Санкт-Петербургский Политехнический Университет
Высшая школа прикладной математики и вычислительной физики, ФизМех
01.03.02 Прикладная математика и информатика

Отчет по лабораторной работе № 14 "Решение краевой задачи для ОДУ 2-ого порядка" Модифицированный метод суперпозиции дисциплина "Численные методы"

Выполнил студент гр. 5030102/20003	Жохов О. Д.
Преподаватель	Козлов К.Н.

Октябрь, 2024

Содержание

1 Формулировка задачи и её формализация

1.1 Формализация

Найти решение краевой задачи ОДУ второго порядка модифицированным методом суперпозиции (+метод Эйлера). Также найти решение заданой точности.

1.2 Поставленные задачи:

1. Иллюстрация работы метода. Построить графики точного и численных решений для двух фиксированных значений шага на отрезке. Построить график ошибки на отрезке для этих решений

2. Исследование точности метода. Заданная точность достигается по правилу для метода на каждом шаге, построить график изменения шага по отрезку. Построить график зависимости фактической погрешности от заданной точности

2 Алгоритм метода

Краевая задача: $r(x)y'' + p(x)y' + q(x)y = f(x), x[a, b]$

$\begin{cases} \alpha_0 y(a) + \alpha_1 y(a) = A \\ \beta_0 y(b) + \beta_1 y(b) = B \end{cases}$

ГУ: $\alpha_0 = 0, \beta_0 \neq 0, \beta_1 \neq 0$

Поиск решения в виде $y(x) = u(x) + cv(x)$, где $L(u) = f, L(v) = 0$

1. Построение двух задач Коши: из ГУ получаем начальные условия: $\begin{cases} u(a) = \frac{\alpha_0}{\alpha_0^2+\alpha_1^2} A \\ u'(a) = \frac{\alpha_1}{\alpha_0^2+\alpha_1^2} A \end{cases}$ и

   $\begin{cases} v(a) = \alpha_1 \\ v'(a) = -\alpha_0 \end{cases}$

2. Построение сеток u^h, v^h с помощью методов Эйлера

3. Вычисление константы $C = \frac{B - \beta_0 u(b) - \beta_1 u'(b)}{\beta_0 v(b) + \beta_1 v'(b)}$

Для достижения заданой точности используется правило Рунге:

1. Берётся шаг в половину длины отрезка

2. Если $|y_(2i+2)(\frac{h}{2}) - y_(i+1)(h)| < \epsilon)$, то шаг уменьшается вдвое. Иначе переходим к следующему элементу сетки

