21374389 牛鹏军 210711

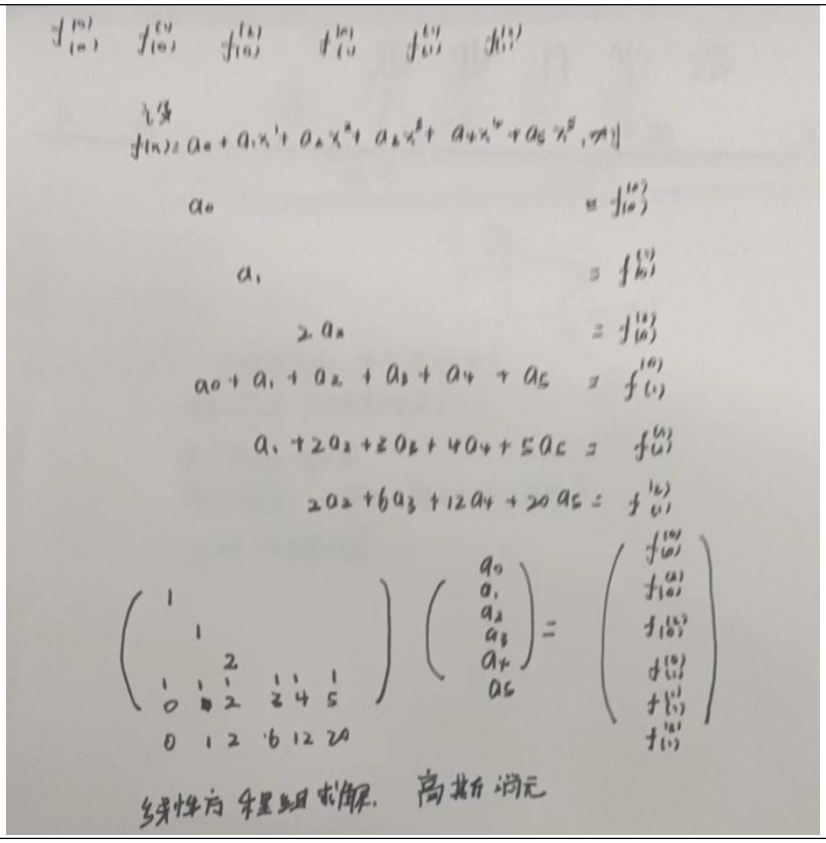
理论分析

作业：

3. 试构造一个多项式 $f(x)$，使之在 0、1 处的函数值、1 阶导数、2 阶导数分别为 $f_{(0)}^{(0)}, f_{(0)}^{(1)}, f_{(0)}^{(2)}, f_{(1)}^{(0)}, f_{(1)}^{(1)}, f_{(1)}^{(2)}$。

此题不难列一个通用形式，求导和二阶导，列个线性方程组等式，高阶全不要了，用点高斯消元就出来了。

算法设计



编程实现

C 语言代码：

```
//编程环境中文注释粘贴会乱码，下边我都用的英文
#include <stdio.h>
#include <stdlib.h>

void gaussianElimination(double matrix[6][6], double* vector, int n) {
    for (int k = 0; k < n - 1; k++) {
        for (int i = k + 1; i < n; i++) {
            // Calculate elimination factor
            double factor = matrix[i][k] / matrix[k][k];

            for (int j = k; j < n; j++) {
                // Perform elimination operation
                matrix[i][j] -= factor * matrix[k][j];
            }

            // Update vector
            vector[i] -= factor * vector[k];
        }
    }

    // Array to store solution vector
    double* solution = (double*)malloc(n * sizeof(double));

    for (int i = n - 1; i >= 0; i--) {
        double sum = 0.0;

        for (int j = i + 1; j < n; j++) {
            // Calculate the sum of products for solution vector
            sum += matrix[i][j] * solution[j];
        }

        // Calculate each element of solution vector
        solution[i] = (vector[i] - sum) / matrix[i][i];
    }

    printf("Solution:\n");

    for (int i = 0; i < n; i++) {
        // Print each element of solution vector
        printf("x%d = %lf\n", i + 1, solution[i]);
    }

    // Free memory allocated for solution vector
```

```
    free(solution);
}

int main() {
    double a, da, dda, b, db, ddb;

    printf("Enter the values of a, da, dda, b, db, ddb: ");
    scanf("%lf %lf %lf %lf %lf %lf", &a, &da, &dda, &b, &db, &ddb);

    int n = 6; // Matrix dimension

    // Initialize vector
    double* vector = (double*)malloc(n * sizeof(double));
    vector[0] = a;
    vector[1] = b;
    vector[2] = da;
    vector[3] = db;
    vector[4] = dda;
    vector[5] = ddb;

    // Coefficient matrix of the linear equation system
    double A[6][6] = {
        {1, 0, 0, 0, 0, 0},
        {1, 1, 1, 1, 1, 1},
        {0, 1, 0, 0, 0, 0},
        {0, 0, 2, 0, 0, 0},
        {0, 1, 2, 3, 4, 5},
        {0, 0, 2, 6, 12, 20}
    };

    // Solve the linear equation system using Gaussian elimination method
    gaussianElimination(A, vector, n);

    // Free memory allocated for vector
    free(vector);

    return 0;
}
```

为了便于验证有效性，采用 Matlab 编写 LU 分解程序：

```
n = input('请输入 Hilbert 矩阵的大小：');
hilbertMatrix = hilb(n);
vector = ones(n, 1);

[L, U, P] = lu(hilbertMatrix);
```

```
y = P * vector;
solution = U \ (L \ y);

disp('Solution:');
for i = 1:n
disp(['x', num2str(i), ' = ', num2str(solution(i))]);
end
```

测试分析

当输入各值分别是 1 0 0 0 0 1 时，得到输出为，1 0 0 -9.5 14 -5.5，对应的分别是 0 到 5 次项多项式的系数，是正确的，再高取什么值都可以，就不求了。

```
Enter the values of a, da, dda, b, db, ddb:
1 0 0 0 0 1
Solution:
x1 = 1.000000
x2 = 0.000000
x3 = -0.000000
x4 = -9.500000
x5 = 14.000000
x6 = -5.500000
```

结论

本次作业难点是通过分析将之转变为易于解决的线性方程组问题，求起来不难。