Práctico 2 - Sistemas de ecuaciones lineales

Los ejercicios 10 y 18 son los “entregables” de este práctico. Todo estudiante inscripto en el curso va a tener asignado un único ejercicio del práctico 1, 2, o 3 para entregar antes del comienzo del primer período de parciales. **El 20 de setiembre vamos a avisar qué ejercicio le corresponde a cada estudiante, por lo que recomendamos fuertemente haber terminado y tener escrito los ejercicios 10 y 18 para esa fecha.**

Métodos directos

Ejercicio 1 (Un sistema esencialmente triangular inferior)**.** Explicar cómo resolver de forma eficiente un sistema lineal de la forma

$$\begin{bmatrix} \mathcal{O} & L_1 \\ L_2 & B \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{c} \end{bmatrix}$$

donde L_1 y L_2 son matrices triangulares inferiores y no singulares, $\mathcal{O}$ es la matriz nula, B es una matriz arbitraria, y los vectores están particionados acordemente. Describir los pasos necesarios en términos de las submatrices y vectores dados.

Ejercicio 2 (Sustitución hacia atrás)**.** Escribir una función `x = atras(U,b)` que tome como entradas una matriz triangular superior U y un vector columna $\mathbf{b}$, y resuelva el sistema $U\mathbf{x} = \mathbf{b}$ mediante sustitución hacia atrás.

Ejercicio 3 (Cómputo de determinantes)**.** La factorización $PA = LU$ se puede utilizar para computar el determinante de A. Tenemos $\det(L)\det(U) = \det(P)\det(A)$. Como L es triangular y tiene unos en la diagonal, $\det(L) = 1$. Al ser U triangular, $\det(U) = u_{11}u_{22}\ldots u_{nn}$. Como P es de permutaciones, $\det(P) = +1$ si la cantidad de intercambios es par y $\det(P) = -1$ si es impar. Por lo tanto,

$$\det(A) = \pm u_{11}u_{22}\ldots u_{nn}.$$

Modificar la función `lutx` de modo que retorne cuatro variables.

```
function [L,U,p,sig] = lutx_modificada(A)
% LU Triangular factorization
% [L,U,p,sig] = lutx_modificada(A) computa una matriz triangular inferior L,
% una matriz triangular superior U, un vector de permutaciones p y
% un escalar sig, de forma que L*U = A(p,:) y sig = +1 o -1 si p
% es una permutacion par o impar.
```

Escribir una función `determinante(A)` que use la función `lutx_modificada` para calcular el determinante de A. El producto $u_{11}u_{22}\ldots u_{nn}$ se puede calcular usando la expresión `prod(diag(U))`.

Ejercicio 4 (Cómputo de inversas)**.** La inversa de una matriz A se puede definir como la matriz X cuyas columnas $\mathbf{x}_j$ resuelven las ecuaciones

$$A\mathbf{x}_j = \mathbf{e}_j,$$

donde $\mathbf{e}_j$ es la j-ésima columna de la matriz identidad.

a) Tomando como punto de partida la función `bslashtx`, escribir una función `X = inversa(A)` que compute la inversa de A. Dicha función debe llamar a `lutx` solamente una vez y no debe usar ni las funciones `inv` ni \ (backslash).

b) Comparar los resultados obtenidos con esta función con las inversas obtenidas utilizando la función `inv(A)` en algunas matrices.

Ejercicio 5 (Muchos sistemas)**.** Para una matriz $A \in \mathcal{M}_n(\mathbb{R})$ con $n = 100$, generar aleatoriamente 10 vectores $\mathbf{b}$ diferentes y resolver los 10 sistemas correspondientes de dos formas distintas:

- usando la función `bslashtx` para cada sistema;
- usando la función `lutx` una vez y sustitución hacia adelante y atrás para cada sistema.

Comparar el trabajo total realizado. Las funciones `tic` y `toc` pueden ser útiles para este fin.

