Компьютерные технологии

Карусевич А. А.

Задание 11 Создайте модель процесса остывания стеклянного стакана с горячим кофе при комнатных условиях. Постройте графики изменения температуры с учетом теплопроводности, конвекции и испарения, а также оцените точность интегрирования в зависимости от схемы интегрирования и величины шага интегрирования.

1 Алгоритм решения

Для моделирования процесса остывания стакана с кофе необходимо учесть несколько эффектов. Во-первых, теплопроводность.

Учет теплопроводности Теплопроводность описывается уравнением теплопроводности (Heat Equation) и в общем виде выглядит следующим образом:

$$\frac{\partial u}{\partial t} = a^2 \Delta u + f(r,t),$$

где $u(x,y,t)$ - температура в пространстве и времени, a^2 — коэффициентом температуропроводности, Δ — оператор Лапласа и $f(r,t)$ — функция тепловых источников. В нашем случае кофе наливается горячим, а после остывает при комнатных температурах, где источники отсутствуют, поэтому $f = 0$.

Чтобы описать систему кофе-стакан-воздух, введем зависимость коэффициента температуропроводности a^2 от координат. Соответственно в тех местах, где расположен кофе, будет значение температуропроводности кофе, где стекло - там температуропроводность стекла и т.д.

Рассматривать задачу будем в двух измерениях X, Y, поскольку при использовании цилиндрической системы координат от угла ничего не будет за-

висеть. В двумерной декартовой системе координат оператор Лапласа запишется как

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}.$$

Для численного расчета Лапласиана будем применять метод конечных разностей, например для частной производной по x:

$$\frac{\partial^2 u(x_i, t)}{\partial x^2} = \frac{u(x_{i+1}, t) - 2u(x_i, t) + u(x_{i-1}, t)}{\Delta x^2},$$

где x_i - принадлежит предварительно распределенной координатной сетке с заданным шагом.

Учет конвекции Чтобы учесть конвекционную передачу тепла от жидкости стенке, воспользуемся законом Ньютона — Рихмана, описывающего теплопередачу от жидкости телу. Запишем в следующей форме:

$$\frac{\partial u}{\partial n} = \frac{\alpha}{\lambda}(u_s - u),$$

где $\frac{\partial u}{\partial n}$ - производная по нормали к поверхности тела на границе тело-жидкость, т.е. стенки стакана, u_s - температура поверхности, α - коэффициент теплоотдачи, λ - коэффициент теплопроводности.

В таком виде данный закон выступает граничным условием третьего рода для уравнения теплопроводности, описанного выше.

Учет испарения В общем случае учет испарения является достаточно трудоемкой задачей. В данной работе охлаждение жидкости засчет испарения будет учитываться в зависимости от скорости испарения массы жидкости, а также удельной теплоты испарения, откуда будет находится изменение температуры.

Скорость испарения W (кг/ч) описывается как

$$W = S(b + 0.0174 \cdot V)P_V(T)(1 - \frac{h}{100}),$$

где S - площадь испаряемой жидкости (м^2), b - фактор скорости подвижности окружающего воздуха, V - скорость воздуха на поверхности испаряемой

жидкости, $P_V(T)$ - давление насыщенного пара, зависящее от температуры, h - влажность воздуха в процентах.

Зная массу испаряемой жидкости, а также удельную теплоту испарения $L \simeq 2260$ кДж/кг, можно найти теплоту, потраченную жидкостью на испарение, а значит и падение температуры в верхнем слое.

2 Результаты моделирования

Моделирование системы производится на языке Python. Используется библиотека SciPy, а также встроенный для решения дифференциальных уравнений метод integrate.solve_ivp. В данном методе используется алгоритм Рунге-Кутты 5-го порядка, а шаг интегрирования выбирается так, чтобы ошибка не превышала наперед заданного значения относительной $\varepsilon_r \leq 10^{-3}$ и абсолютной ошибок $\varepsilon_a \leq 10^{-6}$ интегрирования. Исходный код программы приведен в листинге 1.

Результаты моделирования приведены на рисунках 1, 2, 3, 4, 5, 6.

Кофе имеет изначальную температуру 350 K, а стекло и воздух вокруг 300 K. Изначальное распределение температуры приведено на рис. 1. На рисунках 2, 3, 4, 5 показаны распределения температуры в разные моменты времени - 10, 30, 120 секунд с начала моделирования

На рис. 6 приведена зависимость температуры верхней части кофе от времени.

