МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский ядерный университет «МИФИ»
(НИЯУ МИФИ)
Институт интеллектуальных кибернетических систем
Кафедра Кибернетики

Лабораторная работа №1
«Решение системы дифференциальных уравнений методом Адамса-Башфорта третьего порядка»

Выполнил студент группы Б17-511: Чудновец И.В.

Проверил: Саманчук В.Н.

Москва, 2020

Задание

Написать программу для решения системы дифференциальных уравнений по методу Адамса-Башфорта третьего порядка.

$$\begin{cases} \frac{dy_1}{dx} = -1000 * y_1 + 999 * y_2 \\ \frac{dy_2}{dx} = y_1 - 2 * y_2 \end{cases}$$

Начальные условия:

$$\begin{cases} y_{10} = 10 \\ y_{20} = 20 \end{cases}$$

Описание метода

Метод Адамса — конечноразностный многошаговый метод численного интегрирования обыкновенных дифференциальных уравнений первого порядка. В отличие от метода Рунге-Кутты использует для вычисления очередного значения искомого решения не одно, а несколько значений, которые уже вычислены в предыдущих точках.

Метод Адамса-Башфорта третьего порядка имеет вид:

$$y_{n+1} = y_n + \frac{h}{12}(23 f_n - 16 f_{n-1} + 5 f_{n-2}).$$

На вход программе процедуре поступают параметры: n – порядок метода (3), N – число шагов, h - шаг, x – начальное значение x0, y – массив начальных значений y0, foo_m – массив функций правой части системы.

Локальная погрешность метода Адамса 3-го порядка — $O(h^3)$

Листинг программы

```
#include <iostream>

using namespace std;

typedef long double ld;
typedef double (*f)(double, double, double);

void Adams_Bashforth(int order, int n, double h, double x, double* y, f
array_f[2]){
    if(order < 1 || order > 5){
        cout << "Error! Order must be integer in range [1, 5]";
        return;
    }
    double** t = new double*[3];
    for(int i = 0; i < 3; i++){
        t[i] = new double[order];
    }

    t[0][0] = y[0];
    t[1][0] = y[1];
    t[2][0] = x;

    int AB_table[6][5];
    AB_table[0][0] = 1;
    AB_table[0][1] = 2;
    AB_table[0][2] = 12;
    AB_table[0][3] = 24;
    AB_table[0][4] = 720;
    AB_table[1][0] = 1;
    AB_table[1][1] = 3;
    AB_table[1][2] = 23;
    AB_table[1][3] = 55;
    AB_table[1][4] = 1901;
    AB_table[2][1] = -1;
    AB_table[2][2] = -16;
    AB_table[2][3] = -59;
    AB_table[2][4] = -2774;
    AB_table[3][2] = 5;
    AB_table[3][3] = 37;
    AB_table[3][4] = 2616;
    AB_table[4][3] = -9;
    AB_table[4][4] = -1274;
    AB_table[5][4] = 251;

    double sum;

    for(int j = 0; j < order - 1; j++){
        for(int i = 0; i < 2; i++){
            sum = 0;
            for(int k = 0; k < j + 1; k++){
                sum += AB_table[k + 1][j] * array_f[i](x - k * h, t[0][j -
k], t[1][j - k]);
            }
            sum *= h / AB_table[0][j];
            t[i][j + 1] = t[i][j] + sum;
        }
        x += h;
        t[2][j + 1] = x;
```

```
        cout << j << " " << t[2][j] << " " << t[0][j] << " " << t[1][j] <<
endl;
    }

    cout << order - 1 << " " << t[2][order - 1] << " " << t[0][order - 1] <<
" " << t[1][order - 1] << endl;

    double new_values[2];

    for(int j = order; j < n; j++){
        for(int i = 0; i < 2; i++){
            sum = 0;
            for(int k = 0; k < order; k++){
                sum += AB_table[k + 1][order - 1] * array_f[i](x - k * h,
t[0][order - 1 - k], t[1][order - 1 - k]);
            }
            sum *= h / AB_table[0][order - 1];
            new_values[i] = t[i][order - 1] + sum;
        }

        for(int m = 1; m < order; m++){
            t[0][m - 1] = t[0][m];
            t[1][m - 1] = t[1][m];
        }

        t[0][order - 1] = new_values[0];
        t[1][order - 1] = new_values[1];
        x += h;
        t[2][order - 1] = x;

        cout << j << " " << x << " " << t[0][order - 1] << " " << t[1][order
- 1] << endl;
    }
}

double f1(double x, double y_1, double y_2){
    return -1000 * y_1 + 999 * y_2;
}

double f2(double x, double y_1, double y_2){
    return y_1 - 2 * y_2;
}

int main()
{
    double a = 0;
    double b = 0.001;
    double h = 0.00001;

    int N = (b - a) / h + 1;
    double x = a;
    double *y = new double[2];
    y[0] = 10;
    y[1] = 20;
    f foo_m[] = {f1, f2};

    Adams_Bashforth(3, N, h, x, y, foo_m);

    return 0;
}
```

