Московский авиационный институт
(национальный исследовательский университет)

Институт №8 «Компьютерные науки и прикладная математика»

Кафедра 806 «Вычислительная математика и программирование»

Лабораторные работы по курсу «Численные методы»

Студент: О. В. Гребнева
Преподаватель: Д. Е. Пивоваров
Группа: М8О-303Б-21
Дата:
Оценка:
Подпись:

Москва, 2024

3.1. Построение интерполяционных многочленов Лагранжа и Ньютона

1 Постановка задачи

Используя таблицу значений Y_i функции $y = f(x)$, вычисленных в точках $X_i, i = 0, ..., 3$ построить интерполяционные многочлены Лагранжа и Ньютона, проходящие через точки $\{X_i, Y_i\}$. Вычислить значение погрешности интерполяции в точке X_i.

Вариант: 5

$y = ln(x)$,
$a) X_i = 0.2, 0.6, 1.0, 1.4;$
$b) X_i = 0.2, 0.6, 1.0, 1.4;$
$X^* = 0.8$

2 Результаты работы

```
Коэффициенты:
4.1912
-3.9908
-0.0000
0.8762
Значение многочлена Лагранжа в точке x: -0.2078
Абсолютная погрешность интерполяции в точке x: 0.0154

Коэффициенты:
-1.6094
2.7465
-1.8368
1.0766
Значение многочлена Ньютона в точке x: -0.2078
Абсолютная погрешность интерполяции в точке x: 0.0154
```

3 Исходный код

```
#include <iostream>
#include <vector>
#include <fstream>

using namespace std;

double f(double x) {
    return log(x);
}

pair<vector<double>, double> lagrange(const int n, vector<double> x, vector<double> y,
     double starX) {
    double v = 0.0, func = 0.0;
    vector<double> w(n, 0);

    for (int j = 0; j < n; ++j){
        w[j] = y[j];
        func = y[j];
        for (int i = 0; i < n; ++i){
            if (i != j) {
                w[j] = w[j]/(x[j] - x[i]);
                func = func*(starX - x[i])/(x[j] - x[i]);
            }
        }
        v = v + func;
    }

    return make_pair(w, v);
}

pair<vector<double>, double> newton(const int n, vector<double> x, vector<double> y,
    double starX) {
    double v = 0.0, func = 0.0;
    vector<double> w(n, 0);
    vector<vector<double>> d(n, vector<double>(n, 0));

    for (int i = 0; i < n; ++i){
        d[i][0] = y[i];
    }
    for (int j = 1; j < n; j++){
        for (int i = 0; i < (n - j); i++){
            d[i][j] = (d[i][j - 1] - d[i + 1][j - 1])/(x[i] - x[i + j]);
        }
    }
    for (int j = 0; j < n; ++j){
        w[j] = d[0][j];
        func = d[0][j];
```

```
46          for (int i = 0; i < j; ++i){
47              func = func * (starX - x[i]);
48          }
49          v = v + func;
50      }
51
52      return make_pair(w, v);
53  }
54
55  int main() {
56      int n = 4;
57      vector<double> x(n, 0), y(n, 0), w(n, 0);
58      double starX = 0.0, v = 0.0;
59
60      ifstream fin("input.txt");
61      for (int i = 0; i < n; ++i){
62          fin >> x[i];
63          y[i] = f(x[i]);
64      }
65      fin >> starX;
66
67      w = lagrange(n, x, y, starX).first;
68      v = lagrange(n, x, y, starX).second;
69
70      ofstream fout("answer.txt");
71      fout.precision(4);
72      fout << fixed;
73
74      fout << ":" << endl;
75      for (int i = 0; i < n; ++i) {
76          fout << w[i] << endl;
77      }
78
79      double delta = 0.0;
80      delta = abs(v - f(starX));
81
82      fout << "     x: " << v << endl;
83      fout << "     x: " << delta << endl;
84
85      w = newton(n, x, y, starX).first;
86      v = newton(n, x, y, starX).second;
87
88      fout << endl << ":" << endl;
89      for (int i = 0; i < n; ++i) {
90          fout << w[i] << endl;
91      }
92
93      delta = abs(v - f(starX));
94
```

```
95      fout << "      x: " << v << endl;
96      fout << "      x: " << delta << endl;
97
98      return 0;
99  }
```

3.2. Построение кубического сплайна для функции, заданной в узлах интерполяции

1 Постановка задачи

Построить кубический сплайн для функции, заданной в узлах интерполяции, предполагая, что сплайн имеет нулевую кривизну при $x = x_0$ и $x = x_4$. Вычислить значение функции в точке $x = X^*$.