3 Предварительный анализ задачи

Краевая задача: $y'' - sin(x)y' + cos(x)y = 1 - cos(x), x[0, \frac{\pi}{2}]$

Точное решение: $y = cos(x)$

$]\alpha_0 = 1, \alpha_1 = 0, \beta_0 = 1, \beta_1 = 1$, тогда $A = 1, B = -1$

4 Ручной расчёт

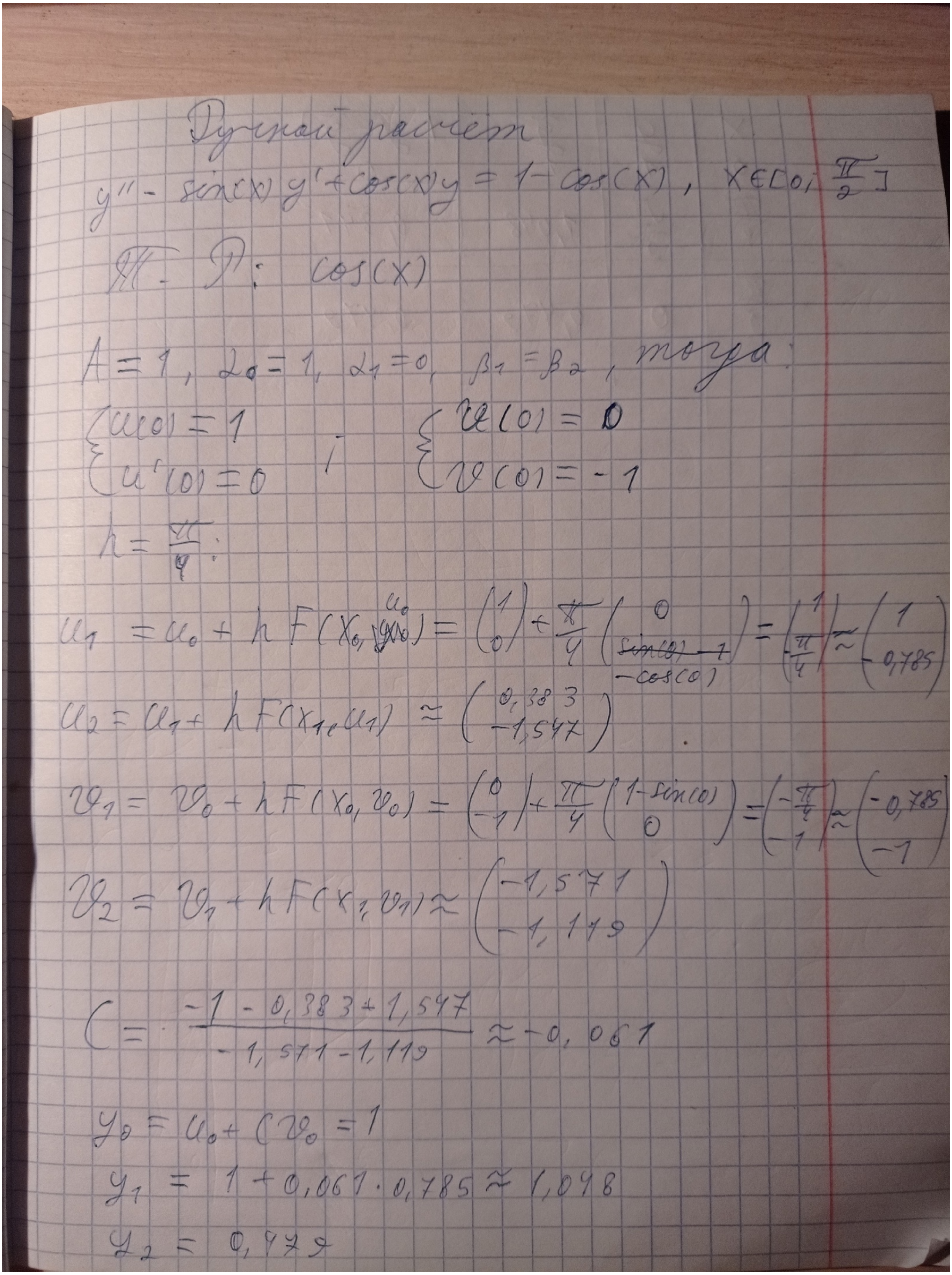
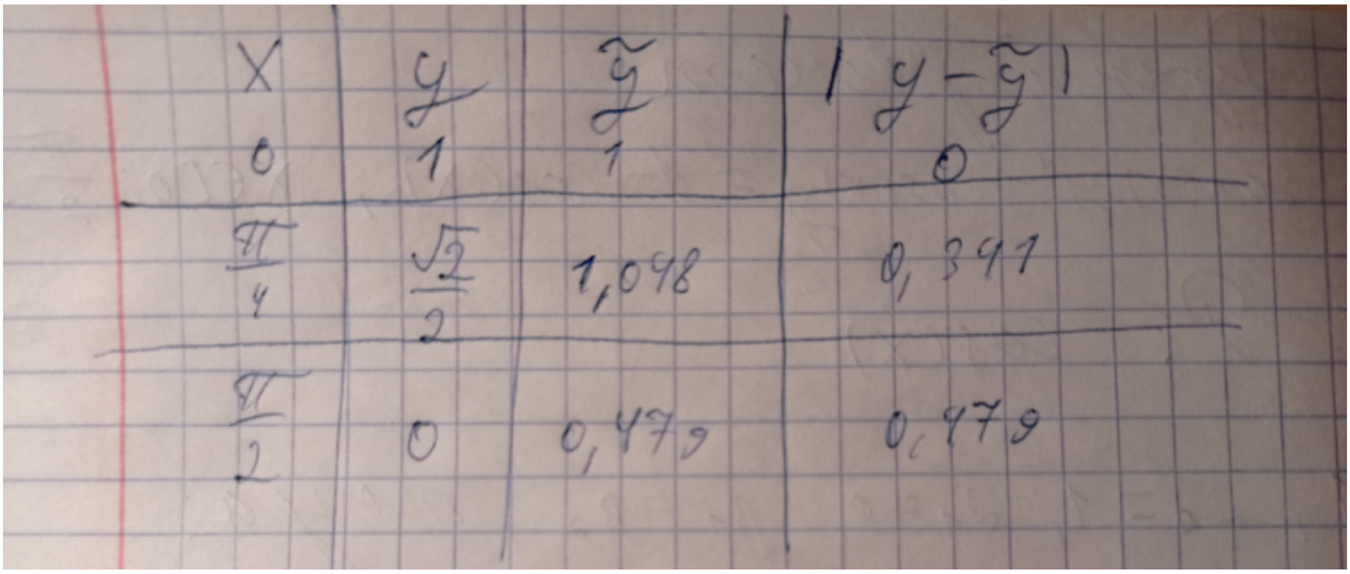