Ejercicio 6 (Fórmula de actualización de la inversa)**.** Dada una matriz $A \in \mathcal{M}_n(\mathbb{R})$ invertible, considerar la matriz $B = A - \mathbf{u}\mathbf{v}^t$, donde $\mathbf{u}$ y $\mathbf{v}$ son columnas de n elementos.

a) Probar que si B es invertible entonces su inversa es

$$B^{-1} = A^{-1} + \alpha(A^{-1}\mathbf{u})(\mathbf{v}^t A^{-1})$$

para un escalar adecuado α que se determinará.
[Sugerencia: $\mathbf{v}^t A^{-1}\mathbf{u}$ es un número distinto de 1.]

b) Aplicar la fórmula anterior para corregir la inversa de A cuando se efectúa un cambio en una de sus columnas. Para ello seguir estos pasos:

- Definir una matriz A de 6×6 y hallar su inversa con el comando `inv`.
- Definir una matriz B igual a A excepto en su columna 4, que será de unos.
- Elegir vectores columna $\mathbf{u}$ y $\mathbf{v}$ de forma que $B = A - \mathbf{u}\mathbf{v}^t$.
- Usar la fórmula anterior para hallar B^{-1} y verificar el resultado.

Ejercicio 7 (Resolución de ecuaciones diferenciales)**.** Consideremos el siguiente problema: hallar $y\colon [a,b] \to \mathbb{R}$ tal que

$$\begin{cases} y''(x) + g(x)y(x) = f(x) & \text{para } x \in (a,b) \\ \quad y(a) = \alpha,\ y(b) = \beta, \end{cases} \tag{E}$$

donde $f, g\colon (a,b) \to \mathbb{R}$ son funciones conocidas. Se desea hallar una aproximación numérica de la solución de (E).

a) Dividir al intervalo $[a,b]$ en N subintervalos de largo $h = \frac{b-a}{N}$ y tomar como incógnitas los valores de y en los puntos de subdivisión interiores a $[a,b]$. Éstos son de la forma $y_i = y(x_i)$, $i = 1, \ldots, N-1$, con

$$x_i = a + ih, \quad i = 0, \ldots, N.$$

En los extremos del intervalo $x_0 = a$ y $x_N = b$ la función y es conocida: $y_0 = y(x_0) = y(a) = \alpha$ e $y_N = y(x_N) = y(b) = \beta$.

b) Para $i = 1, \ldots, N-1$ considerar la aproximación

$$y''(x_i) \simeq \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2}.$$

Usando desarrollos de Taylor alrededor de x_i, y asumiendo que la función y es tan regular como sea necesario, mostrar que el error de aproximación es $O(h^2)$.

c) Imponer la ecuación (E) en cada uno de los puntos x_i para $i = 1, \ldots, N-1$ y obtener un sistema lineal de ecuaciones

$$y_{i-1} + (g_i h^2 - 2)y_i + y_{i+1} = f_i h^2, \quad i = 1, \ldots, N-1$$

donde $f_i = f(x_i)$ y $g_i = g(x_i)$ para $i = 1, \ldots, N-1$, y además $y_0 = \alpha$, $y_N = \beta$.

d) Escribir el sistema anterior en forma matricial $A\mathbf{x} = \mathbf{b}$ con A de dimensiones $(N-1) \times (N-1)$, y $\mathbf{x}, \mathbf{b}$ de dimensiones $(N-1) \times 1$. Resolver en Octave el problema (E) con

$$a = 0, \quad b = 5, \quad \alpha = 0, \quad \beta = \operatorname{sen}(5), \quad f(x) = \operatorname{sen}(x)(e^x - 1), \quad g(x) = e^x.$$

e) Para $N = 50$, graficar el resultado obtenido y compararlo con la solución exacta $y(x) = \operatorname{sen}(x)$.