Вариант: 5

$$X^* = 0.8,$$
$$x_i = \{0.1, 0.5, 0.9, 1.3, 1.7\},$$
$$y_i = \{-2.3026, -0.69315, -0.10536, 0.26236, 0.53063\}$$

2 Результаты работы

```
Коэффициенты:
-2.3026 4.6729 0.0000 -4.0581
-0.6932 2.7250 -4.8697 4.3268
-0.1054 0.9062 0.3225 -0.7245
0.2624 0.8165 -0.5468 0.4557
Значение функции в точке X*: -0.1971
```

3 Исходный код

```
#include <iostream>
#include <vector>
#include <fstream>

using namespace std;

double splain(const int n, vector<double> x, vector<double> y, double starX, vector<
    double>& a, vector<double>& b, vector<double>& c, vector<double>& d){
    vector<double> p(n-2, 0), q(n-2, 0);
    double v = 0.0;
    for (int i = 0; i < n - 1; ++i){
        a[i] = y[i];
    }
    c[0] = 0;
    p[0] = -(x[2] - x[1])/(2*x[2] - 2*x[0]);
    q[0] = 3*((y[2] - y[1])/(x[2] - x[1]) - (y[1] - y[0])/(x[1] - x[0]))/(2*x[2] - 2*x
        [0]);
    for (int i = 1; i < n-2; ++i){
        p[i] = -(x[i+2] - x[i+1])/((2*x[i+2] - 2*x[i]) + (x[i+1] - x[i])*p[i-1]);
        q[i] = (3*((y[i+2] - y[i+1])/(x[i+2] - x[i+1]) - (y[i+1] - y[i])/(x[i+1] - x[i
            ])) - (x[i+1] - x[i])*q[i-1])/((2*x[i+2] - 2*x[i]) + (x[i+1] - x[i])*p[i
            -1]);
    }
    c[n-2] = q[n-3];
    for (int i = n-3; i >= 1; --i){
        c[i] = p[i-1]*c[i+1] + q[i-1];
    }
    for (int i = 0; i < n-2; ++i){
        b[i] = (y[i+1] - y[i])/(x[i+1] - x[i]) - (x[i+1] - x[i])*(c[i+1] + 2*c[i])/3;
        d[i] = (c[i+1] - c[i])/(x[i+1] - x[i])/3;
    }
    b[n-2] = (y[n-1] - y[n-2])/(x[n-1] - x[n-2]) - (x[n-1] - x[n-2])*2*c[n-2]/3;
    d[n-2] = -c[n-2]/(x[n-1] - x[n-2])/3;
    int j=0;
    for (int i = 0; i < n-1; ++i){
        if ((starX >= x[i]) && (starX <= x[i+1])) j = i;
    }
    if (j == 0) return 0;
    starX = starX - x[j];
    v = a[j] + b[j]*starX + c[j]*starX*starX + d[j]*starX*starX*starX;
    return v;
}

int main() {
    int n = 5;
    vector<double> x(n, 0), y(n, 0), w(n, 0);
    double starX = 0.0;
```

```
44      vector<double> a(n-1, 0), b(n-1, 0), c(n-1, 0), d(n-1, 0);
45
46      ifstream fin("input.txt");
47      for (int i = 0; i < n; ++i){
48          fin >> x[i];
49      }
50      for (int i = 0; i < n; ++i){
51          fin >> y[i];
52      }
53      fin >> starX;
54
55      ofstream fout("answer.txt");
56      fout.precision(4);
57      fout << fixed;
58
59      double v = splain(n, x, y, starX, a, b, c, d);
60
61      fout << ":" << endl;
62      for (int i = 0; i < n-1; ++i) {
63          fout << a[i] << " " << b[i] << " " << c[i] << " " << d[i] << endl;
64      }
65
66      fout << "    X*: " << v << endl;
67
68      return 0;
69  }
```

3.3. Нахождение приближающих многочленов таблично заданной функции

1 Постановка задачи

Для таблично заданной функции путем решения нормальной системы МНК найти приближающие многочлены а) 1-ой и б) 2-ой степени. Для каждого из приближающих многочленов вычислить сумму квадратов ошибок. Построить графики приближаемой функции и приближающих многочленов.