Рис. 1: Ручной расчёт, ч1



X	y	$\tilde{y}$	$\|y-\tilde{y}\|$
0	1	1	0
$\frac{\pi}{4}$	$\frac{\sqrt{2}}{2}$	1,098	0,391
$\frac{\pi}{2}$	0	0,479	0,479

Рис. 2: Ручной расчёт, ч2

5 Модульная структура программы

```
typedef struct {
        double y1;
        double y2;
}vector;
```

- структура, описывающая вектор $Y = (y(x), y'(x))^T$

```
double f(double x);
double p(double x);
double q(double x);
```

- функции, соответствующие краевой задаче

```
double F(double x, vector y);
double F_0(double x, vector y);
```

- явная запись уравнения для y'' (для уравнений с нулевым коэффициентом равным 0 и f(x))

```
vector F_vect(double x, vector y);
vector F_vect_0(double x, vector y ;
```

- правые части уравнений $Y = F(x, Y), Y = (y(x), y'(x))^T$

```
vector* find_initials(double A, double alpha_0, double alpha_1);
```

- функция, которая возвращает начальные векторы $u_0 = (u(a), u'(a))^T, v_0 = (v(a), v'(a))^T$

```
double* solve(vector* initials, double betta_0, double betta_1, double B,
double a, double b, vector (*F)(double, vector),
vector (*F_0)(double, vector), int n);
```

- функция, которая ищет решение краевой задачи для фиксированного шага функции F для количества узлов n

```
double** solve_runge(vector* initials, double betta_0,
double betta_1, double B, double a, double b,
vector (*F)(double, vector),
vector (*F_0)(double, vector), double eps, int* n_, int* len_h);
```

- функция, реализующая пошаговый контроль точности eps, возвращает двумерный массив, содержащий сетку x^h, y^h - массив выполненных шагов, массив всех шагов.

