Sobre las rutinas FACTOR y SOLVE.

Para resolver todos los sistemas lineales $Ax = b$ presentados se programaron estas rutinas en su versión de Matlab. Fundamentalmente cada rutina se puede resumir a:

1. FACTOR.
   Esta rutina **recibe** la matriz del sistema A.
   Devuelve las matrices L, U y P que son triangular inferior, triangular superior y de permutación que verifican
   $$PA = LU$$
   se verifica que P es unitaria, por lo tanto $P^{-1} = P^T$. Esta descomposición se realiza a través de la función `lu` de Matlab, misma que emplea una variante de eliminación gaussiana con pivoteo parcial. También se puede **devolver** a P como un vector columna que contiene los mismos intercambios de filas que P (esta opción debe especificarse).

   Adicionalmente FACTOR **devuelve** el número de condición de A calculado como
   $$cond(A) = \|A\|_p \|A^{-1}\|_p$$
   con la función `cond` de Matlab, donde se realiza explícitamente el calculo de A^{-1} y se calculan las normas según se requiera (se puede elegir entre la norma 1, 2, infinita y de Frobenius). Opcionalmente se puede **devolver** también el inverso del número de condición calculado con la función `rcond` de Matlab, pero que no resulta ser un cálculo preciso sino una aproximación al inverso de $cond(A)$.

```
function [L,U,P,ncondi,inv_ncondi] = FACTOR(A)
    %Hacemos eliminación para A y obtenemos L,U
    %P se puede obtener como matriz o como vector (descomentar)
    [L,U,P]=lu(A);
    %%%[L,U,P]=lu(A,'vector');

    %Calculamos el numero de condición de la matriz
    %la función cond acepta la norma que se desea emplear
    ncondi=cond(A,1);
    %Calculamos el inverso de la condición de la matriz (norma 1)
    inv_ncondi=rcond(A);
    return;
end
```

2. SOLVE.
   Esta rutina recibe las matrices L, U, P y b que verifican $PA = LU$ y $Ax = b$. Las tres primeras son resultado de usar `FACTOR(A)` y b es el vector columna dado del lado derecho del sistema.
   Devuelve un solo vector columna x que es la solución de $Ax = b$. La resolución se determina con las soluciones de dos sistemas lineales, los cuales son
   $$Ly = Pb$$
   $$Ux = y$$

Notemos que en ambos casos los sistemas son triangulares (inferior y superior respectivamente), por lo que la resolución se puede realizar a través de sustitución (hacia adelante y atrás respectivamente). Este procedimiento se realiza a través de la función `mldivide` (o el operador \) de Matlab, misma que automáticamente detecta que los sistemas dados son triangulares y procede a resolverlos por sustitución.

Opcionalmente, como ya sabemos que estos sistemas son triangulares, se podría evitar la comprobación que hace `mldivide` utilizando en su lugar la función `linsolve` que puede recibir como parámetro la indicación de resolver directamente el sistema triangular sin realizar ninguna comprobación.
Esta diferencia de rendimiento es más importante para sistemas grandes.

```
function [x] = SOLVE(L,U,P,b)
    %El primer sistema es Ly=Pb para y
    y=L\(P*b);
    %El segundo sistema es Ux=y para x
    x=U\y;
    %%%Opcionalmente se puede usar linsolve (sistemas grandes)
    %opts.LT=true; y=linsolve(L,P*b,opts);
    %opts.UT=true; x=linsolve(U,y,opts);
    return;
end
```

2.13 Los códigos Factor/Solve pueden ser usados para encontrar elementos de A^{-1}. La inversa es usada para estimar ciertos parámetros estadísticos y también para estudiar la sensibilidad de la solución a errores en los datos. Sea x^i la solución de $Ax^i = b^i, i = 1,2,\cdots,n$ donde el i-ésimo lado derecho b^i es $b^i_j = \begin{cases} 0 \; si \; i \neq j \\ 1 \; si \; i = j \end{cases}$.

Si formamos la matriz $X = [x^1 \quad x^2 \quad \cdots \quad x^n]$ es fácil demostrar que $X = A^{-1}$. Usa Factor/Solve para hallar la inversa de la matriz $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9.01 \end{pmatrix}$.

Resolver el sistema $Ax = b$ para distintos lados derechos, en particular para lados derechos $b = [1,0,0]^T, [0,1,0]^T, [0,0,1]^T$, nos dará por soluciones las columnas 1, 2 y 3 de A^{-1} respectivamente.

Se utilizó FACTOR para determinar la factorización LU de A y después se utilizo SOLVE tres veces consecutivas para cada uno de los lados derechos $[1,0,0]^T, [0,1,0]^T, [0,0,1]^T$ (no es necesario volver a usar FACTOR porque ya tenemos determinada la factorización LU, solo se va cambiando el lado derecho a resolver).

```
%Declaramos la matriz
A=[1,2,3;4,5,6;7,8,9.01];
%Calculamos la factorización LU para A usando FACTOR
[L,U,P,condicion]=FACTOR(A);
%Calculemos su inversa obteniendo columna a columna
%Como solo va cambiando el lado derecho, solo usamos SOLVE
%Inicializamos la inversa como una matriz 3x3
inversa=zeros(3,3);
for i=1:3
    %El lado derecho es un vector cuya i-esima entrada es 1
    lado_derecho=zeros(3,1);
    lado_derecho(i)=1;
    %Obtenemos la columna y añadimos la columna a la inversa
    [inversa(:,i)]=SOLVE(L,U,P,lado_derecho);
end
%Mostramos la inversa obtenida y su multiplicación
disp('Inv(A)='); disp(inversa);
%Mostramos la multiplicación A*inversa
disp('A*Inv(A)='); disp(A*inversa);
```

```
Command Window
>> ejercicio_2_13
Inv(A)=
       98.333      -199.33          100
      -198.67       399.67         -200
          100         -200          100

A*Inv(A)=
                1  -1.1369e-13            0
  -1.1369e-13                1            0
            0                0            1
```

En la salida obtenida, se muestra el cálculo de A^{-1} y la multiplicación matricial de AA^{-1} mostrados en formato corto.