Вариант: 5

$$x_i = \{0.1, 0.5, 0.9, 1.3, 1.7, 2.1\},$$
$$y_i = \{-2.3026, -0.69315, -0.10536, 0.26236, 0.53063, 0.74194\}$$

2 Результаты работы

```
1-ая СТЕПЕНЬ
Нормальная система метода наименьших квадратов:
6.0000*a0 + 6.6000*a1 = 2.1296
6.6000*a0 + 10.0600*a1 = 4.9672
Коэффициенты приближающего многочлена:
-0.6762 0.9374
Значение суммы квадратов ошибок: 0.3236

2-ая СТЕПЕНЬ
Нормальная система метода наименьших квадратов:
6.0000*a0 + 6.6000*a1 + 10.0600*a2 = 2.1296
6.6000*a0 + 10.0600*a1 + 17.2260*a2 = 4.9672
10.0600*a0 + 17.2260*a1 + 31.3750*a2 = 9.8887
Коэффициенты приближающего многочлена:
-0.2532 -0.3145 0.5690
Значение суммы квадратов ошибок: 0.0141
```

3 Исходный код

```
#include <iostream>
#include <vector>
#include <fstream>

using namespace std;

void MNK(vector<double> x, vector<double> y, const int n, const int m, vector<double>&
     b, vector<double>& z, vector<vector<double>>& a, double& f){
    double g = 0.0;
    for (int i = 0; i < m; ++i){
        for (int j = 0; j < m; ++j){
            a[i][j] = 0;
            for (int k = 0; k < n; ++k){
                a[i][j] = a[i][j] + pow(x[k], i+j);
            }
        }
        b[i] = 0;
        for (int k = 0; k < n; ++k){
            b[i] = b[i]+y[k]*pow(x[k], i);
        }
    }
    if (m == 1) z[0] = b[0]/a[0][0];
    if (m == 2) {
        g = a[0][0] * a[1][1] - a[1][0] * a[0][1];
        z[0] = (b[0]*a[1][1] - b[1]*a[1][0])/g;
        z[1] = (a[0][0]*b[1] - a[1][0]*b[0])/g;
    }
    if (m == 3) {
        g = a[0][0]*a[1][1]*a[2][2] + a[1][0]*a[2][1]*a[0][2] + a[2][0]*a[0][1]*a[1][2]
             - a[2][0]*a[1][1]*a[0][2] - a[0][0]*a[2][1]*a[1][2] - a[1][0]*a[0][1]*a
            [2][2];
        z[0] = (b[0]*a[1][1]*a[2][2] + b[1]*a[2][1]*a[0][2] + b[2]*a[0][1]*a[1][2] - b
            [2]*a[1][1]*a[0][2] - b[0]*a[2][1]*a[1][2] - b[1]*a[0][1]*a[2][2])/g;
        z[1] = (a[0][0]*b[1]*a[2][2] + a[1][0]*b[2]*a[0][2] + a[2][0]*b[0]*a[1][2] - a
            [2][0]*b[1]*a[0][2] - a[0][0]*b[2]*a[1][2] - a[1][0]*b[0]*a[2][2])/g;
        z[2] = (a[0][0]*a[1][1]*b[2] + a[1][0]*a[2][1]*b[0] + a[2][0]*a[0][1]*b[1] - a
            [2][0]*a[1][1]*b[0] - a[0][0]*a[2][1]*b[1] - a[1][0]*a[0][1]*b[2])/g;
    }

    for (int k = 0; k < n; ++k) {
        g = 0;
        for (int i = 0; i < m; ++i) {
            g = g + z[i]*pow(x[k], i);
        }
        f = f + (g - y[k])*(g - y[k]);
    }
}
```

```
42
43  int main() {
44      int n = 6, m = 2;
45      vector<double> x(n, 0), y(n, 0), b(m, 0), z(m, 0);
46      vector<vector<double>> a(m, vector<double>(m, 0));
47      double f = 0.;
48
49      ifstream fin("input.txt");
50      for (int i = 0; i < n; ++i){
51          fin >> x[i];
52      }
53      for (int i = 0; i < n; ++i){
54          //fin >> y[i];
55          y[i] = x[i] * log(x[i]);
56      }
57
58      ofstream fout("answer.txt");
59      fout.precision(4);
60      fout << fixed;
61
62      MNK(x, y, n, m, b, z, a, f);
63
64      fout << "1- " << endl;
65      fout << "    :" << endl;
66      for (int i = 0; i < m; ++i) {
67          for (int j = 0; j < m; ++j){
68              fout << a[i][j] << "*a" << j;
69              if (j != m-1) fout << " + ";
70              else fout << " = ";
71          }
72          fout << b[i] << endl;
73      }
74
75      fout << "  :" << endl;
76      for (int i = 0; i < m; ++i) {
77          fout << z[i] << " ";
78      }
79      fout << endl;
80
81      fout << "   : " << f << endl;
82
83      m = 3;
84      vector<double> b1(m, 0), z1(m, 0);
85      vector<vector<double>> a1(m, vector<double>(m, 0));
86      f = 0.;
87      MNK(x, y, n, m, b1, z1, a1, f);
88
89      fout << endl << "2- " << endl;
90      fout << "    :" << endl;
```

```
91      for (int i = 0; i < m; ++i) {
92          for (int j = 0; j < m; ++j){
93              fout << a1[i][j] << "*a" << j;
94              if (j != m-1) fout << " + ";
95              else fout << " = ";
96          }
97          fout << b1[i] << endl;
98      }
99
100     fout << "  :" << endl;
101     for (int i = 0; i < m; ++i) {
102         fout << z1[i] << " ";
103     }
104     fout << endl;
105
106     fout << "   : " << f << endl;
107
108     return 0;
109 }
```