6 Численный анализ решения

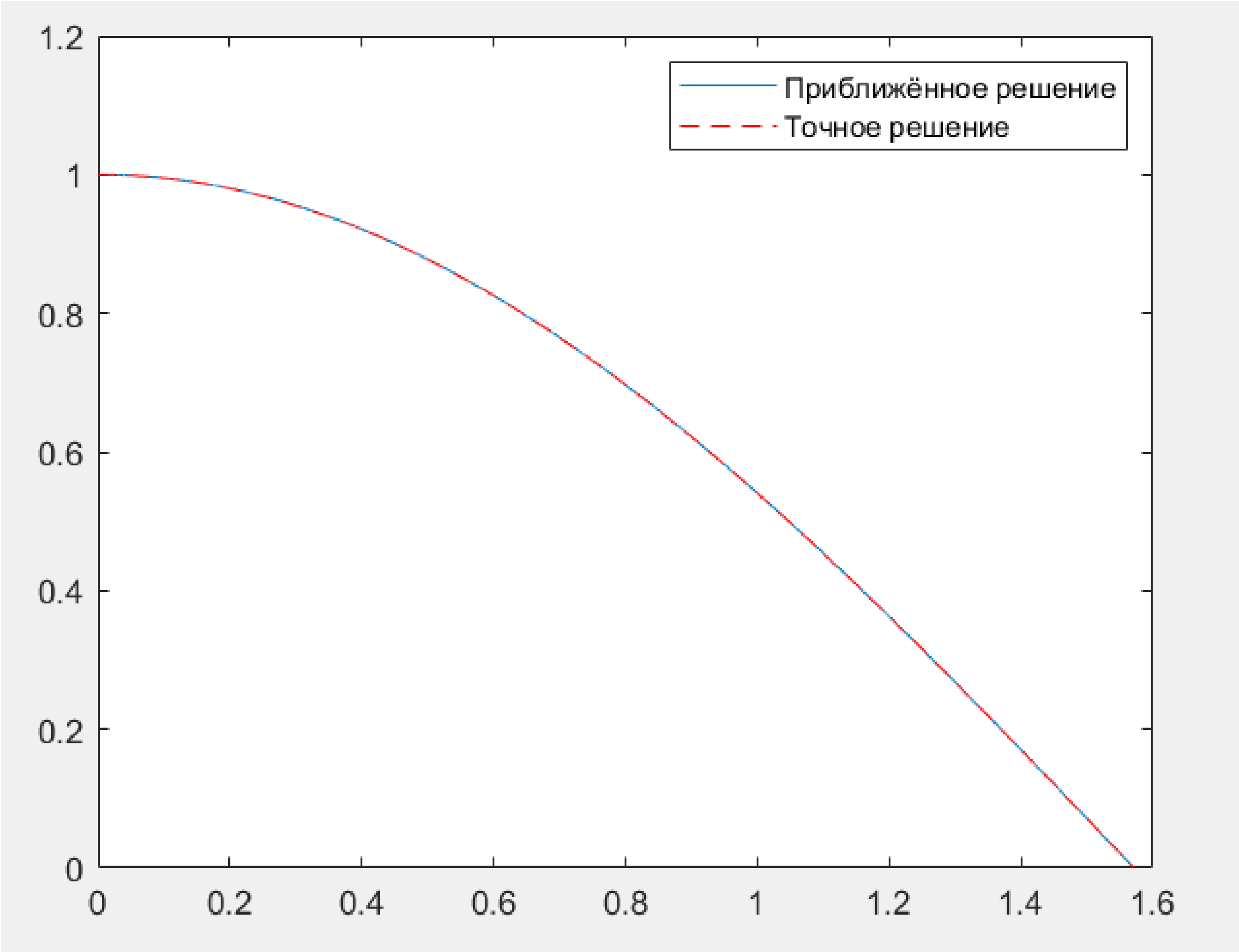


Рис. 3: Графики точного и приближённого по методу решений от x при количестве узлов n = 10000

