|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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 и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Лабораторная работа №*

*По предмету: «Моделирование»*

**Тема:**

*Метод Пикара решения задачи Коши*

Студент: Юмаев Артур Русланович

Группа: ИУ7-65Б

Оглавление

[Введение 3](#_Toc32337396)

[Задача Коши для дифференциального уравнения 1-го порядка 4](#_Toc32337397)

[Метод последовательных приближений (метод Пикара) 5](#_Toc32337398)

[Метод Рунге-Кутты 6](#_Toc32337399)

[Листинг 7](#_Toc32337400)

# Введение

Многообразие реальных процессов порождает большое многообразие прикладных задач, приводящих к дифференциальным уравнениям в обыкновенных и в частных производных, точное решение которых может быть получено лишь в исключительных случаях. Отсюда возникает необходимость приближенного решения таких задач. В настоящее время создано и разработано значительное число приближенных методов решения дифференциальных уравнений, изложить которые можно лишь на примере некоторых модельных задач. Одной из таких задач является задача Коши для дифференциальных уравнений в обыкновенных производных. В этой работе не ставится цель изложения всех методов решения задачи Коши, а рассматриваются лишь методы Пикара и Рунге-Кутта.

# Задача Коши для дифференциального уравнения 1-го порядка

Требуется найти функцию удовлетворяющую при дифференциальному уравнению

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

и начальному условию

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

Условия существования и единственности решения поставленной задачи будем считать выполненными.

Обычно приближенные методы разделяют на классы аналитических и численных. Аналитические методы те, что дают приближенное решение в аналитическом виде, численные в виде значений искомой функции в заранее выбранных узлах.

## Метод последовательных приближений (метод Пикара)

Интегрируя обе части уравнения (1) с учетом условия (2), получаем интегральное уравнение

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

Будем решать (3) методом последовательных приближений: задавшись произвольным начальным приближением, например, , последовательные бриближения найдем согласно следующему итерационному процессу

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

Тогда точное решение уравнения (3) находится как предел при , т. е. дополнительно требуется исследовать сходимость итерационного процесса (4). Если этот процесс сходится, то за приближенное решение задачи Коши (1)-(2) принимается при достаточно большом , определяемым уровнем допустимой точности . Этот метод имеет два существенных недостатка, которые ограничивают его применение на практике, а именно, здесь не только необходимо установление его сходимости, но и оценка скорости сходимости. Второй недостаток его состоит в том, что здесь требуется проведение операции интегрирования.

Пример 1. Найти решение задачи Коши методом Пикара

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

Найдем несколько приближений по методу Пикара. Пусть

|  |  |  |
| --- | --- | --- |
|  | , | (6) |

тогда

|  |  |  |
| --- | --- | --- |
|  |  | (7) |

|  |  |  |
| --- | --- | --- |
|  |  | (8) |

и так далее.

## Метод Рунге-Кутты

Метод Рунге-Кутты – одношаговый метод, состоящий в последовательном вычислении искомой функции задачи (1)-(2) в точках где - некоторый выбранный шаг, по некоторой расчетной формуле.

Рассмотрим детальнее получение расчетной формулы метода Рунге-Кутты для задачи (1)-(2). Проинтегрируем обе части уравнения (1) в пределах от до , получим

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

В последнем сделаем замену

|  |  |  |
| --- | --- | --- |
|  |  | (10) |

Таким образом, задача вычисления сводится к вычислению интеграла в правой части (10).

Опуская вычисления, обозначим через и получим следующую расчетную формулу

|  |  |  |
| --- | --- | --- |
|  |  | (11) |

где .

Заметим, что

|  |  |  |
| --- | --- | --- |
|  |  |  |

и т.д., т.е. – определяются последовательно, а, следовательно, определяется правая часть (11). Приведем расчетную формулу Рунге-Кутты.

|  |  |  |
| --- | --- | --- |
|  |  | (12) |

погрешность которой

# Листинг

|  |  |
| --- | --- |
|  | class Polynom:  def \_\_init\_\_(self, coef, degree):  self.coef = coef  self.degree = degree  def mult\_poly(arr):  # arr: [polynom1, polynom2, ...]  # (a + b)^2 = (a + b)(a + b) = a\*a + a\*b + b\*a + b\*2  elements = []  # Multiplying  for l\_i in arr:  for r\_j in arr:  new\_coef = l\_i.coef \* r\_j.coef  new\_degree = l\_i.degree + r\_j.degree  new\_poly = Polynom(new\_coef, new\_degree)  elements.append(new\_poly)  return elements  def integrate(arr):  t2 = Polynom(1, 2)  arr = [t2] + arr    integrated = []  for p in arr:  new\_poly = Polynom(p.coef / (p.degree + 1),  p.degree + 1)  integrated.append(new\_poly)  return integrated  def y(n):  # Starting point  x = Polynom(0, 0) # coef=1, degree=1  multiplied = [x]  for i in range(n):  multiplied\_new = mult\_poly(multiplied)  integrated\_new = integrate(multiplied\_new)  multiplied = integrated\_new  result = []  for p in multiplied:  if p.coef != 0:  result.append(p)  return result  def compute\_x(polynom, x):  result = 0  for p in polynom:  result += (p.coef \* x \*\* p.degree)  return result  def func(x, y):  return x \*\* 2 + y \*\* 2  def explicit(x, y, h):  return (y + h \* func(x, y));  def implicit(x, y, h):  K1 = func(x, y);  K2 = func(x + h / 2, y + h \* K1 / 2);  K3 = func(x + h / 2, y + h \* K2 / 2);  K4 = func(x + h, y + h \* K3);  return y + h / 6 \* (K1 + 2 \* K2 + 2 \* K3 + K4); |