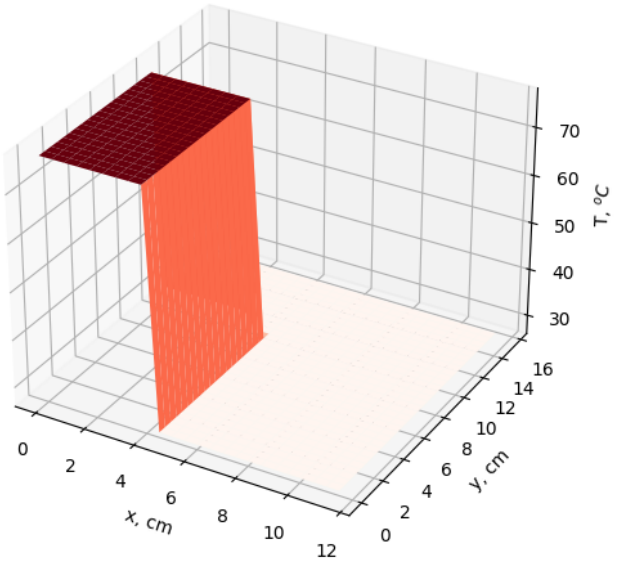

Рис. 1: Начальное распределение температуры. До $x = 4$ см расположено кофе, далее до $x = 5$ см стеклянный стакан. Далее пространство заполнено воздухом