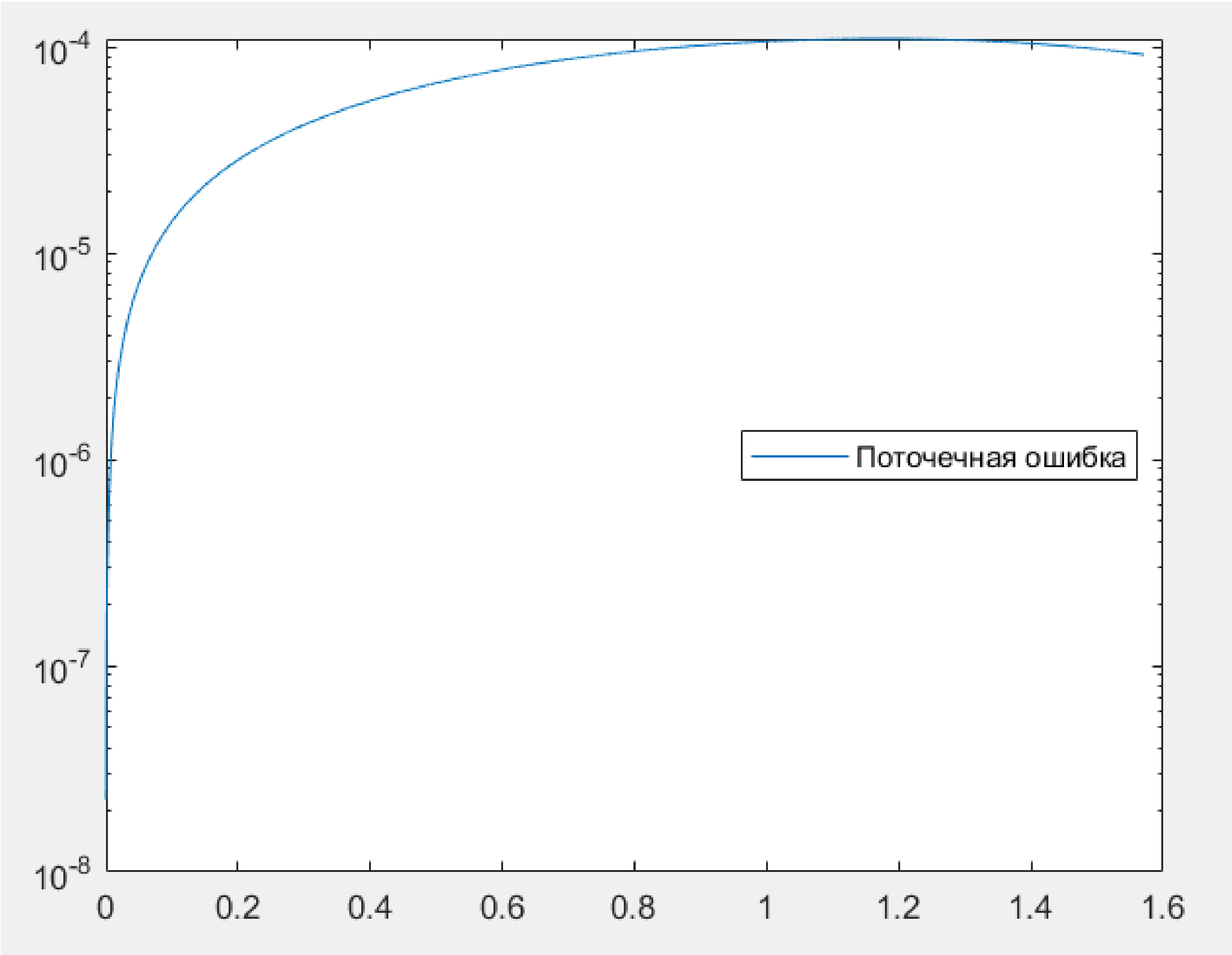


Рис. 4: График поточечной ошибки от x при количестве узлов n = 10000

