SOLUTION OF LINEAR EQUATIONS BY GAUSS JACOBI & GAUSS-SIEDEL METHOD

SCILAB ALGORITHM

```
// gauss Jacobi method and gauss siedel method
clc;clear;
a=[3 -1 1;-1 4 1;-1 1 5]
b=[20;6;7]
n=length(b)
c=0
for i=1:n
   s=0
   for j=1:n
      if j~=i
         s=s+a(i,j)
      end
   end
   if (a(i,i)>s)
      c=c+1
   else break
   end
end
if c==n then
   disp('matrix is diagonally dominant')
else
   disp('matrix is diagonally not dominant')
   break
end
x=input('Initial guess of first value ');
y=input('Initial guess of second value ');
z=input('initial guess of third value ');
n=input('Enter no. of iterations ')
for i=1:n
   x(i+1)=(b(1,1)-(a(1,2)*y(i))-(a(1,3)*z(i)))/a(1,1)
   y(i+1)=(b(2,1)-a(2,1)*x(i+1)-a(2,3)*z(i))/a(2,2)
   z(i+1)=(b(3,1)-a(3,1)*x(i+1)-a(3,2)*y(i+1))/a(3,3)
   if (abs(x(i+1)-(x(i)))<0.001)& (abs(y(i+1)-(y(i)))<0.001) & (abs(z(i+1)-(z(i)))<0.001)
      break
   end
end
disp("No. of iteration",i,x(i),y(i),z(i))
iter = 0:i

scf(0)
plot(iter, x(1:i+1), 'ro-')
plot(iter, y(1:i+1), 'gs-')
plot(iter, z(1:i+1), 'b^-')
legend("x","y","z")
xlabel("Iteration")
ylabel("Value")
title("Convergence of Gauss-Seidel Method")
xgrid()
```

Python

```
#Gauss-jacobi & Gauss-Siedel
import numpy as np
import matplotlib.pyplot as plt
A = np.array([[3, -1, 1],
              [-1, 4, 1],
              [-1, 1, 5]], dtype=float)
b = np.array([20, 6, 7], dtype=float)
n = len(b)
c = 0
for i in range(n):
    s=0
    for j in range(n):
       if j != i:
          s=s+A[i,j]
    if (A[i,i]>s):
        c += 1
    else:
        break
if c == n:
    print("Matrix is diagonally dominant")
else:
    print("Matrix is diagonally not dominant")
x0 = float(input("Initial guess of first value: "))
y0 = float(input("Initial guess of second value: "))
z0 = float(input("Initial guess of third value: "))
max_iter = int(input("Enter number of iterations: "))
x_vals = [x0]
y_vals = [y0]
z_vals = [z0]
for i in range(max_iter):
    x_new = (b[0] - (A[0,1]*y_vals[-1]) - (A[0,2]*z_vals[-1])) / A[0,0]
    y_new = (b[1] - A[1,0]*x_new - A[1,2]*z_vals[-1]) / A[1,1]
    z_new = (b[2] - A[2,0]*x_new - A[2,1]*y_new) / A[2,2]
    x_vals.append(x_new)
    y_vals.append(y_new)
    z_vals.append(z_new)
    if (abs(x_vals[-1] - x_vals[-2]) < 0.001 and
        abs(y_vals[-1] - y_vals[-2]) < 0.001 and
        abs(z_vals[-1] - z_vals[-2]) < 0.001):
        break
print(f"No. of iterations = {i+1}")
print(f"x = {x_vals[-1]:.4f}, y = {y_vals[-1]:.4f}, z = {z_vals[-
1]:.4f}")
iterations = list(range(len(x_vals)))
plt.figure(figsize=(8,6))
plt.plot(iterations, x_vals, marker='o', label='x')
plt.plot(iterations, y_vals, marker='s', label='y')
plt.plot(iterations, z_vals, marker='^', label='z')
plt.xlabel("Iteration")
plt.ylabel("Value")
plt.title("Convergence of Gauss-Seidel Method")
plt.legend()
plt.grid(True)
plt.show()
```