Pràctica 1: Àlgebra Lineal

Les instruccions precises de què (i quan) cal entregar com a resultat d'aquesta pràctica les expliquem al final d'aquest document.

Fixem $n \geq 2$. Donats $x \in \mathbb{R}^n$, $b \in \mathbb{R}^n$ i $A \in \mathbb{R}^{n\times n}$ de rang màxim ($\det(A) \neq 0$), considerem el sistema lineal $Ax = b$. El nostre objectiu és resoldre'l. Per tal d'anar de situacions més senzilles a més complicades ho farem en **TRES** parts.

Primera Part: Sistemes triangulars

Implementeu una funció per resoldre el sistema d'equacions $Ax = b$ quan A és triangular superior (zeros sota de la diagonal principal). La funció tindrà com a capçalera

```
int resoltrisup (int n, double **A, double *b, double *x, double tol)
```

La funció rebrà com a paràmetres la dimensió `n`, la matriu A i el vector b, respectivament i `tol` és la tolerància acceptada, és a dir, qualsevol nombre més petit, en valor absolut, que `tol` el considerem 0.

Per tal de resoldre el sistema, la funció `resoltrisup` suposarà que la matriu A és triangular superior, és a dir, tal que $a_{ij} = 0$ per $0 \leq j < i$, $i = 0, \ldots, n-1$. La solució del sistema es guardarà en el vector `x`. D'altra banda, la funció retornarà l'enter 0 si ha pogut resoldre el sistema i 1 altrament.

Dissenyeu un programa principal (la funció `main`) que llegeixi una matriu $A = (a_{ij})_{0\leq i,j<n}$, llegeixi un vector $b = (b_i)_{0\leq i<n}$ i cridi la funció `resoltrisup`. Una vegada s'hagi calculat la solució x del sistema $Ax = b$, el programa principal escriurà el vector solució x i el valor $\|Ax - b\|_2$ com a indicador de la bondat de la solució x proposada. Com aplicació resoleu els sistemes següents, amb toleràncies 10^{-3}, 10^{-6}, 10^{-9} i 10^{-12},

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 0 & 3 & 2 & 3 \\ 0 & 0 & -1 & 2 \\ 0 & 0 & 0 & 1 \end{pmatrix}, b = \begin{pmatrix} -4 \\ -5 \\ -6 \\ -2 \end{pmatrix}, A = \begin{pmatrix} 1.0234 & 2.0981 & 9.9871 & 1.1 \\ 0 & -6.9876 & 2.2222 & 0.3333 \\ 0 & 0 & -1.9870 & 20.121 \\ 0 & 0 & 0 & 1.1234 \end{pmatrix}, b = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}.$$

Per a calcular $y = Ax$ useu la funció

```
void prodMatVec (int n, double **A, double *x, double *y);
```

Segona Part: El cas general. El mètode de Gauss amb pivotatge

Si la matriu $A \in \mathbb{R}^{n\times n}$ té rang màxim, el sistema $Ax = b$ té una única solució. Pot passar però que el **mètode de Gauss** aplicat directament no ens doni aquesta solució, això és degut a què, encara que la matriu A té rang màxim, en algun pas del mètode es dóna que $|a_{kk}^{(k)}| <$ `tol` i per tant no poguem continuar.

Per solucionar aquest problema podem usar **pivotatge maximal per columnes**. En cada pas k abans de calcular m_{ik} busquem $max_{j=k,\dots,n}|a_{jk}^{(k)}|$. Si denotem per $l \in \{k, k+1, \dots n\}$ la fila tal que $a_{lk}^{(k)}$ assoleix aquest màxim, llavors intercanviem la fila l amb la fila k (també l'element corresponent del vector b) i continuem amb el mètode.

Implementeu una funció que faci aquest algorisme. La funció tindrà com a capçalera

```
int gausspivot(int n, double **A, double *v, double tol)
```

i resoldrà el sistema $Ax = b$ usant el **mètode de Gauss amb pivotatge**

Notes: a) Observeu que un cop triangularitzat el sistema caldrà cridar la funció anterior per resoldre el sistema. b) `A` no conté la matriu original a la sortida de la funció i `v` conté la solució del sistema a la sortida.

Dissenyeu un programa principal (la funció `main`) que llegeixi una matriu $A = (a_{ij})_{0\leq i,j<n}$, llegeixi un vector $b = (b_i)_{0\leq i<n}$ i cridi a la funció `gausspivot`. Una vegada s'hagi calculat la solució x del sistema $Ax = b$, el programa principal escriurà el vector solució x i el valor $\|Ax - b\|_2$ com a indicador de la bondat de la solució x proposada.