Ejercicio 8 (Algoritmo de Thomas)**.** Sea $A = (a_{ij})$ una matriz $n \times n$ tridiagonal, es decir, $a_{ij} = 0$ si $|i - j| > 1$. Se tiene la siguiente descomposición LU de A:

$$\begin{array}{cccc}
A & = & L & U \\
\begin{bmatrix} a_{11} & a_{12} & 0 & \cdots & 0 \\ a_{21} & a_{22} & a_{23} & \ddots & \vdots \\ 0 & a_{32} & a_{33} & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & a_{n-1\,n} \\ 0 & \cdots & 0 & a_{nn-1} & a_{nn} \end{bmatrix} & = & \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_2 & 1 & 0 & \ddots & \vdots \\ 0 & l_3 & 1 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & l_n & 1 \end{bmatrix} & \begin{bmatrix} c_1 & u_1 & 0 & \cdots & 0 \\ 0 & c_2 & u_2 & \ddots & \vdots \\ 0 & 0 & c_3 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & u_{n-1} \\ 0 & \cdots & 0 & 0 & c_n \end{bmatrix}.
\end{array}$$

a) Demostrar que $u_k = a_{kk+1}$ para $k = 1, \ldots, n-1$ y calcular fórmulas para las entradas l_k, $k = 2, \ldots, n$ de L y para c_k, $k = 1, \ldots, n$ de U en función de las entradas a_{ij} de A.

b) Escribir un programa `x = tridiagonal(A,b)` que resuelva el sistema $A\mathbf{x} = \mathbf{b}$ para una matriz tridiagonal. Notar que, usando los resultados obtenidos en la parte anterior, se puede hallar una descomposición LU de forma eficiente. Este método suele ser llamado *algoritmo de Thomas.*

c) Usar el algoritmo de Thomas para resolver, como en el Ejercicio 7, la ecuación diferencial

$$y''(x) = -1{,}5\, y(x) + \operatorname{sen}\left(\frac{\pi}{2} x\right), \quad x \in [0,1], \quad y(0) = y(1) = 1.$$

Comparar la solución de la ecuación diferencial con las aproximaciones obtenidas usando diferentes valores de h (por ejemplo $h = 10^{-k}$, $k = 1, 2, 3, \ldots$).

d) Demostrar que el costo computacional de usar el algoritmo de Thomas para resolver el sistema $A\mathbf{x} = \mathbf{b}$ con $A \in \mathcal{M}_n(\mathbb{R})$ tridiagonal es $O(n)$.

Ejercicio 9 (Descomposición de Cholesky)**.** El algoritmo de Cholesky permite factorizar matrices simétricas definidas positivas de forma eficiente.

Recordemos que una matriz $A \in \mathcal{M}_n(\mathbb{R})$ es *simétrica* si $A = A^t$ y es *definida positiva* si se cumple cualquiera de las siguientes condiciones equivalentes:

- la forma cuadrática $\mathbf{x}^t A\mathbf{x}$ es positiva para todo vector $\mathbf{x}$ no nulo;
- todos los valores propios de A son positivos;
- existe una matriz $R \in \mathcal{M}_n(\mathbb{R})$ triangular superior tal que $A = R^t R$. Esta es la llamada descomposición de Cholesky.

Usando la última condición arriba e igualando los elementos en la fórmula $A = R^t R$, obtenemos

$$a_{ij} = \sum_{k=1}^{i} r_{ki} r_{kj}, \quad i \leq j.$$

Usar estas ecuaciones en un orden adecuado para computar los elementos de R de forma eficiente.

Ejercicio 10 (Matrices de Hilbert)**.** La *matriz de Hilbert* $H_n = (h_{ij})_{i,j=1}^n$ de orden n está definida por

$$h_{ij} = \frac{1}{i+j-1}.$$

Esta matriz es no singular y tiene una inversa explícita. Sin embargo, cuando n crece, el número de condición de H_n crece rápidamente. Las funciones de Octave `hilb(n)` e `invhilb(n)` devuelven H_n y H_n^{-1} respectivamente. Sean $\mathbf{x}_n = (1, 1, \cdots, 1)^t$ y $\mathbf{b}_n = H_n \mathbf{x}_n$. En este problema vamos a examinar dos principios fundamentales respecto a la calidad de la solución computada $\mathbf{x}_n^*$.

