МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В. Ломоносова

ФИЗИЧЕСКИЙ ФАКУЛЬТЕТ

КАФЕДРА ОБЩЕЙ ФИЗИКИ И ВОЛНОВЫХ ПРОЦЕССОВ

ЗАДАНИЕ №2

по курсу «Численные методы в физике»

$u_0 = 7.33$

Выполнил:
Делекторский Никита Юрьевич,
студент 425 группы

Преподаватель:
Шлёнов Святослав Александрович

Москва
2024

Оглавление

1. Постановка задачи

Численно решите следующую задачу Коши:

$$\begin{cases} \dfrac{d^2u}{dt^2} + u = 0 \\ u(t=0) = u_0, \;\; \dot{u}(t=0) = v_0 \end{cases}$$

на отрезке $t = [0,20]$ для $v_0 = 0$ и u_0, вычисляемого в соответствии с правилом*. Используйте двухслойную схему с перешагиваем. Для расчета значений функций в следующем за начальным узлом сетки используйте:

1) Точное решение. Рассмотрите три варианта шага интегрирования: а) на границе устойчивости схемы; б) шаг в два раза меньше предыдущего; в) шаг в 5 раз меньше границы устойчивости.

2) Схему Эйлера. Для основного шага интегрирования в 2 раза меньше границы устойчивости рассмотрите два варианта получения решения в следующим за начальном узле основной сетки: а) шаг в схеме Эйлера совпадает с шагом основной сетки; б) шаг в схеме Эйлера 2 раза меньше шага основной сетки.

Постройте графики численных решений совместно с графиком точного аналитического решения задачи (шкалы на осях обязательны).

Подготовьте отчет о выполненном задании в виде pdf-файла. В отчете следует отразить постановку задачи, методы ее решения и полученные результаты. На титульном листе рядом с ФИО в скобках укажите вычисленное значение u0.

(*) Правило вычисления значения u0. Записываете первую букву вашей фамилии и инициалы –ФИО. Значение u_0= [код(Ф)+код(И)+код(О)]/30. Код буквы выберите из таблицы.

В соответствии с правилом $u_0 = 7.33$.

2. Аналитическое решение

Общее решение начальной задачи для дифференциального уравнения осцилляторного типа:

$$u(t) = A\cos t + B\cos t$$

Подставляя начальные условия $v_0 = 0$ b $u_0 = 7.33$, имеем:

$$u(t) = 7.33 \cdot \cos t$$

3. Численное решение

Обозначим $v = \frac{du}{dt}$. Тогда

$$u_{n+1} = u_{n-1} + v_n \cdot 2\Delta t$$

$$v_{n+1} = v_{n-1} - u_n \cdot 2\Delta t$$

Условие устойчивости схемы:

$$\Delta t \leq \frac{1}{\omega}$$

В поставленной задаче $\omega = 1$, следовательно, схема устойчива при $\Delta t \leq 1$.

