Gradientes Conjugados con Precondicionamiento

Marco teorico

Se propone resolver el sistema de ecuaciones $A.x = b$ con $A \in \mathbb{R}^{nxn}$, x y b $\in \mathbb{R}^n$ se propone usar para esto el método de Gradientes conjugados con matriz de precondicionamiento. La matriz de precondicionamiento es tal que disminuya el número de condicion κ disminuya. Este está definido como $\kappa = \frac{\lambda_{max}}{\lambda_{min}} \geq 1$ donde λ son los autovalores de la matriz λ_{max} el mayor y λ_{min} el menor de estos. La matriz de precondicionamiento se aplica de la siguiente manera:

$$M^{-1}.A.\,x = M^{-1}.\,b \qquad (1)$$

Si se logra que $\kappa\left(M^{-1}.\,A\right) \geq \kappa\,(A)$ entonces la convergencia del sistema (1) será más rápida que la del sistema original.

resultados

Se compararon los números de condición de la matriz A y de la matriz $M^{-1}.A$ y se obtuvieron los siguientes resultados:

Luego se analizó el tiempo de resolución y el número de iteraciones para el método y se comparó con el tiempo para la misma matriz sin precondicionamiento y se obtuvieron los siguientes resultados:

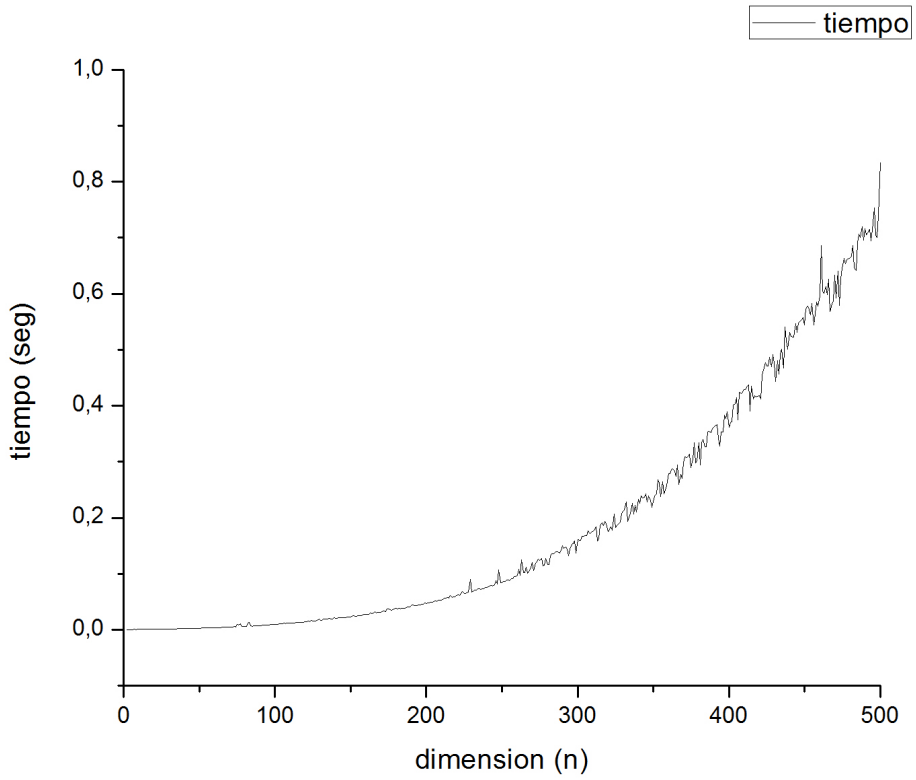

Figura 1: Tiempo de resolución en función de la dimensión n de la matriz. Puede verse que el incremento de tiempo no es lineal con la dimensión