a) Para $n = 5, 10$, definir $\mathbf{x}_n$ usando el comando `ones`, multiplicar $H_n \mathbf{x}_n$ para obtener $\mathbf{b}_n$, y luego calcular $\mathbf{x}_n^*$ con el comando \ de Octave.

b) Computar el *error* $\mathbf{e}_n = \mathbf{x}_n - \mathbf{x}_n^*$, el *residuo* $\mathbf{r}_n = \mathbf{b}_n - H_n \mathbf{x}_n^*$, y sus normas $\|\cdot\|_1, \|\cdot\|_2, \|\cdot\|_\infty$ con el comando `norm`. Extraer conclusiones.

c) Hallar el número de condición $\kappa(H_n) = \|H_n\|\|H_n^{-1}\|$ de H_n para las normas de matrices subordinadas a las normas vectoriales $\|\cdot\|_1, \|\cdot\|_2, \|\cdot\|_\infty$. Para este fin, usar el comando `cond` y compararlo con un cálculo directo de $\kappa(H_n)$ mediante `invhilb(n)` y `norm`.

d) El número de condición da una estimación de la precisión relativa esperable en la solución. Si $k(H_n) \approx 10^t$ con un entero $t \geq 0$, entonces el número de dígitos decimales correctos en la solución se espera que sea $16-t$. ¿Cuántos dígitos decimales correctos se esperan para $n = 5, 10$?

Métodos iterativos

Ejercicio 11 (Convergencia I)**.** Se considera $A = \begin{bmatrix} 3 & -1 \\ 1 & \beta \end{bmatrix}$. Sin calcular $\rho(Q)$, indicar un rango de valores de β que asegure la convergencia de Jacobi y de Gauss-Seidel.

Ejercicio 12 (Convergencia II)**.** Analizar, según $\alpha \in \mathbb{R}$, las propiedades de convergencia de los métodos de Jacobi y Gauss-Seidel para la resolución de un sistema lineal cuya matriz es

$$A = \begin{bmatrix} \alpha & 0 & 1 \\ 0 & \alpha & 0 \\ 1 & 0 & \alpha \end{bmatrix}.$$

Ejercicio 13 (Convergencia III)**.**

a) Sin hallar Q, indicar si el método de Jacobi es convergente para las siguientes matrices:

$$A_1 = \begin{bmatrix} 4 & 1 & 2 \\ -1 & 5 & 2 \\ 5 & -2 & -10 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 4 & 1 & 5 \\ -1 & 5 & 2 \\ 2 & -2 & -10 \end{bmatrix}.$$

b) Verificar que la matriz

$$A_3 = \begin{bmatrix} 5 & -3 \\ 8 & -5 \end{bmatrix},$$

no es diagonal dominante. Demostrar, sin embargo, que el método de Jacobi es convergente para esta matriz.

c) Repetir los puntos anteriores para el método de Gauss-Seidel.

Ejercicio 14 (Implementación)**.**

a) Escribir un programa `Jacobi(A,b,x0,tol,maxiter)` que implemente el método de Jacobi para el sistema $A\mathbf{x} = \mathbf{b}$ utilizando como punto de partida el punto `x0`, una tolerancia para la condición de parada `tol` (se debe elegir una condición adecuada), y que tenga una cota en el número de iteraciones `maxiter`. La salida del programa debe ser `[sol, error]` siendo `sol` la solución encontrada, y `error` un vector conteniendo la distancia entre cada iterado y la solución final `sol`. Graficar y estudiar los resultados obtenidos, probando con diferentes tolerancias y diferentes tipos de condición de parada.

b) Hacer lo mismo que en la parte anterior para el método de Gauss-Seidel.

c) Resolver las ecuaciones diferenciales de los Ejercicios 7 y 8 usando los métodos de Jacobi y de Gauss-Seidel. Comparar y analizar los resultados obtenidos (por ejemplo, número de iteraciones vs. precisión de ambos métodos).