4. Листинг программы

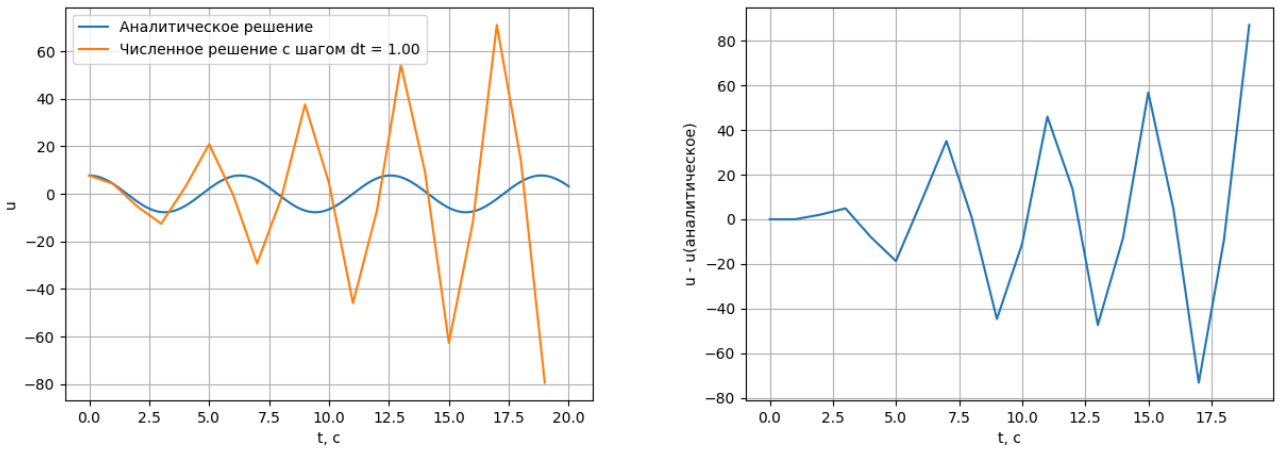
```python
import numpy as np
import matplotlib.pyplot as plt

u_0 = (50 + 150 + 32) / 30
v_0 = 0

# Схема устойчива при dt <= 1
dt = 1

t = np.arange(0, 20, dt)
u = t * 0
v = t * 0

u_analit = u_0 * np.cos(t)
v_analit = -u_0 * np.sin(t)

EULER_SCHEME = False


def euler(num_steps: int) -> np.ndarray:
    '''
    Функция для поиска второго элемента массива значений
    :param num_steps: число, на которое нужно разделить шаг основной сетки
    :return: Возращает значпения вторых координаты и скорости
    '''

    dt_ = dt / num_steps
    v_, u_ = v_0, u_0

    # Ищем второй элемент
    for _ in np.arange(num_steps):
        u_ += v_ * dt_
        v_ -= u_ * dt_

    return u_, v_


def solving(num_steps: int = 1) -> tuple:
    '''
    Функция для решения схемой с перешагиванием
    :param num_steps: Число частей, на которое нужно разделить основной шаг
сетки
    :return: :return: кортеж из двух массивов значений скорости и координат
    '''

    dt_ = dt if EULER_SCHEME else dt/num_steps
    t_ = np.arange(0, 20, dt_)
    u_ = t_ * 0
    v_ = t_ * 0
    u_[0], v_[0] = u_0, v_0

    print(f'dt = {dt_}')

    if EULER_SCHEME:
        u_[1], v_[1] = euler(num_steps)
    else:
        u_[1] = u_0 * np.cos(t_[1])
        v_[1] = -u_0 * np.sin(t_[1])

    for i in np.arange(t_.size - 2):
```

```
        u_[i + 2] = u_[i] + 2 * dt_ * v_[i + 1]
        v_[i + 2] = v_[i] - 2 * dt_ * u_[i + 1]
    return u_


if EULER_SCHEME:
    for num_steps in [1, 2]:
        print(f'num steps = {num_steps}')
        u = solving(num_steps)

        plt.figure(num=f'Решение по схеме Эйлера')
        plt.plot(t, u_analit, label=f'Аналитическое решение с шагом dt = 
{dt:.2f}')
        plt.plot(t, u, label=f'Численное решение с шагом dt = {dt / 
num_steps:.2f}')

        plt.xlabel('t, c')
        plt.ylabel('u')
        plt.grid()
        plt.show()

        plt.figure(num=f'Ошибка в схеме Эйлера')
        plt.plot(t, u_analit - u)
        plt.xlabel('t, c')
        plt.ylabel('u - u(аналитическое)')
        plt.grid()
        plt.show()

else:
    for num_steps in [1, 2, 5]:
        u = solving(num_steps)

        plt.figure(num=f'Точное решение')
        plt.plot(np.arange(0, 20, dt / num_steps),
                 u_0 * np.cos(np.arange(0, 20, dt / num_steps)),
                 label='Аналитическое решение')

        plt.plot(np.arange(0, 20, dt / num_steps),
                 u,
                 label=f'Численное решение с шагом dt = {dt / 
num_steps:.2f}')

        plt.xlabel('t, c')
        plt.ylabel('u')
        plt.grid()
        plt.legend()
        plt.show()

        plt.figure(num=f'График ошибки при dt = {dt / num_steps:.2f}')
        error = u_0 * np.cos(np.arange(0, 20, dt / num_steps)) - u
        plt.plot(np.arange(0, 20, dt / num_steps),
                 error)

        plt.xlabel('t, c')
        plt.ylabel('u - u(аналитическое)')
        plt.grid()
        plt.show()
```

