МІНІСТЕРСТВО ОСВІТИ, НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

Комп'ютерний практикум №6
з дисципліни " Методи обчислень "

Розв'язання задачі Коші методами Рунге-Кутта та Адамса-Башфорта

Варіант № 4

Виконав:
студент 3 курсу
групи ФІ-84
Коломієць Андрій Юрійович
Email: andrew.kolomiets.work@gmail.com

Перевірила:
Стьопочкіна Ірина Валеріївна

Київ – 2021

Варіант завдання згідно варіанту

Рівняння має вигляд: $y' = (1 - x^2)y + F(x)$. Покласти h = 0,1, початкові умови x(0) визначити, використовуючи точне значення розв'язку.

Нехай розв'язок відомий:

$$y = x\cos x$$

Необхідно підставити розв'язок у рівняння та визначити F(x) у правій частині. Таким чином, відомим є вигляд рівняння та його точний розв'язок, за допомогою числових методів далі будуємо наближений розв'язок:

$$(x) = y' - (1 - x^2)\,y \;\; =$$
$$= (\cos(x) - x\sin(x)) - (1 - x^2)\,x\cos(x) =$$
$$= \cos(x)\,(1 - x + x^3) - x\sin(x)$$

Звідси y' набуває вигляду:

$$y' = (1 - x^2)y + \cos(x)\,(1 - x + x^3) - x\sin(x)$$

Розглядаємо інтервал [-1; 1].

Знайдемо наближений розв'язок за допомогою метода Рунге-Кутта четвертого порядку:

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$\begin{cases} k_1 = hf(x_i, y_i), \\ k_2 = hf(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1), \\ k_3 = hf(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_2), \\ k_4 = hf(x_i + h, y_i + k_3). \end{cases}$$

Шукаємо наближений розв'язок методом Адамса-Башфорта четвертого порядку. В якості початкових даних використовуємо дані із попереднього кроку.

$$y_{k+1} = y_k + h(55\,f_k - 59\,f_{k-1} + 37\,f_{k-2} - 9\,f_{k-3})/24$$

Обрахуємо похибки обчислень для обох методів, як різницю отриманого значення та значення точного розв'язку, взяту по модулю.

Результати роботи програми виводяться в Excel

Runge-Kutta methods:

x	y
-1	-0.5403
-0.9	-0.55945
-0.8	-0.55737
-0.7	-0.53539
-0.6	-0.4952
-0.5	-0.43879
-0.4	-0.36842
-0.3	-0.2866
-0.2	-0.19601
-0.1	-0.0995
-1.39E-16	1.45E-06
0.1	0.099502
0.2	0.196015
0.3	0.286603
0.4	0.368426
0.5	0.438793
0.6	0.495203
0.7	0.535392
0.8	0.557368
0.9	0.559451
1	0.540305

Adams-Bashforth methods:

x	y
-1	-0.5403
-0.9	-0.55945
-0.8	-0.55737
-0.7	-0.53539
-0.6	-0.49521
-0.5	-0.4388
-0.4	-0.36844
-0.3	-0.28661
-0.2	-0.19603
-0.1	-0.09952
-1.39E-16	-1.53E-05
0.1	0.099485
0.2	0.195998
0.3	0.286585
0.4	0.368409
0.5	0.438777
0.6	0.495188
0.7	0.535377
0.8	0.557355
0.9	0.55944
1	0.540295

Exect solution:

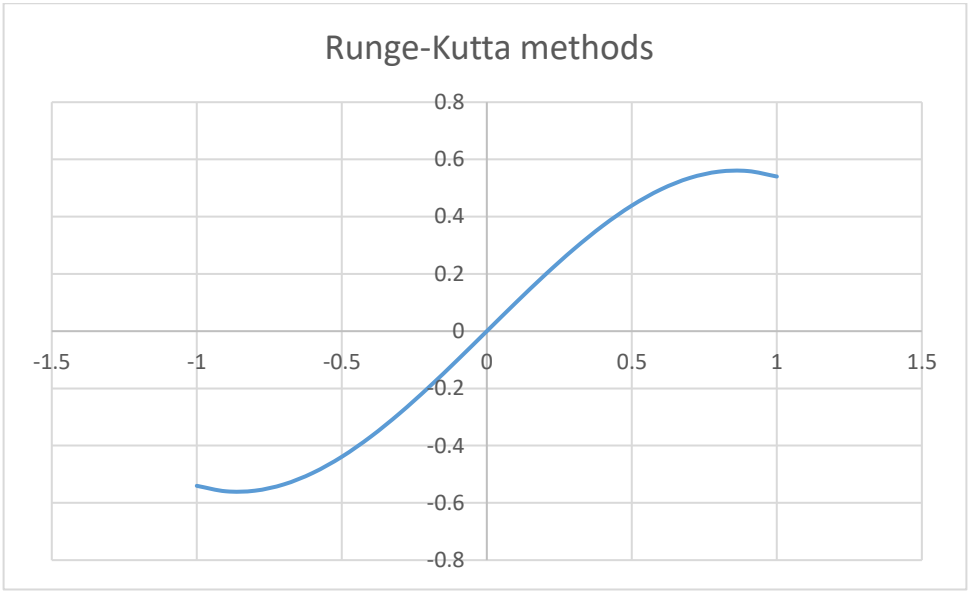
y_0
-0.5403
-0.55945
-0.55737
-0.53539
-0.4952
-0.43879
-0.36842
-0.2866
-0.19601
-0.0995
-1.39E-16
0.0995
0.196013
0.286601
0.368424
0.438791
0.495201
0.53539
0.557365
0.559449
0.540302

