Федеральное государственное автономное образовательное учреждение высшего образования Национальный исследовательский университет ИТМО

Факультет программной инженерии и компьютерной техники

Вычислительная математика

Лабораторная работа №6

Вариант № 1

Выполнил: студент группы P3208, Васильев Н. А.

Преподаватель: Машина Е.А.

Санкт-Петербург 2025

# Текст задания

Численное решение обыкновенных дифференциальных уравнений.

Цель работы

Решить задачу Коши для обыкновенных дифференциальных уравнений численными методами.

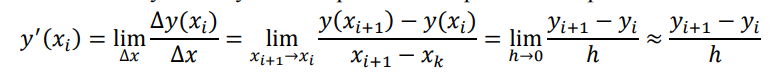
Описание метода, расчётные формулы

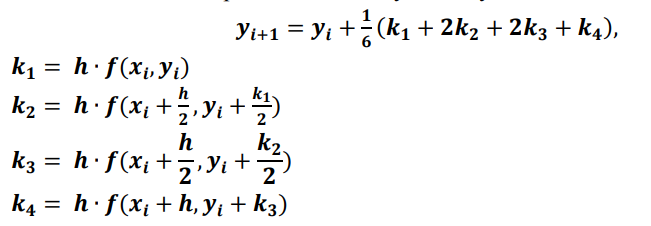


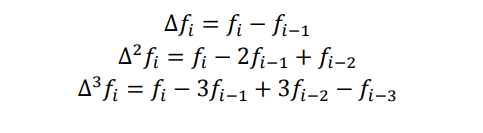














Листинг программы

import math  
  
from examples import ODU  
  
  
class Calculation:  
 def \_\_init\_\_(self, equation, initial, start, finish, h, accuracy):  
 self.equation = equation  
 self.initial = initial  
 self.start = start  
 self.finish = finish  
 self.h = h  
 self.accuracy = accuracy  
  
 def euler(self):  
 x = self.start  
 y = self.initial  
 results = [(0, x, y, self.equation(x, y))]  
 n = int((self.finish - self.start) / self.h)  
  
 for i in range(1, n + 1):  
 y += self.h \* self.equation(x, y)  
 x = self.start + i \* self.h  
 results.append((i, x, y, self.equation(x, y)))  
  
 return results  
  
 def runge\_kutt(self):  
 n = int((self.finish - self.start) / self.h)  
 x = self.start  
 y = self.initial  
 k1 = self.h \* self.equation(x, y)  
 k2 = self.h \* self.equation(x + self.h / 2, y + k1 / 2)  
 k3 = self.h \* self.equation(x + self.h / 2, y + k2 / 2)  
 k4 = self.h \* self.equation(x + self.h, y + k3)  
 results = [(0, x, y, k1, k2, k3, k4)]  
  
 for i in range(1, n + 1):  
 k1 = self.h \* self.equation(x, y)  
 k2 = self.h \* self.equation(x + self.h / 2, y + k1 / 2)  
 k3 = self.h \* self.equation(x + self.h / 2, y + k2 / 2)  
 k4 = self.h \* self.equation(x + self.h, y + k3)  
  
 results.append((i, x + self.h, y + (k1 + 2 \* k2 + 2 \* k3 + k4) / 6, k1, k2, k3, k4))  
 y += (k1 + 2 \* k2 + 2 \* k3 + k4) / 6  
 x += self.h  
  
 return results  
  
 def adams(self):  
 n = int((self.finish - self.start) / self.h)  
 x\_vals = [self.start + i \* self.h for i in range(n + 1)]  
 y\_vals = [0.0] \* (n + 1)  
  
 rk = self.runge\_kutt()  
 for i in range(4):  
 \_, xi, yi, \*\_ = rk[i]  
 x\_vals[i] = xi  
 y\_vals[i] = yi  
  
