| | |
|---|---|
| | **Министерство науки и высшего образования Российской Федерации**<br>**Федеральное государственное бюджетное образовательное учреждение высшего образования**<br>**«Московский государственный технический университет имени Н.Э. Баумана**<br>**(национальный исследовательский университет)»**<br>**(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

**Лабораторная работа № 5**

**Дисциплина: Моделирование**

**Тема: Исследование математической модели на основе технологии вычислительного эксперимента.**

**Студент: Гасанзаде М.А.**

**Группа ИУ7-66Б**

**Оценка (баллы) _______________**

**Преподаватель : Градов В.М.**

Москва.
2020 г.

СОДЕРЖАНИЕ

## I. **АНАЛИТИЧЕСКАЯ ЧАСТЬ.**

**Цель работы**

Получение навыков проведения исследований компьютерной математической модели, построенной на квазилинейном уравнении параболического типа.

Исследование проводится с помощью программы, созданной в лабораторной работе №4.

**Исходные данные**

1. Значения параметров (все размерности согласованы)

$$k(T)=a_1(b_1+c_1T^{m_1}), Вт/см\,К,$$
$$c(T)=a_2+b_2T^{m_2}-\frac{c_2}{T^2}, Дж/см^3\,К.$$

Порядки величин (как в лаб. Работе №4 ):

$$a_1=0.0134, b_1=1, c_1=4.3510^{-4}, m_1=1,$$
$$a_2=2.049, b_2=0.56310^{-3}, c_2=0.52810^5, m_2=1.$$

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

Порядки величин (как в лаб. Работе №4):

$$\alpha_0=0.05\,Вт/см^2\,К,$$
$$\alpha_N=0.01\,Вт/см^2\,К,$$
$$l=10\,см,$$
$$T_0=300\,K,$$
$$R=0.5\,см$$

2. Поток тепла $F$(t) при $x$ = 0

$$F(t)=\frac{F_{max}}{t_{max}}t\,e^{-(\frac{t}{t_{max}}-1)},$$ **(1)** где $F_{max}$ и $t_{max}$ – амплитуда импульса потока и время её достижения (Вт/см$^2$ и с)

*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.

# II. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

ЯП был выбран Python3 из-за простоты работы с графиками и библиотеки matplotlib. Ниже на листингах будет представлена реализация программы:

## Листинг

Листинг 1. Метод прогонки

```
def calc_n_plus_1_ksi(ksi_n, C_n, B_n, A_n):
    return C_n / (B_n - A_n * ksi_n)


def calc_n_plus_1_etta(ksi_n, etta_n, B_n, A_n, F_n):
    return (F_n + A_n * etta_n) / (B_n - A_n * ksi_n)


def calc_y_n(ksi_n_plus_1, etta_plus_1, y_plus_1):
    return ksi_n_plus_1 * y_plus_1 + etta_plus_1


def progonka(A_n, B_n, C_n, F_n, K_0, M_0, P_0, K_N, M_N, P_N):

    ksi = []
    etta = []
    y_result = []

    ksi_1 = -M_0 / K_0
    etta_1 = P_0 / K_0

    ksi.append(ksi_1)
    etta.append(etta_1)

    for i in range(len(A_n)):
        ksi.append(calc_n_plus_1_ksi(ksi[i], C_n[i], B_n[i], A_n[i]))
        etta.append(calc_n_plus_1_etta(ksi[i], etta[i], B_n[i], A_n[i], F_n[i]))

    y_N = (P_N - M_N * etta[i]) / (K_N + M_N * ksi[i])

    y_result.append(y_N)

    for i in range(len(ksi) - 1, -1, -1):
        y_result.append(calc_y_n(ksi[i], etta[i], y_result[-1]))

    y_result.reverse()

    return y_result
```

Листинг 2. Нахождение данных

```
from math import *

class Data:
    def __init__(self):
        self.a_1 = 0.0134
        self.b_1 = 1.0
        self.c_1 = 4.35e-4
        self.m_1 = 1.0
        self.a_2 = 2.049
        self.b_2 = 0.563e-3
        self.c_2 = 0.528e5
```

```
        self.m_2 = 1.0
        self.alpha_0 = 0.05
        self.alpha_N = 0.01
        self.l = 10.0
        self.T_0 = 300.0
        self.R = 0.5
        self.F_0 = 50.0
        self.F_max = 5.0
        self.t_max = 20.0
        self.eps = 1e-4
        self.k_0 = 0.4
        self.k_N = 0.1
        self.a, self.b = self.get_a_b()
        self.c, self.d = self.get_c_d()
        self.t_list = []

    def get_c_d(self):
        d = (self.alpha_N * self.l) / (self.alpha_N - self.alpha_0)
        c = self.alpha_0 * (-d)
        return c, d

    def F_t(self, t):
        return self.F_max / self.t_max * t * exp(-(t / self.t_max - 1))

    def get_a_b(self):
        b = (self.k_N * self.l) / (self.k_N - self.k_0)
        a = self.k_0 * (-b)
        return a, b

    def k_x(self, x):
        return self.a / (x - self.b)

    def X_n_and_half(self, x_n, x_n_and_1):
        return 2.0 * self.k_x(x_n) * self.k_x(x_n_and_1) / (self.k_x(x_n) +
self.k_x(x_n_and_1))


    def alpha_x(self, x):
        return self.c / (x - self.d)

    def f_x(self, x):
        return 2 * self.T_0 * self.alpha_x(x) / self.R

    def p_x(self, x):
        return 2 * self.alpha_x(x) / self.R


    def c_T(self, T):
        return self.a_2 + self.b_2 * pow(T, self.m_2) - self.c_2 / (T * T)

    def t_list_update(self, h_t):
        for i in range(len(self.t_list)):
            self.t_list[i] += h_t
```