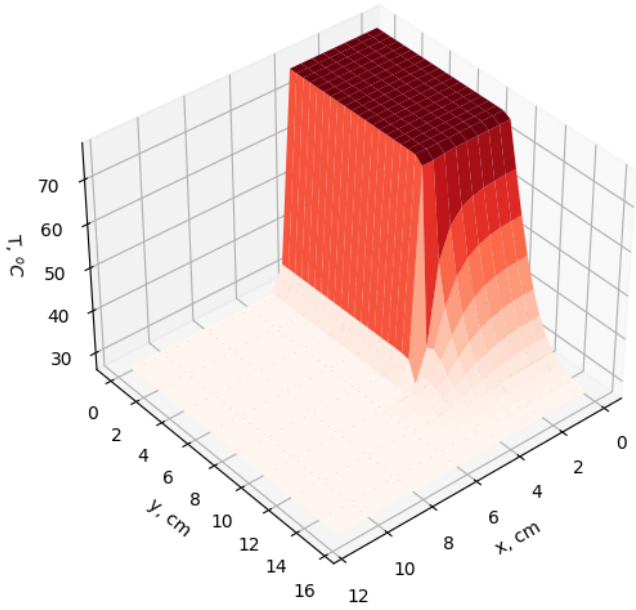

Рис. 2: Распределение температуры в $t = 10$с

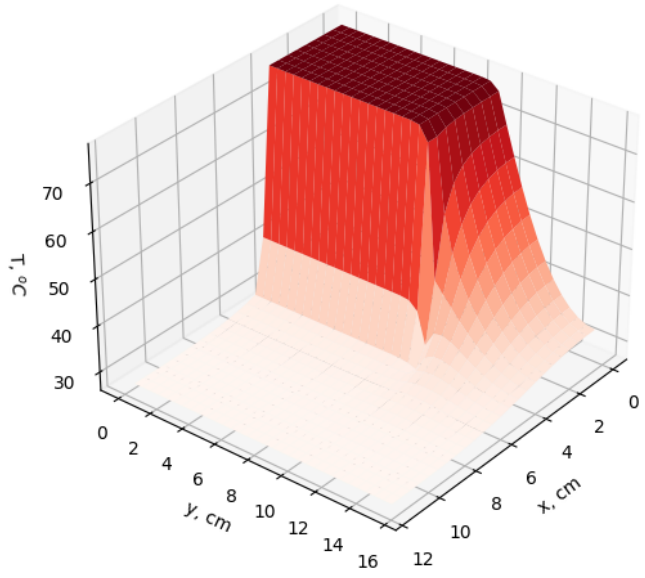

Рис. 3: Распределение температуры в $t = 30$с

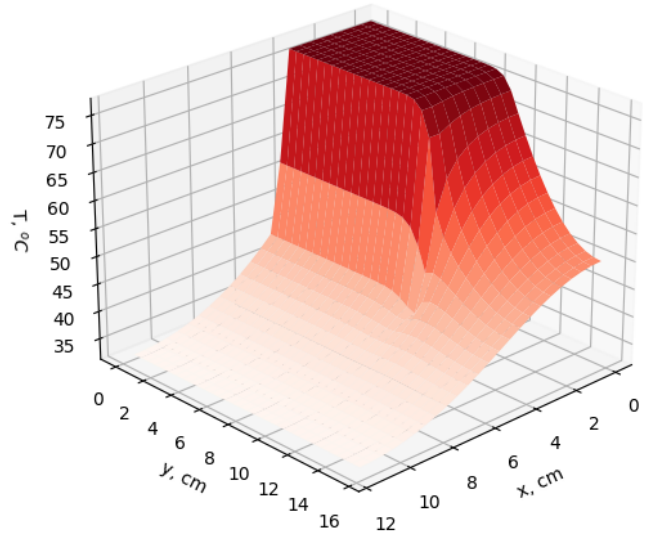

Рис. 4: Распределение температуры в $t = 120$с

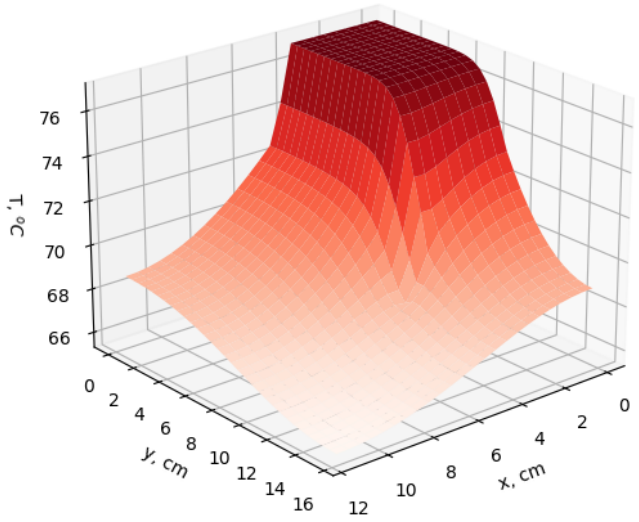

Рис. 5: Распределение температуры в $t = 600$с

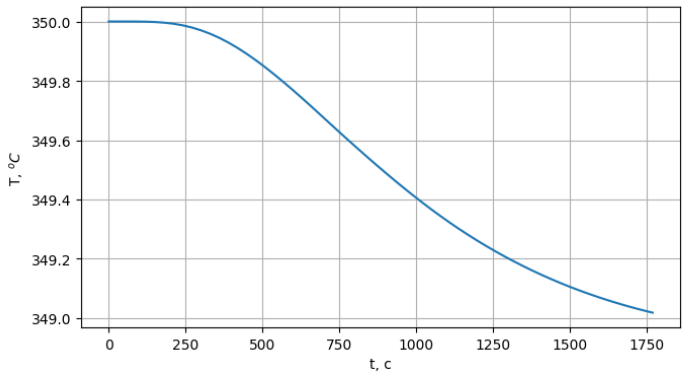

Рис. 6: Зависимость температуры в верхней части стакана кофе от времени ($x = 2, y = 0.97$ см).

