Práctica 4

Introducción

Con la realización de esta práctica se espera que el estudiante adquiera las competencias necesarias para resolver sistemas de ecuaciones lineales mediante la aplicación de métodos directos e iterativos, de forma manual y con el apoyo de funciones (subrutinas o programas) propias o predefinidas utilizando los entornos de software MATLAB o Scilab. Asimismo, que aprenda a diferenciar los métodos directos e iterativos y las dificultades que surgen cuando se utilizan para resolver este tipo de sistemas mediante el uso de la computadora.

La primera sección está planificada de modo que el estudiante diseñe sus propios programas a partir de los algoritmos vistos en clase o disponibles en la bibliografía. Más aún, se insta al estudiante a que trabaje de esta forma, pues ello le proporcionará una visión integral de los métodos. Es decir, al programarlos aprenderá a utilizar de manera eficiente las fórmulas vistas.

En la segunda sección se proponen algunos ejercicios tomados de la bibliografía recomendada. Esto tiene un doble propósito. En primer lugar que el estudiante aprenda a identificar los principales elementos del problema, a saber: (i) la matriz de coeficientes (A), (ii) el vector de términos independientes ($\mathbf{b}$), y (iii) la solución del problema $\mathbf{x}$. En segundo lugar, la codificación del programa y su ejecución le servirá para reconocer e interiorizar cada parte del mismo, depurar cualquier error y tenerlo optimizado para el día del cuestionario en línea.

Finalmente, en las secciones 3 y 4 se presentan las normas y estimaciones del error, así como la sintaxis de las funciones propias de MATLAB: `mldivide (\), chol, lu, det` y `cond`. Algunas de estas funciones simplifican de manera considerable el diseño de los algoritmos propuestos mientras que otras resuelven ciertos problemas considerados en esta unidad. Por otro lado, las normas son necesarias para determinar los errores y entender las dificultades que pueden surgir durante la resolución numérica de sistemas de ecuaciones lineales.

1 Diseño de algoritmos

Dada la matriz de coeficientes (A) y el vector de términos independientes ($\mathbf{b}$) del sistema lineal $A\mathbf{x} = \mathbf{b}$.

a) **Codifique el algoritmo de Crout para matrices tridiagonales**. Es decir, escriba una función de nombre **tridecrout.m** que descomponga una matriz tridiagonal $A \in \mathbb{R}^{n\times n}$ en la forma LU mediante el algoritmo de Crout. La función debe tener como argumento de entrada a la matriz A, y como argumentos de salida las matrices L y U.

b) **Resolución del sistema lineal tridiagonal**. Escribir una función de nombre **trisolve.m** para resolver el sistema de ecuaciones tridiagonal considerado mediante la ejecución de los dos pasos: $L\mathbf{y} = \mathbf{b}$ y $U\mathbf{x} = \mathbf{y}$, donde L y U son las matrices que resultan del algoritmo de descomposición de Crout (`tridecrout.m`). Los argumentos de entrada de la función deben ser: (i) las matrices L y U de la descomposición de Crout, y (ii) el vector de términos independientes $\mathbf{b}$. El argumento de salida debe ser la solución del sistema, $\mathbf{x}$.

c) **Método de Jacobi**. Escribir una función de nombre **Jacobi.m** para resolver un sistema de ecuaciones $A\mathbf{x} = \mathbf{b}$ utilizando el método iterativo de Jacobi. La función deberá tener como argumentos de entrada: (i) la matriz A, (ii) el vector de términos independientes $\mathbf{b}$, (iii) el vector de aproximación inicial $\mathbf{x}^{(0)}$, (iv) el valor de la tolerancia deseada tol, y (v) el número de iteraciones máximo $itmax$. Como argumento de salida deberá devolver la solución aproximada $\mathbf{x}$.

d) **Método de Gauss Seidel y SOR**. Escribir una función de nombre **SOR.m** para resolver un sistema de ecuaciones $A\mathbf{x} = \mathbf{b}$ utilizando el método iterativo SOR. Utilice como argumentos de entrada A, $\mathbf{b}$, $\mathbf{x}^{(0)}$, el valor de la tolerancia deseada tol, el parámetro ω y el número de iteraciones máximo $itmax$, y como argumento de salida la solución del sistema $\mathbf{x}$.

Nota. Recuerde que la ecuación para el cálculo de la i-ésima componente del vector $\mathbf{x}^{(k)}$ para el método SOR es:

$$x_i^{(k)} = (1-\omega)x_i^{(k-1)} + \frac{\omega}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}\right]$$

Por lo tanto, cuando $\omega = 1$ se obtiene el método de Gauss-Seidel. Es decir, con esta subrutina tendremos las soluciones de los métodos de Gauss-Seidel y SOR.

Observación. Es necesario que definan las funciones según las instrucciones dadas, pues al evaluar los scripts correspondientes a la asignación 2 si las funciones no se pueden invocar del modo predefinido se considerará no realizada esta pauta de la asignación acarreando una penalización sobre la nota.

