МЕТОД СТРЕЛЬБЫ

Вагапов И.В., студент,
Бирский филиал УУНиТ, г. Бирск, Россия
Русинов А.А., к.ф.-м.н., доцент,
Бирский филиал УУНиТ, г. Бирск, Россия
Чиглинцева А.С., д.ф.-м.н., доцент,
УГНТУ, г. Уфа, Россия

Аннотация. Метод стрельбы является эффективным численным методом для решения краевых задач обыкновенных дифференциальных уравнений (ОДУ). В данной статье рассматривается суть метода и его реализации. Приводится пример использования метода стрельбы для решения краевой задачи на языке программирования Python, а также обсуждаются преимущества и ограничения метода.

Ключевые слова: метод стрельбы, краевая задача, обыкновенные дифференциальные уравнения, численные методы, метод Рунге-Кутты.

Краевые задачи для ОДУ возникают во многих приложениях, таких как физика, инженерия и биология. Решение таких задач часто невозможно получить в аналитическом виде, что делает численные методы незаменимыми инструментами. Метод стрельбы — один из таких подходов, который эффективно справляется с задачами, имеющими заданные граничные условия. Основная идея метода заключается в преобразовании краевой задачи в задачу начальных условий и последующем подборе параметров, обеспечивающих выполнение граничных условий[2].

Рассмотрим частную, но довольно распространенную граничную задачу. Ищется решение уравнения[4]:

$$L_y \equiv y'' - p(x)y = f(x) \text{ на } [0, X], (1)$$

при граничных условиях

$$y(0) = a, y(0) = b. (2)$$

Зададимся шагом $h = \frac{X}{N}, N -$ целое, точки $x_j = jh$ примем за узлы сетки, $y_j -$ приближения к значениям $y(x_j)$. После замены производной $y''(x_j)$ на разностное отношение $\frac{\Delta^2 y_j}{h^2} = \frac{y_{j+1}-2y_j+y_{j-1}}{h^2}$ получим систему уравнений[4]:

$$l(y_j) = \frac{y_{j+1} - 2y_j + y_{j-1}}{h^2} - p_j y_j = f_j, j = 1,2, \dots, N - 1. (3)$$

Здесь $p_j = p(x_j), f_j = f(x_j)$, граничные условия заменим соотношениями[4]:

$$y_0 = a, y_N = b. (4)$$

Можно утверждать, что при условии $p(x) \geq 0$ у системы уравнений (3) и (4) существует решение. Однородная версия системы, соответствующая этим уравнениям, обладает лишь тривиальным решением. Поскольку количество уравнений в (3) и (4) совпадает с числом переменных, неоднородная система (3) и (4) имеет единственное решение[4].

При численном решении задачи Коши значения решения вычисляются в последовательных узлах с помощью рекуррентных формул. Однако в случае граничной задачи такая методика недоступна, так как значения решения зависят от условий на границах диапазона интегрирования[4].

Метод пристрелки.

Возьмем частное решение$y_0(x)$ неоднородного уравнения[3]:

$$y_0'' - p(x)y_0 = f(x)$$

и два линейнонезависимых решения однородного уравнения

$$y_i'' - p(x)y_i = 0, i = 1,2.$$

Общее решение неоднородного уравнения запишется в виде

$$y_0(x) + C_1 y_1(x) + C_2 y_2(x);$$

постоянные C_1 и C_2 определяем из граничных условий. Приближения кфункциям $y_1(x), y_2(x)$ находим каким-либо численным методом задачиКоши; затем находим C_i, где $i = 1, 2$ и получаем нужное решение[3].

Более рациональным будет поступить следующим образом. Начнем с нахождения частного решения неоднородного уравнения.$y_0'' - p(x)y_0 = f(x)$, удовлетворяющее условию $y_0(0) = a$, и частное решение однородного уравнения $y_1(x)$,удовлетворяющее условию $y_1(0) = 0$. Общее решение неоднородного уравнения,удовлетворяющее условию $y(0) = a$ имеет вид $y_0(x) + Cy_1(x)$; значение C определяется из условия $y_0(X) + Cy_1(X) = b$. Метод решенияграничной задачи, следующий этой схеме, принято называть методомстрельбы или методом пристрелки[4].

Сеточный аналог этого метода заключается в следующем. Задаемся$y_0^0 = a, y_0^1 = 0$, произвольными $y_1^1 \neq 0$ и y_1^0, и из уравнений:

$$\frac{y_{n+1}^0 - 2y_n^0 + y_{n-1}^0}{h^2} - p_n y_n^0 = f_n;$$

$$\frac{y_{n+1}^1 - 2y_n^1 + y_{n-1}^1}{h^2} - p_n y_n^1 = 0,$$

последовательно определяем $y_2^0, \dots, y_N^0, y_2^1, \dots, y_N^1$. Затем находим C из уравнения $y_N^0 + \mathrm{C} y_N^1 = b$ и полагаем $y_n = y_n^0 + C y_n^1$; функция y_n являются требуемым решением[4].

