Integrantes:

- Juan Jose Bolaños
- David Saavedra
- David Duarte

5) Dado el sistema lineal de la forma AX=b donde la matriz de coeficientes inicialmente está dado por:

$$\text{Si } A = \begin{bmatrix} 5 & 1 & -2 & 0 \\ 1 & 2 & 0 & 0 \\ -2 & 0 & 4 & 1 \\ 0 & 0 & 1 & 3 \end{bmatrix} \quad b = \begin{bmatrix} 1 \\ 5 \\ 14 \\ 15 \end{bmatrix}$$

Código en R:

```
library(pracma)
library(Matrix)
library(Rlinsolve)
N = 4
A = matrix(c(5, 1, -2, 0, 1, 2, 0, 0, -2, 0, 4, 1, 0, 0, 1, 3), nrow=4, byrow=TRUE)
b = matrix(c(1,5,14,15), nrow = 4, ncol = 1, byrow = TRUE)
```

¿Es diagonalmente dominante?

Respuesta: Una matriz se dice matriz diagonalmente dominante, si en cada uno de los renglones,el valor absoluto del elemento de la diagonal principal es mayor que la suma de los valores absolutos de los elementos restantes del mismo renglón.
Por lo tanto:

|Cij| > Para todo $\sum_{j=1}^{n}$ |Cij| para todo i =1,...,n con $j \neq i$

Fila 1: |5| > |1| + |-2| + |0| = 3 VERDADERO
Fila 2: |2| > |1| = 1 VERDADERO
Fila 3: |4| > |-2| + |0| + |1| = 3 VERDADERO
Fila 4: |3| > |0| + |0| + |1| = 1 VERDADERO

Por lo tanto la matriz si es diagonalmente dominante.

Código en R:

```
Diago = function(matriz){
  matriz[col(matriz) != row(matriz)] = 0
```

```
  return (matriz)
}
cat("Matriz Triangular Inferior")
Diago(A) - tril(A)
cat("Matriz Triangular Superior")
Diago(A) - triu(A)
cat("Matriz Diagonal")
Diago(A)
```

a. Calcule el radio espectral ρ(λ) de la matriz de transición por el método de Gauss-Seidel.
Sea el sistema Ax = b convergente, entonces la ecuación matricial iterativa está dada por:

$$X^{k+1} = N^{-1}PX^{K} + N^{-1}b$$

Para resolver un sistema se realizan las iteraciones siendo el método convergente si y sólo si el radio espectral ρ(λ) < 1, .

Radio espectral:
El radio espectral de T se denota y se define p(T) = max{|λ|}, tal que λ es valor propio de T (λ satisface det(T-λI) = 0)

Código en R:

```
r = max(abs(eig(A)))
cat("El radio espectral es:", r)
```

b. Utilice el método de Gauss-Seidel para aproximar la solución (utilice 16 cifras significativas), determine el número máximo de iteraciones.

Descomposición de Cholesky
La descomposición de Cholesky lleva el nombre gracias a el matemático André-Louis Cholesky, esta descomposición consiste en tener una matriz de coeficientes de un sistema de ecuaciones, la llamamos A, comprobamos que esta matriz sea simétrica, porque de lo contrario no es factorizable por Cholesky. Cuando A es simétrica podemos tratar de factorizar en la forma A = L*L(t), L(t) quiere decir la matriz L traspuesta, cuando la tenemos factorizada ya podemos resolver el sistema de ecuaciones.
solución por descomposición de Cholesky:

Código en Python:

```
import  numpy as np

def cholesky(a):
    a = np.array(a, float)
    L = np.zeros_like(a)
    n,_ = np.shape(a)
    for j in range (n):
      for i in range (j,n):
        if i == j:
          L[i,j] = np.sqrt(a[i,j]-np.sum(L[i,:j]**2))
        else:
          L[i,j] = (a[i,j] - np.sum(L[i,:j]*L[j,:j]))/L[j,j]
    return L

def solveLU (L,U,b):
  L = np.array(L,float)
  U = np.array(U,float)
  b = np.array(b,float)
  n,_ = np.shape(L)
  y = np.zeros(n)
  x = np.zeros(n)

  for i in range(n):
    sumj = 0
    for j in range(i):
      sumj += L[i,j] * y[j]
    y[i] = (b[i]-sumj)/L[i,i]




  for i in range(n-1,-1,-1):
      sumj = 0
      for j in range(i+1,n):
            sumj += U[i,j] * x[j]
      x[i] = (y[i]-sumj)/U[i,i]



  return x



a = [[5,1,-2,0],
    [1,2,0,0],
    [-2,0,4,1],
    [0,0,1,3]]
```

```
b = [1,5,14,15]


l = cholesky(a)
x = solveLU(l,np.transpose(l),b)
#SOLUCION CON cholesky
print(x)

#SOLUCION SIN cholesky
print(np.linalg.solve(a,b))
```

