In [ ]:
```
Computational Mathematic Lab 4 Iterative methods of solving SLAE
Nurshanov Dias
IT3-2208
```

In [1]:
```
#1) code implementation

def gauss_elimination(A, b):
    n = len(b)
    for i in range(n):
        max_row = i
        for k in range(i + 1, n):
            if abs(A[k][i]) > abs(A[max_row][i]):
                max_row = k

        A[i], A[max_row] = A[max_row], A[i]
        b[i], b[max_row] = b[max_row], b[i]

        for k in range(i + 1, n):
            factor = A[k][i] / A[i][i]
            b[k] -= factor * b[i]
            for j in range(i, n):
                A[k][j] -= factor * A[i][j]

    x = [0] * n
    for i in range(n - 1, -1, -1):
        x[i] = b[i] / A[i][i]
        for k in range(i - 1, -1, -1):
            b[k] -= A[k][i] * x[i]
    return x

def jacobi_method(A, b, tol=1e-4, max_iterations=1000):
    n = len(b)
    x = [0.0 for _ in range(n)]
    x_new = [0.0 for _ in range(n)]

    for _ in range(max_iterations):
        for i in range(n):
            sum_Ax = sum(A[i][j] * x[j] for j in range(n) if i != j)
            x_new[i] = (b[i] - sum_Ax) / A[i][i]

        if all(abs(x_new[i] - x[i]) < tol for i in range(n)):
            break

        x = x_new[:]

    return x

def gauss_seidel_method(A, b, tol=1e-4, max_iterations=1000):
    n = len(b)
    x = [0.0 for _ in range(n)]

    for _ in range(max_iterations):
        x_old = x[:]

        for i in range(n):
            sum_Ax = sum(A[i][j] * x[j] for j in range(n) if i != j)
            x[i] = (b[i] - sum_Ax) / A[i][i]

        if all(abs(x[i] - x_old[i]) < tol for i in range(n)):
            break

    return x
```

In [2]:
```
# 2) Output of the code with comparison

A = [
    [10, -1, 2, 0],
    [-3, 8, -1, 1],
    [2, -1, 9, -2],
    [-4, 1, -2, 7]
]
b = [6, 21, -12, 15]

A_ge = [row[:] for row in A]
b_ge = b[:]

jacobi_solution = jacobi_method([row[:] for row in A], b[:])
print(f"Jacobi solution: {jacobi_solution}")

gauss_seidel_solution = gauss_seidel_method([row[:] for row in A], b[:])
print(f"Gauss-Seidel solution: {gauss_seidel_solution}")
```

```
gauss_solution = gauss_elimination(A_ge, b_ge)
print(f"Gaussian Elimination solution: {gauss_solution}")

jacobi_error = [abs(jacobi_solution[i] - gauss_solution[i]) for i in range(len(b))]
gauss_seidel_error = [abs(gauss_seidel_solution[i] - gauss_solution[i]) for i in range(len(b))]

print(f"Jacobi method error: {jacobi_error}")
print(f"Gauss-Seidel method error: {gauss_seidel_error}")
```

```
Jacobi solution: [1.0234734051024659, 2.6440166721273313, -0.795233427218206, 2.122710265449941]
Gauss-Seidel solution: [1.0234725957604556, 2.644053037616012, -0.795280904946252, 2.1227536479333295]
Gaussian Elimination solution: [1.023457071282168, 2.644044636757003, -0.7952630380323389, 2.122751081758141]
Jacobi method error: [1.6333820297864676e-05, 2.796462967191715e-05, 2.9610814132885466e-05, 4.081630820040871e-05]
Gauss-Seidel method error: [1.5524478287565202e-05, 8.400859008883543e-06, 1.7866913913144877e-05, 2.5661751883454542e-06]
```