Листинг 3. Модуль решений

```
from progon import *
from math import *
import numpy as np


def get_abs_dif(y_n_s_minus_1, y_n_s):
    return fabs((y_n_s - y_n_s_minus_1) / y_n_s)

def get_max_dif_from_result(T_list, T_new_list):
    max_dif = 0
    for i in range(len(T_list)):
        dif = get_abs_dif(T_list[i], T_new_list[i])
        if (max_dif < dif):
            max_dif = dif
    return max_dif



def calc_A_n(T_n, T_n_plus_1, data, h_x, h_t):
    return data.X_n_and_half(T_n, T_n_plus_1) * h_t / h_x


def calc_C_n(T_n, T_n_minus_1, data, h_x, h_t):
    return data.X_n_and_half(T_n, T_n_minus_1) * h_t / h_x


def calc_B_n(T_n, data, A, C, h_x, h_t, cur_x):
    return A + C + data.c_T(T_n) * h_x + data.p_x(cur_x) * h_x * h_t

def calc_F_n(T_n, data, h_x, h_t, cur_x, T_time_ago):
    return data.f_x(cur_x) * h_x * h_t + data.c_T(T_n) * T_time_ago * h_x

def calc_coeff(data, T_list, h_x, h_t, T_time_ago_list):
    A_list, B_list, C_list, F_list = [], [], [], []

    for i in range(1, len(T_list) - 1):
        cur_x = i * h_x

        A = calc_A_n(cur_x, cur_x + h_x, data, h_x, h_t)

        C = calc_C_n(cur_x, cur_x - h_x, data, h_x, h_t)

        #A = calc_A_n(T_list[i], T_list[i + 1], data, h_x, h_t)

        #C = calc_C_n(T_list[i], T_list[i - 1], data, h_x, h_t)

        B = calc_B_n(T_list[i], data, A, C, h_x, h_t, cur_x)
        F = calc_F_n(T_list[i], data, h_x, h_t, cur_x, T_time_ago_list[i])

        A_list.append(A)
        C_list.append(C)
        B_list.append(B)
        F_list.append(F)

    return A_list, B_list, C_list, F_list
```

```python
def get_sum_from_all_pulses(data, t):
    sum_F_t = data.F_t(t)
    for pulse_t in data.t_list:
        sum_F_t += data.F_t(pulse_t)
    return sum_F_t

def calc_left_condition(data, T_list, h_x, h_t, T_old_list, t):
    c_0 = data.c_T(T_list[0])
    c_1 = data.c_T(T_list[1])
    p_0 = data.p_x(0)
    p_1 = data.p_x(h_x)
    p_half = (p_0 + p_1) / 2
    c_half = (c_0 + c_1) / 2
    X_half = data.X_n_and_half(0, h_x)
    #X_half = data.X_n_and_half(T_list[0], T_list[1])
    y_0 = T_old_list[0]
    y_1 = T_old_list[1]
    f_0 = data.f_x(0)
    f_1 = data.f_x(h_x)
    f_half = (f_0 + f_1) / 2

    F_t = get_sum_from_all_pulses(data, t)

    K_0 = h_x * (c_half / 8 +
                 (c_0 / 4) +
                 (h_t * p_half / 8) +
                 (h_t * p_0 / 4)) + \
          X_half * h_t / h_x

    M_0 = h_x * c_half / 8 - \
          X_half * h_t / h_x + \
          h_t * h_x * p_half / 8

    P_0 = h_x * (
        c_half * (y_0 + y_1) / 8 +
        c_0 * y_0 / 4 +
        h_t * (f_half + f_0) / 4
    ) + F_t * h_t

    return K_0, M_0, P_0


def calc_right_condition(data, T_list, h_x, h_t, T_old_list):
    N = len(T_list)
    c_N = data.c_T(T_list[N - 1])
    c_N_minus_1 = data.c_T(T_list[N - 2])
    p_N = data.p_x(data.l)
    p_N_minus_1 = data.p_x(data.l - h_x)
    p_N_minus_half = (p_N + p_N_minus_1) / 2
    c_N_minus_half = (c_N + c_N_minus_1) / 2
    X_N_minus_half = data.X_n_and_half(data.l, data.l - h_x)
    #X_N_minus_half = data.X_n_and_half(T_list[N - 1], T_list[N - 2])
    y_N = T_old_list[N - 1]
    y_N_minus_1 = T_old_list[N - 2]
    f_N = data.f_x(data.l)
 f_N_minus_1 = data.f_x(data.l - h_x)

    K_N = h_t * (X_N_minus_half / h_x + data.alpha_N + h_x / 4 * p_N + h_x / 8 *
p_N_minus_half) +\
          h_x * c_N / 4 + h_x * c_N_minus_half / 8
```

```
    M_N = - h_t * (X_N_minus_half / h_x - h_x * p_N_minus_half / 8) +\
          h_x * c_N_minus_half / 8

    P_N = data.alpha_N * data.T_0 * h_t +\
          h_t * h_x * (3 * f_N + f_N_minus_1) / 8 +\
        h_x * c_N * y_N / 4 +\
        h_x * c_N_minus_half * (y_N + y_N_minus_1) / 8


    return K_N, M_N, P_N


def get_T_list_for_cur_time(data, T_old_list, h_x, h_t, t):

    T_list = T_old_list
    max_dif = 1

    while (max_dif > data.eps):

        A_list, B_list, C_list, F_list = calc_coeff(data, T_list, h_x, h_t,
T_old_list)

        K_0, M_0, P_0 = calc_left_condition(data, T_list, h_x, h_t, T_old_list,
t)

        K_N, M_N, P_N = calc_right_condition(data, T_list, h_x, h_t, T_old_list)

        T_new_list = progonka(A_list, B_list, C_list, F_list, K_0, M_0, P_0,
K_N, M_N, P_N)

        max_dif = get_max_dif_from_result(T_list, T_new_list)

        T_list = T_new_list

    return T_list


def solve_task(data, h_x, h_t, t_max, frequency):

    T_list_list = []

    T_base_list = [data.T_0 for i in np.arange(0, data.l, h_x)]

    max_dif = 1

    T_list_list.append(T_base_list)

    period = 1 / frequency

    #цикл по времени
    t = h_t
    sum_t = t
    while (True):
        if (t >= period):
            data.t_list.append(t)
            T = 0
        T_new_list = get_T_list_for_cur_time(data, T_base_list, h_x, h_t, t)
        T_list_list.append(T_new_list)

        #max_dif = get_max_dif_from_result(T_base_list, T_new_list)
```

```
        T_base_list = T_new_list
        data.t_list_update(h_t)
        t += h_t
        sum_t += h_t

        check = data.F_t(sum_t) / data.F_max
        if (check < 0.05):
            print("F(t_u)/F_max: " + str(check))
            print("t_u: " + str(sum_t) + "\n")

        if (sum_t > t_max):
            break

    return T_list_list
```

