



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика, искусственный интеллект и системы управления»

КАФЕДРА «Теоретическая информатика и компьютерные технологии»

ОТЧЕТ

по лабораторной работе № 4

по курсу «Численные методы»

на тему: «Численное решение краевой задачи для линейного
дифференциального уравнения второго порядка методом прогонки»

Вариант № 4

Студент ИУ9-61Б
(Группа)

(Подпись, дата)

Афанасьев И.
(И. О. Фамилия)

Преподаватель

(Подпись, дата)

Домрачева А. Б.
(И. О. Фамилия)

2024 г.

1 Постановка задачи

1. Написать и отладить процедуру для решения трёхдиагональной линейной системы методом прогонки.
2. Решить аналитически задачу Коши

$$y'' + py' + qy = f(x), \quad y(0) = y_0, \quad y'(0) = y'_0$$

и по найденному решению задачи Коши $y(x)$ вычислить $b = y(1)$.

3. Найти численное решение (x_i, y_i) , $i = 0, \dots, n$, краевой задачи для того же уравнения с краевыми условиями $y(0) = a$, $y(1) = b$ при $n = 10$.
4. Найти погрешность численного решения $\|y - \tilde{y}\| = \max_{0 \leq i \leq n} |y(x_i) - \tilde{y}_i|$.

Индивидуальный вариант: $p = -1$, $q = 0$, $f(x) = 2(1 - x)$, $y(0) = 1$, $y'(0) = 1$.

2 Основные теоретические сведения

Краевая задача для линейного дифференциального уравнения второго порядка имеет вид

$$y'' + p(x)y' + q(x)y = f(x), \quad (2.1)$$

$$y(0) = a, \quad y(1) = b. \quad (2.2)$$

Требуется найти частное решение уравнения 2.1, отвечающее краевым условиям 2.2. Приближенным численным решением задачи 2.1, 2.2 называется *сеточная функция* (x_i, y_i) , $i = 0, \dots, n$, заданная в $(n + 1)$ -й точке $x_i = ih$, $h = \frac{1}{n}$.

Обозначим через $p_i = p(x_i)$, $q_i = q(x_i)$, $f_i = f(x_i)$ значения коэффициентов уравнения в точках x_i (узлах сетки), $i = 0, \dots, n$. Применяя разностную аппроксимацию производных по формулам численного интегрирования, получим приближенную систему уравнений относительно ординат сеточной функции y_i :

$$\frac{y_{i+1} - 2y_i + 2y_{i-1}}{h^2} + p_i \frac{y_{i+1} - y_{i-1}}{2h} + q_i y_i = f_i,$$

или после преобразований

$$y_{i-1}(1 - \frac{h}{2}p_i) + y_i(h^2q_i - 2) + y_{i+1}(1 + \frac{h}{2}p_i) = h^2f_i, \quad i = 1, \dots, n-1, \quad (2.3)$$

с краевыми условиями

$$y_0 = a, \quad y_n = b. \quad (2.4)$$

Система 2.3 является *разностной системой с краевыми условиями* 2.4 и представляет собой трёхдиагональную систему линейных алгебраических уравнений $(n + 1)$ -го порядка. Трёхдиагональную линейную систему следует решать методом прогонки.

3 Реализация

В листинге 3.1 представлен исходный код программы на языке C++.

Листинг 3.1 – Исходный код программы на языке C++

```
1 // clang-format off
2 #include <sprout/math/exp.hpp>
3 // clang-format on
4
5 #include <algorithm>
6 #include <cassert>
7 #include <cmath>
8 #include <iomanip>
9 #include <limits>
10 #include <numbers>
11 #include <stdexcept>
12 #include <vector>
13
14 namespace {
15
16 bool CheckDiagonalPredominanceConditions(
17     const std::vector<std::vector<double>>& mat) {
18     const auto n = mat.size();
19
20     if (std::abs(mat[0][1] / mat[0][0]) > 1) {
21         return false;
22     }
23
24     if (std::abs(mat[n - 1][n - 2] / mat[n - 1][n - 1]) > 1) {
25         return false;
26     }
27
28     for (auto i = 1; i < n; i++) {
29         if (std::abs(mat[i][i]) <
30             std::abs(mat[i][i - 1]) + std::abs(mat[i][i + 1])) {
31             return false;
32         }
33     }
34
35     return true;
36 }
37
```

```

38 std::vector<double> RunThrough(const
    std::vector<std::vector<double>>& mat,
39                                const std::vector<double>& cfs) {
40     if (!CheckDiagonalPredominanceConditions(mat)) {
41         throw std::runtime_error("Run-through predominance
            conditions failed");
42     }
43
44     const auto n = cfs.size();
45
46     std::vector<double> as(n - 1);
47     std::vector<double> bs(n - 1);
48
49     as[0] = -mat[0][1] / mat[0][0];
50     bs[0] = cfs[0] / mat[0][0];
51
52     for (auto i = 1; i < n - 1; i++) {
53         as[i] = -mat[i][i + 1] / (mat[i][i - 1] * as[i - 1] +
            mat[i][i]);
54         bs[i] = (cfs[i] - mat[i][i - 1] * bs[i - 1]) /
            (mat[i][i - 1] * as[i - 1] + mat[i][i]);
55     }
56
57
58     std::vector<double> res(n);
59     res[n - 1] = (cfs[n - 1] - mat[n - 1][n - 2] * bs[n - 2]) /
60         (mat[n - 1][n - 2] * as[n - 2] + mat[n - 1][n -
            1]);
61
62     for (int i = n - 2; i >= 0; i--) {
63         res[i] = as[i] * res[i + 1] + bs[i];
64     }
65
66     return res;
67 }
68
69 void PrintResult(const std::vector<double>& xs, const
    std::vector<double>& ys,
70                  const std::vector<double>& errs,
71                  const std::function<double(const double)>& y) {
72     static constexpr std::size_t kWidth = 10;
73