Veamos que $AA^{-1} \neq I$ por el comportamiento de las entradas (1,2) y (2,1), sin embargo, su tamaño pequeño puede darnos una idea de que esta aproximación a la inversa es buena, ya que si exponente es 10^{-13}, mismo que puede ser ocasionado por el error de redondeo.

2.14 Considera un sistema $Ax = b$ con A la matriz de Hilbert de orden 3.

a) Usando el método anterior, encuentra la inversa de A. Calcula el número de condición exacto de A.
b) Usa FACTOR/SOLVE para $Ax = b$ con A la matriz de Hilbert con entradas truncadas a dos dígitos y $b = [1,0,0]^T$. ¿Cuánto vale COND? ¿La siguiente desigualdad es válida para este problema? Considera $\|\Delta A\| = 0.003333$.

$$\frac{\|\Delta x\|}{\|x\|} \leq \frac{cond(A)}{1 - cond(A)\frac{\|\Delta A\|}{\|A\|}}\left(\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta \mathrm{b}\|}{\|b\|}\right), \qquad \text{válida para } \|A^{-1}\|\|\Delta A\| < 1$$

a) Para representar la matriz de Hilbert de orden 3 se utilizó la función `hilb(3)` de Matlab. Se utilizaron las rutinas `FACTOR/SOLVE` como se hizo anteriormente para calcular la inversa de la matriz.
   Una vez obtenida esta última se utilizó la función `norm` (configurada como norma infinita) de Matlab y la condición devuelta por `FACTOR` para obtener el número de condición exacto de A.
b) Para la matriz de Hilbert de orden 3 truncada a dos dígitos (A) se ingresaron las entradas manualmente y con el proceso anterior se determinó la inversa. Con este resultado se calculó el factor $\|A^{-1}\|\|\Delta A\|$, el cual debe ser menor a 1 para la validez e la desigualdad.

```
%A es la matriz de hilbert 3x3
A=hilb(3);
%Factorizando LU a la matriz A
[L,U,P,condicion]=FACTOR(A);
%Encontremos la inversa columna a columna con la tecnica de resolver para
%distintos lados derechos
inversa=zeros(3,3);
for i=1:3
    lado_derecho=zeros(3,1); lado_derecho(i)=1;
    [inversa(:,i)]=SOLVE(L,U,P,lado_derecho);
end
%Mostramos la inversa obtenida
disp('Inversa de Hilb(3)'); disp(inversa);
%Mostramos el número de condicion provisto por FACTOR (Matlab) y el
%calculado con la inversa obtenida y la rutina de norma_maximo
disp('Números de condición para Hilb(3)');
disp(condicion); disp(norm(A,'Inf')*norm(inversa,'Inf'));

%Ahora A es la matriz de Hilbert con decimales truncados a 2 digitos
A=[1.00,0.50,0.33;0.50,0.33,0.25;0.33,0.25,0.20];
%A traves del proceso anterior obtenemos la inversa
[L,U,P,condicion]=FACTOR(A);
inversa=zeros(3,3);
for i=1:3
```

```
    lado_derecho=zeros(3,1); lado_derecho(i)=1;
    [inversa(:,i)]=SOLVE(L,U,P,lado_derecho);
end
%Esta vez calculamos el numero de condición con nuestra rutina
disp('Número de condición para Hilb(3) truncada')
disp(norm(A,'Inf')*norm(inversa,'Inf'));
%Calculamos el factor para la validez de la desigualdad
disp('Factor para validez de desigualdad');
disp(norm(inversa,'Inf')*0.003333);
```

```
Command Window
>> ejercicio_2_14
Inversa de Hilb(3)
          9.00000000000003         -36.0000000000001          30.0000000000001
         -36.0000000000001          192.000000000001         -180.000000000001
          30.0000000000001         -180.000000000001          180.000000000001

Números de condición para Hilb(3)
          748.000000000004

          748.000000000003

Número de condición para Hilb(3) truncada
          5623.61904761876

Factor para validez de desigualdad
          10.2423619047614
```

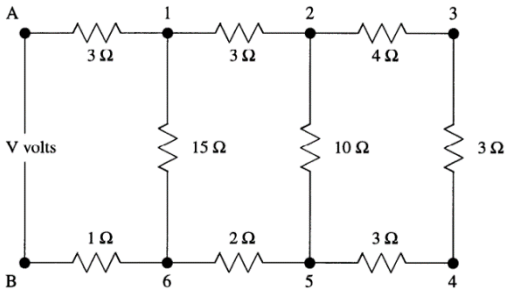
En la salida obtenida se muestra la inversa calculada de la matriz de Hilbert de orden 3 con un error por redondeo mínimo. Los dos números de condición calculados solo difieren en aproximadamente 1×10^{-12}, por lo que concluimos que el número de condición de la matriz de Hilbert de orden 3 es $\mathbf{748}$.

Por otro lado, el número de condición de la matriz de Hilbert de orden 3 para cuando truncamos a dos dígitos se dispara, resultando en aproximadamente $\mathbf{5623.619}$. Por último, tenemos que $\|\boldsymbol{A}^{-1}\|\|\boldsymbol{\Delta A}\| \approx \mathbf{10.242}$ lo cual claramente es mayor que uno, invalidando la desigualdad presentada. Nótese que no necesitamos resolver el sistema completo.

2.15 Suponga que tenemos una red eléctrica y deseamos encontrar los potenciales eléctricos en las uniones de la (1) a la (6) como se muestra en la figura. El potencial aplicado entre A y B es de V volts. Denotemos a los potenciales como $v_1, v_2, \cdots, v_6$. Aplicando la ley de Ohm y la ley de corrientes de Kirchhoff obtenemos el siguiente conjunto de ecuaciones lineales para cada voltaje:

$$\begin{aligned} 11v_1 - 5v_2 - v_6 &= 5V \\ -20v_1 + 41v_2 - 15v_3 - 6v_5 &= 0 \\ -3v_2 + 7v_3 - 4v_4 &= 0 \\ -v_3 + 2v_4 - v_5 &= 0 \\ -3v_2 - 10v_4 + 28v_5 - 15v_6 &= 0 \\ -2v_1 - 15v_5 + 47v_6 &= 0 \end{aligned}$$

Resuelve para $V = 50$.