 for i in range(3, n):  
 f0 = self.equation(x\_vals[i], y\_vals[i])  
 f1 = self.equation(x\_vals[i - 1], y\_vals[i - 1])  
 f2 = self.equation(x\_vals[i - 2], y\_vals[i - 2])  
 f3 = self.equation(x\_vals[i - 3], y\_vals[i - 3])  
  
 y\_vals[i + 1] = (  
 y\_vals[i]  
 + self.h \* (55 \* f0 - 59 \* f1 + 37 \* f2 - 9 \* f3) / 24  
 )  
  
 results = []  
 for i in range(n + 1):  
 fxy = self.equation(x\_vals[i], y\_vals[i])  
 results.append((i, x\_vals[i], y\_vals[i], fxy))  
  
 return results  
  
 def exact\_solution(self, x):  
 if self.equation.\_\_code\_\_.co\_code == ODU[0].\_\_code\_\_.co\_code:  
 return -math.exp(x) / (x \* math.exp(x) - (  
 self.start \* math.exp(self.start) \* self.initial + math.exp(self.start)) / self.initial)  
 elif self.equation.\_\_code\_\_.co\_code == ODU[1].\_\_code\_\_.co\_code:  
 return (math.exp(self.start) \* self.initial + (-self.start \*\* 2 + 2 \* self.start - 2) \* math.exp(  
 self.start)) / (math.exp(x)) + x \*\* 2 - 2 \* x + 2  
 elif self.equation.\_\_code\_\_.co\_code == ODU[2].\_\_code\_\_.co\_code:  
 return (2 \* math.exp(self.start) \* self.initial - math.exp(2 \* self.start)) / (2 \* math.exp(x)) + (  
 math.exp(x) / 2)  
 else:  
 raise NotImplementedError("Точное решение не определено для этого уравнения.")  
  
 def error\_adams\_vs\_exact(self):  
 adams\_results = self.adams()  
 return max(abs(self.exact\_solution(x) - y) for (\_, x, y, \*\_rest) in adams\_results)  
  
 def error\_runge\_rule\_euler(self):  
 def method(step):  
 calc = Calculation(self.equation, self.initial, self.start, self.finish, step, self.accuracy)  
 return calc.euler()  
  
 coarse = method(self.h)  
 fine = method(self.h / 2)  
  
 errors = [  
 abs(y1 - y2)  
 for (\_, x1, y1, \*\_), (\_, x2, y2, \*\_) in zip(coarse, fine[::2])  
 if abs(x1 - x2) < 1e-8  
 ]  
 return max(errors) / (2 \*\* 1 - 1)  
  
 def error\_runge\_rule\_rk4(self):  
 def method(step):  
 calc = Calculation(self.equation, self.initial, self.start, self.finish, step, self.accuracy)  
 return calc.runge\_kutt()  
  
 coarse = method(self.h)  
 fine = method(self.h / 2)  
  
 errors = [  
 abs(y1 - y2)  
 for (\_, x1, y1, \*\_), (\_, x2, y2, \*\_) in zip(coarse, fine[::2])  
 if abs(x1 - x2) < 1e-8  
 ]  
 return max(errors) / (2 \*\* 4 - 1)

Примеры и результаты работы программы

**Пример 1:** Функция ln(x) на интервале [1; 4], разбиение на 10 точек, точка интерполяции 2:

Начальный y₀=y(x₀): -1

Начальную границу интервала: 1

Конечную границу интервала: 1.5

Шаг: 0.1

Точность: 0.01

Решение методом Эйлера:

----------------------------------------------------

| i | x | y | f(x, y) |

----------------------------------------------------

| 0 | 1.000000 | -1.000000 | 1.000000 |

----------------------------------------------------

| 1 | 1.100000 | -0.900000 | 0.801000 |

----------------------------------------------------

| 2 | 1.200000 | -0.819900 | 0.659019 |

----------------------------------------------------

| 3 | 1.300000 | -0.753998 | 0.553582 |

----------------------------------------------------

| 4 | 1.400000 | -0.698640 | 0.472795 |

----------------------------------------------------

| 5 | 1.500000 | -0.651360 | 0.409316 |

----------------------------------------------------

Погрешность метода Эйлера (правило Рунге):

