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 № 4

Тема: Программно- алгоритмическая реализация моделей на основе дифференциальных уравнений в частных производных с краевыми условиями II и III рода.

Студент Оберган Т.М.

Группа ИУ7-65Б

Оценка (баллы) ________________

Преподаватель Градов В.М.

Москва.
2020 г.

Оглавление

Цель работы: получение навыков разработки алгоритмов решения смешанной краевой задачи при реализации моделей, построенных на квазилинейном уравнении параболического типа.

Исходные данные

1. Задана математическая модель.

Уравнение для функции $T(x,t)$

$$c(T)\frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(k(T)\frac{\partial T}{\partial x}\right) - \frac{2}{R}\alpha(x)T + \frac{2T_0}{R}\alpha(x) \tag{1}$$

Краевые условия

$$\begin{cases} t = 0, \quad T(x,0) = T_0, \\ x = 0, \quad -k(T(0))\dfrac{\partial T}{\partial x} = F_0, \\ x = l, \quad -k(T(l))\dfrac{\partial T}{\partial x} = \alpha_N\,(T(l) - T_0) \end{cases} \tag{2}$$

В обозначениях уравнения (14.1) лекции №14

$$p(x) = \tfrac{2}{R}\alpha(x) \quad f(u) \equiv f(x) = \tfrac{2T_0}{R}\alpha(x) \tag{3}$$

2. Разностная схема с разностным краевым условием при $x = 0$

$$\left(\frac{h}{8}\widehat{c_{\frac{1}{2}}} + \frac{h}{4}\,\widehat{c_0} + \widehat{X_{\frac{1}{2}}}\frac{\tau}{h} + \frac{\tau h}{8}\,p_{\frac{1}{2}} + \frac{\tau h}{4}p_0\right)\widehat{y_0} + \left(\frac{h}{8}\,\widehat{c_{\frac{1}{2}}} - \widehat{X_{\frac{1}{2}}}\frac{\tau}{h} + \frac{\tau h}{8}p_{\frac{1}{2}}\right)\widehat{y_1} = \frac{h}{8}\,\widehat{c_{\frac{1}{2}}}\,(y_0 + y_1) + \frac{h}{4}\,\widehat{c_0}\,y_0 + \hat{F}\,\tau + \frac{\tau h}{4}\,(\widehat{f_{\frac{1}{2}}} + \widehat{c_0}\,) \tag{4}$$

При получении разностного аналога краевого условия при $x = l$ необходимо учесть, что поток

$$F_N = \alpha_N(\widehat{y_N} - T_0\,)$$
$$F_{N-\frac{1}{2}} = \widehat{X_{N-\frac{1}{2}}}\,\frac{\widehat{y_{N-1}} - \widehat{y_N}}{h} \tag{5}$$

3. Значения параметров для отладки (все размерности согласованы)

$k(T) = a_1(b_1 + c_1 T^{m_1})$, Вт/см К,

$c(T) = a_2 + b_2 T^{m_2} - \frac{c_2}{T^2}$, Дж/см^{3}К.

a_1 =0.0134, b_1 =1, c_1 =4.35 10^{-4}, m_1 =1,

a_2 =2.049, b_2 =0.563 10^{-3}, c_2 =0.528 10^5, m_2 =1.

$\alpha(x) = \frac{c}{x-d}$,

$\alpha_0 = 0.05$ Вт/см^2 К,

$\alpha_N = 0.01$ Вт/см^2 К,

$l =$ 10 см,

$T_0 = 300$К,

$R = 0.5$ см,

$F(t) = 50$ Вт/см^2 (для отладки принять постоянным).

Физическое содержание задачи

Постановки задач в данной лабораторной работе и работе №3 во многом совпадают. Отличия заключаются в следующем:

1. Сформулированная в данной работе математическая модель описывает нестационарное температурное поле $T(x,t)$, зависящее от координаты x и меняющееся во времени.
2. Свойства материала стержня привязаны к температуре, т.е. теплоемкость и коэффициент теплопроводности $c(T), k(T)$ зависят от Т, тогда как в работе №3 k(x) зависит от координаты, а c = 0.
3. При $x = 0$ цилиндр нагружается тепловым потоком $F(t)$, в общем случае зависящим от времени, а в работе №3 поток был постоянный.