Ejercicio 15 (Radio espectral grande)**.** El objetivo de este ejercicio es analizar qué ocurre con una sucesión $\{x^k\}$ generada mediante la iteración estacionaria

$$\mathbf{x}^{k+1} = Q\mathbf{x}^k + \mathbf{r}, \quad \mathbf{x}^0 \in \mathbb{R}^n$$

en el caso en que $\rho(Q) \geq 1$.

a) Sea $\mathbf{e}^k = \mathbf{x}^k - \mathbf{x}^*$ el error en el paso k-ésimo. Demostrar que se cumple la ecuación $\mathbf{e}^k = Q^k\mathbf{e}^0$ para todo $k \in \mathbb{N}$.

b) Probar que si $\mathbf{e}^0$ es un vector propio de Q con valor propio asociado λ, entonces $\mathbf{e}^k = \lambda^k \mathbf{e}^0$ para todo $k \in \mathbb{N}$.

c) Probar que, si Q tiene un valor propio λ tal que

a) $|\lambda| < 1$, entonces existe un $\mathbf{x}^0$ tal que $\mathbf{x}^k$ converge a $\mathbf{x}^*$;

b) $|\lambda| = 1$, entonces existe un $\mathbf{x}^0$ tal que $\|\mathbf{e}^k\|_2$ permanece constante, y por lo tanto la iteración no converge;

c) $|\lambda| > 1$, entonces existe un $\mathbf{x}^0$ tal que $\|\mathbf{e}^k\|_2 \to \infty$, y por lo tanto la iteración diverge.

Ejercicio 16 (Divergencia del método SOR)**.** Demostrar que $\det(Q_{SOR}) = (1-\omega)^n$ y deducir que el método SOR solamente puede ser convergente si ω está en el intervalo $(0, 2)$.

Ejercicio 17 (Métodos iterativos para matrices simétricas y definidas positivas)**.** En este ejercicio, vamos a usar el siguiente resultado.

Teorema (Householder-John). Si $A, B \in \mathcal{M}_n(\mathbb{R})$ son matrices tales que A y $B + B^t - A$ son simétricas y definidas positivas, entonces la matriz $H = I_n - B^{-1}A$ cumple $\rho(H) < 1$.

a) Demostrar que si A es una matriz simétrica y definida positiva, entonces el método de Gauss-Seidel aplicado a A es convergente.
[Sugerencia: demostrar que si A es definida positiva, entonces los elementos de su diagonal son estrictamente positivos.]

b) Demostrar que si A es una matriz simétrica y definida positiva, entonces el método SOR aplicado a A es convergente si y solo si $0 < \omega < 2$. (Notar que el que sea una condición necesaria se deduce del ejercicio anterior.)

c) Verificar que la matriz

$$A = \begin{bmatrix} 2 & -1 & 1 \\ -1 & 2 & -1 \\ 1 & -1 & 2 \end{bmatrix}$$

es simétrica y definida positiva. Esta verificación puede hacerse computacionalmente o a mano, y en el último caso puede ser útil observar que 1 es valor propio de A. Experimentar en Octave con el método de Jacobi aplicada a A y extraer conclusiones.

Ejercicio 18 (Relajar para converger)**.** Consideremos el sistema $A\mathbf{x} = \mathbf{b}$, donde

$$A = \begin{bmatrix} -3 & 4 & 0 \\ 1 & -2 & 3 \\ -1 & 9 & 3 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} -3 \\ 4 \\ 2 \end{bmatrix}.$$

a) Verificar que la solución del sistema es $\mathbf{x} = [1, 0, 1]^t$.

b) Implementar la resolución del sistema de arriba mediante los métodos de Jacobi y de Gauss-Seidel, y verificar computacionalmente que ambas iteraciones son divergentes.

c) Calculando los valores propios de la matriz de iteración Q_{GS}, demostrar que efectivamente el método de Gauss-Seidel es divergente para este problema.

d) Considerar los métodos de Jacobi relajado, de Gauss-Seidel relajado, y SOR, todos ellos con parámetros de relajación $\omega \in (0,1)$. Verificar computacionalmente que para este problema las tres iteraciones son convergentes si ω es suficientemente pequeño. Hallar el parámetro de relajación óptimo para cada uno de ellos y comparar la cantidad de iteraciones necesarias para la convergencia comenzando con $\mathbf{x}^0 = \mathbf{0}$ y con tolerancia `tol = 1e-6`.