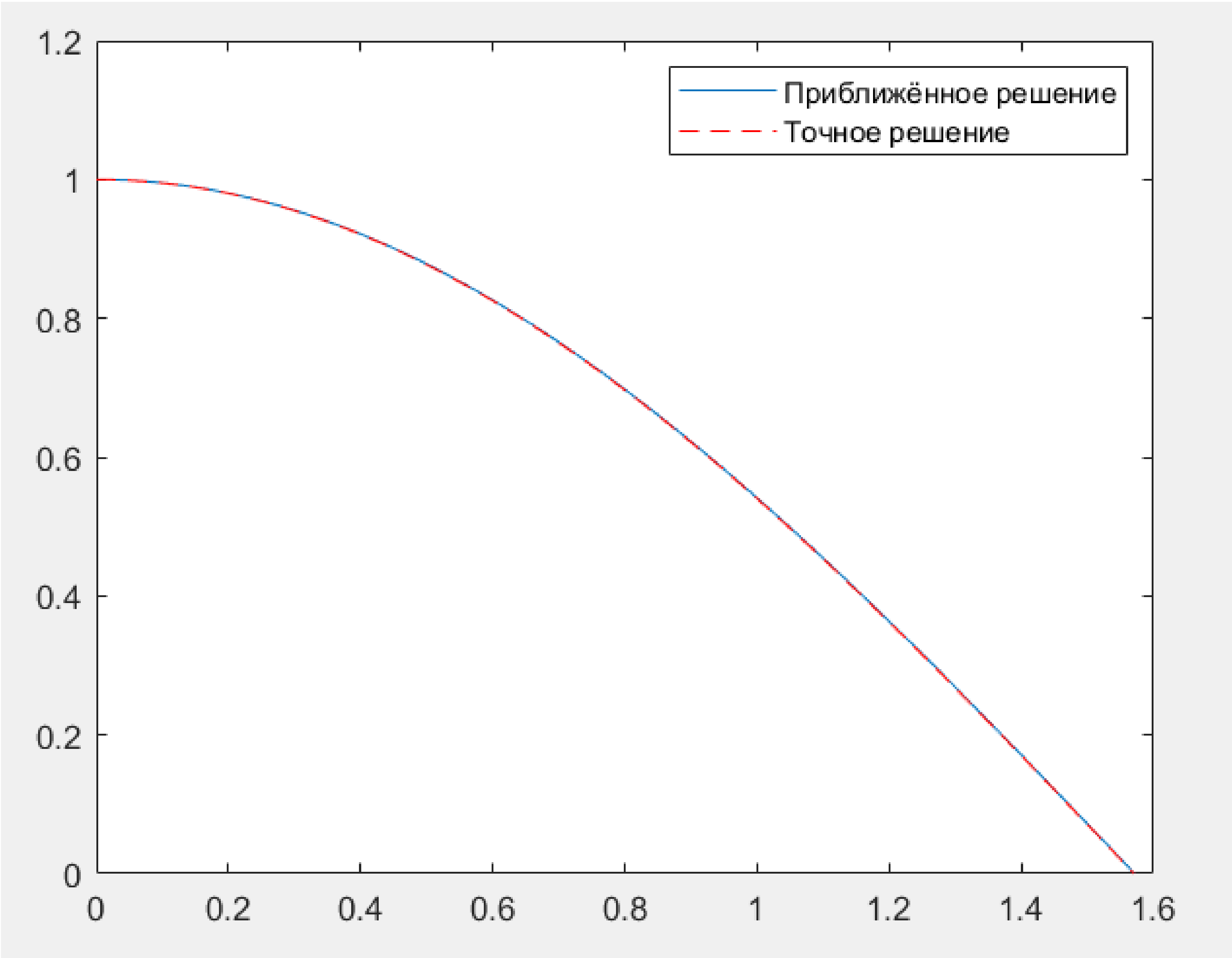


Рис. 5: Графики точного и приближённого по методу решений от x при количестве узлов n = 1000