0.008266163763751444

Решение методом Рунге-Кутта 4-го порядка:

--------------------------------------------------------------------------------------

| i | x | y | k1 | k2 | k3 | k4 |

--------------------------------------------------------------------------------------

| 0 | 1.000000 | -1.000000 | 0.100000 | 0.090012 | 0.091463 | 0.082489 |

--------------------------------------------------------------------------------------

| 1 | 1.100000 | -0.909093 | 0.100000 | 0.090012 | 0.091463 | 0.082489 |

--------------------------------------------------------------------------------------

| 2 | 1.200000 | -0.833337 | 0.082645 | 0.075124 | 0.076154 | 0.069339 |

--------------------------------------------------------------------------------------

| 3 | 1.300000 | -0.769234 | 0.069445 | 0.063640 | 0.064395 | 0.059098 |

--------------------------------------------------------------------------------------

| 4 | 1.400000 | -0.714289 | 0.059173 | 0.054599 | 0.055166 | 0.050968 |

--------------------------------------------------------------------------------------

| 5 | 1.500000 | -0.666670 | 0.051021 | 0.047354 | 0.047790 | 0.044405 |

--------------------------------------------------------------------------------------

Погрешность метода Рунге-Кутта (правило Рунге):

2.339298935079744e-07

Решение методом Адамса:

-------------------------------------------------------

| i | x | y (corrector) | f(x,y) |

-------------------------------------------------------

| 0 | 1.000000 | -1.000000 | 1.000000 |

-------------------------------------------------------

| 1 | 1.100000 | -0.909093 | 0.826453 |

-------------------------------------------------------

| 2 | 1.200000 | -0.833337 | 0.694454 |

-------------------------------------------------------

| 3 | 1.300000 | -0.769234 | 0.591725 |

-------------------------------------------------------

| 4 | 1.400000 | -0.714439 | 0.510577 |

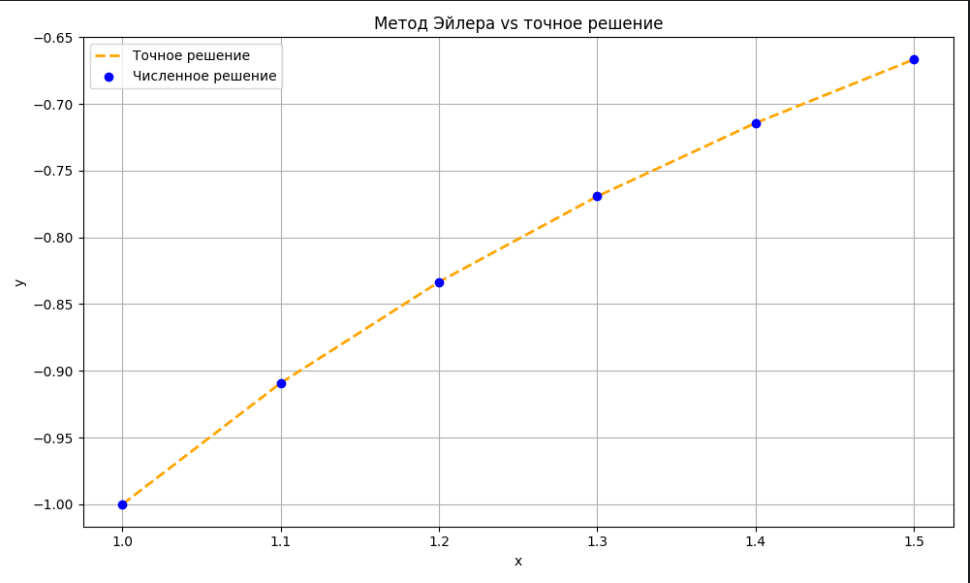
-------------------------------------------------------