## III. ЭКСПЕРИМЕНТАЛЬНАЯ ЧАСТЬ

### Задание №1.

Провести исследование по выбору оптимальных шагов по времени $\tau$ и пространству $h$. Шаги должны быть максимально большими при сохранении устойчивости разностной схемы и заданной точности расчета.

Рассмотреть влияние на получаемые результаты амплитуды импульса $F_{max}$ и времени $t_{max}$ (определяют крутизну фронтов и длительность импульса).

Точность расчета можно оценить разными способами:

1. Уменьшая шаги и наблюдая сходимость решений, как это делалось в лаб. работе №1.

2. Проверяя, соблюдается ли при выбранных $h$, $\tau$ баланс мощности после выхода на стационарное распределение температуры (в установившемся режиме), реализующееся при $F(t) = const$, т.е. в этом режиме должно выполняться условие: подводимая мощность равна отводимой. Имеем:

$$\pi R^2(F_0 - F_N) = 2\pi R \int_0^l \int \alpha [T(x, t_m) - T_0] dx \quad ,$$

окончательно

$$\frac{F_0 - F_N}{\frac{2}{R}\int_0^l \int \alpha [T(x, t_m) - T_0] dx} - 1 \le \varepsilon \qquad \textbf{(2)}$$

Задать точность $\varepsilon$ примерно $10^{-2}$. Здесь $t_m$ - время выхода на стационарный режим, т.е. когда температура перестает меняться с заданной точностью (см. лаб. работу №4).

### Решение

Исследование будет проводиться по выбору оптимальных шагов по времени и пространству, проверяя выполнение неравенства (2) по следующему диапазону шагов: <u>по $t$:$\tau$ = 1, 2, …, 10 сек</u>, по <u>$x$: $h$ = 0.1, 0.2, ..., 1 см</u>. Учитывая: <u>$F(t) = const = F_0$</u>.

Ниже, на рис. 1 будут представлены результаты выполнения/не выполнения неравенства (2) при заданном диапазоне шагов в виде множества точек графика $\tau\ (h)$, где точки $(hi\ ,\ \tau i)$ означают, что при шагах $h = hi$ , $\tau = \tau i$ неравенство выполняется.