Если в настоящей работе задать поток постоянным, т.е. F(t) = const, то будет происходить формирование температурного поля от начальной температуры T_0 до некоторого установившегося (стационарного) распределения

T(x, t). Это поле в дальнейшем с течением времени меняться не будет и должно совпасть с температурным распределением T(x), получаемым в лаб. работе №3, если все параметры задач совпадают, в частности, вместо k(T) надо использовать k(x) из лаб. работы №3. Это полезный факт для тестирования программы.

Если после разогрева стержня положить поток F(t) = 0, то будет происходить остывание, пока температура не выровняется по всей длине и не станет равной T_0.

При произвольной зависимости потока F(t) от времени температурное поле будет как-то сложным образом отслеживать поток.

Замечание. Варьируя параметры задачи, следует обращать внимание на то, что решения, в которых температура превышает примерно 2000К, физического смысла не имеют и практического интереса не представляют.

Листинг

```
# Лабораторная по моделированию №4
# Реализация модели на основе ДУ в частных производных
# с краевыми условиями II и  III рода.

import matplotlib.pyplot as plt
import numpy as np
from math import fabs
from mpl_toolkits.mplot3d import Axes3D


# Исходные данные модели
def k(T):
    return a1 * (b1 + c1 * T**m1)

def c(T):
    return a2 + b2 * T**m2 - (c2 / T**2)

def alpha(x):
    d = (alphaN*l) / (alphaN-alpha0)
    c = - alpha0 * d
    return c / (x-d)

def p(x) :
    return (2/R) * alpha(x)

def f(x):
    return (2*T0/R) * alpha(x)


def A(T):
    return t/h * approc_minus_half(k, T, t)

def D(T):
    return t/h * approc_plus_half(k, T, t)

def B(x, T):
    return A(T) + D(T) + h*c(T) + h*t*p(x)

def F(x, T):
    return h*t*f(x) + T*h*c(T)


# Простая аппроксимация
def approc_plus_half(func, n, step):
    return (func(n) + func(n + step)) / 2

def approc_minus_half(func, n, step):
    return (func(n) + func(n - step)) / 2


# Краевые условия
# При x = 0
def left_boundary_condition(T_prev):
    T_prev_0 = T_prev[0]
    c_plus = approc_plus_half(c, T_prev_0, t)
    k_plus = approc_plus_half(k, T_prev_0, t)
    c0 = c(T_prev_0)

    K0 = h/8 * c_plus + h/4 * c0 + t/h * k_plus + \
         t * h/8 * p(h/2) + t * h/4 * p(0)
```

```python
    M0 = h/8 * c_plus - t/h * k_plus + t * h/8 * p(h/2)

    P0 = h/8 * c_plus * (T_prev_0 + T_prev[1]) + \
         h/4 * c0 * T_prev_0 + F0 * t + t * h/8 * (3 * f(0) + f(h))

    return K0, M0, P0

# При x = N
def right_boundary_condition(T_prev):
    T_prev_N = T_prev[-1]
    c_minus = approc_minus_half(c, T_prev_N, t)
    k_minus = approc_minus_half(k, T_prev_N, t)
    cN = c(T_prev_N)

    KN = h/8 * c_minus + h/4 * cN + t/h * k_minus + t * alphaN + \
         t * h/8 * p(l - h/2) + t * h/4 * p(l)

    MN = h/8 * c_minus - t/h * k_minus + t * h/8 * p(l - h/2)

    PN = h/8 * c_minus * (T_prev_N + T_prev[-2]) + \
         h/4 * cN * T_prev_N + t * alphaN * T0 + t * h/4 * (f(l) + f(l - h/2))

    return KN, MN, PN


def get_T_new(T_prev):
    K0, M0, P0 = left_boundary_condition(T_prev)
    KN, MN, PN = right_boundary_condition(T_prev)

    eps = [0, -M0 / K0]
    eta = [0, P0 / K0]

    x = h
    n = 1
    while (x + h < l):
        T_prev_n = T_prev[n]
        denominator = (B(x, T_prev_n) - A(T_prev_n) * eps[n])

        next_eps = D(T_prev_n) / denominator
        next_eta = (F(x, T_prev_n) + A(T_prev_n) * eta[n]) / denominator

        eps.append(next_eps)
        eta.append(next_eta)

        n += 1
        x += h

    T_new = [0] * (n + 1)
    T_new[n] = (PN - MN*eta[n]) / (KN + MN*eps[n])

    for i in range(n - 1, -1, -1):
        T_new[i] = eps[i+1] * T_new[i+1] + eta[i+1]

    return T_new
```

```python
# Метод простых итераций
def simple_iter():
    step1 = int(l / h)
    T = [T0] * (step1 + 1)
    T_new = [0] * (step1 + 1)
    ti = 0
    res = []
    res.append(T)
    lent = len(T)

    while True:
        T_prev = T
        while True:
            T_new = get_T_new(T_prev)

            cur_max = fabs((T[0] - T_new[0]) / T_new[0])
            for i in range(lent):
                d = fabs(T[i] - T_new[i]) / T_new[i]
                if d > cur_max:
                    cur_max = d

            if cur_max < 1:
                break
            T_prev = T_new

        res.append(T_new)
        ti += t

        flag_eps_ok = True
        for i in range(lent):
            if fabs((T[i] - T_new[i]) / T_new[i]) > 1e-2:
                flag_eps_ok = False
        if flag_eps_ok:
            break
        T = T_new

    return res, ti


if __name__ == "__main__":
    # Исходные данные
    a1 = 0.0134
    b1 = 1
    c1 = 4.35e-4
    m1 = 1
    a2 = 2.049
    b2 = 0.563e-3
    c2 = 0.528e5
    m2 = 1

    alpha0 = 0.05
    alphaN = 0.01
    l = 10
    T0 = 300
    R = 0.5
    F0 = 50

    h = 1e-3
    t = 1

    # Расчеты
    res, ti = simple_iter()
```

```
# Построение графиков
lenres = len(res)
last = int(len(res[0]) / 7) # оставим только седьмую часть графика
res_cutted = [i[0:last:] for i in res] # обрезаем неинтересную часть графика

# Трехмерный график
x, y = np.mgrid[0:lenres:1, 0:last:1]
z = np.array([np.array(i) for i in res_cutted])

fig_3d = plt.figure()
xyz = fig_3d.add_subplot(111, projection='3d')
xyz.plot_surface(x, y, z, cmap='inferno')
fig_3d.show()


# Проекции
fig, (first_graph, second_graph) = plt.subplots(
    nrows=1, ncols=2,
    figsize=(8, 4))

# Первая cm - K
x = list(np.arange(0, l, h))
x_cutted = x[:last:]
step = 3
for i in res_cutted[::step]:
    first_graph.plot(x_cutted, i)
first_graph.plot(x_cutted, res_cutted[-1])
first_graph.set_xlabel("x, cm")
first_graph.set_ylabel("T, K")
first_graph.grid()

# Вторая sec - K
te = list(range(0, ti, t))
for i in np.arange(0, l/3, 0.2):
    line = [j[int(i/h)] for j in res]
    second_graph.plot(te, line[:-1])
second_graph.set_xlabel("t, sec")
second_graph.set_ylabel("T, K")
second_graph.grid()
fig.show()
```