Para este ejercicio se introdujeron manualmente la matriz A y b del sistema $Av = b$ que se está planteando. Simplemente se utilizó `FACTOR/SOLVE` para determinar v. Adicionalmente se muestra el número de condición de A.

```
%El sistema Av=b para los voltajes es
A=[11,-5,0,0,0,-1; -20,41,-15,0,-6,0; 0,-3,7,-4,0,0;
    0,0,-1,2,-1,0; 0,-3,0,-10,28,-15; -2,0,0,0,-15,47];
disp('Matriz A del sistema Av=b:'); disp(A);
b=[5*50;0;0;0;0;0];
%Resolviendo para hallar los voltajes v
[L,U,P,condicion]=FACTOR(A);
[v]=SOLVE(L,U,P,b);
%Mostramos el resultado y el numero de condicion del sistema
disp ('Voltajes:'); disp(v);
disp('Condición del sistema:'); disp(condicion);
```

```
Command Window
>> ejercicio_2_15
Matriz A del sistema Av=b:
    11    -5     0     0     0    -1
   -20    41   -15     0    -6     0
     0    -3     7    -4     0     0
     0     0    -1     2    -1     0
     0    -3     0   -10    28   -15
    -2     0     0     0   -15    47

Voltajes:
    35
    26
    20
  15.5
    11
     5

Condición del sistema:
   148.7808
```

Se obtuvo como solución

$$\begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{pmatrix} = \begin{pmatrix} 35 \\ 26 \\ 20 \\ 15.5 \\ 11 \\ 5 \end{pmatrix}$$

Y se tiene que el número de condición del sistema es **148.7808.**

2.17 Analizando muestras ambientales tomadas de la atmosfera, un modelo simple con m muestras y n fuentes y químicos produce $AX = B$, donde a_{jk} es la concentración promedio del elemento i proveniente de la fuente k, x_{ki} es la masa de las partículas provenientes de la fuente k contribuyendo a la muestra j, b_{ij} es la concentración del elemento i en la muestra j, y $1 \leq i \leq n$, $1 \leq k \leq n$, $1 \leq j \leq m$. Si $m = 4, n = 3$, entonces

$$A = \begin{pmatrix} 0.172 & 0.013 & 0.144 \\ 0.368 & 0.681 & 0.271 \\ 0.099 & 0.510 & 0.329 \end{pmatrix}, B = \begin{pmatrix} 1.44 & 4.35 & 1.32 & 3.95 \\ 2.84 & 9.30 & 2.90 & 8.29 \\ 2.36 & 3.45 & 3.25 & 7.35 \end{pmatrix}$$

a) ¿Cuál es X? ¿Qué te dice la condición del sistema acerca de la fiabilidad de este resultado? Primero asume datos exactos, entonces las entradas de A y B son redondeadas a los valores mostrados.
b) Usa el método del ejercicio *2.13* para computar A^{-1}. ¿Cuál es el número de condición exacto?
c) Considera $\Delta x_i = \sum_{p=1}^{n} \alpha_{ip} \Delta b_p \, , i = 1, \ldots$ con α_{ij} las entradas de A^{-1} y Δb_p las componentes del vector Δb, entonces Δx_i son las componentes del vector Δx, obteniendo la relación $\Delta x = A^{-1} \Delta b$.
¿Qué te dice esta expresión acerca de la sensibilidad de x_{21} a los cambios de b_{11}? Reemplaza b_{11} por 1.43 y recalcula x_{21}. ¿Las respuestas numéricas confirman la teoría? Debes considerar cambios relativos entre los datos y la solución.

a) Para encontrar la matriz X solución al sistema matricial $AX = B$ se optó por solucionar varios sistemas lineales de la forma $Ax^i = b^i$ con $i = 1,2,3,4$, donde x^i y b^i representan la i-ésima columna de las matrices X y b. En pocas palabras, se optó por resolver el sistema columna a columna.
Para esto se factorizó LU con `FACTOR` una sola vez y después se resolvieron los cuatro sistemas mencionados, encontrando las columnas de X.

También se calculó el número de condición del sistema de manera exacta, obteniendo $cond(A) = 13.648$, al ser relativamente pequeño nos dice que la X encontrada es fiable.

b) Se utilizó el método de hallar la inversa columna por columna como en los ejercicios anteriores. El número de condición exacto ya se ha calculado, pero se recalculó tomando la norma de la inversa que acabamos de obtener.

c) La entrada b_{11} fue cambiada de 1.44 a 1.43 y se recalculó la matriz X. Para claridad en este inciso escribamos las matrices obtenidas en el programa. La A^{-1} obtenida es:

$$A^{-1} = \begin{pmatrix} 2.7819 & 2.2414 & -3.0639 \\ -3.0542 & 1.3719 & 0.20676 \\ 3.8974 & -2.8011 & 3.6410 \end{pmatrix}$$

Y la perturbación ΔB de B es

$$\Delta B = \begin{pmatrix} 0.01 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

La expresión $\Delta x_i = \sum_{p=1}^{n} \alpha_{ip} \Delta b_p \, , i = 1, \dots, 4$ nos permite obtener la perturbación que sufre todo un renglón de X como consecuencia de haber perturbado B.
aplicándola específicamente para el renglón 2 y la columna 1 (ya que estudiamos X_{21}, entonces $i = 2$ y solo considerando la columna 1 de ΔB, es decir, $\Delta b_p = \Delta B_{p1}$), tenemos que el efecto consecuente de haber perturbado B_{11} es

$$\Delta x_2 = \sum_{p=1}^{4} \alpha_{2p} \Delta B_{1p} = \alpha_{21} \Delta B_{11} + \alpha_{22} \Delta B_{21} + \alpha_{23} \Delta B_{31}$$
$$= (-3.0542)(-0.01) + (1.3719)(0) + (0.20676)(0) = 0.030542$$