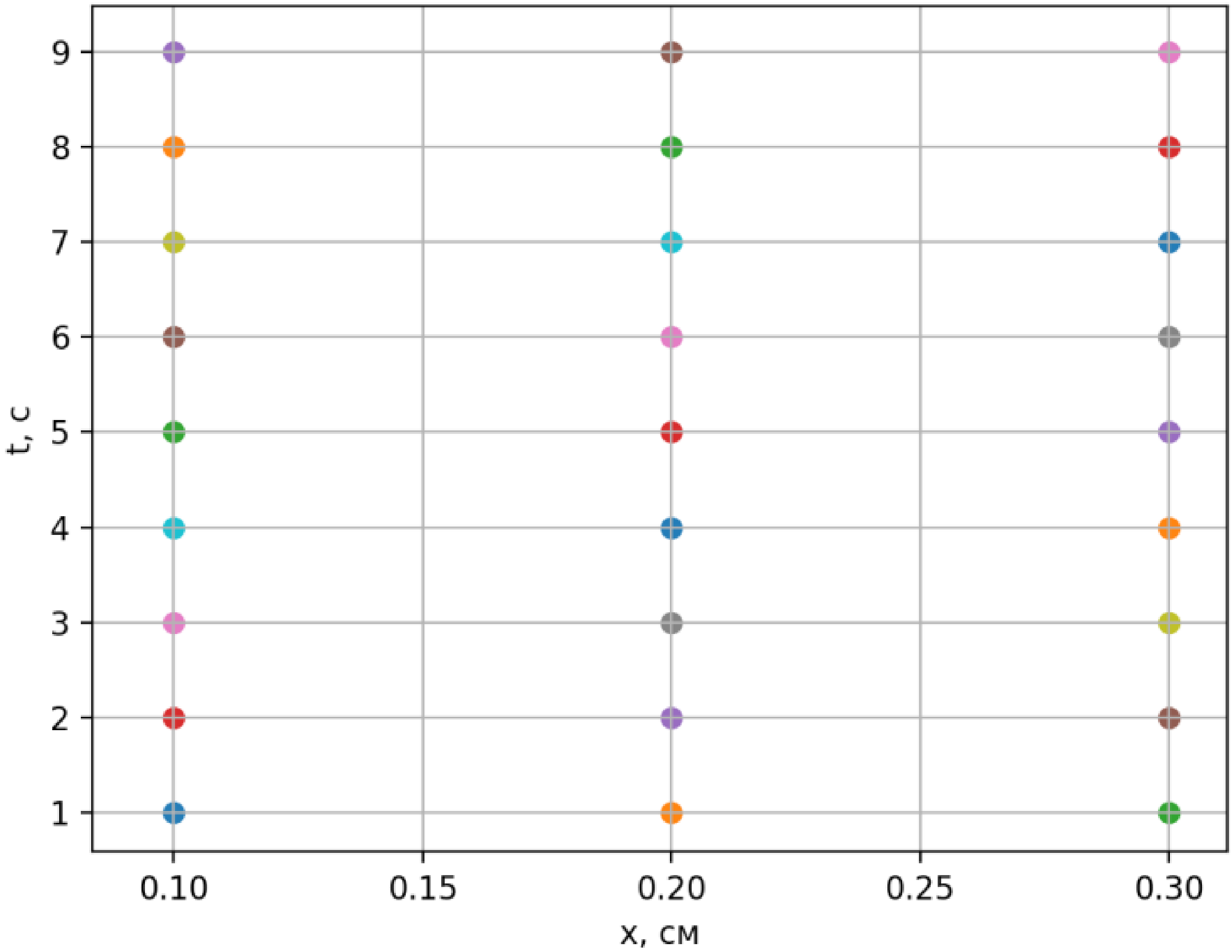


Рис. 1: Выполнение неравенства (2) при некоторых шагах

По графику видно, что при $h > 0.3$ см, баланс мощности после выхода на стационарное распределение температуры не соблюдается. Причём, по данного эксперименту вытекает, что баланс мощности после выхода на стационарное распределение температуры не зависимо от шага по времени. Это получается из того, что параметры неравенства (2) не зависят от $\tau$. С этого момента остановимся на условии $h = 0.3$ см. Далее пробуем уменьшать шаги во времени. По формуле (1), подбирая $F_{max} = 30$ Вт/см$^2$, $t_{max} = 20$ с.

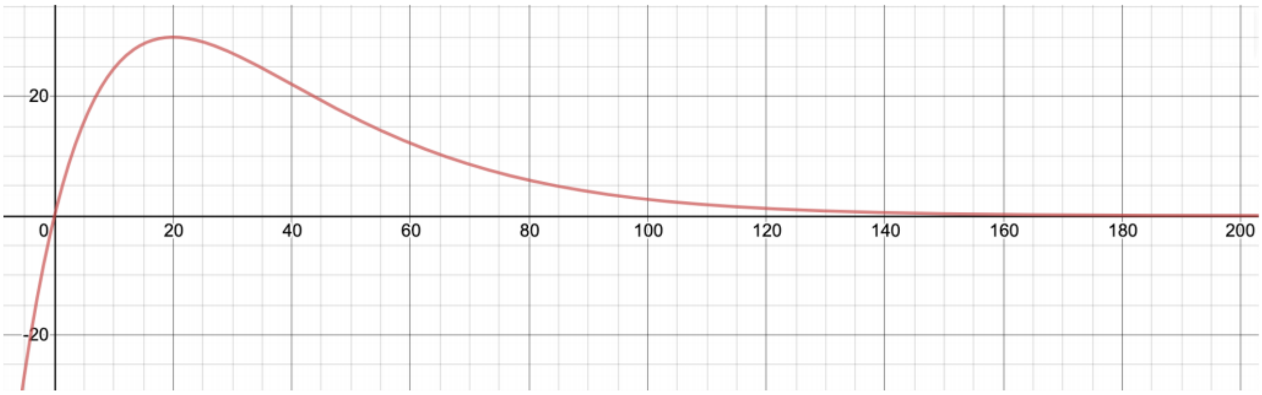


Рис. 2: График функции $F(t)=\frac{30}{20}te^{-\left(\frac{t}{20}-1\right)}$

Поскольку график функции $F(t)=\frac{30}{20}te^{-\left(\frac{t}{20}-1\right)}\approx 0$ при $t>160$ с., то будем проводить измерения при $\underline{t_i = 18;72;144}$ с. (*так как при значении $t > 160$ стержень практически полностью остывает*).

| x,см | T(t_i=18c), tau = 9c | T(t_i=18c), tau = 6c | T(t_i=18c), tau = 3c | T(t_i=18c), tau = 1c | T(t_i=18c), tau = 0.5c | T(t_i=18c), tau = 0.25c |
|---|---|---|---|---|---|---|
| 0.00 | 698.6470 | 704.7200 | 711.0016 | 715.4807 | 716.6590 | 717.2604 |
| 0.30 | 418.3414 | 420.1297 | 421.8280 | 423.0352 | 423.3699 | 423.5453 |
| 0.60 | 336.0523 | 336.0652 | 335.7837 | 335.4113 | 335.2971 | 335.2377 |
| 0.90 | 311.1787 | 310.8767 | 310.3318 | 309.7716 | 309.5989 | 309.5070 |
| 1.20 | 303.5058 | 303.2774 | 302.9144 | 302.5578 | 302.4482 | 302.3897 |
| 1.50 | 301.1088 | 300.9848 | 300.8021 | 300.6329 | 300.5826 | 300.5560 |
| 1.80 | 300.3532 | 300.2950 | 300.2157 | 300.1482 | 300.1294 | 300.1197 |
| 2.10 | 300.1132 | 300.0881 | 300.0568 | 300.0330 | 300.0269 | 300.0239 |
| 2.40 | 300.0365 | 300.0263 | 300.0147 | 300.0070 | 300.0053 | 300.0044 |
| 2.70 | 300.0118 | 300.0078 | 300.0037 | 300.0014 | 300.0010 | 300.0008 |

Рис. 3.1: Таблица $T(x, 18)$. $h = 0.3$ см.

| x,см | T(t_i=72c), tau = 9c | T(t_i=72c), tau = 6c | T(t_i=72c), tau = 3c | T(t_i=72c), tau = 1c | T(t_i=72c), tau = 0.5c | T(t_i=72c), tau = 0.25c |
|---|---|---|---|---|---|---|
| 0.00 | 475.2750 | 475.2583 | 475.0150 | 474.7235 | 474.6338 | 474.5865 |
| 0.30 | 377.2207 | 377.5177 | 377.6289 | 377.5903 | 377.5658 | 377.5512 |
| 0.60 | 334.5280 | 334.9302 | 335.2299 | 335.3617 | 335.3849 | 335.3951 |
| 0.90 | 315.6171 | 315.9582 | 316.2606 | 316.4326 | 316.4709 | 316.4893 |
| 1.20 | 307.1179 | 307.3534 | 307.5833 | 307.7299 | 307.7652 | 307.7826 |
| 1.50 | 303.2566 | 303.4000 | 303.5499 | 303.6530 | 303.6790 | 303.6920 |
| 1.80 | 301.4907 | 301.5704 | 301.6581 | 301.7218 | 301.7385 | 301.7469 |
| 2.10 | 300.6810 | 300.7220 | 300.7688 | 300.8043 | 300.8138 | 300.8187 |
| 2.40 | 300.3099 | 300.3296 | 300.3525 | 300.3703 | 300.3752 | 300.3777 |
| 2.70 | 300.1403 | 300.1491 | 300.1593 | 300.1673 | 300.1696 | 300.1707 |
| 3.00 | 300.0632 | 300.0668 | 300.0708 | 300.0740 | 300.0748 | 300.0753 |
| 3.30 | 300.0283 | 300.0296 | 300.0309 | 300.0319 | 300.0322 | 300.0323 |
| 3.60 | 300.0126 | 300.0130 | 300.0133 | 300.0134 | 300.0134 | 300.0134 |
| 3.90 | 300.0056 | 300.0056 | 300.0056 | 300.0055 | 300.0054 | 300.0054 |
| 4.20 | 300.0024 | 300.0024 | 300.0023 | 300.0022 | 300.0021 | 300.0021 |