Результаты работы

1. Представить разностный аналог краевого условия при x=l и его краткий вывод интегро-интерполяционным методом.

Разностный аналог краевого условия при x = l получается путем интегрирования (1) на отрезке $[X_{N-\frac{1}{2}},\ X_N]$, с учетом (5).

Введем следующее обозначение:

$$F = -k(u)\frac{\partial T}{\partial x} \tag{6}$$

Проинтегрируем (1):

$$\int_{X_{N-\frac{1}{2}}}^{X_N} dx \int_{t_m}^{t_{m+1}} c(t)\frac{\partial T}{\partial t}dt = -\int_{t_m}^{t_{m+1}} dt \int_{X_{N-\frac{1}{2}}}^{X_N} \frac{\partial F}{\partial x}\,dx - \int_{X_{N-\frac{1}{2}}}^{X_N} dx \int_{t_m}^{t_{m+1}} p(x)T dt + \int_{X_{N-\frac{1}{2}}}^{X_N} dx \int_{t_m}^{t_{m+1}} f(x)dt \tag{7}$$

Интегрируя аналогично разностному аналогу краевого условия при $x = 0$ (из лекции 14) получим, учтя (5):

$$\frac{h}{4}\left(\widehat{c_N}\,(\widehat{y_N} - y_N) - \widehat{c_{N-\frac{1}{2}}}\left(\frac{\widehat{y_N} + \widehat{y_{N-1}}}{2} - \frac{y_N + y_{N+1}}{2}\right)\right) = -\tau\left(\alpha_N\,(\widehat{y_N} - T_0) - \widehat{X_N}\,\frac{\widehat{y_N} + \widehat{y_{N-1}}}{h}\right) - \tau\frac{h}{4}\left(p_N\widehat{y_N} - p_{n-\frac{1}{2}}\,\frac{\widehat{y_N} + \widehat{y_{N-1}}}{2} + \left(\widehat{f_N} - \widehat{f_{N-\frac{1}{2}}}\right)\right) \tag{8}$$