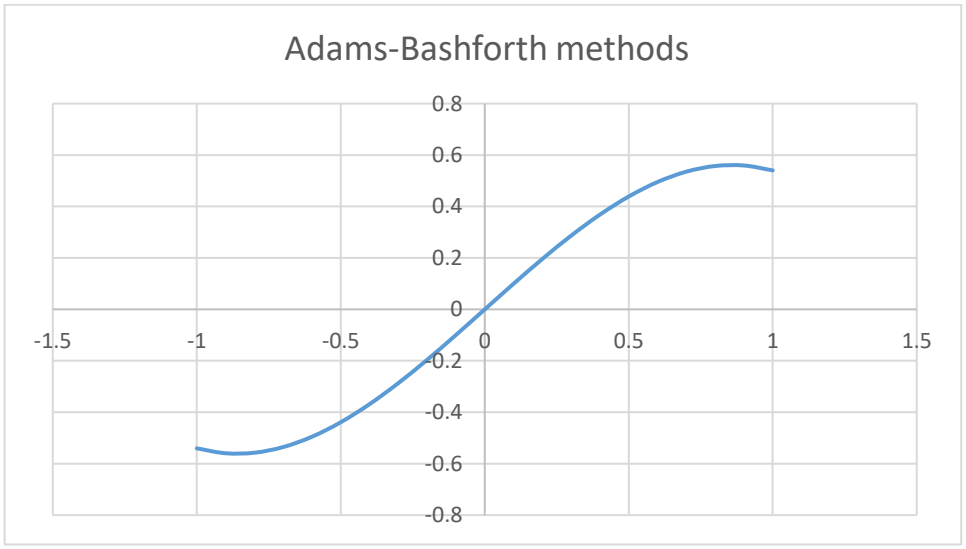
Errors of Runge-Kutta methods:

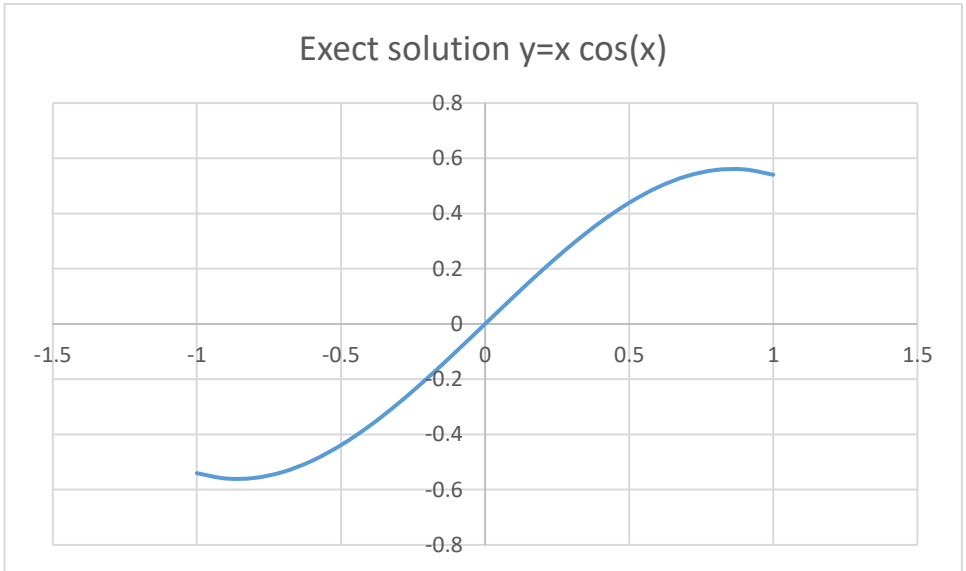
0
1.17E-07
3.14E-07
5.46E-07
7.77E-07
9.82E-07
1.15E-06
1.27E-06
1.35E-06
1.40E-06
1.45E-06
1.50E-06
1.58E-06
1.68E-06
1.81E-06
1.96E-06
2.13E-06
2.28E-06
2.42E-06
2.50E-06
2.50E-06