Рис. 3.2: Таблица $T(x, 72)$. $h = 0.3$ см.

| x,см | T(t_i=144c), tau = 9c | T(t_i=144c), tau = 6c | T(t_i=144c), tau = 3c | T(t_i=144c), tau = 1c | T(t_i=144c), tau = 0.5c | T(t_i=144c), tau = 0.25c |
|---|---|---|---|---|---|---|
| 0.00 | 310.8666 | 310.6081 | 310.3689 | 310.2208 | 310.1853 | 310.1678 |
| 0.30 | 305.2961 | 305.0905 | 304.8990 | 304.7802 | 304.7516 | 304.7375 |
| 0.60 | 302.7241 | 302.5850 | 302.4533 | 302.3707 | 302.3508 | 302.3410 |
| 0.90 | 301.4626 | 301.3757 | 301.2913 | 301.2375 | 301.2244 | 301.2179 |
| 1.20 | 300.8112 | 300.7600 | 300.7085 | 300.6748 | 300.6665 | 300.6624 |
| 1.50 | 300.4604 | 300.4319 | 300.4018 | 300.3814 | 300.3763 | 300.3738 |
| 1.80 | 300.2653 | 300.2503 | 300.2336 | 300.2218 | 300.2188 | 300.2173 |
| 2.10 | 300.1542 | 300.1469 | 300.1382 | 300.1317 | 300.1300 | 300.1291 |
| 2.40 | 300.0899 | 300.0867 | 300.0826 | 300.0793 | 300.0784 | 300.0779 |
| 2.70 | 300.0524 | 300.0513 | 300.0496 | 300.0480 | 300.0476 | 300.0474 |
| 3.00 | 300.0304 | 300.0302 | 300.0297 | 300.0292 | 300.0290 | 300.0289 |
| 3.30 | 300.0176 | 300.0177 | 300.0177 | 300.0176 | 300.0176 | 300.0176 |
| 3.60 | 300.0101 | 300.0103 | 300.0105 | 300.0106 | 300.0106 | 300.0106 |
| 3.90 | 300.0057 | 300.0059 | 300.0062 | 300.0063 | 300.0063 | 300.0063 |

Рис. 3.3: Таблица $T(x, 144)$. $h = 0.3$ см.

На рисунках 3.1-3.3 приведены таблицы при значениях $t_i = 18;72;144$ с. Колонки соответствуют значениям функции $T(x, t_i)$ при шагах $\tau_k$ по $t$ и $h = 0.3$ см по $x$.

Соответственно, фиксируя $x_j$ и сравнивая на этой строке значения $T(x_j, t_i)$ при разных шагах $\tau$ в каждой таблице, можем заметить, что наиболее оптимальным шагом по времени является $\tau = 1$ с., так как при $\tau > 1$ погрешность $T$ увеличивается, а также при $\tau < 1$ погрешность уже незначительная и наблюдается сходимость решений.

Пример:

Зафиксируем $x_j = 0.3$ см;

$$T(t_i = 18) = 418.3414, 420.1297, 421.8280, 423.0352, 423.3699, 423.5453\,K$$

при значении $\tau = 9, 6, 3, 1, 0.5, 0.25\,c.$ соответственно.

Далее начинаем уменьшать шаги по пространству $h$ при $\tau = 1$ с.

| x,см | T(t_i=18c),h=0.3см | T(t_i=18c),h=0.1см | T(t_i=18c),h=0.05см | T(t_i=18c),h=0.01см | T(t_i=18c),h=0.005см | T(t_i=18c),h=0.001см |
|---|---|---|---|---|---|---|
| 0.00 | 715.4807 | 722.8159 | 716.3603 | 709.4603 | 708.5146 | 707.7466 |
| 0.30 | 423.0352 | 481.9886 | 444.1038 | 414.0602 | 413.8101 | 413.6110 |
| 0.60 | 335.4113 | 349.4057 | 338.4945 | 330.1204 | 330.0570 | 330.0082 |
| 0.90 | 309.7716 | 312.6328 | 309.6169 | 307.7659 | 307.5653 | 307.4095 |
| 1.20 | 302.5578 | 302.9841 | 302.2030 | 301.6601 | 301.6566 | 301.6541 |
| 1.50 | 300.6329 | 300.3788 | 300.3495 | 300.3373 | 300.3365 | 300.3360 |
| 1.80 | 300.1482 | 300.0725 | 300.0650 | 300.0621 | 300.0619 | 300.0618 |
| 2.10 | 300.0330 | 300.0128 | 300.0110 | 300.0104 | 300.0104 | 300.0104 |
| 2.40 | 300.0070 | 300.0038 | 300.0024 | 300.0016 | 300.0016 | 300.0016 |

Рис. 4.1: Таблица $T(x, 18)$. $\tau = 1$ с.

