Tarea 5: Método de Gradiente Conjugado Precondicionado

Juan Gerardo Fuentes Almeida

Abstract—En esta práctica se implementa el algoritmo de Gradiente Conjugado con Precondicionamiento de Fourier en el problema de regularización de imágenes. Asimismo se hacen comparaciones en tiempo e iteraciones contra el método no precondicionado.

1 Introducción

Los métodos de Gradiente Conjugado se encuentran entre los más útiles para resolver sistemas grandes de ecuaciones, no requieren almacenamiento de matrices y nos permiten resolver problemas de optimización cuadrática.

La forma lineal del método de Gradiente Conjugado consiste en un método iterativo para resolver sistemas lineales con matrices definidas positivas. Su desempeño esta determinado por la distribución de los eigenvalores de la matriz, esta distribución se puede trasformar o pre-condicionar para mejorar significativamente la convergencia del método.

2 Teoría

2.1 Precondicionamiento

Es posible acelerar la convergencia del método de gradiente conjugado transformando el sistema lineal para mejorar la distribución de los eigenvalores de la matriz A. Un sistema de ecuaciones $Ax = b$ es considerado "bien condicionado" si un pequeño cambio en b resulta en un pequeño cambio en x, si un pequeño cambio en b resulta en un cambio grande en x, decimos que el sistema esta mal condicionado. Esta caracteristica esta definida por el numero de condición $\eta = \frac{\lambda_n}{\lambda_1}$, donde λ_n y λ_1 son los eigenvalores máximo y mínimo de A, respectivamente. Al reducir el número de condición de un sistema, se puede acelerar la velocidad de convergencia del gradiente conjugado.

Entonces, en lugar de resolver el sistema

$\mathbf{Ax} - \mathbf{b} = 0$
Resolvemos el sistema

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

con $\mathbf{M}^{-1}$ una matriz cuadrada simétrica y definida positiva, que recibe el nombre de precondicionador

El mejor precondicionador seria claramente $\mathbf{M}^{-1} = \mathbf{A}$, así $\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}$, y el gradiente conjugado convergería en un solo paso.

Algoritmo 1: Gradiente Conjugado Precondicionado

Dado x_0, precondicionador M;
$r_0 \leftarrow Ax_0 - b$;
Resolver $My_0 = r_0$ para y_0;
$p_0 = -y_0, k \leftarrow 0$;
while $r_k \neq 0$

$$\alpha_k \leftarrow \frac{r_k^T y_k}{p_k^T A p_k};$$
$$x_{k+1} \leftarrow x_k + \alpha_k p_k;$$
$$r_{k+1} \leftarrow r_k + \alpha_k A p_k;$$
Resolver $My_{k+1} = r_{k+1}$;
$$\beta_{k+1} \leftarrow \frac{r_{k+1}^T y_{k+1}}{r_k^T y_k};$$
$$p_{k+1} \leftarrow -y_{k+1} + \beta_{k+1} p_k;$$
$$k \leftarrow k + 1;$$

end (while)

2.2 Transformada de Fourier

Para esta práctica se implmentó un precondicionador basado en la transformada de Fourier del sistema, utilizando la transformada discreta de Fourier en 2 dimensiones definida de la siguiente manera para una imagen de M X N:

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-2\pi i (\frac{ux}{M} + \frac{vy}{N})}$$

y la transformada inversa de Fourier:

$$f(x,y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{-2\pi i (\frac{ux}{M} + \frac{vy}{N})}$$

3 Implementación

Se implementan versiones de Gradiente Conjugado Lineal sin precondicionamiento y con precondicionamiento de Fourier. Con estas implementaciones se resuelve el

problema de suavizamiento o "denoising" mediante la minimización de la función de regularización definida por la siguiente expresión:

$$U(I_1) = \frac{1}{2}\sum_{x\in\Omega}(I_2(x) - I_1(x))^2 + \frac{\lambda}{2}\sum_{l\in N_x}\rho(I_1(x) - I_1(l))$$

donde N_x es la vecindad del pixel $I_1(x)$ de 4 vecinos, $\rho(z) = z^2$ es la función de regularización con $z = I_1(x) - I_1(l)$, $I_1(x)$ es la imagen suavizada, $I_2(x)$ es la imagen observada con ruido, $x = (i, j)$ son las coordenadas de un pixel, Ω es el conjunto de pixeles en común a ambas imágenes y λ es el parámetro de de regularización o penalización.