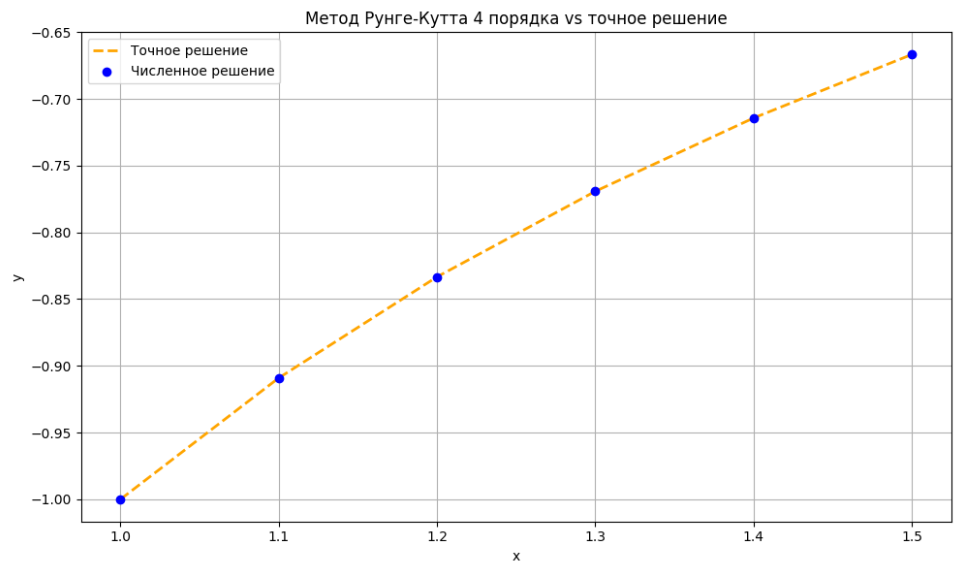
| 5 | 1.500000 | -0.666828 | 0.444821 |

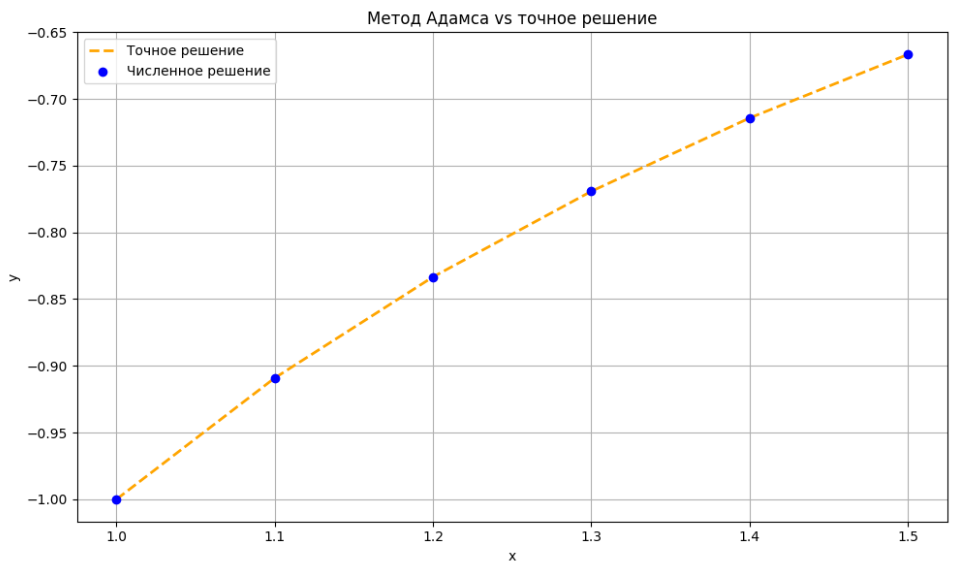
-------------------------------------------------------

Погрешность метода Адамса (сравнение с точным решением):

0.00016150509790202605







**Пример 2:** y′ = x² − y  
Начальный y₀=y(x₀): 1

Начальную границу интервала: 3

Конечную границу интервала: 6

Шаг: 0.2

Точность: 0.01

Решение методом Эйлера:

----------------------------------------------------

| i | x | y | f(x, y) |

----------------------------------------------------

| 0 | 3.000000 | 1.000000 | 8.000000 |

----------------------------------------------------

| 1 | 3.200000 | 2.600000 | 7.640000 |

----------------------------------------------------

| 2 | 3.400000 | 4.128000 | 7.432000 |

----------------------------------------------------

| 3 | 3.600000 | 5.614400 | 7.345600 |

----------------------------------------------------

| 4 | 3.800000 | 7.083520 | 7.356480 |

----------------------------------------------------

| 5 | 4.000000 | 8.554816 | 7.445184 |

----------------------------------------------------

| 6 | 4.200000 | 10.043853 | 7.596147 |

----------------------------------------------------

| 7 | 4.400000 | 11.563082 | 7.796918 |

----------------------------------------------------

| 8 | 4.600000 | 13.122466 | 8.037534 |

----------------------------------------------------

| 9 | 4.800000 | 14.729973 | 8.310027 |

----------------------------------------------------

| 10 | 5.000000 | 16.391978 | 8.608022 |

----------------------------------------------------

| 11 | 5.200000 | 18.113582 | 8.926418 |

----------------------------------------------------

| 12 | 5.400000 | 19.898866 | 9.261134 |

----------------------------------------------------

| 13 | 5.600000 | 21.751093 | 9.608907 |

----------------------------------------------------

| 14 | 5.800000 | 23.672874 | 9.967126 |

----------------------------------------------------

| 15 | 6.000000 | 25.666299 | 10.333701 |

----------------------------------------------------

Погрешность метода Эйлера (правило Рунге):

0.06837509666421937

Решение методом Рунге-Кутта 4-го порядка:

--------------------------------------------------------------------------------------

| i | x | y | k1 | k2 | k3 | k4 |

--------------------------------------------------------------------------------------

| 0 | 3.000000 | 1.000000 | 1.600000 | 1.562000 | 1.565800 | 1.534840 |

--------------------------------------------------------------------------------------

| 1 | 3.200000 | 2.565073 | 1.600000 | 1.562000 | 1.565800 | 1.534840 |

--------------------------------------------------------------------------------------

| 2 | 3.400000 | 4.078715 | 1.534985 | 1.511487 | 1.513837 | 1.496218 |

--------------------------------------------------------------------------------------

| 3 | 3.600000 | 5.564749 | 1.496257 | 1.484631 | 1.485794 | 1.479098 |

--------------------------------------------------------------------------------------

| 4 | 3.800000 | 7.042682 | 1.479050 | 1.477145 | 1.477336 | 1.479583 |

--------------------------------------------------------------------------------------

| 5 | 4.000000 | 8.528482 | 1.479464 | 1.485517 | 1.484912 | 1.494481 |

--------------------------------------------------------------------------------------

| 6 | 4.200000 | 10.035226 | 1.494304 | 1.506873 | 1.505616 | 1.521180 |

--------------------------------------------------------------------------------------

| 7 | 4.400000 | 11.573618 | 1.520955 | 1.538859 | 1.537069 | 1.557541 |

--------------------------------------------------------------------------------------

| 8 | 4.600000 | 13.152423 | 1.557276 | 1.579549 | 1.577322 | 1.601812 |

--------------------------------------------------------------------------------------

| 9 | 4.800000 | 14.778816 | 1.601515 | 1.627364 | 1.624779 | 1.652560 |

--------------------------------------------------------------------------------------

| 10 | 5.000000 | 16.458674 | 1.652237 | 1.681013 | 1.678135 | 1.708610 |

--------------------------------------------------------------------------------------

| 11 | 5.200000 | 18.196805 | 1.708265 | 1.739439 | 1.736321 | 1.769001 |

--------------------------------------------------------------------------------------

| 12 | 5.400000 | 19.997148 | 1.768639 | 1.801775 | 1.798462 | 1.832947 |

--------------------------------------------------------------------------------------

| 13 | 5.600000 | 21.862928 | 1.832570 | 1.867313 | 1.863839 | 1.899803 |

--------------------------------------------------------------------------------------

| 14 | 5.800000 | 23.796784 | 1.899414 | 1.935473 | 1.931867 | 1.969041 |

--------------------------------------------------------------------------------------

| 15 | 6.000000 | 25.800877 | 1.968643 | 2.005779 | 2.002065 | 2.040230 |

--------------------------------------------------------------------------------------