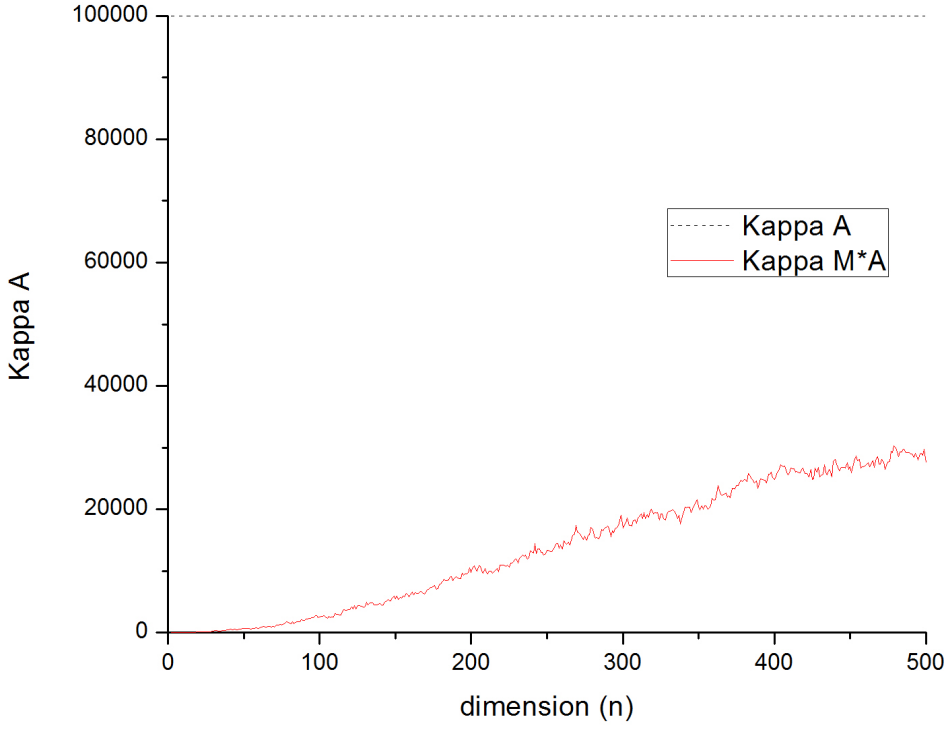

Figura 2: número de condición κ para la matriz con condiciónamiento (línea rayada) y sin condicionamiento (línea entera) Puede verse que el número de condición es mucho menor para la matriz a la cual se le aplica el condicionamiento.

Codigo del programa

A continuación se presentará el código del programa utilizado para resolver el sistema y medir los tiempos:

```
for n=2:500
tol=1E-10;
rand("seed", 1.058);
v = 5/(n-1) * (0:n-1);
D = (10 .** v);
A = eye(n) + 0.01*rand(n,n); % A = I + pequena perturbacion.;
[Q R] = qr(A); % Descomposicion QR (Q unitaria, R triangular);
A = Q'*diag(D)*Q;
b = ones (1,n)/(n + 1)**2;
x=ones(n,1);
M=zeros(n);
C=zeros(n);
for l=1:n
        M(l,l)=1/A(l,l);
        C(l,l)=sqrt(M(l,l));
endfor
l1=eig(A);
l2=eig(M*A);
```

```
kapa1(n)=max (l1)/min(l1);
kapa2(n)=max (l2)/min(l2);
t1=time;
k=0;
r= b' - A*x;
d=M*r;
while(norm(r)>tol && k<=n)
        k=k+1;
        r1=r;
        alfa=(r'*M*r)/(d'*A*d);


        x=x+alfa*d;
        r=r-alfa*A*d;
        bet=(r'*M*r)/(r1'*M*r1);
        d=M*r+bet*d;
endwhile
T(n)=time-t1;
it(n)=k;
endfor

q(:,1)=T;
q(:,2)=it;
q(:,3)=kapa1;
q(:,4)=kapa2;
save Datos2P4E3.dat q;
printf("fin del programa")
```