Приведем уравнение к виду $K_N\widehat{y_N} + M_N\widehat{y_{N-1}} = P_N$

$$\left(\frac{h}{4}\,\widehat{c_N} + \frac{h}{8}\widehat{c_{N-\frac{1}{2}}} + \tau\,\alpha_N + \frac{\tau}{h}\,\widehat{X_{N-\frac{1}{2}}} + \frac{h}{4}\tau\,p_N + \frac{h}{8}\tau\,p_{N-\frac{1}{2}}\right)\widehat{y_N} + \left(\frac{h}{8}\widehat{c_{N-\frac{1}{2}}} - \frac{\tau}{h}\widehat{X_{N-\frac{1}{2}}} + \frac{h}{8}\,\tau\,p_{N-\frac{1}{2}}\right)\widehat{y_{N-1}} = \alpha_N\,\tau\,T_0 + \frac{h}{4}\widehat{C_N}y_N + \frac{h}{8}\,\widehat{c_{N-\frac{1}{2}}}\,(y_N + y_{N-1}) + \frac{h}{4}\tau\,(\widehat{f_N} + \widehat{f_{N-\frac{1}{2}}})$$

(9)

Будет принята простая аппроксимация (для функций c, X, p):

$$p_{N-\frac{1}{2}} = \frac{p_{N-1} + p_N}{2}$$

(10)

Из (4) можно получить $K_0, M_0, P_0, K_N, M_N, P_N$:

$$\begin{cases} \widehat{K_0}\,\widehat{y_0} + \widehat{M_0}\,\widehat{y_1} = \widehat{P_0} \\ \widehat{A_n}\,\widehat{y_{n-1}} - \widehat{B_n}\,\widehat{y_n} + \widehat{D_n}\,\widehat{y_{n+1}} = -\widehat{F_n} \\ \widehat{K_N}\,\widehat{y_N} + \widehat{M_{N-1}}\,\widehat{y_{N-1}} = \widehat{P_N} \end{cases}$$

(11)

Систему (11) можно решить методом итераций.
Пусть s – номер итерации.

$$A_n^{s-1}\,y_{n+1}^s - B_n^{s-1}\,y_n^s + D_n^{s-1}\,y_{n-1}^s = -F_n^{s-1}$$

(12)

2. График зависимости температуры $\boldsymbol{T}(\boldsymbol{x}, \boldsymbol{t_m})$ от координаты х при нескольких фиксированных значениях времени $\boldsymbol{t_m}$ при заданных выше параметрах.