Погрешность метода Рунге-Кутта (правило Рунге):

1.5995649121691712e-06

Решение методом Адамса:

-------------------------------------------------------

| i | x | y (corrector) | f(x,y) |

-------------------------------------------------------

| 0 | 3.000000 | 1.000000 | 8.000000 |

-------------------------------------------------------

| 1 | 3.200000 | 2.565073 | 7.674927 |

-------------------------------------------------------

| 2 | 3.400000 | 4.078715 | 7.481285 |

-------------------------------------------------------

| 3 | 3.600000 | 5.564749 | 7.395251 |

-------------------------------------------------------

| 4 | 3.800000 | 7.042376 | 7.397624 |

-------------------------------------------------------

| 5 | 4.000000 | 8.528066 | 7.471934 |

-------------------------------------------------------

| 6 | 4.200000 | 10.034643 | 7.605357 |

-------------------------------------------------------

| 7 | 4.400000 | 11.573021 | 7.786979 |

-------------------------------------------------------

| 8 | 4.600000 | 13.151777 | 8.008223 |

-------------------------------------------------------

| 9 | 4.800000 | 14.778205 | 8.261795 |

-------------------------------------------------------

| 10 | 5.000000 | 16.458068 | 8.541932 |

-------------------------------------------------------

| 11 | 5.200000 | 18.196249 | 8.843751 |

-------------------------------------------------------

| 12 | 5.400000 | 19.996622 | 9.163378 |

-------------------------------------------------------

| 13 | 5.600000 | 21.862454 | 9.497546 |

-------------------------------------------------------

| 14 | 5.800000 | 23.796347 | 9.843653 |

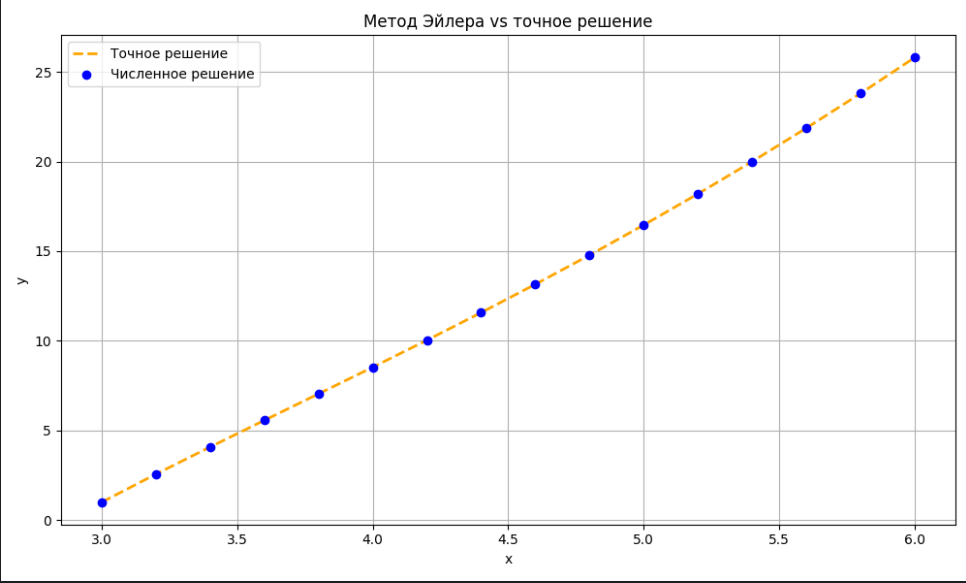
-------------------------------------------------------