Obtuvimos que la entrada $X(2,1)$ antes y después del cambio tiene los valores -0.0139 y 0.0166 respectivamente, entonces basta ver que

$$|-0.0139 - 0.0166| = 0.0305$$

¡Que resulta ser aproximadamente igual a Δx_2! (no se muestra la igualdad completamente por el número de decimales mostrados, se está mostrando en formato corto), es decir, la respuesta numérica concuerda en buena medida con la respuesta teórica.
Como nota extra, Δx_2 nos dice el cambio en todo el renglón, sin embargo, podemos ver que la matriz X recalculada mantiene sus entradas iguales en el resto del renglón, esto se debe a que no perturbamos en sus respectivas columnas en B.

```
%Guardamos A y B del sistema AX=B
A=[0.172,0.013,0.144; 0.368,0.681,0.271; 0.099,0.510,0.329];
B=[1.44,4.35,1.32,3.95; 2.84,9.30,2.90,8.29; 2.36,3.45,3.25,7.35];
%Para determinar X debemos reducir AX=B a sistemas de la forma Ax=b con
%x y b vectores columna de X y B respectivamente
%Factorizamos LU a A inicialmente
[L,U,P,condicion]=FACTOR(A);
%Inicializamos la matriz resultado X como una 3x4
X=zeros(3,4);
%Generamos las columnas de X una a una
for i=1:4
    %Resolvemos Ax=b con x y b las i-esimas columnas de X y B
    [X(:,i)]=SOLVE(L,U,P,B(:,i));
end
%Mostramos la solución obtenida X
disp('X:'); disp(X);

%Obtención de la inversa de A (matriz 3x3) columna a columna
inversa=zeros(3,3);
for i=1:3
```

```
    lado_derecho=zeros(3,1); lado_derecho(i)=1;
    [inversa(:,i)]=SOLVE(L,U,P,lado_derecho);
end
disp('Inv(A):'); disp(inversa);
%Mostramos los numeros de condición provistos por FACTOR y calculados
disp('Número de condición provisto por Matlab:'); disp(condicion);
disp('Número de condición calculado: ');
disp(norm(A,'Inf')*norm(inversa,'Inf'));

%Guardamos X(2,1) antes del cambio
x_2_1_antes=X(2,1);
%Cambiamos la entrada B(1,1) de 1.44 a 1.43
B(1,1)=1.43;
%Resolvemos una vez más
X=zeros(3,4);
for i=1:4
    [X(:,i)]=SOLVE(L,U,P,B(:,i));
end
%Guardamos X(2,1) despues del cambio
x_2_1_despues=X(2,1);
%Mostramos X recalculada tras la perturbación
disp('X calculado tras la perturbación en B(1,1):'); disp(X);
%Mostramos la diferencia entre X(2,1) antes y despues de perturbar
disp('|X(2,1)_antes - X(2,1)_despues|:');
disp(abs(x_2_1_antes-x_2_1_despues));
```

Command Window

```
>> ejercicio_2_17
X:
       3.1408       22.376       0.2146       7.0503
    -0.013943      0.18607      0.61889      0.82852
       6.2498       3.4647       8.8545       18.935

Inv(A):
       2.7819       2.2414      -3.0639
      -3.0542       1.3719      0.20676
       3.8974      -2.8011        3.641

Número de condición provisto por Matlab:
       13.648

Número de condición calculado:
       13.648

X calculado tras la perturbación en B(1,1):
        3.113       22.376       0.2146       7.0503
     0.016599      0.18607      0.61889      0.82852
       6.2108       3.4647       8.8545       18.935

|X(2,1)_antes - X(2,1)_despues|:
     0.030542
```

2.18 Considera el sistema lineal

$$\begin{pmatrix} 0.217 & 0.732 & 0.414 \\ 0.508 & 0.809 & 0.376 \\ 0.795 & 0.886 & 0.338 \end{pmatrix} x = \begin{pmatrix} 0.741 \\ 0.613 \\ 0.485 \end{pmatrix}$$

a) Resuelve para x usando FACTOR/SOLVE.
b) Si cada entrada de A y b pudieran tener un error de ± 0.0005, ¿Qué tan fiable resulta ser x?
c) Has cambios arbitrarios de ± 0.0005 en los elementos de A para obtener $A + \Delta A$ y en los elementos de b para obtener $b + \Delta b$. Resuelve el sistema $(A + \Delta A)(x + \Delta x) = (b + \Delta b)$ para obtener $(x + \Delta x)$. Calcula $\frac{\|\Delta x\|}{\|x\|}$. ¿Es esto consistente con el inciso b)? ¿Cuál es el cambio relativo en cada x_i?

a) Se determinó x utilizando FACTOR/SOLVE sencillamente.

b) Si se perturbaran las entradas de A y b en ± 0.0005 entonces

$$\Delta A = \begin{pmatrix} \pm 0.0005 & \pm 0.0005 & \pm 0.0005 \\ \pm 0.0005 & \pm 0.0005 & \pm 0.0005 \\ \pm 0.0005 & \pm 0.0005 & \pm 0.0005 \end{pmatrix}, \qquad \Delta b = \begin{pmatrix} \pm 0.0005 \\ \pm 0.0005 \\ \pm 0.0005 \end{pmatrix}$$

De donde $\|\Delta A\| = |\pm 0.0005| \times 3 = 0.0015$ y $\|\Delta b\| = |\pm 0.0005| = 0.0005$ (usando la norma infinito para matrices). El error relativo de x calculado con estas perturbaciones, i.e. $\frac{\|\Delta x\|}{\|x\|}$, está acotado por la expresión

$$\frac{\|\Delta x\|}{\|x\|} \leq \frac{cond(A)}{1 - cond(A)\frac{\|\Delta A\|}{\|A\|}} \left(\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta \mathrm{b}\|}{\|b\|} \right), \qquad \text{válida para } \|A^{-1}\|\|\Delta A\| < 1$$