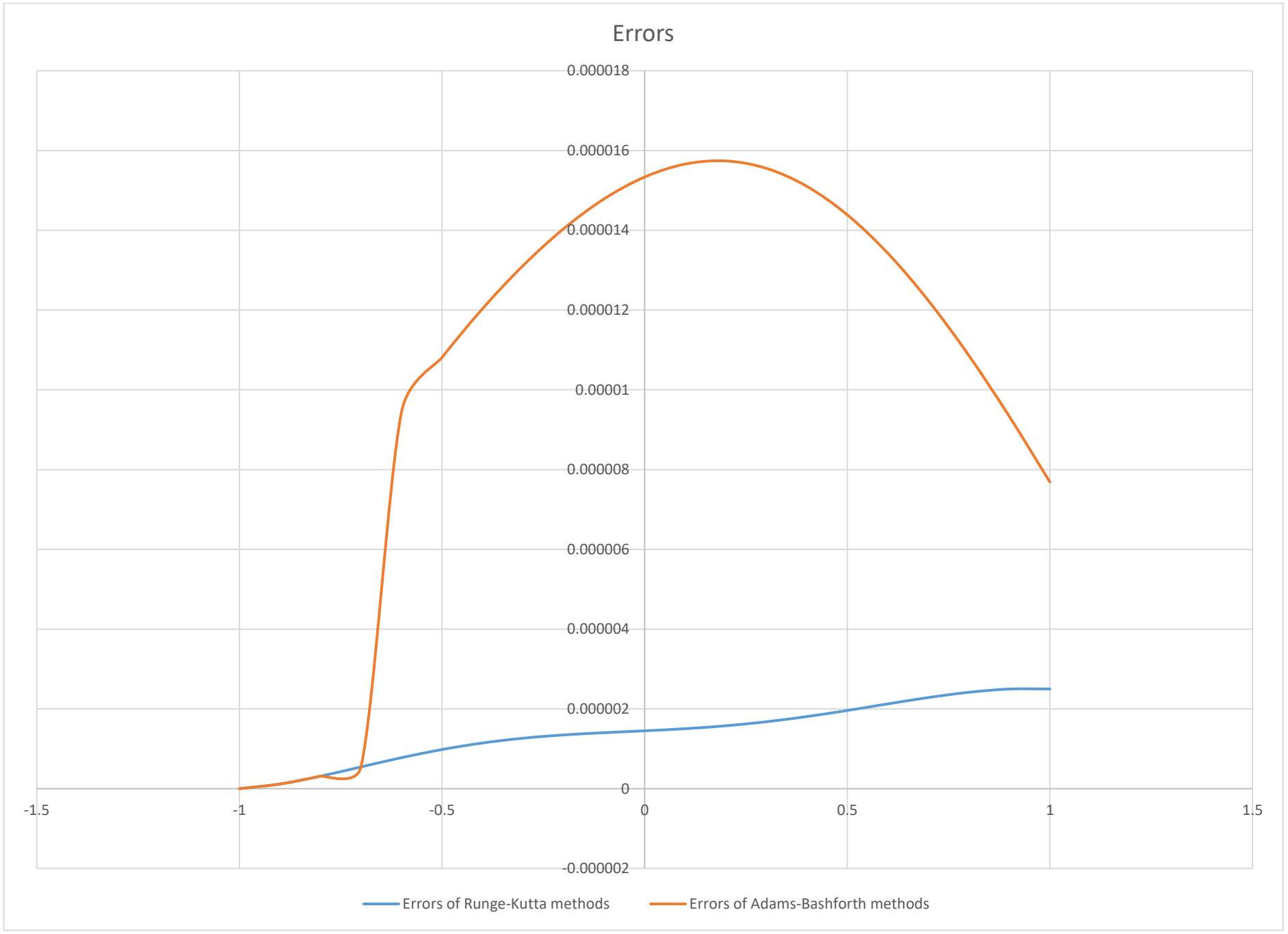
Errors of Adams-Bashforth methods:

0
1.17E-07
3.14E-07
5.46E-07
9.44E-06
1.08E-05
1.20E-05
1.31E-05
1.40E-05
1.48E-05
1.53E-05
1.57E-05
1.57E-05
1.56E-05
1.51E-05
1.44E-05
1.34E-05
1.22E-05
1.09E-05
9.33E-06
7.69E-06


Runge-Kutta methods
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1.5
-1
-0.5
0
0.5
1
1.5


Adams-Bashforth methods
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1.5
-1
-0.5
0
0.5
1
1.5


Exect solution y=x cos(x)
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1.5
-1
-0.5
0
0.5
1
1.5