c. Qué pasa con la solución anterior si $a_{13} = -2$, explique su respuesta.
Respuesta: si igualamos a(13) =0, no vamos a obtener una solución correcta ya que tenemos como condición necesaria y suficiente para poder usar este métodos, tener una matriz A que sea simétrica y definida positiva.
Para los valores iniciales del enunciado cumplimos con esos requisitos sin embargo al cambiar a(13) =0, la matriz A no cumple con el requisito de ser definida positivamente. por ende, la función genera resultados inexactos.

d. Evalúe la matriz de transición del método SOR y determine varias soluciones aproximadas, para 15 valores de ω. Utilice 10 cifras significativas.

La matriz **A** se re-escribe como la suma de las siguientes matrices:

$$A = L + D + S - D$$

D es una matriz diagonal con elementos iguales a los de la diagonal principal de **A**
L es una matriz triangular inferior con ceros en la diagonal principal y los otros elementos iguales a los elementos respectivos de la matriz **A**
S es una matriz triangular superior con todos sus elementos iguales a los elementos respectivos de la matriz **A**

Entonces:

Matriz de transición para el método de Relajación

$$X^{(k+1)} = X^{(k)} + \omega(D^{-1}B - D^{-1}LX^{(k+1)} - D^{-1}SX^{(k)}), \quad k = 0,1,2,...$$

De donde se obtiene:

$$T = -(I + \omega D^{-1}L)^{-1}(I - \omega D^{-1}S)$$

Se tomó k como x0.

Código en R :

```
A <- matrix(c(5,1,2,0,1,2,0,0,2,0,4,1,0,0,1,3), nrow = 4, ncol = 4)
B <- c(1,5,14,20)

Nmax = 100
x0 = 1
tol = 1e-10
i = 1
error = tol+1

w = 1
S <-A
L <-A
D <-solve(A)
I <- diag(4)
S[lower.tri(S,diag = TRUE)] <-0
L[upper.tri(L,diag = TRUE)] <-0

P <- (-1*(I+w*D*L))
Pr <- solve(P)


Tw<- Pr*(I-w*D*S)
Cw<- (w*(D-w*L))*(-1*B)

while(i <= Nmax || error > tol){

x = Tw*x0+Cw
error = sqrt(x*x-x0*x0) //Normalización
i = i+1
x0=x
}
if(error < tol){
  cat("La solución es ", x)
}
else{
cat("Diverge")
}
```

e. Genere una tabla que tenga 20 iteraciones del método de Jacobi con vector inicial de $x_0 = [1, 5, 14, 15]$ y calcular el error en cada iteración, (utiliza entre 5 y 9 cifras significativas).

Método de Jacobi

Este método inicia de una aproximación lineal, con base en esta se recalcula los valores de x al despejar de la ecuación, esto se desarrolla para cada una de las incógnitas de x y cada proceso se realiza con los valores de la aproximación anterior. Se define matricialmente de la siguiente manera:

$$X(k+1) = Tj\ Xk + Cj$$

Donde:

$$Tj = D - 1(L+U)\ y\ Cj = D - 1b$$

- D: matriz diagonal con los elementos de A
- L: inversos aditivos de los elementos por debajo de la diagonal principal de A
- U: inversos aditivos de los elementos por encima de la diagonal principal de A

El método se itera hasta que || X(k+1)- Xk||<tolerancia.

Código en R:

```
x0 = c(1, 5, 14, 15)
cont = 0

while (cont < 20){
  x1 = itersolve(A, b, x0, nmax=cont, tol = 1e-9, method="Jacobi")
  x2 = itersolve(A, b, x0, nmax=cont + 1, tol = 1e-9, method="Jacobi")
  err = (x2$x - x1$x)
  cat("Iteración: ", cont+1, " Error: ")
  print(err)
  cont = cont + 1
}
```

REFERENCIAS

1. https://es.slideshare.net/miguelmosciaro/metodo-de-cholesky
2. https://www.metodosnumericosunitec.com/post/la-vida-bajo-el-microscopio-lo-que-no-se-puede-ver
3. https://sites.google.com/site/procesosnumericosproyectos/practica02/marco-teorico-1/gauss-seidel
4. https://www.youtube.com/watch?v=qNKyw5ED7eM