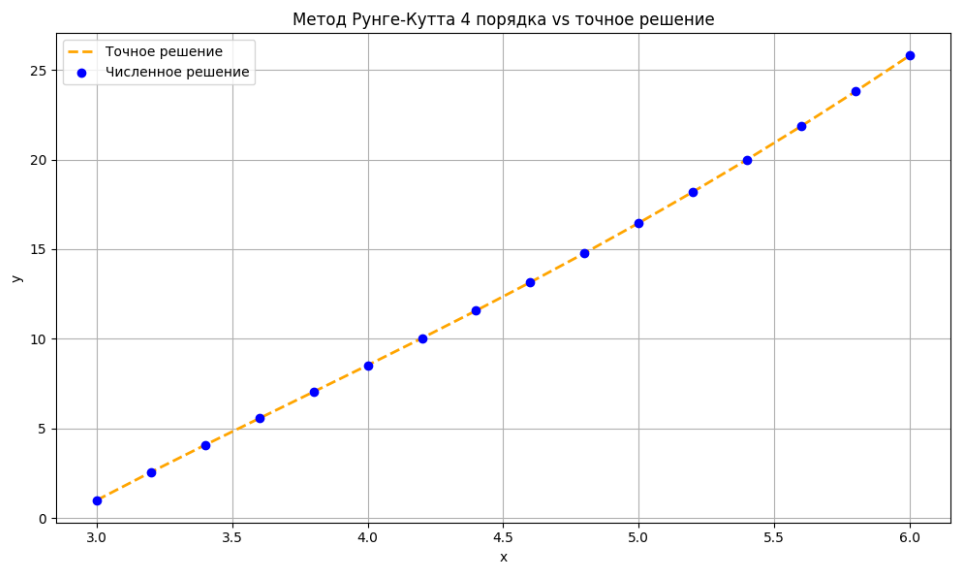
| 15 | 6.000000 | 25.800488 | 10.199512 |

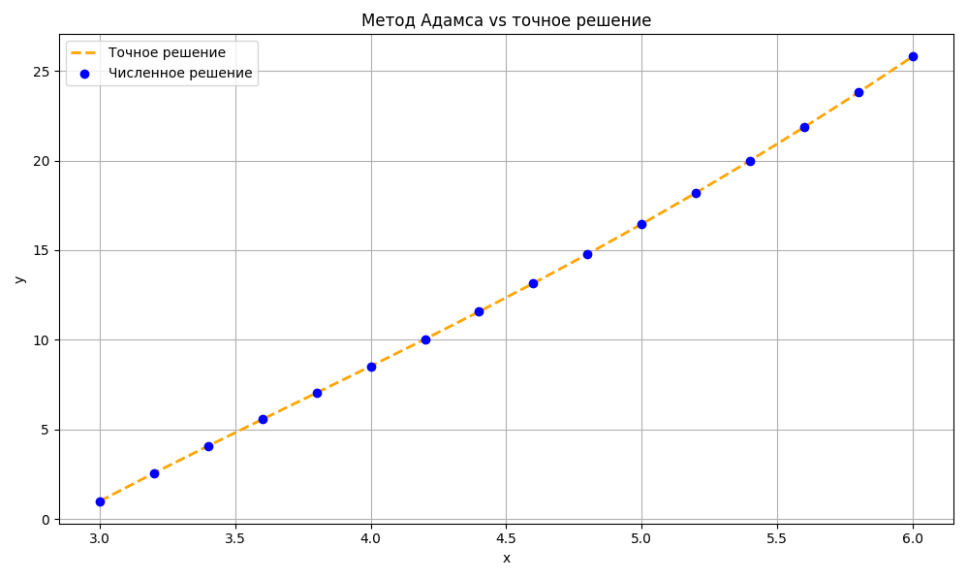
-------------------------------------------------------

Погрешность метода Адамса (сравнение с точным решением):

0.0006366995496520644







Вывод

В ходе лабораторной работы были реализованы и сравнены три численных метода решения задачи Коши для ОДУ с заданным начальным условием.