```

```

74     const auto n = xs.size();
75     assert(ys.size() == n);
76
77     std::cout << std::setw(kWidth) << "i";
78     for (std::size_t i = 0; i < n; i++) {
79         std::cout << std::setw(kWidth) << i;
80     }
81     std::cout << "\n";
82
83     std::cout << std::setw(kWidth) << "x_i";
84     for (std::size_t i = 0; i < n; i++) {
85         std::cout << std::setw(kWidth) << std::fixed << xs[i];
86     }
87     std::cout << "\n";
88
89     std::cout << std::setw(kWidth) << "y(x_i)";
90     for (std::size_t i = 0; i < n; i++) {
91         std::cout << std::setw(kWidth) << std::fixed << y(xs[i]);
92     }
93     std::cout << "\n";
94
95     std::cout << std::setw(kWidth) << "y_i";
96     for (std::size_t i = 0; i < n; i++) {
97         std::cout << std::setw(kWidth) << std::fixed << ys[i];
98     }
99     std::cout << "\n";
100
101     std::cout << std::setw(kWidth) << "error";
102     for (std::size_t i = 0; i < n; i++) {
103         std::cout << std::setw(kWidth) << std::fixed << errs[i];
104     }
105     std::cout << "\n";
106
107     const auto max_err = std::max_element(errs.begin(),
108         errs.end());
109     std::cout << "max error = " << *max_err << "\n";
110 }
111 } // namespace
112
113 int main() {

```

```

114 constexpr auto y = [](const double x) {
115     return x * x + 9 * sprout::exp(x) - 8;
116 };
117
118 constexpr double a = 1;
119 constexpr auto b = y(1);
120 constexpr auto p = [](const double x) { return -1; };
121 constexpr auto q = [](const double x) { return 0; };
122 constexpr auto f = [](const double x) { return 2 * (1 - x); };
123
124 constexpr std::size_t n = 10;
125 constexpr auto h = 1 / static_cast<double>(n);
126
127 // Fill xs, ps, qs, fs
128 std::vector<double> xs, ps, qs, fs;
129
130 xs.reserve(n + 1);
131 ps.reserve(n + 1);
132 qs.reserve(n + 1);
133 fs.reserve(n + 1);
134
135 double x = 0;
136 for (std::size_t i = 0; i <= n; ++i) {
137     xs.push_back(x);
138     ps.push_back(p(x));
139     qs.push_back(q(x));
140     fs.push_back(f(x));
141
142     x += h;
143 }
144
145 // Fill matrix, free coefficients
146 std::vector<std::vector<double>> mat(n - 1,
147     std::vector<double>(n - 1));
148
149 std::vector<double> cfs;
150 cfs.reserve(n - 1);
151
152 const auto h_sqr = h * h;
153 const auto h_hlf = h / 2;

```

```

154 mat[0][0] = h_sqr * qs[1] - 2;
155 mat[0][1] = 1 + h_hlf * ps[1];
156
157 cfs.push_back(h_sqr * fs[1] - a * (1 - h_hlf * ps[1]));
158
159 for (std::size_t i = 1, end = n - 2; i < end; ++i) {
160     mat[i][i - 1] = 1 - h_hlf * ps[i + 1];
161     mat[i][i] = h_sqr * qs[i + 1] - 2;
162     mat[i][i + 1] = 1 + h_hlf * ps[i + 1];
163
164     cfs.push_back(h_sqr * fs[i + 1]);
165 }
166
167 mat[n - 2][n - 3] = 1 - h_hlf * ps[n - 1];
168 mat[n - 2][n - 2] = h_sqr * qs[n - 1] - 2;
169
170 cfs.push_back(h_sqr * fs[n - 1] - b * (1 + h_hlf * ps[n - 1]));
171
172 // Run-through
173 const auto sol = RunThrough(mat, cfs);
174
175 std::vector<double> ys;
176 ys.reserve(n + 1);
177
178 ys.push_back(a);
179 for (std::size_t i = 0, end = n - 1; i < end; ++i) {
180     ys.push_back(sol[i]);
181 }
182 ys.push_back(b);
183
184 // Error
185 std::vector<double> errs;
186 errs.reserve(n + 1);
187
188 for (std::size_t i = 0; i <= n; ++i) {
189     errs.push_back(std::abs(y(xs[i]) - ys[i]));
190 }
191
192 PrintResult(xs, ys, errs, y);
193 }

```


4 Результаты

На рисунке 4.1 представлен вывод программы.

```
      i      0      1      2      3      4      5      6      7      8      9      10
x_i  0.000000  0.100000  0.200000  0.300000  0.400000  0.500000  0.600000  0.700000  0.800000  0.900000  1.000000
y(x_i) 1.000000  1.956538  3.032625  4.238729  5.586422  7.088491  8.759069  10.613774  12.669868  14.946428  17.464536
y_i  1.000000  1.956119  3.031829  4.237614  5.585060  7.086975  8.757512  10.612316  12.668679  14.945711  17.464536
error 0.000000  0.000420  0.000796  0.001115  0.001362  0.001516  0.001557  0.001458  0.001189  0.000717  0.000000
max error = 0.001557
```

Рисунок 4.1 – Вывод программы