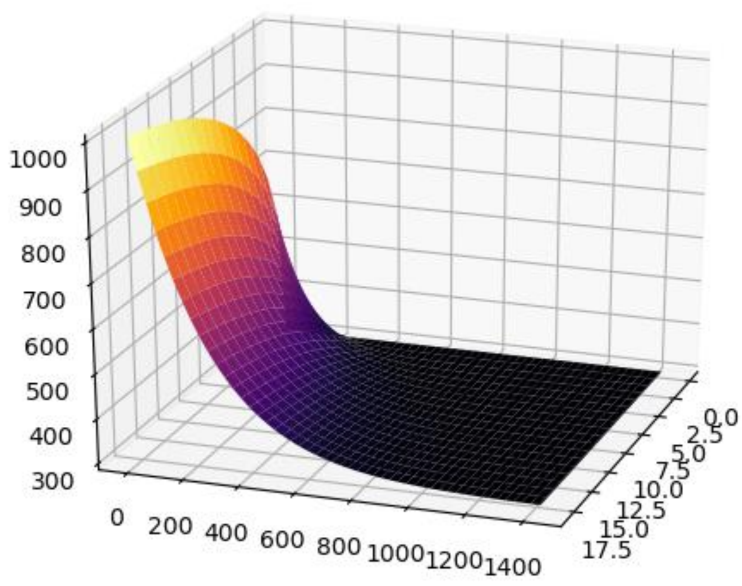


Рис. 1 – трехмерный график изменения температуры по времени и длине

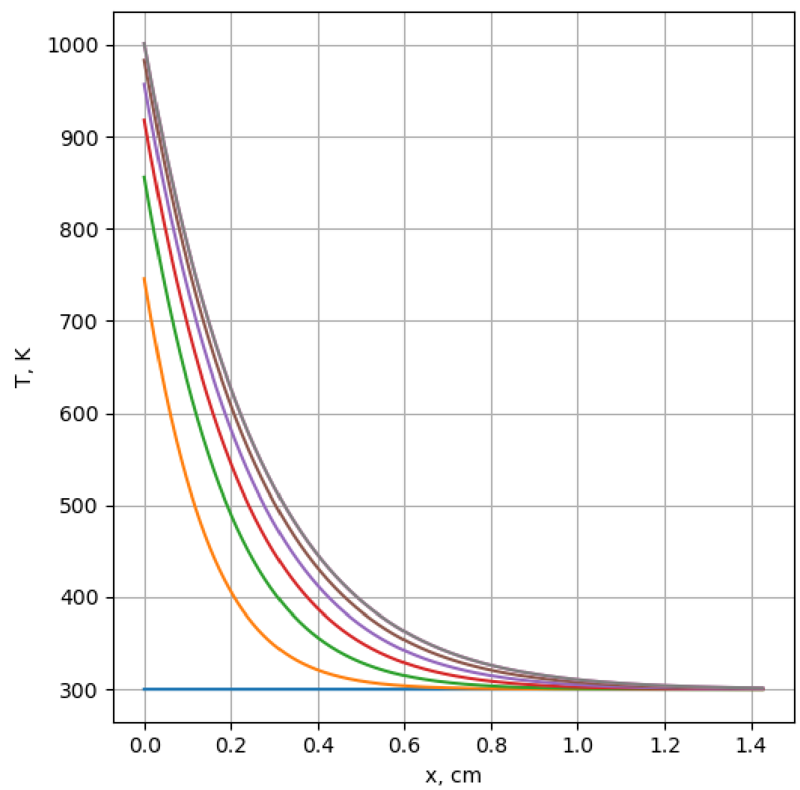


Рис. 2 – проекции трехмерного графика; зависимость температуры и длины в разные моменты времени. Синий график – в начальный момент времени

3. График зависимости $\boldsymbol{T}(\boldsymbol{x_n}, \boldsymbol{t})$ при нескольких фиксированных значениях координаты $\boldsymbol{x_n}$. Обязательно представить случай n=0, т.е. $\boldsymbol{x} = \boldsymbol{x_0} = \boldsymbol{0}$.