3 Исходный код

```
# Задание 11. Создайте модель процесса остывания стеклянного стакана с горячим кофе при
# комнатных условиях. Постройте графики изменения температуры с учетом теплопроводности,
# конвекции и испарения, а также оцените точность интегрирования в зависимости от схемы
# интегрирования и величины шага интегрирования.

from matplotlib.colors import Colormap
import numpy as np
from scipy.integrate import solve_ivp
from scipy.optimize import minimize
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

AMBIENT_TEMP = 300 # Комнатная температура, K
HUMIDITY = 85 # Влажность, %
V_AIR = 0.1 # Скорость воздуха m/s

# Размеры сетки, в метрах
X_ROOM = 0.12 # m
Y_ROOM = 0.16 # m

# Температуропроводности веществ
T_COND_WATER = 0.143*10**(-6)
T_COND_GLASS = 3.4*10**(-7)
T_COND_AIR = 1.9*10**(-5)

L = 2260000 # Удельная теплота парообразования джкг/
Cp = 4200 # Удельная теплоемкость, ДжкгК//
rho = 1000 # Плотность воды кгм/3

# Давление насыщенного пара воды, ммртст..
P_v_exp = np.array([4.585, 6.545, 9.212, 12.79, 17.54, 23.77, 31.84, 42.20, 55.37, 71.93, 92.59, 118.1])
P_v_exp_temps = np.array([0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55]) + 273

# Функция для расчета давления пара для заданой температуры
def get_Pv(T):
    return np.interp(T, P_v_exp_temps, P_v_exp)

coffee_start_temp = 350 # Начальная температура кофе, K

# Внутренние размеры жидкости
x_coffee = 0.04 #m
y_coffee = 0.1 #m

# Координаты стакана
x_glass1 = 0.04
```

```
46  x_glass2 = 0.05
47  y_glass1 = 0
48  y_glass2 = y_coffee
49
50  # Шаг координатной сетки
51  x_step = 0.005
52  y_step = 0.005
53
54
55  def get_laplacian(T):
56      ''' Рассчет лапласиана для уравнения теплопроводности
57      '''
58      d2Tdx2 = np.zeros_like(T)
59      d2Tdy2 = np.zeros_like(T)
60      size_x = len(T[0])
61      size_y = len(T[:,0])
62      for j in range(0, size_y):
63          for i in range(0, size_x):
64              if i==0:
65                  d2Tdx2[j, i] = ( 2*T[j, i+1] - 2*T[j, i]) / x_step**2
66              elif i==size_x-1:
67                  d2Tdx2[j, i] = ( -2*T[j, i] + 2*T[j, i-1] ) / x_step**2
68              else:
69                  d2Tdx2[j, i] = ( T[j, i+1] - 2*T[j, i] + T[j, i-1] ) / x_step**2
70
71              if j==0:
72                  d2Tdy2[j, i] = ( 2*T[j+1, i] - 2*T[j, i]) / y_step**2
73              elif j==size_y-1:
74                  d2Tdy2[j, i] = ( -2*T[j, i] + 2*T[j-1, i] ) / y_step**2
75              else:
76                  d2Tdy2[j, i] = ( T[j+1, i] - 2*T[j, i] + T[j-1, i] ) / y_step**2
77
78      return d2Tdx2 + d2Tdy2
79
80
81  def get_evaporation_speed(T):
82      # Скорость испарения, кгс/
83      W = x_coffee*x_step * (0.022 + 0.0174 * V_AIR)*get_Pv(T)/1000*(1 - HUMIDITY/100) / 3600
84      # W = np.pi*x_coffee**2 * (0.022 + 0.0174 * V_AIR)*get_Pv(T)*(1 - HUMIDITY/100) / 3600
85      return W
86
87
88  def heat_equation(t, params):
89      T = params[0:-1]
90      T = T.reshape(initial_shape)
91      m = params[-1]
92      # Учет испарения
93      dmdt = get_evaporation_speed(T[j_bnd, int(i_bnd/2)])
```

```
    dT = np.zeros_like(T)
    dT_evap = L*m/Cp/(rho*np.pi*x_coffee**2*y_step)
    dT[j_bnd,0:i_bnd] = dT_evap
    # Учет конвекции со стенкой
    lamb = 0.6
    for j in range(0, len(T[:,0])-1):
        if j < j_bnd:
            T[j, i_bnd] = (1 + x_step*120/lamb) * T[j, i_bnd-1] / 2

    dTdt = T_COND*get_laplacian(T) - dT

    return np.hstack((dTdt.flatten(), dmdt))


x = np.arange(0, X_ROOM, x_step)
y = np.arange(0, Y_ROOM, y_step)

i_bnd = None
j_bnd = None

X, Y = np.meshgrid(x, y)
start_temp_distr = np.zeros((len(y), len(x))) + AMBIENT_TEMP
T_COND = np.zeros_like(start_temp_distr)
# Задаем начальное распределение температуры, температуропроводности
for i, valx in enumerate(x):
    if not i_bnd:
        if valx>=x_coffee:
            i_bnd = i
            print(i_bnd)
    for j, valy in enumerate(y):
        if not j_bnd:
            if valy>=y_coffee:
                j_bnd = j
                print(j_bnd)
        if valx<x_coffee and valy<y_coffee:
            start_temp_distr[j, i] = coffee_start_temp
            T_COND[j, i] = T_COND_WATER
        elif valx>=x_glass1 and valx<=x_glass2 and valy>=y_glass1 and valy<=y_glass2:
            T_COND[j, i] = T_COND_GLASS
        else:
            T_COND[j, i] = T_COND_AIR

# Время окончания моделирования, с
t_end = 60
time_points = np.arange(0, t_end, 30)
initial_shape = start_temp_distr.shape
print("Init shape", initial_shape)
sol1 = solve_ivp(heat_equation, [0, t_end], np.hstack((start_temp_distr.flatten(), 0)),
```

```
                t_eval=time_points,
                method='Radau')
T = sol1.y
t = sol1.t
Ts = sol1.y[:,-1]
Ts = Ts[0:-1].reshape(initial_shape)


fig = plt.figure(figsize=(6,6))
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X*100, Y*100, Ts-273, cmap='Reds',
                    linewidth=0, antialiased=True)

ax.set_ylabel('y, cm')
ax.set_xlabel('x, cm')
ax.set_zlabel('T, $^oC$')

plt.show()
```

Листинг 1: Исходный код задания