n2}\ a_{n3}\ .\ .\ .\ a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ . \\ . \\ b_n \end{bmatrix}. \tag{A.5}$$

El sistema de ecuaciones (A.5) simbólicamente puede escribirse de la siguiente manera:

$$\mathbf{Ax} = \mathbf{b}, \tag{A.6}$$

en donde **A** se llama *matriz del sistema*. La matriz formada por **A**, a la que se le ha agregado el vector de términos independientes como última columna, se le llama *matriz ampliada del sistema*, que se representa con (A, b).

Campo aleatorio.

Sea $F = \{F_1, ..., F_m\}$, una familia de variables aleatorias definidas en S, donde cada variable aleatoria F_i toma valores f_i en L. Se llama *campo aleatorio* a la familia de variables aleatorias F. Se utilizará $F_i = f_i$ para denotar que el suceso F_i toma el valor f_i.