Para maximizar la función definimos la derivada para cierto elemento x_{ij}:

$$\frac{\partial}{\partial x_{ij}}U(I_1) = -[I_2(x_{ij}) - I_1(x_{ij})] + \lambda\sum_{l\in N_x}[I_1(x_{ij}) - I_1(l)]$$

Así generamos un sistema de ecuaciones para todo el dominio Ω:

$$I_1(x_{ij})[1 + \lambda|N_x|] - \lambda\sum_{l\in N_x} I_1(l) = I_2(x_{ij})$$

donde $|N_x|$ representa el número de vecinos del pixel; en esta implementación estamos considerando un dominio continuo sobre la imagen, es decir, el vecino de un pixel en los extremos de la imagen, será el pixel del lado opuesto. Por tanto, cada pixel tendrá exactamente 4 vecinos.

$$\begin{bmatrix} 1+4\lambda & -\lambda & 0 & \cdots & 0 \\ -\lambda & \ddots & -\lambda & \ddots & \vdots \\ 0 & -\lambda & 1+4\lambda & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -\lambda \\ 0 & \cdots & 0 & -\lambda & 1+4\lambda \end{bmatrix} * \begin{bmatrix} I_x(1,1) \\ \vdots \\ I_x(i,j) \\ \vdots \\ I_x(N,M) \end{bmatrix} = \begin{bmatrix} I_y(1,1) \\ \vdots \\ I_y(i,j) \\ \vdots \\ I_y(N,M) \end{bmatrix}$$

En este punto implementamos la transformada de Fourier como precondicionador en cada iteración del algoritmo de gradiente conjugado. Para cada renglón del sistema de ecuaciones tenemos que:

$$\boldsymbol{DFT}[(1 + 4\lambda)I_1(x_{ij}) - \lambda(I_1(x_{i-1,j}) + I_1(x_{i+1,j}) + I_1(x_{i,j-1}) + I_1(x_{i,j+1})) = I_2(x_{ij})]$$

$$\Rightarrow (1+4\lambda)F_{uv} - \lambda F_{uv}(e^{\frac{2\pi ui}{n}} + e^{\frac{-2\pi ui}{n}} + e^{\frac{2\pi vi}{n}} + e^{\frac{-2\pi vi}{n}}) = G_{uv}$$

y aplicamos la fórmula de Euler:

$$e^{i\theta} + e^{-i\theta} = 2\cos\theta$$

$$\Rightarrow (1 + 4\lambda)F_{uv} - \lambda F_{uv}(2\cos(\tfrac{2\pi ui}{n}) + 2\cos(\tfrac{2\pi vi}{n})) = G_{uv}$$

de esta manera obtenemos una expresión que nos permite resolver el sistema $F_{uv} = H_{uv}G_{uv}$ para F_{uv}:

$$F_{uv} = \frac{1}{1+4\lambda-2\lambda(cos(\frac{2\pi ui}{n})+\cos(\frac{2\pi vi}{n}))}G_{uv}$$

Esta fórmula es la que aplicaremos en cada paso de precondicionamiento, es decir, transformaremos el vector g_{xy} a G_{uv} en el dominio de Fourier y después de resolver para F_{uv}, transformaremos a f_{xy} en el espacio cartesiano.

