Métodos Numéricos
Segundo cuatrimestre 2015
Taller 1
Image denoising:
Eliminando ruido en imágenes

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Introducción

Un problema típico que se encuentra al trabajar con imágenes digitales es la existencia de "ruido" en las mismas. En pocas palabras, podemos decir que el ruido ocurre cuando el valor de uno o más píxeles de la imagen, no se corresponden con la realidad. La mayoría de las veces, esto se debe a la calidad del equipo electrónico utilizado para tomar las fotografías, o bien a posibles perturbaciones introducidas al momento de transmitir la información. Un caso muy común de imágenes con ruido son las fotografías satelitales.

Una forma de corregir (o reducir) este fenómeno en las imágenes es mediante la aplicación de filtros, con el objetivo de suavizar las mismas para obtener resultados más cercanos a la realidad. Hoy en día, existen muchas técnicas de filtrado de imágenes, y muchas de ellas están basadas en modelos matemáticos que en general se resuelven mediante métodos numéricos.

Se puede pensar el problema de filtrar una imagen con ruido como la minimización del siguiente funcional:

$$\Pi = \int_{\Omega} \frac{\lambda}{2} |u - \tilde{u}|^2 + \frac{1}{2} \|\nabla u\|^2 d\Omega, \tag{1}$$

donde $u : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}$ describe la imagen filtrada y $\tilde{u} : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}$ la imagen a filtrar (con ruido). De esta manera, el primer término *pesa* cuanto ruido tiene $\tilde{u}$ y el segundo *pesa* la suavidad de la imagen obtenida. La constante λ controla la importancia relativa de los dos términos.

La minimización del funcional de la ecuación (1) da lugar a la siguiente ecuación diferencial:

$$\lambda (u - \tilde{u}) - \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) = 0. \tag{2}$$

La solución de la ecuación (2) que representa la imagen filtrada se puede aproximar de manera discreta utilizando el método de diferencias finitas, lo cual conduce al siguiente sistema de ecuaciones:

$$\lambda u_{i,j} - (u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{i,j}) = \lambda \tilde{u}_{i,j} \tag{3}$$

donde ahora $u, \tilde{u} : \Omega \subset \mathbb{Z}^2 \rightarrow [0 \ldots 255]$ son la versiones discretas de la imagen filtrada y la imagen original, respectivamente. Viendo la imagen u como una matriz, i, j son los índices de fila y columna de cada elemento (píxel) de la matriz, donde el 0 es representado por el color negro y el 255 por el blanco[1].

Enunciado

El objetivo principal de este taller es implementar un programa para eliminar (o reducir) el ruido en imágenes digitales. Para ello, el programa deberá tomar como entrada una imagen (supuestamente con ruido) y resolver la ecuación (2) por el método de diferencias finitas (resolviendo el sistema de ecuaciones dado por las ecuaciones (3)). Finalmente, el programa deberá devolver la versión filtrada de la imagen.

Ejercicios:

1. Repasar la definición de factorización LU y matriz Simétrica Definida Positiva (SDP). ¿Bajo qué condiciones podemos garantizar que una matriz tiene factorización LU?

[1] Este modelo de filtrado de imágenes se puede extender a imágenes color RGB, repitiendo el proceso descripto para cada componente de color.

2. ¿Es cierto que si una matriz es inversible entonces tiene factorización LU?. Y si tengo una matriz que tiene factorización LU, ¿entonces es no singular? Demostrar o dar un contraejemplo.

3. Sea $A \in \mathbb{R}^{n \times n}$. Determinar *Verdadero* o *Falso* y, según corresponda, demostrar o dar un contraejemplo:

   - AA^t es una matriz simétrica.
   - Si A es no singular, entonces $A^t A$ es SDP.

4. Rellenar el archivo CheckCondLU.m para verificar que una matriz tiene factorización LU, usando para esto el ejercicio 7 de la práctica 2. De forma análoga, rellenar el archivo CheckCondChol.m para verificar que una matriz es SDP (sugerencia: utilizar la funcion `det` de Matlab). Evaluar ambas rutinas, por ejemplo de la siguiente forma:

```
>> A = rand(5,5)
>> CheckCondLU(A)
>> CheckCondChol(A'*A)
```

   Considerar distintas matrices, por ejemplo usando también la rutina `hilb`, y variar el tamaño de las mismas.

5. Rellenar el archivo CholFromLU.m para que tome una matriz A y devuelva la matriz L correspondiente a su factorización de Cholesky. En este caso se comienza teniendo la factorización LU de A y se quiere generar la factorización de Cholesky. Se debe incluir código para chequear la factorización con respecto al comando *chol* de Matlab.

6. Rellenar el archivo CholFromBlocks.m para que tome una matriz A y devuelva la matriz L correspondiente a su factorización de Cholesky. En este caso se comienza teniendo la factorización de Cholesky de la submatriz principal de A que omite la última fila y la última columna, y se quiere generar la factorización de Cholesky. Se debe incluir código para chequear la factorización con respecto al comando *chol* de Matlab.

Evaluación:

- Coloquio con los docentes durante la clase
- En caso de no asistir a clase, se debe entregar la resolución del taller (código + mini-informe) hasta el Viernes 11 de Septiembre.