Pero no se cumple la condición que la hace válida. Se obtuvo la norma de la matriz inversa en Matlab y al multiplicarla por $\|\Delta A\| = 0.0015$ se tiene que $\|\Delta A\|\|A^{-1}\| \approx 4.8940 > 1$. Por lo tanto, no disponemos de una forma analítica de acotar el error relativo de x tras la perturbación a A y b.
En el siguiente inciso veremos que una condición alternativa para la validez de esta expresión tampoco se cumple.
Este comportamiento se debe a que **la perturbación que estamos ejerciendo es muy grande**, haciendo más grande a $\|\Delta A\|$.

c) Para generar matrices de incrementos arbitrarios $\|\Delta A\|$ y $\|\Delta b\|$ se optó por generar cada una de estas con entradas aleatorias que toman valores de ± 0.0005. Se utilizó la función `randi` de Matlab que permite generar matrices de enteros aleatorios, combinada con operaciones aritméticas para tener solo enteros ± 1 y finalmente

multiplicar cada entrada por 0.0005. En cada ejecución del programa estos valores cambian y se muestran para saber las correspondientes $\|\Delta A\|$ y $\|\Delta b\|$.

Una vez generadas las matrices de perturbación se resolvió el sistema $(A + \Delta A)(x + \Delta x) = (b + \Delta b)$ para $x + \Delta x$, mismo que se muestra para cada ejecución.
Con esto, se calculó el error relativo de esta solución con respecto a la solución original x como $\frac{\|x+\Delta x-x\|}{\|x\|} = \frac{\|\Delta x\|}{\|x\|}$. Esto no es consistente con el inciso b) pues ahí llegamos a la conclusión de que no podíamos acotar este error ya que $\|\Delta A\|\|A^{-1}\| \approx 4.8940$.

Sin embargo, utilizando la expresión

$$\frac{\|\Delta x\|}{\|x\|} \leq \frac{cond(A)}{1 - \|A^{-1}\Delta A\|}\left(\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta \mathrm{b}\|}{\|b\|}\right), \qquad \text{válida para } \|A^{-1}\Delta A\| < 1$$

Podemos ocuparla solo cuando $\|A^{-1}\Delta A\| < 1$ y para esto es necesario realizar la multiplicación de A^{-1} con la matriz generada aleatoriamente de ΔA. Este producto se muestra en la ejecución del programa y si resulta verificarse la condición se calcula la cota superior del error (lado derecho de la desigualdad), misma que corresponde correctamente con acotar el error, aunque de una manera no muy precisa, pues el lado derecho suele distar en gran medida del error (lado izquierdo).

Como último punto, el cambio relativo para cada renglón de la solución perturbada $x + \Delta x$ con respecto a la original x también se muestra en cada ejecución del programa directamente después de $\frac{\|\Delta x\|}{\|x\|}$, resultando en que la primer entrada de la solución perturbada resulta estar increíblemente alejada de la original.

```
%Declaramos A y b del sistema dado Ax=b
A=[0.217,0.732,0.414; 0.508,0.809,0.376; 0.795,0.886,0.338];
b=[0.741;0.613;0.485];
%Resolvemos el sistema para x
[L,U,P,cond_A]=FACTOR(A);
[x]=SOLVE(L,U,P,b);
%Mostramos el resultado x
disp('x:'); disp(x);
%Mostramos cond(A)
disp('cond(A):'); disp(cond(A));
disp('||A||:'); disp(norm(A,'Inf'));
disp('||b||:'); disp(norm(b,'Inf'));
%Mostramos el factor de validez de la desigualdad
factor_val_desigualdad_1=norm(inv(A),'Inf')*0.0015;
disp('Factor ||inv(A)||*||Delta A||:'); disp(factor_val_desigualdad_1);

%Generamos y mostramos incrementos ALEATORIOS de +-0.0005 para A y b
deltaA=0.0005.*(2*randi([0,1],3,3)-1); disp('DeltaA:'); disp(deltaA);
deltab=0.0005.*(2*randi([0,1],3,1)-1); disp('DeltaB:'); disp(deltab);
```

```
%Resolvemos ahora (A+deltaA)(x+deltax)=(b+deltab) para x+deltax
[L,U,P,cond_deltaA]=FACTOR(A+deltaA);
[xmasdeltax]=SOLVE(L,U,P,b+deltab);

%Mostramos el resultado (x+deltax) del sistema perturbado
disp('x+Delta x:'); disp(xmasdeltax);
%Error relativo de la solución perturbada con la original
disp('||Delta x||/||x||:'); disp(norm(xmasdeltax-x,'Inf')/norm(x,'Inf'));
%Error relativo de cada entrada de x+Delta x con respecto a x
errores_rel_fila=zeros(3,1);
for i=1:3
    errores_rel_fila(i)=(xmasdeltax(i)-x(i))/x(i);
end
disp('Errores relativos por filas:'); disp(errores_rel_fila);

%Opcional: factor ||A^-1*DeltaA|| para validez de desigualdad
factor_val_desigualdad_2=norm(inv(A)*deltaA,'Inf');
disp('Factor ||inv(A)*Delta A||:'); disp(factor_val_desigualdad_2);
if factor_val_desigualdad_2<1
    %Si es menor que 1, calculamos la parte derecha de la desigualdad
    disp('Cota superior del error relativo ||Delta x||/||x||')
    aux=(norm(deltaA,'Inf')/norm(A,'Inf'))+(norm(deltab,'Inf')/norm(b,'Inf'));
    disp((cond(A)/(1-norm(inv(A)*deltaA,'Inf')))*(aux));
else
    disp('No se puede calcular la cota por la expresión');
end
```

```
Command Window
  x:
     0.000000000000019
    -0.416021710724738
     2.525429691426338

  cond(A):
       4.436593854304055e+03

  ||A||:
     2.019000000000000

  ||b||:
     0.741000000000000

  Factor ||inv(A)||*||Delta A||:
     4.894009448186260

  DeltaA:
     1.0e-03 *

    -0.500000000000000   0.500000000000000   0.500000000000000
    -0.500000000000000   0.500000000000000   0.500000000000000
    -0.500000000000000  -0.500000000000000   0.500000000000000

  DeltaB:
     1.0e-03 *

     0.500000000000000
    -0.500000000000000
    -0.500000000000000
```

```
DeltaB:
   1.0e-03 *

   0.500000000000000
  -0.500000000000000
  -0.500000000000000

x+Delta x:
  -0.312800984027612
   0.251203936110621
   1.508358334960068

||Delta x||/||x||:
   0.402732002367422

Errores relativos por filas:
   1.0e+13 *

  -1.604297143932496
  -0.000000000000160
  -0.000000000000040

Factor ||inv(A)*Delta A||:
   0.824056856635593

Cota superior del error relativo ||Delta x||/||x||
  35.748964627134363
```