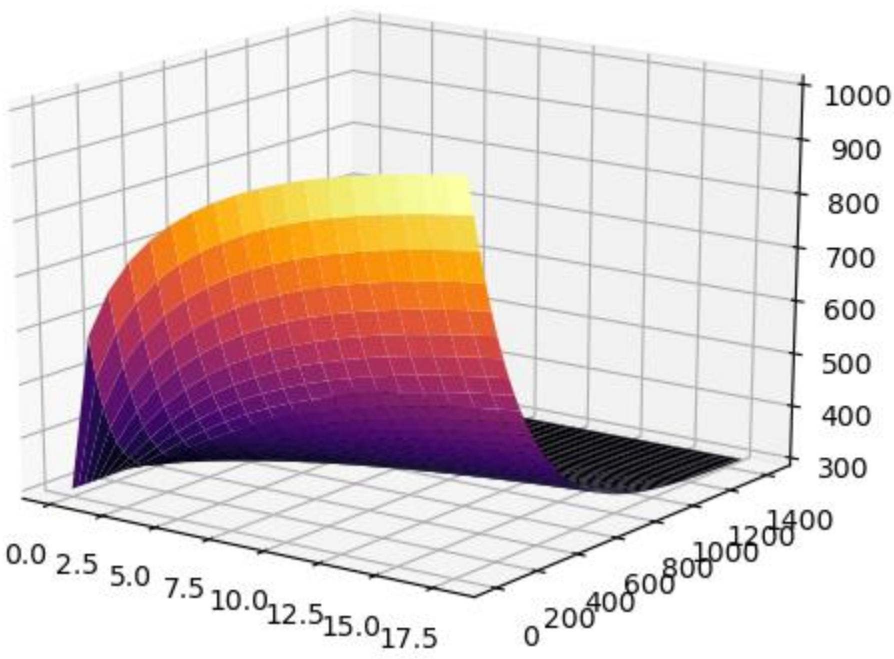


Рис. 3 – трехмерный график изменения температуры по времени и длине

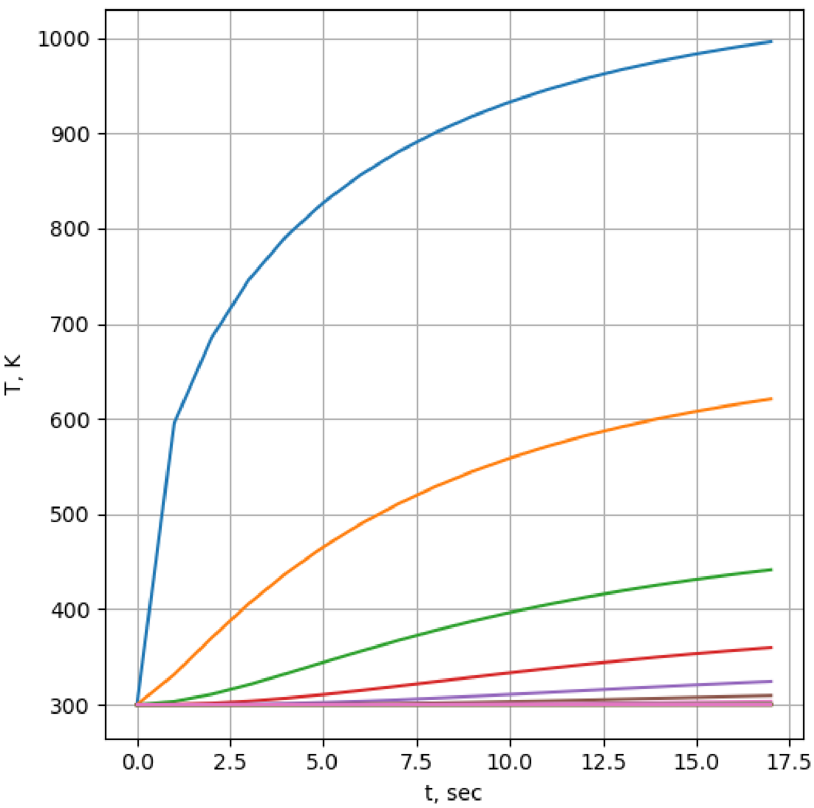


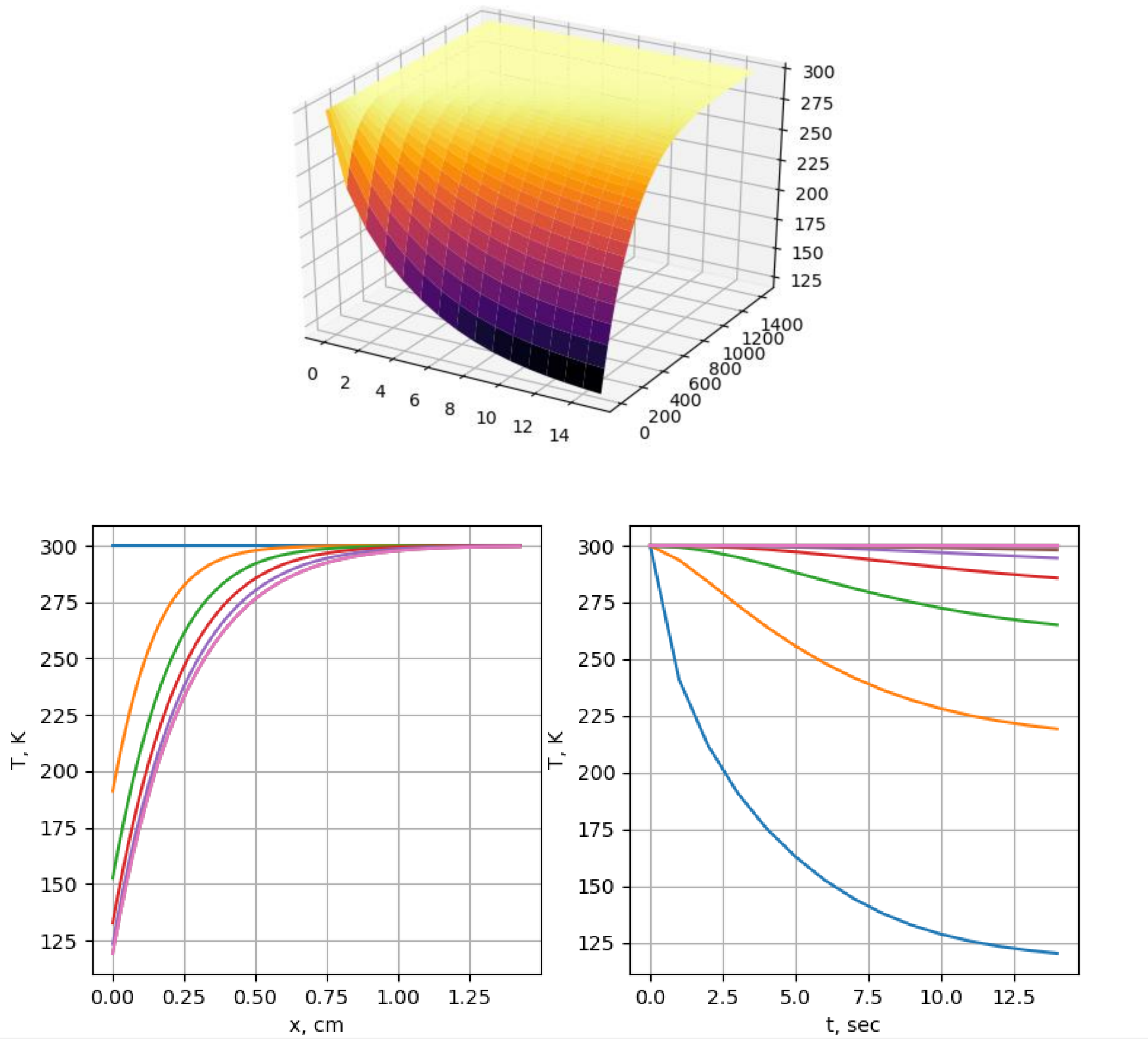
Рис. 4 – проекции трехмерного графика; зависимость температуры по времени в разных частях стержня. Синий график – при x = 0

