Práctica Final Algebra Lineal: Refinamiento Iterativo

Elena del Pilar Fernández Wyzynska (hecho individualmente)

Introducción a los contenidos del trabajo

En esta sección se explicarán los diferentes aspectos fuera del problema en sí, para comprender en su totalidad el ejercicio tanto a nivel algebraico como a nivel de programación.

- **Matriz de Vandermonde:**
  Se denomina una Matriz de Vandermonde, cuando sigue un esquema geométrico en cada fila, esta comienza con 1 en la primera posición siendo la siguiente un número elevado a 1, siguiente a ^2 y así consecutivamente…

- **Gauss Triangular Superior e Inferior**
  Este sistema/ código nos sirve para encontrar las soluciones de una matriz triangulada tanto superiormente o inferiormente respectivamente con un vector, este nos devuelve un vector de soluciones.

- **Factorización LU**
  Teniendo una matriz ampliada (A|b) podemos encontrar la solución x aunque esta es la manera menos precisa comparada con las otras dos (factorización por columnas PA =LU y pivotaje maximal PAQ=LU).
  Lo que nos propone la Factorización LU es triangulación la matriz A superiormente, dándonos como resultado L (siendo una matriz que parte de la Identidad donde se colocan todos los -a de (i,j;a)provocados en cada posición correspondientemente) y U siendo la matriz A totalmente triangulada superiormente.

- **Resolución de una matriz ampliada con la Factorización LU**
  Teniendo la A=LU y un vector b procedemos a realizar en este orden estas operaciones:
  1. Lc=b;
  2. Ux=c;

  x es la solución de Ax=b mediante la Factorización LU

- **Dibujar gráfica en el código**
  La gráfica ha sido implementada en el código con las funciones plot() y hold con esta última para poder acumular más de una línea en la gráfica.

Método de Refinamiento

El método de Refinamiento es usado para poder conseguir resultados más exactos, de aquellos métodos que nos dan resultados con pérdida de precisión, en nuestro caso tenemos el método de Factorización LU el cual vamos a intentar subir la precisión de los resultados en base al método de refinamiento.

Teniendo una matriz A, un vector b y una solución exacta
Resolvemos con el método de Factorización LU obteniendo la solución x.
Ahora aplicamos la fórmula r= b-Ax, si el método de factorización no perdiese precision r sería igual a 0, pero supongamos que r es distinto de 0,en ese caso volvemos a resolver

con la Factorización pero esta vez el sistema Ae=r, con la e devuelta le añadimos el error a x para tener una solución más exacta a la anterior x2 = x + e.
El método de refinamiento tiene como finalidad mejorar el resultado obtenido, no que r=0, por tanto el trabajo finaliza cuando ya no hay una mejora entre xn y xn+1.

Código Implementado

```
b=[1; 2; -3; 4; -3; 2; 1];
x0=[-(729857845599855204562360795095096734135/161522902359886581763421436897166183618486056);
    (1141920845551633879786052007720493144881679/585227907101038339722541438033210810211190600);
    -(3602541025701077010017638143991772358920603 51/623640549652071744260314428174386809337784 00)
    (1325202990256272231817290041849909246806604584 9377/1346024186332388181361845307476384863487383 80000);
    (3823104946524030429816556524407289735121134688 2671/2019036279498582272042767961214577295231075 70000);
    -(2212502000088855754980314852695301655663528 69/3511489581374277839303572230711637439965 00);
    (8405881176031813838315838200925760183294735/673012093166194090680922653738192431743691 9)]
unos=[1, 1, 1, 1,1, 1, 1];
a=0;    %Numero de iteraciones hechas
X=[];  % guardar las soluciones
R= []; %guardar las bondades de r
E = []; % guardar las mejoras e
mj=1;   %mejora

[L,U]=LUFact(A)
c = GaussTrianguloInferior(L,b)
[x1]=GaussTrianguloSuperior(U,c)
X=[X,x1];


plot(x1);
hold on
%usamos la formula
while mj !=0

r1=b-A*x1;
%resolvemos otra vez pero esta vez A*e1=r1
[L,U]=LUFact(A)
c = GaussTrianguloInferior(L,r1)
[e1]=GaussTrianguloSuperior(U,c)


x2 = x1+e1  %de esta manera x2 es mas preciso
X=[X,x2];
plot(x2);
hold on
a=a+1;
mj = unos*x2-unos*x1;
x1=x2;    %llamamos a x1 x2 para repetir el bucle

% guardamos soluciones
bondr= norm(r1);
R=[R,bondr];
E=[E,e1];

endwhile
```

*Para una mejor visualización del código por favor descargar el archivo

Lo que se hace en este código es que después de la primera resolución de la Factorización LU, entre en bucle el método de refinamiento hasta que las soluciones no muestran un cambio.
Dentro del bucle se realiza la

1. fórmula de r =b-Ax.
2. Resolución Ae=r.
3. x2=x+e.

Luego de esto lo que hacemos es llamar a x1 = x2 para que el bucle continúe y nos aproxime la solución.

Guardamos las diferentes soluciones en cada iteración y llamamos a plot() para que nos muestre una gráfica.