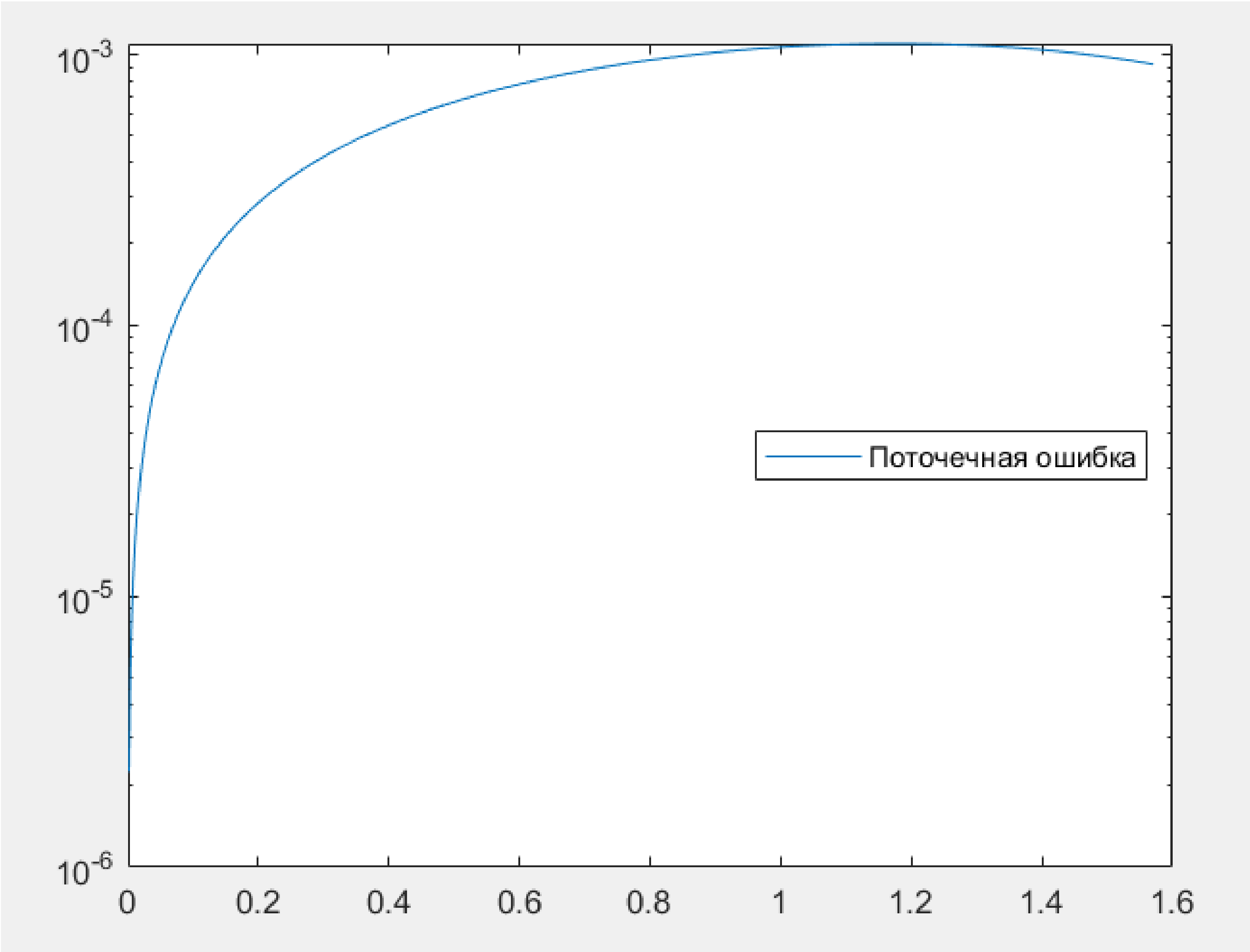


Рис. 6: График поточечной ошибки от x при количестве узлов n = 1000

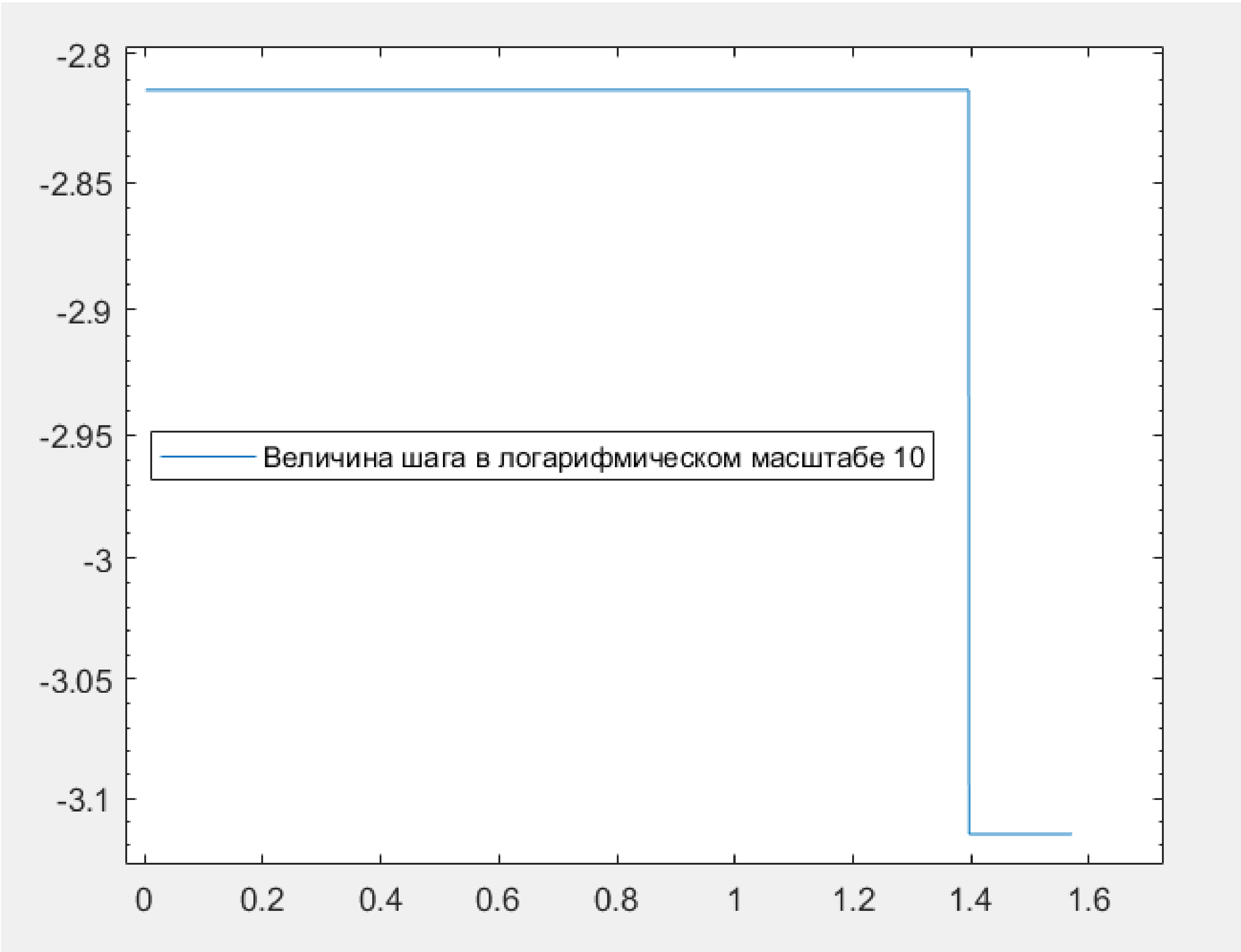


Рис. 7: График величины шага по методу рунге с заданной точносью $\epsilon = 10^{-3}$