3.4. Вычисление производных таблично заданной функции

1 Постановка задачи

Вычислить первую и вторую производную от таблично заданной функции $y_i = f(x_i), i = 0, 1, 2, 3, 4$ в точке $x = X^*$.

Вариант: 5

$$X^* = 2.0$$
$$x_i = \{0.0, 1.0, 2.0, 3.0, 4.0\},$$
$$y_i = \{0.0, 1.0, 1.4142, 1.7321, 2.0\}$$

2 Результаты работы

```
Первая производная:0.3214
Вторая производная:-0.0594
```

3 Исходный код

```
#include <iostream>
#include <vector>
#include <fstream>

using namespace std;

pair<double, double> derivatives(const vector<double> x, const vector<double> y, const
     int n, const double starX){
    vector<vector<double>> d(n, vector<double>(n, 0));
    double f = 0., g = 0.;
    double v = 0., w = 0.;

    for (int i = 0; i < n; ++i) {
        d[i][0] = y[i];
    }
    for (int j = 1; j < n; ++j) {
        for (int i = 0; i < n-j; ++i) {
            d[i][j] = (d[i][j-1] - d[i+1][j-1])/(x[i] - x[i+j]);
        }
    }

    for (int j = 1; j < n; ++j) {
        f = 0.;
        for (int k = 0; k < j; ++k) {
            g = d[0][j];
            for (int i = 0; i < j; ++i) {
                if ( i != k) g = g*(starX - x[i]);
            }
            f = f + g;
        }
        v = v + f;
    }

    for (int j = 3; j <= n; ++j) {
        f = 0.;
        for (int k = 1; k <= (j-1)*(j-2); ++k) {
            g = d[0][j - 1];
            for (int i = 1; i < j; ++i) {
                if (( i != 1 + (k-1)/(j-2)) && (i != 1 + (k + (k-1)/(j-1)) % (j-1)))
                    g = g * (starX - x[i-1]);
            }
            f = f + g;
        }
        w = w + f;
    }

    return make_pair(v, w);
```

```
47 }
48
49 int main() {
50     int n = 5;
51     vector<double> x(n, 0), y(n, 0);
52     double starX = 0.;
53
54     ifstream fin("input.txt");
55     for (int i = 0; i < n; ++i){
56         fin >> x[i];
57     }
58     for (int i = 0; i < n; ++i){
59         fin >> y[i];
60     }
61     fin >> starX;
62
63     ofstream fout("answer.txt");
64     fout.precision(4);
65     fout << fixed;
66
67     fout << " :" << derivatives(x, y, n, starX).first << endl;
68     fout << " :" << derivatives(x, y, n, starX).second << endl;
69
70     return 0;
71 }
```

3.5. Вычисление определённого интеграла методами прямоугольников, трапеций и Симпсона

1 Постановка задачи

Вычислить определенный интеграл $F = \int_{X_0}^{X_1} ydx$, методами прямоугольников, трапеций, Симпсона с шагами h_1, h_2. Оценить погрешность вычислений, используя Метод Рунге-Ромберга.