| x,см | T(t_i=72c),h=0.3см | T(t_i=72c),h=0.1см | T(t_i=72c),h=0.05см | T(t_i=72c),h=0.01см | T(t_i=72c),h=0.005см | T(t_i=72c),h=0.001см |
|---|---|---|---|---|---|---|
| 0.00 | 474.7235 | 475.1732 | 473.8360 | 472.4725 | 472.2882 | 472.1389 |
| 0.30 | 377.5903 | 399.6132 | 385.7095 | 373.8286 | 373.7317 | 373.6542 |
| 0.60 | 335.3617 | 343.7813 | 337.6780 | 332.5178 | 332.4680 | 332.4285 |
| 0.90 | 316.4326 | 319.6933 | 316.9351 | 315.0193 | 314.7970 | 314.6218 |
| 1.20 | 307.7299 | 308.9947 | 307.7198 | 306.6657 | 306.6531 | 306.6432 |
| 1.50 | 303.6530 | 303.1922 | 303.1060 | 303.0495 | 303.0433 | 303.0385 |
| 1.80 | 301.7218 | 301.4615 | 301.4168 | 301.3887 | 301.3858 | 301.3835 |
| 2.10 | 300.8043 | 300.6610 | 300.6383 | 300.6246 | 300.6232 | 300.6222 |
| 2.40 | 300.3703 | 300.3856 | 300.3237 | 300.2756 | 300.2750 | 300.2745 |
| 2.70 | 300.1673 | 300.1685 | 300.1402 | 300.1221 | 300.1200 | 300.1185 |
| 3.00 | 300.1673 | 300.0717 | 300.0590 | 300.0511 | 300.0502 | 300.0495 |
| 3.30 | 300.0740 | 300.0296 | 300.0241 | 300.0207 | 300.0203 | 300.0200 |
| 3.60 | 300.0319 | 300.0118 | 300.0095 | 300.0081 | 300.0079 | 300.0078 |
| 3.90 | 300.0134 | 300.0045 | 300.0036 | 300.0030 | 300.0030 | 300.0029 |
| 4.20 | 300.0055 | 300.0017 | 300.0013 | 300.0011 | 300.0011 | 300.0011 |

Рис. 4.2: Таблица $T(x, 72)$. $\tau = 1$ с.

| x,см | T(t_i=144c),h=0.3см | T(t_i=144c),h=0.1см | T(t_i=144c),h=0.05см | T(t_i=144c),h=0.01см | T(t_i=144c),h=0.005см | T(t_i=144c),h=0.001см |
|---|---|---|---|---|---|---|
| 0.00 | 310.2208 | 310.4836 | 310.4870 | 310.4809 | 310.4799 | 310.4791 |
| 0.30 | 304.7802 | 306.1619 | 305.4092 | 304.7569 | 304.7562 | 304.7556 |
| 0.60 | 302.3707 | 302.9319 | 302.5935 | 302.3011 | 302.3006 | 302.3002 |
| 0.90 | 301.2375 | 301.4774 | 301.3153 | 301.2013 | 301.1880 | 301.1775 |
| 1.20 | 300.6748 | 300.7816 | 300.6994 | 300.6288 | 300.6286 | 300.6284 |
| 1.50 | 300.3814 | 300.3554 | 300.3513 | 300.3491 | 300.3489 | 300.3488 |
| 1.80 | 300.2218 | 300.2038 | 300.2010 | 300.1995 | 300.1993 | 300.1992 |
| 2.10 | 300.1317 | 300.1193 | 300.1174 | 300.1163 | 300.1162 | 300.1162 |
| 2.40 | 300.0793 | 300.0842 | 300.0758 | 300.0687 | 300.0687 | 300.0686 |
| 2.70 | 300.0480 | 300.0502 | 300.0451 | 300.0416 | 300.0412 | 300.0408 |
| 3.00 | 300.0480 | 300.0300 | 300.0269 | 300.0247 | 300.0245 | 300.0243 |
| 3.30 | 300.0292 | 300.0178 | 300.0159 | 300.0146 | 300.0145 | 300.0144 |
| 3.60 | 300.0176 | 300.0105 | 300.0094 | 300.0086 | 300.0085 | 300.0084 |

Рис. 4.3: Таблица $T(x, 144)$. $\tau = 1$ с.

Соответственно фиксируя $x_j$ и сравнивая на этой строке значения $T(x_j\ , t_i)$ при разных шагах $h$ в каждой таблице, можем заметить, что наиболее оптимальным шагом по времени будет <u>$h = 0.01$ см</u>, так как при $h > 1$ погрешность $T$ увеличивается, а также при $h < 1$ различия уже минимальные и <u>наблюдается сходимость решений</u>.

Пример:

Зафиксируем $x_j = 0.6$ см;

$$T(t_i = 18) = 335.4113, 349.4057, 338.4945, 330.1204, 330.0570, 330.0082\, K$$

при значении $h = 0.3, 0.1, 0.05, 0.01, 0.005, 0.001\, см.$ соответственно.

Ответ: <u>оптимальные шаги: $\tau = 1.0$ с. $h = 0.01$ см</u>

При этом $F_{max}$ и $t_{max}$ будут влиять на полученные результаты только при одном случае, если задать их близкими к 0. При уменьшении $t_{max}$ будет меняться выбор фиксируемых $t_i$ для $T(x_j\ , t_i)$. Поскольку, если задать $t_{max} \approx 0$, то стержень очень быстро остынет и невозможно будет успеть сравнить большие $t_i$.

Пример:

Показать это очень легко на графике, $t_{max} = 0.7$ с. График функции $F(t)$ будет выглядеть:


Рис. 5: График функции $F(t)=\frac{30}{0.7}t\,e^{-\left(\frac{t}{20}-1\right)}$

Итак, здесь видим, что уже при значении $t = 9$ с. $F(t) \approx 0$. А это значит уже при $t > 20$, при практически полном остывании стрежня, температура по всей длине стержня будет практически равна 300 К. Поэтому нужно сравнивать только при малых $t_i$. Следуя из этого, если задать $t_{max} < \tau$, то сравнивать будет нечего.

То же самое касается и $F_{max}$. Если задать его значение близким к 0, то стержень вообще не будет нагреваться и не будет данных для сравнения.