In [ ]:

```
#3) Solution by hand (3 iterations)
System of Equations:

10x1 - x2 + 2x3 = 6
-3x1 + 8x2 - x3 + x4 = 21
2x1 - x2 + 9x3 - 2x4 = -12
-4x1 + x2 - 2x3 + 7x4 = 15

Initial Guess:
x1(0) = 0, x2(0) = 0, x3(0) = 0, x4(0) = 0

JACOBI METHOD (FIRST 3 ITERATIONS)

Iteration 1:
x1(1) = (6 + x2(0) - 2x3(0)) / 10 = 6 / 10 = 0.6
x2(1) = (21 + 3x1(0) + x3(0) - x4(0)) / 8 = 21 / 8 = 2.625
x3(1) = (-12 - 2x1(0) + x2(0) + 2x4(0)) / 9 = -12 / 9 = -1.3333
x4(1) = (15 + 4x1(0) - x2(0) + 2x3(0)) / 7 = 15 / 7 = 2.1429

After Iteration 1:
x1(1) = 0.6
x2(1) = 2.625
x3(1) = -1.3333
x4(1) = 2.1429

Iteration 2:
x1(2) = (6 + x2(1) - 2x3(1)) / 10 = (6 + 2.625 - 2(-1.3333)) / 10 = 1.2333
x2(2) = (21 + 3x1(1) + x3(1) - x4(1)) / 8 = (21 + 3(0.6) + (-1.3333) - 2.1429) / 8 = 2.6655
x3(2) = (-12 - 2x1(1) + x2(1) + 2x4(1)) / 9 = (-12 - 2(0.6) + 2.625 + 2(2.1429)) / 9 = -0.6980
x4(2) = (15 + 4x1(1) - x2(1) + 2x3(1)) / 7 = (15 + 4(0.6) - 2.625 + 2(-1.3333)) / 7 = 2.0036

After Iteration 2:
x1(2) = 1.2333
x2(2) = 2.6655
x3(2) = -0.6980
x4(2) = 2.0036

Iteration 3:
x1(3) = (6 + x2(2) - 2x3(2)) / 10 = (6 + 2.6655 - 2(-0.6980)) / 10 = 1.5333
x2(3) = (21 + 3x1(2) + x3(2) - x4(2)) / 8 = (21 + 3(1.2333) + (-0.6980) - 2.0036) / 8 = 2.7932
x3(3) = (-12 - 2x1(2) + x2(2) + 2x4(2)) / 9 = (-12 - 2(1.2333) + 2.6655 + 2(2.0036)) / 9 = -0.3776
x4(3) = (15 + 4x1(2) - x2(2) + 2x3(2)) / 7 = (15 + 4(1.2333) - 2.6655 + 2(-0.6980)) / 7 = 2.0855

After Iteration 3:
x1(3) = 1.5333
x2(3) = 2.7932
x3(3) = -0.3776
x4(3) = 2.0855




GAUSS-SEIDEL METHOD (FIRST 3 ITERATIONS)

Iteration 1:
x1(1) = (6 + x2(0) - 2x3(0)) / 10 = 6 / 10 = 0.6
x2(1) = (21 + 3x1(1) + x3(0) - x4(0)) / 8 = (21 + 3(0.6)) / 8 = 2.7375
x3(1) = (-12 - 2x1(1) + x2(1) + 2x4(0)) / 9 = (-12 - 2(0.6) + 2.7375) / 9 = -1.5125
x4(1) = (15 + 4x1(1) - x2(1) + 2x3(1)) / 7 = (15 + 4(0.6) - 2.7375 + 2(-1.5125)) / 7 = 2.2304

After Iteration 1:
x1(1) = 0.6
x2(1) = 2.7375
x3(1) = -1.5125
x4(1) = 2.2304
```

```
Iteration 2:
x1(2) = (6 + x2(1) - 2x3(1)) / 10 = (6 + 2.7375 - 2(-1.5125)) / 10 = 1.2763
x2(2) = (21 + 3x1(2) + x3(1) - x4(1)) / 8 = (21 + 3(1.2763) + (-1.5125) - 2.2304) / 8 = 2.6786
x3(2) = (-12 - 2x1(2) + x2(2) + 2x4(1)) / 9 = (-12 - 2(1.2763) + 2.6786 + 2(2.2304)) / 9 = -0.6980
x4(2) = (15 + 4x1(2) - x2(2) + 2x3(2)) / 7 = (15 + 4(1.2763) - 2.6786 + 2(-0.6980)) / 7 = 2.0196

After Iteration 2:
x1(2) = 1.2763
x2(2) = 2.6786
x3(2) = -0.6980
x4(2) = 2.0196

Iteration 3:
x1(3) = (6 + x2(2) - 2x3(2)) / 10 = (6 + 2.6786 - 2(-0.6980)) / 10 = 1.5331
x2(3) = (21 + 3x1(3) + x3(2) - x4(2)) / 8 = (21 + 3(1.5331) + (-0.6980) - 2.0196) / 8 = 2.7894
x3(3) = (-12 - 2x1(3) + x2(3) + 2x4(2)) / 9 = (-12 - 2(1.5331) + 2.7894 + 2(2.0196)) / 9 = -0.3771
x4(3) = (15 + 4x1(3) - x2(3) + 2x3(3)) / 7 = (15 + 4(1.5331) - 2.7894 + 2(-0.3771)) / 7 = 2.0857

After Iteration 3:
x1(3) = 1.5331
x2(3) = 2.7894
x3(3) = -0.3771
x4(3) = 2.0857
```