Вариант: 5

$$y = \frac{1}{(2x+7)(3x+4)},$$
$$X_0 = -1, X_k = 1, h_1 = 0.5, h_2 = 0.25$$

2 Результаты работы

```
Левые прямоугольники с шагом h1: 0.1626
Левые прямоугольники с шагом h2: 0.1308
Погрешность: 0.0158

Правые прямоугольники с шагом h1: 0.0705
Правые прямоугольники с шагом h2: 0.0848
Погрешность: 0.0149

Средние прямоугольники с шагом h1: 0.0991
Средние прямоугольники с шагом h2: 0.1029
Погрешность: 0.0003

Трапеции с шагом h1: 0.1165
Трапеции с шагом h2: 0.1078
Погрешность: 0.0004

Метод Симпсона с шагом h1: 0.1049
Метод Симпсона с шагом h2: 0.1045
Погрешность: 0.0001
```

3 Исходный код

```
#include <iostream>
#include <vector>
#include <fstream>

using namespace std;

double f(double x){
    return 1 / ((2*x + 7)*(3*x + 4));
}

void methods(double x0, double xk, double& l, double& r, double& m, double& t, double&
     w, double h){
    int n = (xk - x0) / h + 1;
    vector<double> x(n, 0);

    for (int i = 0; i < n; ++i){
        x[i] = x0;
        x0 = x0 + h;
    }
    vector<double> y(2*n-1, 0);

    for (int i = 0; i < n; i++) {
        y[i] = f(x[i]);
    }

    for (int i = n; i < 2*n - 1; i++){
        y[i] = f((x[i - n] + x[i - n + 1]) / 2);
    }

    for (int i = 0; i < n - 1; i++) {
        l = l + y[i]*h;
        r = r + y[i + 1]*h;
        m = m + y[i + n]*h;
        t = t + (h/2) * (y[i] + y[i + 1]);
        w = w + (h/6)*(y[i] + y[i + 1] + 4*y[i + n]);
    }
}

double RuRoRi(double a, double aa){
    double RuR = 0;
    double integral = 0.10447;
    RuR = a -(aa - a)/3;
    return abs(RuR - integral);
}

int main() {
    double x0, xk, h1, h2;
```

```
47      double l = 0, r = 0, m = 0, t = 0, w = 0;
48
49      ifstream fin("input.txt");
50      fin >> x0;
51      fin >> xk;
52      fin >> h1;
53      fin >> h2;
54
55      ofstream fout("answer.txt");
56      fout.precision(4);
57      fout << fixed;
58
59      methods(x0, xk, l, r, m, t, w, h1);
60      double ll = l, rr = r, mm = m, tt = t, ww = w;
61
62      l = 0, r = 0, m = 0, t = 0, w = 0;
63      methods(x0, xk, l, r, m, t, w, h2);
64
65
66      fout << "    h1: " << ll << endl;
67      fout << "    h2: " << l << endl;
68      fout << ": " << RuRoRi(l, ll) << endl;
69
70      fout << endl << "    h1: " << rr << endl;
71      fout << "    h2: " << r << endl;
72      fout << ": " << RuRoRi(r, rr) << endl;
73
74      fout << endl << "    h1: " << mm << endl;
75      fout << "    h2: " << m << endl;
76      fout << ": " << RuRoRi(m, mm) << endl;
77
78      fout << endl << "   h1: " << tt << endl;
79      fout << "   h2: " << t << endl;
80      fout << ": " << RuRoRi(t, tt) << endl;
81
82      fout << endl << "    h1: " << ww << endl;
83      fout << "    h2: " << w << endl;
84      fout << ": " << RuRoRi(w, ww) << endl;
85
86      return 0;
87  }
```