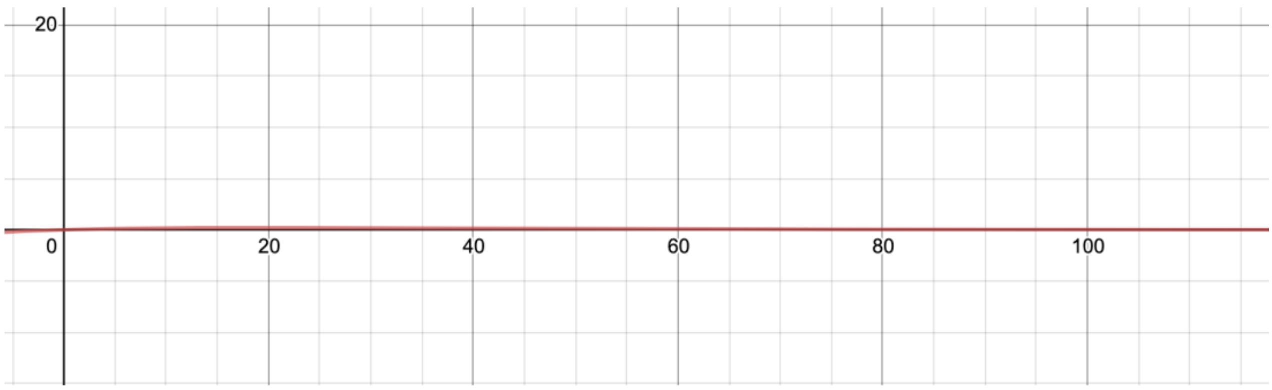


Рис. 6: График функции $F(t)=\frac{0.2}{19}t\,e^{-\left(\frac{t}{20}-1\right)}$, график по сути параллелен Ох.

**Задание №2.**

График зависимости температуры $T(0,\ t)$ при 3-4 значениях параметров $a_2$ и/или $b_2$ теплоемкости.

**Решение**

Ниже приведён графики температуры $T(0,\ t)$ при 4 значениях параметров $a_2$ теплоемкости

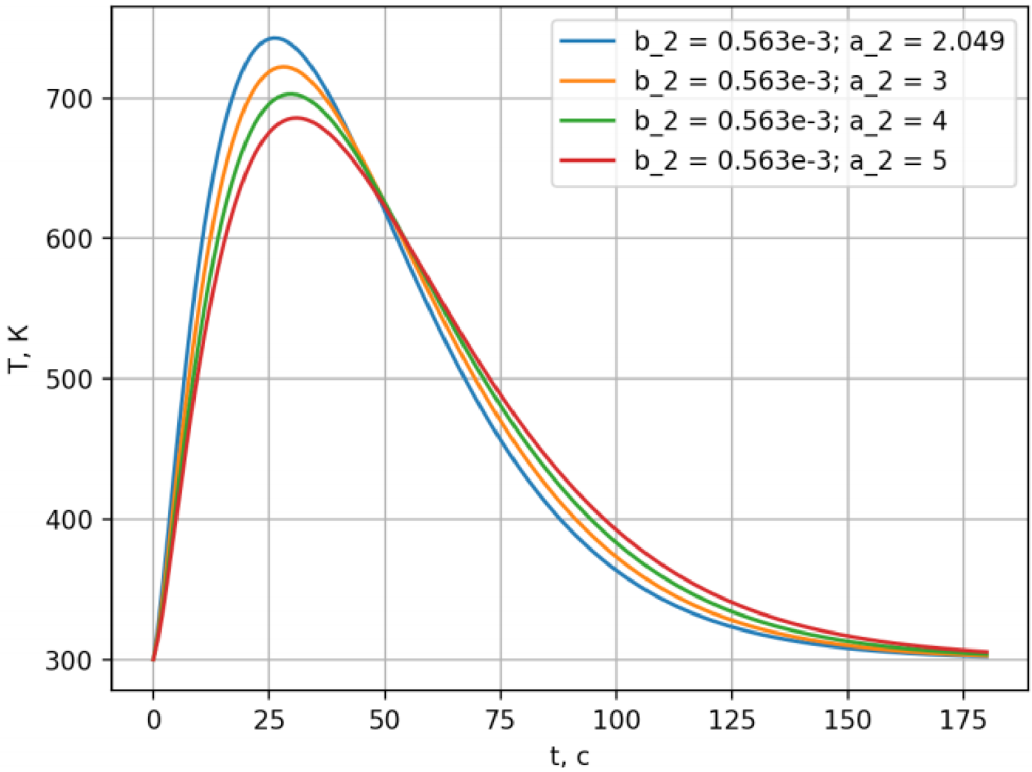


Рис. 7: График функции $T(0,\ t)$ при значениях параметров $a_2$ и/или $b_2$ теплоемкости.

*Справка*. С ростом теплоемкости темп нарастания температуры снижается (видно по графику).

**Задание №3.**

График зависимости температуры $T(0, t)$ (т.е. при $x = 0$ ) в частотном режиме теплового нагружения. Импульсы следуют один за другим с заданной частотой $\nu$ (частота определяется количеством импульсов в 1 секунду).

Показать, что при большом количестве импульсов температурное поле начинает в точности воспроизводиться от импульса к импульсу.

Продемонстрировать, как по мере роста частоты импульсов размах колебаний температуры уменьшается (вплоть до нуля), т.е. реализуется квазистационарный режим, при котором в торец поступает постоянный поток $F_c = \nu \int_0^{t_{uF}} (t) dt$ . Здесь $t_u$ - длительность импульса, определяемая как момент времени, когда $\frac{F(t_u)}{F_{max}} \approx 0.05$ . Если взять прямоугольные импульсы длительностью $t_u$, т.е. $F(t) = const = F_0$, то $F_c = \nu F_0 t_u$.

*Справка*. Полученное температурное поле должно совпасть с результатом расчета $T(x)$ по программе лаб. работы №3 при $F_0 = F_c$, разумеется при всех одинаковых параметрах модели, в частности, вместо $k(T)$ надо использовать $k(x)$ из лаб. работы №3.