Пример работы программы

```
0 0 10 20
1 1e-005 10.0998 19.9997
2 2e-005 10.1981 19.9994
3 3e-005 10.2954 19.9991
4 4e-005 10.3918 19.9988
5 5e-005 10.4872 19.9985
6 6e-005 10.5816 19.9982
7 7e-005 10.6751 19.9979
8 8e-005 10.7677 19.9976
9 9e-005 10.8593 19.9973
10 0.0001 10.95 19.997
11 0.00011 11.0399 19.9968
12 0.00012 11.1288 19.9965
13 0.00013 11.2168 19.9962
14 0.00014 11.304 19.9959
15 0.00015 11.3902 19.9956
16 0.00016 11.4757 19.9953
17 0.00017 11.5602 19.995
18 0.00018 11.644 19.9948
19 0.00019 11.7269 19.9945
20 0.0002 11.8089 19.9942
21 0.00021 11.8902 19.9939
22 0.00022 11.9706 19.9936
23 0.00023 12.0502 19.9933
24 0.00024 12.1291 19.9931
25 0.00025 12.2071 19.9928
26 0.00026 12.2844 19.9925
27 0.00027 12.3609 19.9922
28 0.00028 12.4366 19.992
29 0.00029 12.5116 19.9917
30 0.0003 12.5858 19.9914
31 0.00031 12.6593 19.9911
32 0.00032 12.7321 19.9909
33 0.00033 12.8041 19.9906
34 0.00034 12.8754 19.9903
35 0.00035 12.946 19.99
36 0.00036 13.0159 19.9898
37 0.00037 13.0851 19.9895
38 0.00038 13.1536 19.9892
39 0.00039 13.2214 19.989
40 0.0004 13.2885 19.9887
41 0.00041 13.355 19.9884
42 0.00042 13.4208 19.9882
43 0.00043 13.4859 19.9879
44 0.00044 13.5504 19.9876
45 0.00045 13.6143 19.9874
46 0.00046 13.6775 19.9871
47 0.00047 13.7401 19.9869
48 0.00048 13.802 19.9866
49 0.00049 13.8634 19.9863
50 0.0005 13.9241 19.9861
51 0.00051 13.9842 19.9858
52 0.00052 14.0437 19.9855
53 0.00053 14.1027 19.9853
54 0.00054 14.161 19.985
55 0.00055 14.2187 19.9848
56 0.00056 14.2759 19.9845
57 0.00057 14.3325 19.9843
58 0.00058 14.3885 19.984
59 0.00059 14.444 19.9837
60 0.0006 14.4989 19.9835
61 0.00061 14.5533 19.9832
62 0.00062 14.6071 19.983
63 0.00063 14.6604 19.9827
64 0.00064 14.7132 19.9825
65 0.00065 14.7654 19.9822
66 0.00066 14.8171 19.982
67 0.00067 14.8683 19.9817
68 0.00068 14.919 19.9815
69 0.00069 14.9692 19.9812
70 0.0007 15.0188 19.981
71 0.00071 15.068 19.9807
72 0.00072 15.1167 19.9805
73 0.00073 15.1649 19.9802
74 0.00074 15.2126 19.98
75 0.00075 15.2598 19.9797
76 0.00076 15.3066 19.9795
77 0.00077 15.3529 19.9792
78 0.00078 15.3987 19.979
79 0.00079 15.4441 19.9787
80 0.0008 15.489 19.9785
81 0.00081 15.5335 19.9783
82 0.00082 15.5775 19.978
83 0.00083 15.6211 19.9778
84 0.00084 15.6643 19.9775
85 0.00085 15.707 19.9773
86 0.00086 15.7493 19.977
87 0.00087 15.7911 19.9768
88 0.00088 15.8326 19.9766
89 0.00089 15.8736 19.9763
90 0.0009 15.9142 19.9761
91 0.00091 15.9545 19.9758
92 0.00092 15.9943 19.9756
93 0.00093 16.0337 19.9754
94 0.00094 16.0727 19.9751
95 0.00095 16.1113 19.9749
96 0.00096 16.1496 19.9746
97 0.00097 16.1874 19.9744
98 0.00098 16.2249 19.9742
99 0.00099 16.262 19.9739
100 0.001 16.2988 19.9737
```