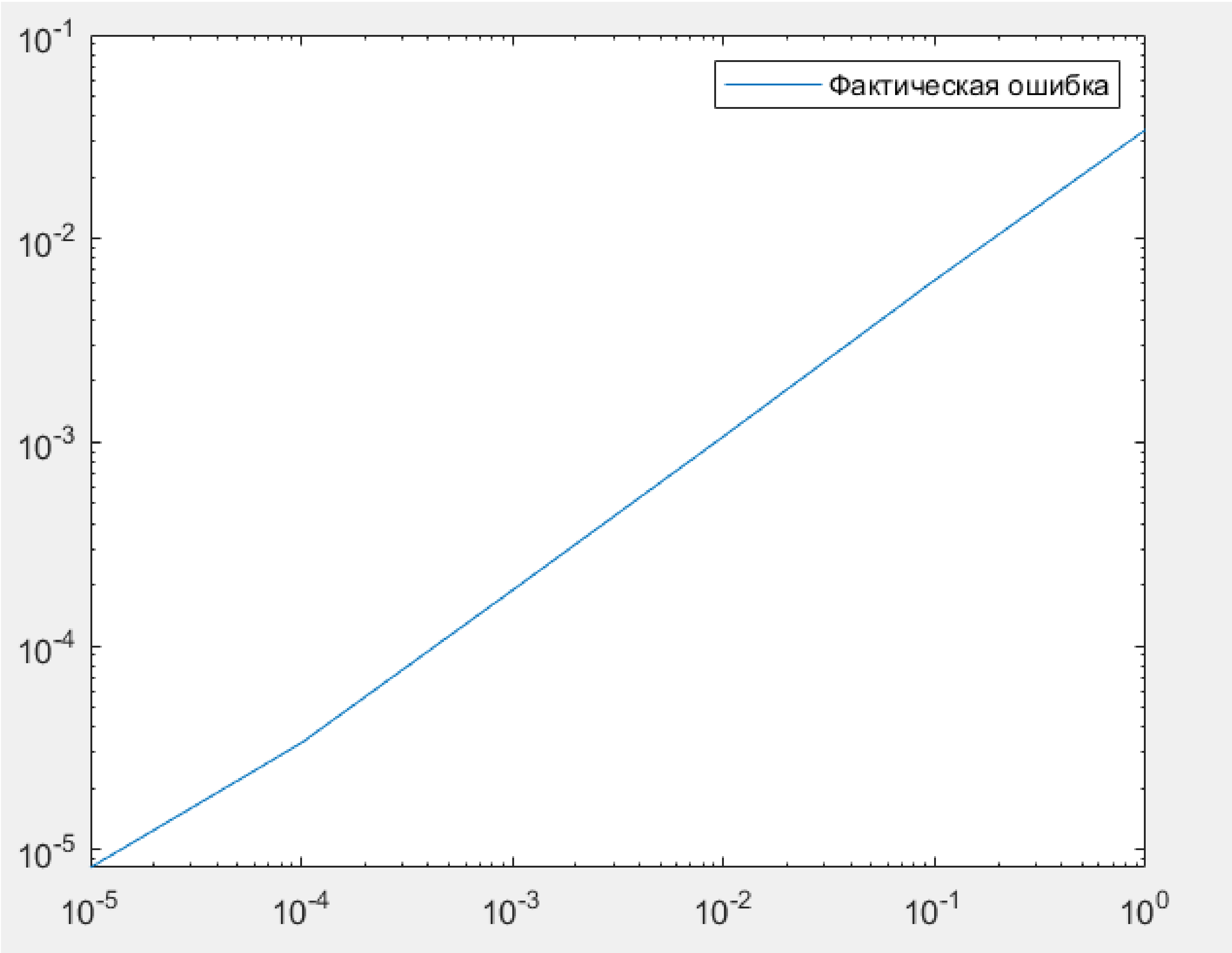


Рис. 8: График величиныфактической ошибки по методу рунге с заданной точностью от $\epsilon = 10^{-5}$ до $\epsilon = 10^{0}$

7 Выводы

1. Из рисунков (4) и (6) видно, что с уменьшением шага в 10 раз максимальная ошибка уменьшается в 10 раз (что соответсвует методу Эйлера). Чем точнее решение мы хотим получить, тем больше итераций надо произвести.

2. Из рисунка (8) видно, что фактическая ошибка никогда не превосходит заданной точности.