- Resultados obtenidos
  x1 = **(sol final x)**

  1.3476e-05
  -5.8205e-02
  1.7281e+01
  -3.0883e+02
  -2.7485e+02
  4.8126e+01
  8.1191e-01

  a = 11 **(número de iteraciones)**

  Matriz X donde estan todas las x guardadas

```
X =

  -5.4578e-03   8.1499e-06   1.3620e-05   1.3476e-05   1.3476e-05   1.3476e-05   1.3476e-05   1.3476e-05   1.3476e-05   1.3476e-05   1.3476e-05   1.3476e-05
   3.2974e+00  -3.9283e-02  -5.8094e-02  -5.8204e-02  -5.8205e-02  -5.8205e-02  -5.8205e-02  -5.8205e-02  -5.8205e-02  -5.8205e-02  -5.8205e-02  -5.8205e-02
  -4.5045e+01   1.6914e+01   1.7279e+01   1.7281e+01   1.7281e+01   1.7281e+01   1.7281e+01   1.7281e+01   1.7281e+01   1.7281e+01   1.7281e+01   1.7281e+01
  -3.5931e+02  -3.0912e+02  -3.0883e+02  -3.0883e+02  -3.0883e+02  -3.0883e+02  -3.0883e+02  -3.0883e+02  -3.0883e+02  -3.0883e+02  -3.0883e+02  -3.0883e+02
  -2.6009e+02  -2.7477e+02  -2.7485e+02  -2.7485e+02  -2.7485e+02  -2.7485e+02  -2.7485e+02  -2.7485e+02  -2.7485e+02  -2.7485e+02  -2.7485e+02  -2.7485e+02
   4.7678e+01   4.8123e+01   4.8126e+01   4.8126e+01   4.8126e+01   4.8126e+01   4.8126e+01   4.8126e+01   4.8126e+01   4.8126e+01   4.8126e+01   4.8126e+01
   8.1347e-01   8.1192e-01   8.1191e-01   8.1191e-01   8.1191e-01   8.1191e-01   8.1191e-01   8.1191e-01   8.1191e-01   8.1191e-01   8.1191e-01   8.1191e-01
```

matriz R con las bondades de r

```
>> R
R =

   1.8990e+19   2.5333e+15   7.0470e+14   2.4547e+12   3.1395e+10   2.3048e+08   3.1830e+06   1.6732e+03   7.7572e+03   6.7773e+03   4.7420e+02
```

Matriz E donde se muestran todos los incrementos e

```
>> E
E =

   8.1140e+01   4.7740e-01   2.8067e-03   1.6501e-05   9.7011e-08   5.7028e-10   3.4071e-12   7.5450e-14   5.3413e-14   8.0017e-14   2.6499e-14
```

Error cometido comparación ||x-x1||

- **Factorización LU con pivotaje maximal**
  añadimos estas lineas de código :

```
err = norm(x0 - x1);
[P3,Q3,L3,U3] = PAQ_LUFact(A);
c3 = GaussTrianguloInferior(L3,P3*b)
[z3]=GaussTrianguloSuperior(U3,c3)
sol3= Q3*z3;
errpiv = norm(x0 - sol3);
```

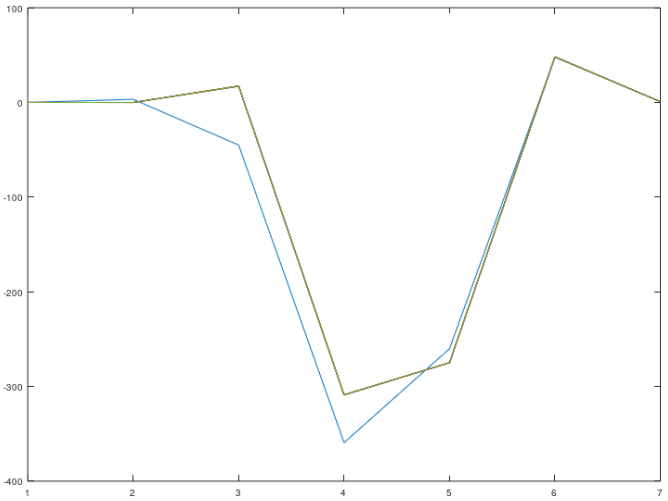
  ```
  >> errpiv
  errpiv = 627.89
  ```

- **Factorización LU método de refinamiento**

```
>> err
err = 627.89
```

- Gráfica obtenida

La gráfica nos muestra como ambas soluciones tanto x1 y x2 se asemejan pero presentan ciertas variaciones, estas se deben por el ajuste de e acercando la precisión más.



- Conclusiones obtenidas

El método de Refinamiento iterativo, nos propone una mejora de precisión en el resultado de aquellos métodos que pierden precisión, tal como puede ser la factorización LU, esto lo hemos podido comprobar con los resultados obtenidos que hemos ido guardando en la matriz X.
Pero, ¿Cómo de preciso puede llegar a ser? Los resultados nos muestran que llega a ser tan preciso como el método de factorización LU con pivotaje máximal,cuya precisión es la mayor de los tres métodos.
La iteración del refinamiento nos permite llegar a soluciones precisas sin necesidad de hacer cálculos complejos, ni cambios de fila ni de columna, un método que puede resultar más intuitivo, pero cuya finalidad no está muy clara si desde el principio se conseguía el mismo resultado con la factorización LU con pivotaje maximal. Al no haber una mejora en el resultado, quizá no sea muy útil realizar este proceso que en este caso nos ha hecho hacer 11 iteraciones, lo mismo a hacer 11 Factorizaciones LU cuyo resultado se obtenía con 1 Factorización de pivotaje maximal.