Для наглядности приведём реализацию метода стрельбы на языке Python. Код позволяет решить краевую задачу для обыкновенного дифференциального уравнения второго порядка$y'' + 2y = 0$ с граничными условиями $y(0) = 0$ и $y(\pi) = 0$:

```
import numpy as np
from scipy.integrate import solve_ivp
# Определение системы уравнений
# Пример: y" = -2y, преобразуется в систему y1' = y2, y2' = -2y1
def odes(x, y, p):
    """
    y: вектор [y1, y2], где y1 = y, y2 = y'
p: параметр для метода стрельбы
    Возвращает систему уравнений в форме [y2, y2'].
```

```
"""
    return [y[1], -2 * y[0]]  # Пример: y" + 2y = 0
# Граничные условия
x0, x1 = 0, np.pi  # Границы диапазона
b1, b2 = 0, 0  # Значения y на x0 и x1
# Методстрельбы
def shooting_method(odes, x0, x1, b1, b2, p_guess, tol=1e-6):
"""
    Реализация метода стрельбы.
odes: функция, задающая систему ОДУ
    x0, x1: точки границ
    b1, b2: граничные условия на x0 и x1
p_guess: начальное приближение для параметра стрельбы
tol: допустимая погрешность
    Возвращает решение задачи на сетке.
    """
def boundary_difference(p):
        """
        Функция для поиска разницы между расчетным и целевым значением
y в x1.
"""
        sol = solve_ivp(odes, [x0, x1], [b1, p], t_eval=[x1], args=(p,))
returnsol.y[0, -1] - b2
    # Итеративный процесс подбора параметра p
p_low, p_high = p_guess - 1, p_guess + 1
    while boundary_difference(p_low) * boundary_difference(p_high) > 0:
p_low -= 1
p_high += 1
p_final = None
    while p_high - p_low>tol:
```

```
p_mid = (p_low + p_high) / 2
    if boundary_difference(p_mid) * boundary_difference(p_low) < 0:
p_high = p_mid
    else:
p_low = p_mid
p_final = (p_low + p_high) / 2
# Решение с найденным параметром p_final
solution = solve_ivp(odes, [x0, x1], [b1, p_final], t_eval=np.linspace(x0, x1, 100), args=(p_final,))
  return solution
# Пример использования
p_initial_guess = 1
solution = shooting_method(odes, x0, x1, b1, b2, p_initial_guess)
# Вывод результатов в виде численных данных
for x, y in zip(solution.t, solution.y[0]):
print(f"x = {x:.5f}, y = {y:.5f}")
```

```
x = 3.07813, y = -1.32436
x = 3.10986, y = -1.34526
x = 3.14159, y = -1.36345
```

Рисунок 1. Решение к данной задаче.

Этот код реализует метод стрельбы для решения краевой задачи для обыкновенного дифференциального уравнения второго порядка с заданными граничными условиями. Из-за большего количества итераций были сделаны выводы, что нужно показатьлишь концовку решения.

А что же по преимуществам и недостаткам данного метода?

Преимущества метода стрельбы[1]:

- Простота идеи и реализации.
- Высокая точность при правильном подборе параметров.

Недостатки метода[1]:

- Не всегда срабатывает, если начальные условия сильно отличаются от искомых.
- Для сложных задач может потребоваться много итераций.

Итак, метод стрельбы представляет собой эффективный инструмент для решения краевых задач в области обыкновенных дифференциальных уравнений. Хотя у него есть свои ограничения, его универсальность и сравнительно простой алгоритм делают его популярным. Он находит применение в таких областях, как физика и инженерия, где важны граничные условия. При решении более сложных задач метод стрельбы часто комбинируется с другими методами, например, методом конечных разностей или методом Галёркина. Такая интеграция позволяет нивелировать его недостатки и существенно расширить сферы его использования.

Литература

1. Shootingmethod – Wikipedia [Электронный ресурс]. Режим доступа: https://en.wikipedia.org/wiki/Shooting_method, - (дата обращения: 29.11.2024).
2. Гереева Т.Р. Учебно пособие по дисциплине «Численные методы» – Махачкала, 2014 - 109с.
3. Метод стрельбы – Рувики: Интернет-энциклопедия [Электронный ресурс]. Режим доступа: https://ru.ruwiki.ru/wiki/Метод_стрельбы, - (дата обращения: 29.11.2024).
4. Савчук В.Ф. Методы численного анализа: электрон. курс лекций для студ. специальностей «Прикладная математика» и «Экономическая кибернетика» физ.-мат. фак. / Матысик О.В.; Брест. гос. ун-т им. А.С. Пушкина, каф. ПМ и ТП, каф. алгебры и геометрии. – Брест: электрон. издание БрГУ, 2013. – 246 с.