Errors
0.000018
0.000016
0.000014
0.000012
0.00001
0.000008
0.000006
0.000004
0.000002
0
-0.000002
-1.5
-1
-0.5
0
0.5
1
1.5
Errors of Runge-Kutta methods
Errors of Adams-Bashforth methods

Код програми

```
#include <iostream>
#include <math.h>
#include <string>
#include <fstream>

using namespace std;

const double P = 3.141592653589793;

double function(double x, double y)
{
	return (1 - pow(x,2))*y + cos(x)* (1 - x + pow(x,3)) - x*sin(x);
}

double* null_vector(double* array, int size)
{
	for (int i = 0; i < size; i++)
	{
		array[i] = 0;
	}
	return array;
}

int main()
{
	fstream fout;		fout.open("graph_addition.xls", ios::out);

	double h = 0.1;

	int n = (2 * 1) / h + 1;

	double* x_i = new double[n];	null_vector(x_i, n);	x_i[0] = -1;
	double* y_i = new double[n];	null_vector(y_i, n);

	double* f_i = new double[n];	null_vector(f_i, n);
```

```
for (int j = 1; j < n; j++)
{
      x_i[j] = x_i[j - 1] + h;
}

y_i[0] = x_i[0]*cos(x_i[0]);

double k1, k2, k3, k4;

fout << endl << "_____________________________________________________________" << endl;

fout <<endl<< "Runge-Kutta methods: " << endl;

for (int j = 0; j < n - 1; j++)
{
      k1 = h * function(x_i[j], y_i[j]);
      k2 = h * function(x_i[j] + (h / 2), y_i[j] + (k1 / 2));
      k3 = h * function(x_i[j] + (h / 2), y_i[j] + (k2 / 2));
      k4 = h * function(x_i[j] + h, y_i[j] + k3);

      y_i[j + 1] = y_i[j] + (k1 + 2 * k2 + 2 * k3 + k4) / 6;
}


fout << endl << "x \t y" << endl;

for (int j = 0; j < n; j++)
{
   fout << x_i[j] << " \t " << y_i[j] << endl;
}

fout << endl<< "x_i:" << endl;

for (int j = 0; j < n; j++)
{
      fout << x_i[j] << endl;
}

fout << "y_i:" << endl;
```

```
	for (int j = 0; j < n; j++)
	{
		fout << y_i[j] << endl;
	}


	fout << endl << "_____________________________________________________________" << endl;

	fout << "Adams-Bashforth methods" << endl;

	double* z_i = new double[n];  null_vector(z_i, n);

	for (int j = 0; j < n; j++)
	{
		if (j < 4)
		{
			z_i[j] = y_i[j];
		}
		else {
			z_i[j] = y_i[j - 1] + (h / 24) * (55 * function(x_i[j - 1], y_i[j - 1]) - 59 * function(x_i[j - 2],
y_i[j - 2]) + 37 * function(x_i[j - 3], y_i[j - 3]) - 9 * function(x_i[j - 4], y_i[j - 4]));
		}
	}

	fout << "x \t y" << endl;

	for (int j = 0; j < n; j++)
	{
		fout << x_i[j] << "\t" << z_i[j] << endl;
	}

    fout <<endl<< "y_i:" << endl;

	for (int j = 0; j < n; j++)
	{
		fout << z_i[j] << endl;
	}


	fout << endl << "_____________________________________________________________" << endl;
```

```
fout << "Original function:" << endl;

double* y_0 = new double[n];        null_vector(y_0, n);

for (int j = 0; j < n; j++)
{
      y_0[j] = x_i[j]*cos(x_i[j]);
}

fout << "y_0" << endl;

for (int j = 0; j < n; j++)
{
      fout << y_0[j] << endl;
}

fout << endl << "_____________________________________________________________" << endl;

fout << endl << "Errors" << endl;

double* Errors_of_Runge_Kutta = new double[n];       null_vector(Errors_of_Runge_Kutta, n);
double* Errors_of_Adams_Bashforth = new double[n];   null_vector(Errors_of_Adams_Bashforth, n);

for (int i = 0; i < n; i++)
{
      Errors_of_Runge_Kutta[i] = abs(y_i[i] - y_0[i]);
      Errors_of_Adams_Bashforth[i] = abs(z_i[i] - y_0[i]);
}

fout << "Errors of Runge-Kutta methods" << endl;

for (int i = 0; i < n; i++)
{
      fout << Errors_of_Runge_Kutta[i] << endl;
}


fout <<endl<< "Errors of Adams-Bashforth methods" << endl;
```

```
	for (int i = 0; i < n; i++)
	{
		fout << Errors_of_Adams_Bashforth[i] << endl;
	}


	fout << endl << "_____________________________________________________________" << endl;

	fout.close();

	return 0;
}
```