Вопросы при защите лабораторной работы

1. Приведите результаты тестирования программы. Учесть опыт выполнения лабораторной работы №3.

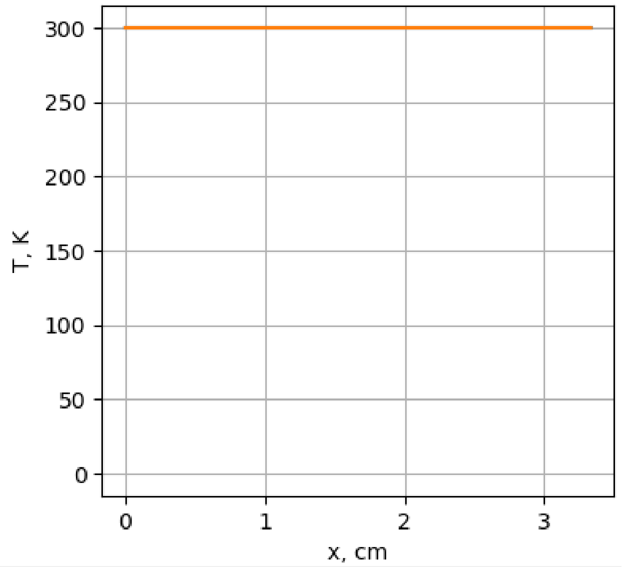
Если принять F0 = -10

При отрицательном тепловом потоке слева идет съем тепла.

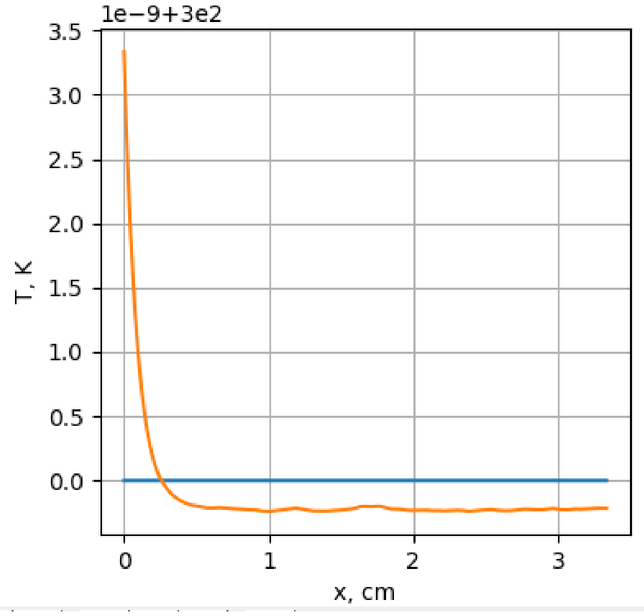


Если принять F0 = 0.

Тепловое нагружение отсутствует, причин для нагрева нет, температура стержня должна быть равна температуре окружающей среды T0.