2 Solución de problemas

a) Dado el sistema de ecuaciones:

$$\begin{aligned} 0.5x_1 - \quad x_2 &= -9.5 \\ 0.26x_1 - 0.5x_2 &= \text{-4.7} \end{aligned}$$

(i) Utilice el método de eliminación Gaussiana para determinar la solución exacta $\mathbf{x}$ del sistema. (ii) Utilice eliminación Gaussiana para calcular la solución $\hat{\mathbf{x}}$ del problema perturbado:

$$\begin{aligned} 0.55x_1 - \quad x_2 &= -9.5 \\ 0.26x_1 - 0.5x_2 &= \text{-4.7} \end{aligned}$$

(iii) Explique por qué una pequeña perturbación en la matriz de coeficientes del sistema genera un gran cambio en la solución. (**Ver material sobre normas y estimación del error, y álgebra lineal numérica y MATLAB agregado en la última sección de esta práctica**).

b) **Use las subrutinas tridecrout y trisolve** para resolver el sistema de ecuaciones:

$$\begin{aligned} 10x_1 + 5x_2 &= 6 \\ 5x_1 + 10x_2 - 4x_3 &= 25 \\ -4x_2 + 8x_3 - x_4 &= -11 \\ -x_3 + 5x_4 &= -11 \end{aligned}$$

c) Use la subrutina Jacobi para resolver el SEL:

$$\begin{aligned} 4x_1 + x_2 - x_3 &= -2 \\ x_1 + 4x_2 - x_3 - x_4 &= -1 \\ -x_1 - x_2 + 5x_3 + x_4 &= 0 \\ x_1 - x_2 + x_3 + 3x_4 &= 1 \end{aligned}$$

con $tol = 10^{-3}$ en la norma ℓ_∞ y aproximación inicial $\mathbf{x}^{(0)} = \mathbf{0}$.

d) De los tres conjuntos siguientes de ecuaciones lineales, identifique aquel(los) que no podrá resolver con el uso de un método iterativo tal como el de Gauss-Seidel. Demuestre que su solución no converge utilizando cualquier número de iteraciones que sea necesario. Enuncie con claridad su criterio de convergencia (es decir, ¿cómo sabe que no está convergiendo?).

Conjunto uno	**Conjunto dos**	**Conjunto tres**
$9x+3y+z = 13$	$x+y+6z = 8$	$-3x+4y+5z = 6$
$-6x+8z = 2$	$x+5y-z = 5$	$-2x+2y-3z = -3$
$2x+5y-z = 6$	$4x+2y-2z = 4$	$2y-z = 1$

e) Emplee la subrutina SOR (con $\omega = 1.2$) para resolver el SEL:

$$\begin{aligned} 4x_1 + x_2 + x_3 + x_5 &= 6 \\ -x_1 - 3x_2 + x_3 + x_4 &= 6 \\ 2x_1 + x_2 + 5x_3 - x_4 - x_5 &= 6 \\ -x_1 - x_2 - x_3 + 4x_4 &= 6 \\ 2x_2 - x_3 + x_4 + 4x_5 &= 6 \end{aligned}$$

con una tolerancia de 10^{-3} en la norma ℓ_∞ y aproximación inicial $\mathbf{x}^{(0)} = \mathbf{0}$.

f) Dado el SEL:

$$\begin{aligned} 4x_1 + x_2 + x_3 + x_4 &= 0.65 \\ x_1 + 3x_2 - x_3 + x_4 &= 0.05 \\ x_1 - x_2 + 2x_3 &= 0 \\ -x_1 + x_2 + 2x_4 &= 0.5 \end{aligned}$$

(i) Verifique que la matriz de coeficientes A es definida positiva. (ii) Utilice el comando de MATLAB: `L= chol(A, 'lower')` para hallar la matriz L de la descomposición de Cholesky. (iii) Resuelva el SEL utilizando el esquema de 2 fases: $L\mathbf{y} = \mathbf{b}$ y $L^T\mathbf{x} = \mathbf{y}$.

3 Normas y Estimaciones del Error

Definición 1 *Una norma vectorial en* $\mathbb{R}^n$ *es una función* $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ *que satisface las siguientes propiedades:*

(i) $\|\mathbf{x}\| \geq 0$ *para todo* $\mathbf{x} \in \mathbb{R}^n$.

(ii) $\|\mathbf{x}\| = 0$ *si y solo si* $\mathbf{x} = [0 \;\; 0 \;\; \ldots \;\; 0]^T \equiv \mathbf{0}$.

(iii) $\|\alpha\mathbf{x}\| = |\alpha|\|\mathbf{x}\|$ *para todo* $\alpha \in \mathbb{R}$ *y* $\mathbf{x} \in \mathbb{R}^n$.

(iv) $\|\mathbf{x}+\mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ *para todo* $\mathbf{x},\ \mathbf{y} \in \mathbb{R}^n$.