Ejercicios opcionales

Ejercicio 19 (Del examen de diciembre de 2023)**.** Sean

$$A = \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix}$$

y $\mathbf{b} \in \mathbb{R}^2$. Para resolver el sistema $A\mathbf{x} = \mathbf{b}$, consideramos el método iterativo

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha(\mathbf{b} - A\mathbf{x}^k).$$

a) La iteración es convergente si y sólo si α está en un cierto intervalo. ¿Cuál es ese intervalo?

b) Hallar el valor de α para el que se espera que la iteración converja en la menor cantidad de pasos.

[Puede ser útil tener en cuenta que un número $\lambda \in \mathbb{R}$ es valor propio de A si y sólo si $1 - \lambda$ es valor propio de $I - A$, y que $\lambda \in \mathbb{R}$ es valor propio de A si y sólo si $\alpha\lambda$ es valor propio de αA ($\alpha \neq 0$).]

Ejercicio 20 (Método del gradiente)**.** El método del gradiente es un algoritmo bastante popular en la práctica[1]. Vamos a considerar sistemas

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

donde $A \in \mathcal{M}_n(\mathbb{R})$ es una matriz simétrica y definida positiva y $\mathbf{b} \in \mathbb{R}^n$.

a) Denotamos por $\cdot$ el producto interno usual en $\mathbb{R}^n$. Consideremos la función $\Phi\colon \mathbb{R}^n \to \mathbb{R}$, $\Phi(\mathbf{x}) = \frac{1}{2}A\mathbf{x}\cdot\mathbf{x} - \mathbf{b}\cdot\mathbf{x}$. Demostrar que para todo $\mathbf{x} \in \mathbb{R}^n$ se tiene $\nabla\Phi(\mathbf{x}) = A\mathbf{x} - \mathbf{b}$. Deducir que Φ tiene un único punto crítico $\mathbf{x}^*$, que es la solución al sistema $A\mathbf{x} = \mathbf{b}$.

b) Verificar que en todo $\mathbf{x} \in \mathbb{R}^n$ la matriz hessiana de Φ es A (puede ser útil notar que Φ es una función cuadrática) y deducir que el único punto crítico de Φ es un mínimo.

c) El método de descenso por gradiente para hallar mínimo de Φ consiste en explotar el hecho de que $\nabla\Phi(\mathbf{x})$ "apunta" en la dirección de máximo crecimiento.

Dado $\mathbf{x}^0$, para cada $k \geq 0$ computar $\mathbf{s}^k := \mathbf{b} - A\mathbf{x}^k = -\nabla\Phi(\mathbf{x}^k)$ y tomar $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k\mathbf{s}^k$, donde $\alpha_k \in \mathbb{R}$ es un número a determinar.

Para definir α_k, una estrategia es buscar mínimos a la función de $\mathbb{R} \to \mathbb{R}$ que manda $t \mapsto \Phi(\mathbf{x}^k + t\mathbf{s}^k)$. Derivar esta función y deducir que da lugar a tomar $\alpha_k := -\frac{(A\mathbf{x}^k - \mathbf{b})\cdot\mathbf{s}^k}{A\mathbf{s}^k\cdot\mathbf{s}^k}$.

d) Implementar este algoritmo, eligiendo adecuadamente criterios de parada. Experimentar con matrices con diferentes números de condición, y determinar cuántas iteraciones se necesitan para alcanzar los criterios de parada en cada caso.

[1] De hecho, se lo usa más como algoritmo de minimización que como método para resolver sistemas lineales, y se lo conoce como método de *descenso por gradiente*. También existen variantes, como el método de gradiente conjugado, que se pueden consultar en el libro de Heath.