Nota: **Ejecutar varias veces el programa genera distintos resultados que corresponden a los distintos análisis presentados en la solución.**

2.19 Cuenta las operaciones aritméticas requeridas por el algoritmo para un sistema lineal de n ecuaciones con una matriz de coeficientes tridiagonal y m lados derechos. Compara esto con cuanto es requerido para el sistema general.

El algoritmo para la resolución de un sistema tridiagonal $Ax = d$

$$\begin{pmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & b_{n-1} & a_{n-1} & c_{n-1} \\ & & & b_n & a_n \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_{n-1} \\ d_n \end{pmatrix}$$

toma su fundamento en la eliminación Gaussiana y factorización LU. Aplicando este algoritmo, pero sin la realización del pivoteo (sin intercambiar columnas) obtenemos la siguiente matriz:

$$\begin{pmatrix} f_1 & c_1 & & & \\ m_2 & f_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & m_{n-1} & f_{n-1} & c_{n-1} \\ & & & m_n & f_n \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_{n-1} \\ e_n \end{pmatrix}$$

Recordemos que los multiplicadores $m_i, i = 2, \cdots, n$ no se almacenan en la matriz mostrada, sino que conforman L y los coeficientes $f_i, i = 1, \cdots, n$ y $c_i, i = 1, \cdots, n$ conforman U, de manera que se verifica $A = LU$. Algo importante que remarcar es que en la eliminación no se alteraron las entradas a_1 ni e_1, por lo que $\boldsymbol{a_1 = f_1}$ y $\boldsymbol{e_1 = d_1}$.

Entonces, en el k-ésimo paso de la eliminación tendremos que $m_{k+1} = b_{k+1}/f_k$, por lo que

$$\begin{aligned} f_{k+1} &= a_{k+1} - m_{k+1}c_k && \text{(se altera)} \\ c_{k+1} &= c_{k+1} - m_{k+1}(0) = c_{k+1} && \text{(no se altera)} \\ e_{k+1} &= d_{k+1} - m_{k+1}d_k && \text{(se altera)} \end{aligned}$$

Una vez finalizada la eliminación simplemente se resuelven los sistemas con sustitución hacia adelante y hacia atrás respectivamente.

Podemos entonces resumir el algoritmo en tres secciones.

Eliminación	Sustitución hacia adelante	Sustitución hacia atrás
$f_1 = a_1$ for $k = 1, \cdots, n-1$ begin $m_{k+1} = b_{k+1}/f_k$ $f_{k+1} = a_{k+1} - m_{k+1}c_k$ end	$e_1 = a_1$ for $k = 1, \cdots, n-1$ begin $e_{k+1} = d_{k+1} - m_{k+1}e_k$ end	$x_n = e_n/f_n$ for $k = n-1, \cdots, 1$ begin $x_k = (e_k - c_k x_{k+1})/f_k$ end
$\text{Flops} = (n-1)(1+2)$ $= 3n - 3$	$\text{Flops} = (n-1)(2) = 2n - 2$	$\text{Flops} = 1 + (n-1)(2+1)$ $= 3n - 3 + 1$ $= 3n - 2$

De modo que para n ecuaciones y m lados derechos tendremos un conteo total de

$$\text{Flops} = (3n-3) + m(2n-1+3n-2) = 3n - 3 + m(5n-3) = (3+5m)n - 3m - 3$$
$$\boldsymbol{\text{Flops} = (3+5m)n - 3m - 3 \approx 5mn - 3m}$$

2.20 Un análisis de elementos finitos de cierto soporte de carga produce el siguiente sistema de ecuaciones rígido:

$$\begin{pmatrix} \alpha & 0 & 0 & 0 & \beta & -\beta \\ 0 & \alpha & 0 & -\beta & 0 & -\beta \\ 0 & 0 & \alpha & \beta & \beta & 0 \\ 0 & -\beta & \beta & \gamma & 0 & 0 \\ \beta & 0 & \beta & 0 & \gamma & 0 \\ -\beta & -\beta & 0 & 0 & 0 & \gamma \end{pmatrix} x = \begin{pmatrix} 15 \\ 0 \\ -15 \\ 0 \\ 25 \\ 0 \end{pmatrix}$$

Donde $\alpha = 482{,}317.$, $\beta = 2{,}196.05$ y $\gamma = 6{,}708.43$. Aquí x_1, x_2 y x_3 son los desplazamientos laterales y x_4, x_5 y x_6 los desplazamientos rotacionales en tres dimensiones, correspondientes a la fuerza aplicada (lado derecho del sistema).

a) Resuelva para x.
b) ¿Qué tan confiable es el computo? Asuma datos exactos, entonces $\frac{\|\Delta A\|}{\|A\|} = 5 \times 10^{-7}$.

a) Se resolvió el sistema mediante `FACTOR/SOLVE` y se calculó el número de condición del sistema utilizando `cond`, obteniendo $cond(A) \approx 73.65$.
b) Para ver que tan confiable es el cómputo analicemos el error relativo con la expresión

$$\frac{\|\Delta x\|}{\|x\|} \leq \frac{cond(A)}{1 - cond(A)\frac{\|\Delta A\|}{\|A\|}}\left(\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta \mathrm{b}\|}{\|b\|}\right), \qquad \text{válida para } \|A^{-1}\|\|\Delta A\| < 1$$