Per testejar la funció resoleu els sistemes

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, b = \begin{pmatrix} 2 \\ 5 \\ 8 \end{pmatrix}, A = \begin{pmatrix} 1.1 & 2.3 & 4.1 & 0.3 \\ 2.3 & -4.2 & 3.2 & -1 \\ 6.5 & -0.3 & 8.27 & 0 \\ 9.71 & 4.21 & -34.7 & 1.23 \end{pmatrix}, b = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}.$$

Tercera (i última) Part: Càlcul de la matriu inversa

Tal com s'explica en els apunts de teoria, per calcular númericament la inversa d'una matriu no singular A de mida n cal resoldre n sistemes lineals del tipus $AX = b_j,\ j = 1, \ldots n$ on cadascun dels vectors b_j corresponen al vector j-èssim de la base canònica de $\mathbb{R}^n$; és a dir

$$b_j(0, \ldots, 0, 1, 0, \ldots 0),\ j = 1, \ldots n$$

on l'1 correspon a la component j del vector b_j.

És fàcil observar que per resoldre aquest problema NO CAL cridar la funció anterior n cops ja que la major part dels càlculs són repetits. La idea és doncs resoldre els n sistemes de forma simultània però on el terme independent és ara una matriu B de tamany n i cada columna, al principi del procés, és el vector b_j esmentat anteriorment.

Implementeu una funció que faci aquest algorisme. La funció tindrà com a capçalera

```
int inversaA(int n, double **A, double **B, double tol)
```

i resoldrà el sistema $AX = B$ usant el **mètode de Gauss amb pivotatge**. A l'entrada la matriu B serà la identitat. A cada pas k del procés de Gauss caldrà modificar TOTS els termes de la matriu B de la fila k, a diferència d'abans quan només havíem de tocar un element del terme independent (observeu que, altre cop, un cop triangularitzat tot el sistema caldrà cridar la funció anterior per resoldre els sistemes; un per cada columna de la matriu B). Podeu guardar la matriu inversa de la matriu A en la matriu B.

Dissenyeu un programa principal (la funció `main`) que llegeixi la matriu $A = (a_{ij})_{0 \le i,j < n}$ i cridi a la funció `inversaA`.

Per testejar la funció calculeu la matriu inversa de les matrius següents:

$$A_1 = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix},\ A_2 = \begin{pmatrix} 1.1 & 2.3 & 4.1 & 0.3 \\ 2.3 & -4.2 & 3.2 & -1 \\ 6.5 & -0.3 & 8.27 & 0 \\ 9.71 & 4.21 & -34.7 & 1.23 \end{pmatrix}.$$

Un cop fet això comproveu, fent ús de la funció de producte de matrius que ja teniu feta de la part introductòria, per comprovar que el producte us dóna, bàsicament, la identitat.

Per a calcular $C = AB$ useu la funció

```
void prodMatMat (int n, double **A, double **B, double **C);
```

Instruccions per a l'entrega

Abans de començar a fer la pràctica heu de crear un subdirectori anomenat:

Grup-Cognom1Cognom2Nom-X

on

- Grup: és el vostre grup de pràctiques en majúscules (pot ser A, B, C o D).
- Cognom1Cognom2Nom: és el vostre primer cognom, segon cognom i nom.
- X: identifica el número de la pràctica (1, 2, 3, etc).

Exemple: `A-LopezPerezMaria-1` correspon a una alumna del grup A que fa la pràctica 1.
Aquest directori contindrà els arxius .c corresponents a les diverses parts:

a) Arxiu `prac1funs.c` que conté les funcions `resoltrisup` , `prodMatVec`, `prodMatMat`.

b) Arxiu `prac1a.c` que conté el programa pricipal de la primera part.

c) Arxiu `prac1b.c` que conté el programa pricipal de la segona part i la funció `gausspivot`.

d) Arxiu `prac1c.c` que conté el programa principal de la tercera part i la funció `inversaA`.

Es crearà un arxiu comprimit del directori amb la comanda

```
tar -czvf A-LopezPerezMaria-1.tgz A-LopezPerezMaria-1
```

executada des del directori pare.

Entregar la pràctica vol dir el següent:

(1) Es penjarà el fitxer comprimit (.tgz) al campus virtual abans del **31 d'octubre de 2014**.

Tots els arxius .c lliurats hauran de començar amb les dades de l'alumne en un comentari de la forma

/ COGNOM1: COGNOM2: NOM: DNI: */*

Tots els programes hauran de compilar amb les opcions: `-ansi`, `-pedantic`, `-O` i `-Wall`.
Lliurar un programa sense les dades personals (usant l'estil anterior), o amb algun error o avís (warning) de compilació, serà avaluat amb la qualificació mínima.

(2) Entregar un document (imprès) durant l'hora del laboratori de l'**11 de novembre de 2014** on es doni resposta argumentada a les diferents qüestions que es proposin amb relació a la pràctica.