Definición 2 *Una norma matricial sobre el conjunto de todas las matrices* $M_{n\times n}$ *es una función* $\|\cdot\| : M_{n\times n} \to \mathbb{R}$ *que satisface las siguientes propiedades:*

(i) $\|A\| \geq 0$

(ii) $\|A\| = 0$ *si y solo si* $A = 0_{n\times n}$, *la matriz con todas las entradas cero o matriz nula.*

(iii) $\|\alpha A\| = |\alpha|\|A\|$.

(iv) $\|AB\| \leq \|A\|\|B\|$.

En esencia, una norma proporciona una medida del tamaño o longitud de entidades matemáticas como los vectores y las matrices. Por ejemplo, la distancia entre las matrices A y B respecto a la norma matricial ℓ_1 es $\|A - B\|_1$, donde la norma ℓ_1, junto a otras normas vectoriales y matriciales clásicas, se define en la Tabla 1 y se presenta, además, la sintaxis utilizada en MATLAB para calcular dichas normas.

Cuadro 1: Normas Vectoriales y Matriciales

Norma	ℓ_1	ℓ_2	ℓ_∞
Vectorial	$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} \|x_i\|$	$\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$	$\|\mathbf{x}\|_\infty = \max_{1\leq i\leq n} \|x_i\|$
Matricial	$\|A\|_1 = \max_{1\leq j\leq n} \sum_{i=1}^{n} \|a_{ij}\|$	$\|A\|_2 = \sqrt{\lambda_{max}}$ [a]	$\|A\|_\infty = \max_{1\leq i\leq n} \sum_{j=1}^{n} \|a_{ij}\|$
Sintaxis MATLAB	`norm(x,1)` `norm(A,1)`	`norm(x)` `norm(A)`	`norm(x,inf)` `norm(A,inf)`

[a] $\lambda_{max} = \max\{|\lambda_k \mid \lambda_k$ es un autovalor de $A^T A|\}$.

Definición 3 *Dada la matriz $A \in M_{n\times n}$, se define el número de condición de una matriz no singular A respecto de la norma $\|\cdot\|$ mediante:*

$$K(A) = \|A\|\|A^{-1}\|$$

Observaciones.

1. Para toda matriz no singular A y toda norma natural $\|\cdot\|$ se verifica,

$$1 = \|I\| = \|AA^{-1}\| \leq \|A\|\|A^{-1}\| = K(A)$$

2. Los errores de aproximación absoluto y relativo vienen dados por:

$$\|\mathbf{x} - \tilde{\mathbf{x}}\| \leq K(A)\frac{\|\mathbf{r}\|}{\|A\|}, \qquad \frac{\|\mathbf{x} - \tilde{\mathbf{x}}\|}{\|\mathbf{x}\|} \leq K(A)\frac{\|\mathbf{r}\|}{\|\mathbf{b}\|}$$

   donde $\mathbf{x}$ denota la solución exacta, $\tilde{\mathbf{x}}$ la solución aproximada, y $\mathbf{r}$ es el vector residual dado por $\mathbf{r} = A\tilde{\mathbf{x}} - \mathbf{b}$.

3. Una matriz A está bien condicionada si $K(A)$ se aproxima a 1 y está mal condicionada si $K(A)$ es relativamente mayor que 1. Dentro de este contexto, el término condición se refiere a la seguridad relativa de que un vector residual pequeño ($\mathbf{r} \approx \mathbf{0}$) implica que el error de aproximación es pequeño (es decir, $\tilde{\mathbf{x}} \approx \mathbf{x}$).

4 Álgebra Lineal Numérica y MATLAB

A continuación algunas funciones de MATLAB que permiten aproximar la solución de problemas del álgebra lineal.

1. **Solución del sistema** $A\mathbf{x} = \mathbf{b}$. Dado el sistema de ecuaciones lineales $A\mathbf{x} = \mathbf{b}$, su solución se obtiene mediante el comando: `x=A\b`.

2. **Condición de una matriz**. Dada la matriz A, la condición de A se obtiene mediante el comando: `cond(A)`.

3. **Descomposición LU**. Dada la matriz A, la descomposición LU de A se obtiene con el comando: `[L,U]=lu(A)`.

4. **Descomposición de Cholesky**.Si A es una matriz definida positiva, el algoritmo de Cholesky permite descomponer la matriz A como el producto LL^T, con L una matriz triangular inferior. El comando de MATLAB que permite llevar a cabo tal descomposición es: `L=chol(A)`. Si A no es definida positiva el algoritmo da un mensaje de error.

5. **Autovalores**. Dada la matriz A, los autovalores de A se obtienen mediante el comando: `eig(A)`.

6. **Radio espectral**. El radio espectral $\rho(A)$ de la matriz A se obtiene mediante el comando: `rho=max(abs(eig(A)))`.

7. **Extraer la diagonal de una matriz**. La diagonal de una matriz A se obtiene mediante el comando: `D=diag(diag(A))`

8. **Extraer la parte triangular superior de una matriz**. La parte estrictamente triangular superior de una matriz A se obtiene mediante el comando: `U=triu(A,1)`.

9. **Extraer la parte triangular inferior de una matriz**. La parte estrictamente triangular inferior de una matriz A se obtiene mediante el comando: `L=tril(A,-1)`