Como los datos son exactos entonces consideremos $\frac{\|\Delta A\|}{\|A\|} = 5 \times 10^{-7}$ y $\frac{\|\Delta b\|}{\|b\|} = 0$. Observemos que se cumple la condición para la validez de la desigualdad ya que, utilizando el número de condición del inciso anterior

$$\|A^{-1}\|\|\Delta A\| = \frac{\|A\|\|A^{-1}\|\|\Delta A\|}{\|A\|} = cond(A)\frac{\|\Delta A\|}{\|A\|} \approx 3.6825 \times 10^{-5} < 1$$

Por tanto, el error relativo esperado esta acotado por

$$\frac{\|\Delta x\|}{\|x\|} \leq \frac{cond(A)}{1 - cond(A)\frac{\|\Delta A\|}{\|A\|}}\left(\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta \mathrm{b}\|}{\|b\|}\right) = \frac{73.65}{1 - 3.6825 \times 10^{-5}}(5 \times 10^{-7} + 0)$$
$$= 3.6826 \times 10^{-5}$$
$$\therefore \frac{\|\boldsymbol{\Delta x}\|}{\|\boldsymbol{x}\|} \leq \mathbf{3.6826 \times 10^{-5}}$$

Por lo tanto, podemos concluir que el cómputo de la solución es confiable dado que su error relativo tiene un orden de 10^{-5}, añadido al pequeño número de condición de A.

```
format long;
%Introducimos el sistema de ecuaciones rigido
a=482317.0;
b=2196.05;
c=6708.43;
A=[a,0,0,0,b,-b;0,a,0,-b,0,-b;0,0,a,b,b,0;
    0,-b,b,c,0,0;b,0,b,0,c,0;-b,-b,0,0,0,c];
b=[15;0;-15;0;25;0];
%Resolviendo Ax=b para x
[L,U,P,condicion]=FACTOR(A);
[x]=SOLVE(L,U,P,b);
%Mostramos la solución x junto con cond(A)
disp('x:'); disp(x);
disp('Número de condición de A:'); disp(condicion);
```

Command Window

```
>> ejercicio_2_20
x:
   0.000014102301710
   0.000000093124932
  -0.000048190579813
   0.000015805997798
   0.003737813402410
   0.000004646969064

Número de condición de A:
  73.650002921707596
```

2.21 Wang considers a statically indeterminate pin-jointed truss. With this problem is associated a statics matrix A that defines the configuration of the framework, a member stiffness matrix S that relates the elastic properties of the constituent members, and an external force vector p that describes the applied forces at the joints. A displacement vector bfx that accounts for the displacement at each degree of freedom and an internal force vector f acting on each member satisfies

$$Kx = p, K = ASA^T, f = (SA^T)x$$

Como ejemplo

$$A = \begin{pmatrix} 0.6 & -1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.6 & 0.0 & 0.0 \\ 0.8 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.8 & 0.0 & 0.0 \\ 0.0 & 1.0 & -0.6 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.6 & 0.0 \\ 0.0 & 0.0 & 0.8 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.8 & 1.0 \\ 0.0 & 0.0 & 0.0 & 1.0 & -1.0 & 0.0 & 0.0 & 0.0 & -0.6 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -1.0 & 0.0 & -0.8 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & -1.0 & 0.0 & 0.6 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.6 & 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{pmatrix}$$

La matriz S tiene todas sus entradas iguales a cero excepto a lo largo de la diagonal, donde tiene {4800,10000,4800,10000,10000,10000,3000,4800,4800,3000}.

Escribe un programa para computar productos matriciales y determina los elementos de K. Resuelve para x usando los tres vectores p

$$p = \begin{pmatrix} 0.0 \\ -1.0 \\ 0.0 \\ 0.0 \\ 0.0 \\ 0.0 \\ 0.0 \\ 0.0 \end{pmatrix}, \begin{pmatrix} 0.0 \\ 0.0 \\ 0.0 \\ -1.0 \\ 0.0 \\ 0.0 \\ 0.0 \\ 0.0 \end{pmatrix}, \begin{pmatrix} 0.0 \\ 0.0 \\ 0.0 \\ 0.0 \\ 0.0 \\ -1.0 \\ 0.0 \\ 0.0 \end{pmatrix}$$

Encuentra los vectores f correspondientes.

Se computaron las matrices A y S en Matlab y se realizaron los cálculos correspondientes para obtener tres vectores f_1, f_2, f_3 a través de

$$f_i = SA^T x_i, \quad Kx_i = p_i, \quad K = ASA^T, \qquad i = 1,2,3,4$$

```
%Introducimos las matrices A y S
A=[0.6,-1.0,0.0,0.0,0.0,0.0,0.0,-0.6,0,0;
    0.8,0.0,0.0,0.0,0.0,0.0,1.0,0.8,0.0,0.0;
    0.0,1.0,-0.6,0.0,0.0,0.0,0.0,0.0,0.6,0.0;
    0.0,0.0,0.8,0.0,0.0,0.0,0.0,0.0,0.8,1.0;
    0.0,0.0,0.0,1.0,-1.0,0.0,0.0,0.0,-0.6,0.0;
    0.0,0.0,0.0,0.0,0.0,0.0,0.0,-1.0,0.0,-0.8,0.0;
    0.0,0.0,0.0,0.0,0.0,1.0,-1.0,0.0,0.6,0.0,0.0;
    0.0,0.0,0.6,0.0,0.0,1.0,0.0,0.0,0.0,0.0];
S=eye(10); S(1,1)=4800; S(2,2)=10000; S(3,3)=4800;
S(4,4)=10000; S(5,5)=10000; S(6,6)=10000; S(7,7)=3000;
S(8,8)=4800; S(9,9)=4800; S(10,10)=3000;
%Calculamos K como K=ASA'
K=A*S*A';
%Resolvemos el sistema Kx=p con tres lados derechos distintos p1,p2,p3
p1=zeros(8,1); p1(2)=-1.0;
p2=zeros(8,1); p2(4)=-1.0;
p3=zeros(8,1); p3(6)=-1.0;
%Aplicando FACTOR/SOLVE
[L,U,P,cond]=FACTOR(K);
[x1]=SOLVE(L,U,P,p1);
[x2]=SOLVE(L,U,P,p2);
[x3]=SOLVE(L,U,P,p3);
%Encontrando los vectores f correspondientes f=(SA')x e imprimiendolos
f1=S*A'*x1; disp('f1:'); disp(f1);
f2=S*A'*x2; disp('f2:'); disp(f2);
f3=S*A'*x3; disp('f3:'); disp(f3);
```