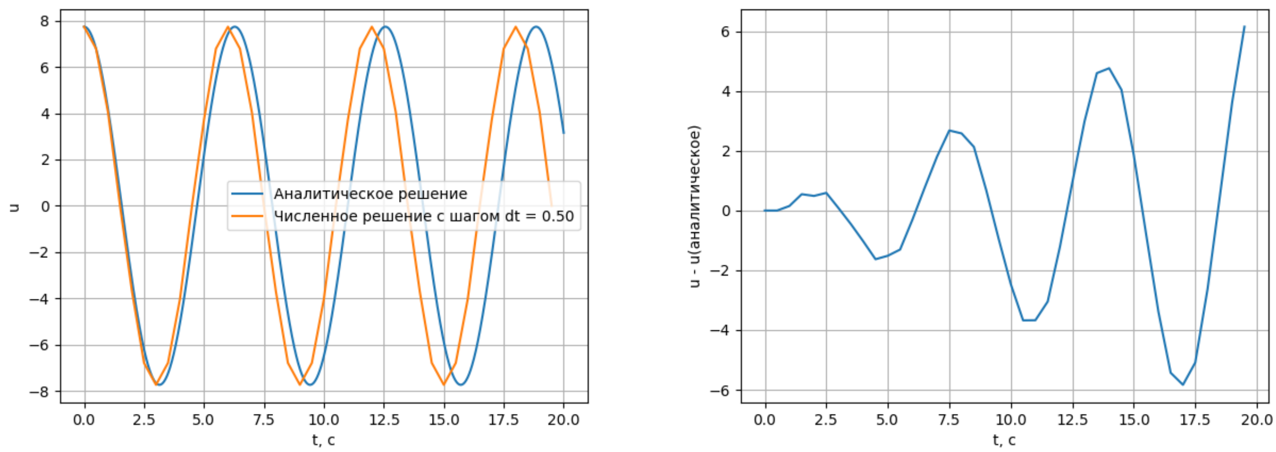
5. Результаты вычислений

Для начала рассмотрим точное решение с тремя разными шагами интегрирования.

а)



б)



в)