4 Resultados

A continuación se muestran las imágenes que se obtuvieron durante la ejecución de la práctica, para diferentes valores de λ en una imagen de prueba:

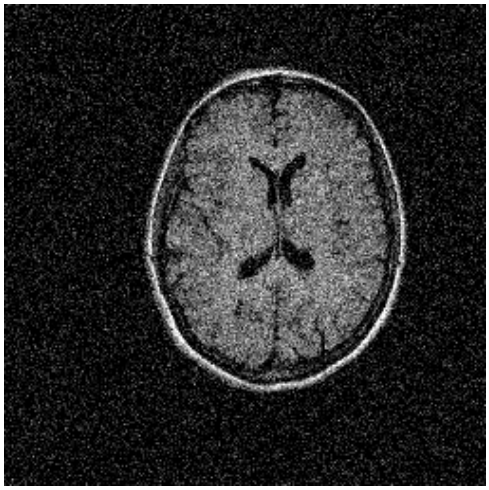
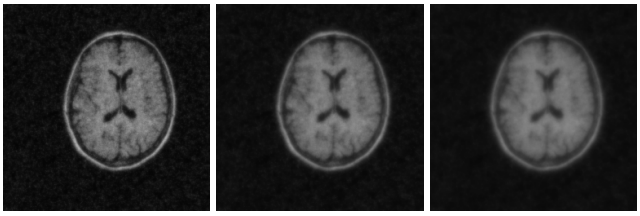
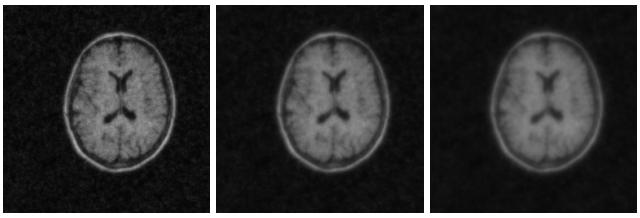


Imagen observada



De derecha a izquierda: Resultados de Gradiente Conjugado no precondicionado para $\lambda = 2, 5, 10$, respectivamente.



De derecha a izquierda: Resultados de Gradiente Conjugado precondicionado con Transformada de Fourier para $\lambda = 2, 5, 10$, respectivamente.

El programa implementado en esta práctica recibe el nombre del archivo de la imagen, el parámetro λ y un indicador de la función a realizar, luego ejecuta los algoritmos de Gradiente Conjugado para suavizar la imagen con este parámetro.

Se imprime en pantalla el valor de λ, el tiempo de ejecución del algoritmo y el valor final de la función objetivo.

Se muestra también una tabla donde se comparan ambos métodos con relación a su tiempo de ejecución, número de iteraciones y valor de la función objetivo:

	Iteraciones		Tiempo		Error	
Lambda	GC	GCP	GC	GCP	GC	GCP
1	34	1	0.562154	0.120807	29622323.46672	29622323.46672
5	76	1	1.21729	0.096197	37532150.06834	37532150.06834
10	107	1	1.702475	0.09365	40132563.91056	40132563.91056
15	131	1	2.079894	0.093617	41511818.11594	41511818.11594
20	152	1	2.40743	0.092614	42474610.24453	42474610.24453
25	170	1	2.69013	0.092973	43236987.9716	43236987.9716
30	185	1	2.935297	0.092975	43883717.96095	43883717.96095

Tabla comparativa entre los algoritmo de Gradiente Conjugado implementados.

5 Conclusiones

Se observa por la tabla mostrada anteriormente que en general el algoritmo de Gradiente Conjugado con precondicionamiento de Fourier resulta ser más eficiente, ya que siempre converge a una iteración puesto que estamos prácticamente resolviendo todo el sistema en el dominio de Fourier. Por otro lado en cuestión del valor de la función objetivo ambos arrojan resultados similares, lo cual se puede confirmar empíricamente al comparar las imágenes, ya que casi no muestran diferencias apreciables para una misma λ.

References

[1] Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. Springer series in operations research and financial engineering. Springer, New York, NY, 2. ed. edition, 2006.

Appendix

El programa está implementado tomando en cuenta todas estandarizaciones indicadas en el curso.

Un *makefile* ha sido generado, el cual soporta los comandos *make*, *run* and *clean*. El programa recibe como primer argumento el nombre del archivo de la imagen inicial, como segundo argumento el nombre del archivo de la imagen de referencia y por último el indicador del método, 1 indica que se ejecutara Gradiente Conjugado sin precondicionamiento y 2 indica que se ejecutara con precondicionamiento.