Command Window

```
>> ejercicio_2_21
f1:
   -0.6190
   -0.0922
    0.0120
    0.3714
    0.2720
   -0.0072
   -0.1325
   -0.4654
    0.1656
   -0.1421
```

Command Window

```
f2:
   -0.1979
   -0.0863
   -0.3957
    0.1187
    0.2699
    0.2374
    0.2015
   -0.0541
   -0.2519
   -0.4819
```

Command Window

```
f3:
   -0.6409
   -0.2959
   -0.0318
    0.3845
    0.1077
    0.0191
    0.6309
   -0.1478
    0.4613
   -0.3437
```

2.23 Ocasionalmente es deseable computar el determinante de una matriz A con n filas y columnas. Usando la factorización $PA = LU$ se puede demostrar que

$$\det A = (-1)^{\text{Número de intercambios de filas}} \times \text{Producto de los pivotes}$$

Usa esta fórmula para calcular el determinante de las siguientes matrices

a) $A_{13} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9.01 \end{pmatrix}$ b) $A_{15} = \begin{pmatrix} 11 & -5 & 0 & 0 & 0 & -1 \\ -20 & 41 & -15 & 0 & -6 & 0 \\ 0 & -3 & 7 & -4 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & -3 & 0 & -10 & 28 & -15 \\ -2 & 0 & 0 & 0 & -15 & 47 \end{pmatrix}$

Para este ejercicio se utilizó la modificación de la función `FACTOR` para devolver a la matriz de permutaciones P como un vector columna de n filas que contiene toda la información de los intercambios. Por ejemplo, si $P = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$ entonces como vector $P = \begin{pmatrix} 2 \\ 1 \\ 4 \\ 3 \end{pmatrix}$. Esta modificación, como se dijo al principio, se realiza únicamente especificando a Matlab un parámetro adicional en la llamada a la función `lu`.

Los pivotes a los que hace referencia el ejercicio son las entradas de la diagonal principal de la matriz U, en otras palabras, los números usados para la eliminación con pivoteo en la descomposición.

Para recuperar el número de intercambios hechos durante la eliminación es necesario restaurar el vector P a su forma original, es decir, $P_i = i, i = 1, \cdots, n$. Ilustrando con un ejemplo:

Si $P = \begin{pmatrix} 3 \\ 1 \\ 4 \\ 2 \end{pmatrix} \Rightarrow \begin{pmatrix} 4 \\ 1 \\ 3 \\ 2 \end{pmatrix} \Rightarrow \begin{pmatrix} 2 \\ 1 \\ 3 \\ 4 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$, por lo tanto, el número de intercambios hechos es **3**

Vale la pena mencionar lo interesante que es este algoritmo para recuperar el número de intercambios, como resulta inmutable la modificación de la función `lu` de Matlab para que nos dé el número de intercambios se debe recurrir a una estrategia de este estilo, sin embargo, no se asegura que sea la forma más optima de hacerlo.

Dicho todo esto, se calcularon los determinantes requeridos y se corroboró su valor utilizando la función `det` de Matlab que, según la documentación, realiza también una descomposición LU de la matriz, aplicando un procedimiento similar al aquí expuesto, obteniendo

$$\mathbf{\det A_{13} = -0.03} \qquad \mathbf{\det A_{15} = 1,500,000}$$

```
%Declaramos las matrices a computar
A_13=[1,2,3;4,5,6;7,8,9.01];
A_15=[11,-5,0,0,0,-1;-20,41,-15,0,-6,0;0,-3,7,-4,0,0;
    0,0,-1,2,-1,0;0,-3,0,-10,28,-15;-2,0,0,0,-15,47];
%Se ha hecho la modificación adecuada en la función FACTOR

%Obteniendo el determinante de A_13
[L,U,P]=FACTOR(A_13);
prod_pivotes=prod(diag(U));
%Recuperar el número de intercambios a partir de la info que da P
no_intercambios=0;
for i=1:length(A_13)
    %Mientras i no esté en su posición adecuada
    while P(i)~=i
        %Intercambia las posiciones que guarda
        aux=P(i); P(i)=P(aux); P(aux)=aux;
        no_intercambios=no_intercambios+1;
    end
end
determinante=prod_pivotes*((-1)^no_intercambios);
disp('Determinante de A_13:'); disp(determinante); disp(det(A_13));

%Obteniendo el determinante de A_13
[L,U,P]=FACTOR(A_15);
prod_pivotes=prod(diag(U));
%Recuperar el número de intercambios a partir de la info que da P
no_intercambios=0;
for i=1:length(A_15)
    %Mientras i no esté en su posición adecuada
    while P(i)~=i
        %Intercambia las posiciones que guarda
        aux=P(i); P(i)=P(aux); P(aux)=aux;
        no_intercambios=no_intercambios+1;
    end
end
determinante=prod_pivotes*((-1)^no_intercambios);
disp('Determinante de A_15:'); disp(determinante); disp(det(A_15));
```

Command Window

```
>> ejercicio_2_23
Determinante de A_13 calculado por este método y por Matlab:
  -0.030000000000000

  -0.030000000000000

Determinante de A_15 calculado por este método y por Matlab:
     1.499999999999999e+06

     1.499999999999999e+06
```