На следующем графике видна погрешность:



*Синий график – в начальный момент времени

2. Выполните линеаризацию уравнения (14.8) по Ньютону, полагая для простоты, что все коэффициенты зависят только от одной переменной $\widehat{y_n}$. Приведите линеаризованный вариант уравнения и опишите алгоритм его решения. Воспользуйтесь процедурой вывода, описанной в лекции №8.

Канонический вид системы квазилинейных разностных уравнений (14.8):

$$\begin{cases} \widehat{K_0}\widehat{y_0} + \widehat{M_0}\widehat{y_1} = \widehat{P_0} \\ \widehat{A_n}\widehat{y_{n-1}} - \widehat{B_n}\widehat{y_n} + \widehat{D_n}\widehat{y_{n+1}} = -\widehat{F_n}\,, \quad 1 \le n \le N-1 \\ \widehat{K_N}\widehat{y_n} + \widehat{M_{N-1}}\widehat{y_{N-1}} = \widehat{P_N} \end{cases} \tag{13}$$

В условии дано:

$$\widehat{A_n}=\widehat{A_n}(\widehat{y_n}); \quad \widehat{B_n}=\widehat{B_n}(\widehat{y_n}); \quad \widehat{C_n}=\widehat{C_n}(\widehat{y_n}); \quad \widehat{D_n}=\widehat{D_n}(\widehat{y_n}) \tag{14}$$

Выполняя линеаризацию (13) по Ньютону, зная (14), получим:

$$\begin{gathered} \left(\widehat{A_n}\widehat{y_{n-1}} - \widehat{B_n}\widehat{y_n} + \widehat{D_n}\widehat{y_{n+1}} + \widehat{F_n}\right)\Big|_{s-1} + \\ + \frac{1}{\partial\widehat{y_{n-1}}}\left(\partial\widehat{A_n}\widehat{y_{n-1}} - \partial\widehat{B_n}\widehat{y_n} + \partial\widehat{D_n}\widehat{y_{n+1}} + \partial\widehat{F_n}\right)\Big|_{s-1} * \Delta\widehat{y_{n-1}}^{s} + \\ + \frac{1}{\partial\widehat{y_n}}\left(\partial\widehat{A_n}\widehat{y_{n-1}} - \partial\widehat{B_n}\widehat{y_n} + \partial\widehat{D_n}\widehat{y_{n+1}} + \partial\widehat{F_n}\right)\Big|_{s-1} * \Delta\widehat{y_n}^{s} + \\ + \frac{1}{\partial\widehat{y_{n+1}}}\left(\partial\widehat{A_n}\widehat{y_{n-1}} - \partial\widehat{B_n}\widehat{y_n} + \partial\widehat{D_n}\widehat{y_{n+1}} + \partial\widehat{F_n}\right)\Big|_{s-1} * \Delta\widehat{y_{n+1}}^{s} = 0 \end{gathered} \tag{15}$$

Преобразуем к:

$$\begin{gathered} \left(\widehat{A_n}\widehat{y_{n-1}} - \widehat{B_n}\widehat{y_n} + \widehat{D_n}\widehat{y_{n+1}} + \widehat{F_n}\right)\Big|_{s-1} + \\ + \left(\widehat{A_n}\right)\Big|_{s-1}\Delta\widehat{y_{n-1}}^{s} + \\ + \left(\frac{\partial\widehat{A_n}}{\partial\widehat{y_n}}\widehat{y_{n-1}} - \frac{\partial\widehat{B_n}}{\partial\widehat{y_n}}\widehat{y_n} - \widehat{B_n} + \frac{\partial\widehat{D_n}}{\partial\widehat{y_n}}\widehat{y_{n+1}} + \frac{\partial\widehat{F_n}}{\partial\widehat{y_n}}\right)\Bigg|_{s-1} * \Delta\widehat{y_n}^{s} + \\ \left(\widehat{D_n}\right)\Big|_{s-1}\Delta\widehat{y_{n+1}}^{s} = 0 \end{gathered} \tag{16}$$

Уравнение (16) решается методом прогонки. В результате находятся все $\Delta\widehat{y_n}^{(s)}$, после чего определяются значения искомой функции в узлах на s-итерации:

$$\widehat{y_n}^{(s)} = \widehat{y_n}^{(s-1)} + \widehat{y_n}^{(s)} \tag{17}$$

Итерационный процесс завершится при достижении условия:

$$\max\left|\frac{\Delta\widehat{y_n}^{(s)}}{\widehat{y_n}^{(s)}}\right| \leq \varepsilon, \qquad n = 0{,}1 \ldots N \tag{18}$$