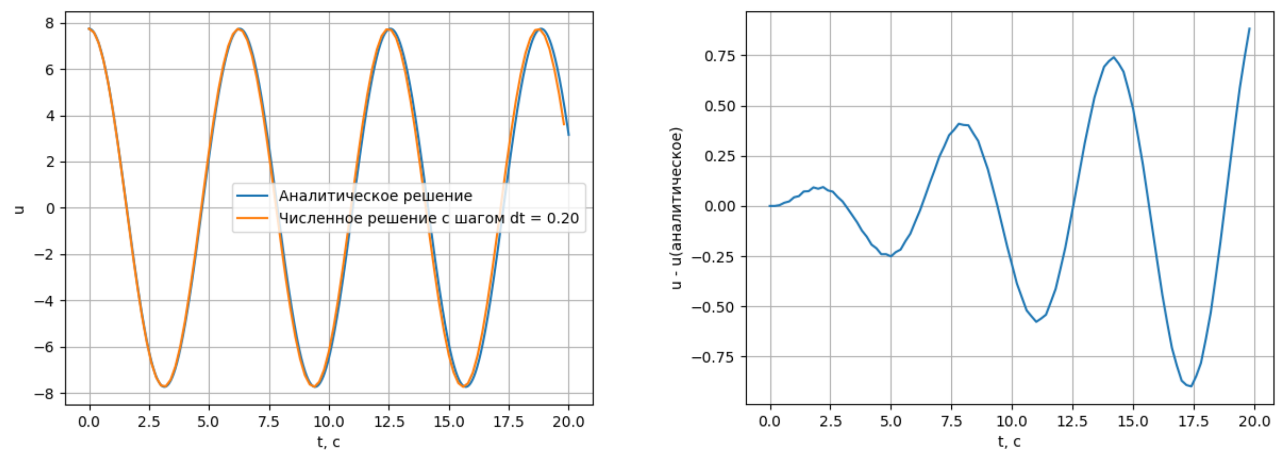
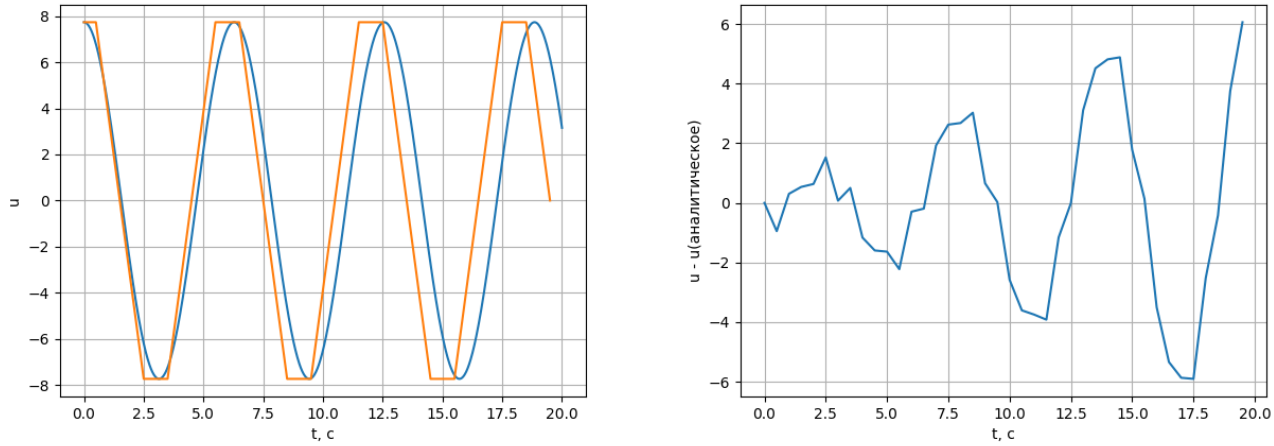


Рис. 1. Графики точных решений с шагом по времени, равным а) 1 секунде, б) 0.5 секунды, в) 0.2 секунды и ошибки каждого решения

Рассмотрим решения с использованием схемы Эйлера.

а)



б)

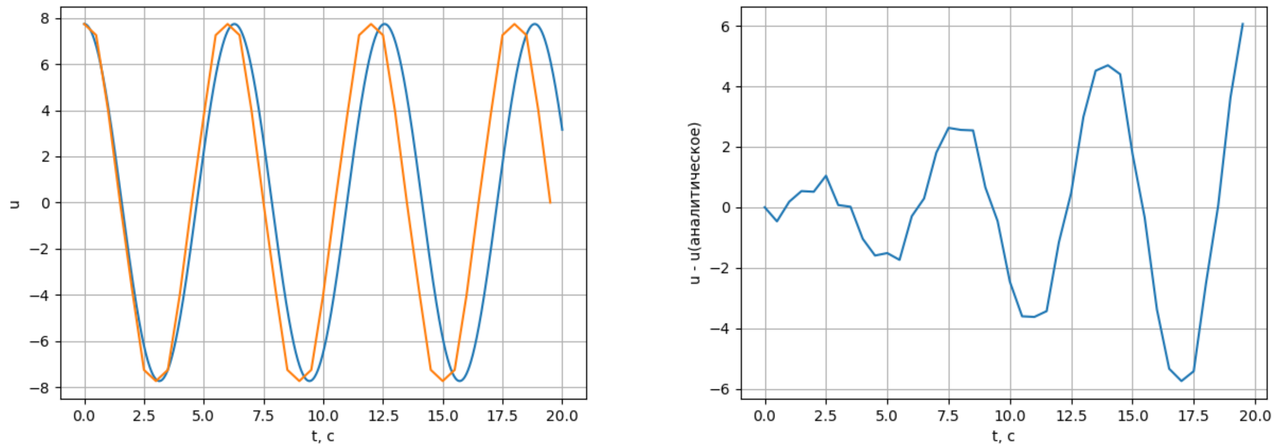


Рис. 2. Графики решений с использованием схемы Эйлера с шагом а) 0.5 секунды, б) 0.25 секунды