**Решение**

Частотный режим реализован следующим образом, заводится массив $\underline{\texttt{pulses}_t}$ для $t_i$, который будет содержать информацию о времени пройденном с момента запуска $i$ (*импульса*). Далее в момент времени $T = \frac{1}{\nu}$ в массиве сохраняется текущее значение времени $t$. При следующем импульсе происходит обнуление $t$. $F(t)$ находится как сумма текущей волны и всех предыдущих, состояние которых лежат в массиве $\underline{\texttt{pulses}_t}$. При окончании каждой итерации, происходит увеличение всех элементов массива $pulses_t$ на $\tau$ (*обновляется состояние каждого импульса*).

Далее все измерения будут сделаны при оптимальных шагах $h$ = 0.01 см.; $\tau$ = 1 с., которые мы получили в задании №1.

Далее приведен график зависимости $T(0, t)$ в частотном режиме теплого нагружения при $\nu$ = 0.01 (т. е. второй импульс запускается, при $t$ = 100 с.); $F_{max}$ = 50 Вт/см$^2$ ; $t_{max}$ = 20 с.:

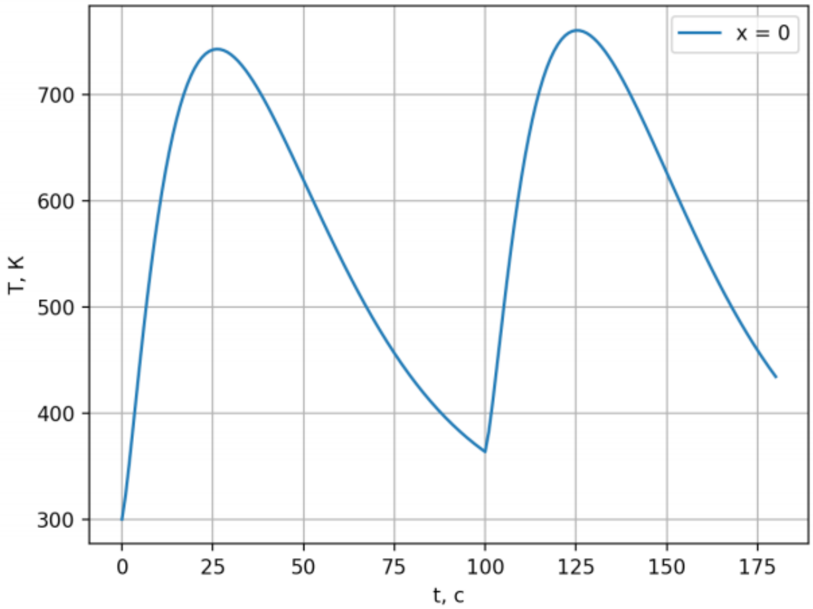


Рис. 8: График функции $T(0, t)$ при выше заданных значениях.

По графику видим, что во время достижения амплитуды второго импульса, температура больше, чем во время достижения первого импульса. Это говорит о том, что к этому моменту времени первый импульс еще не успел до конца затухнуть.

Ниже приведены графики зависимости $T(0, t)$ в частотном режиме теплого нагружения с постепенным увеличением частоты от $\nu$ = 0.05 с. до $\nu$ = 0.5 1/с. с шагом 0.05 1/с.. $F_{max}$ = 5 Вт/см$^2$ так как по условию задачи упомянуто, что решения, в которых температура превышает значения примерно 2000К, физического смысла не имеют и практического интереса не представляют.; $t_{max}$ = 20 с.:

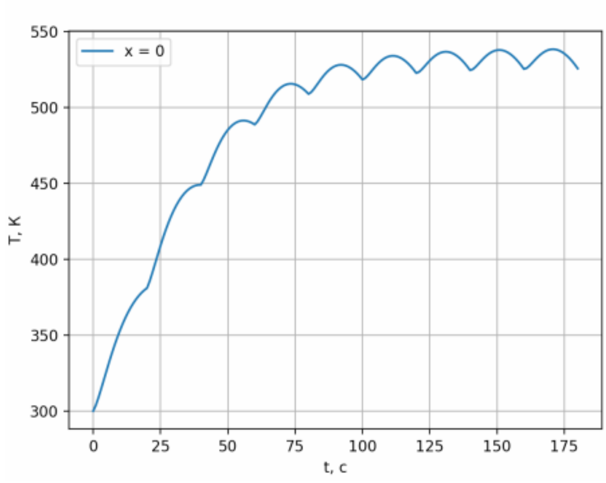


Рис. 9: График функции $T(0, t)$ при $\nu = 0.05$ 1/с. $F_{max} = 5$ Вт/см$^2$ ; $t_{max} = 20$ с.

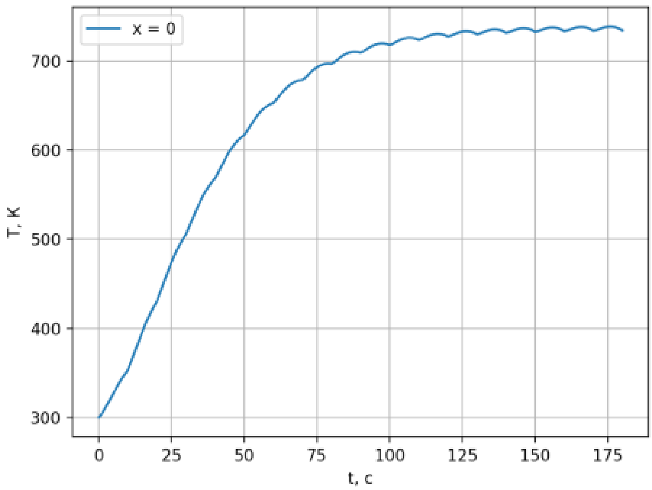


Рис. 10: График функции $T(0, t)$ при $\nu = 0.1$ 1/с. $F_{max} = 5$ Вт/см$^2$ ; $t_{max} = 20$ с.

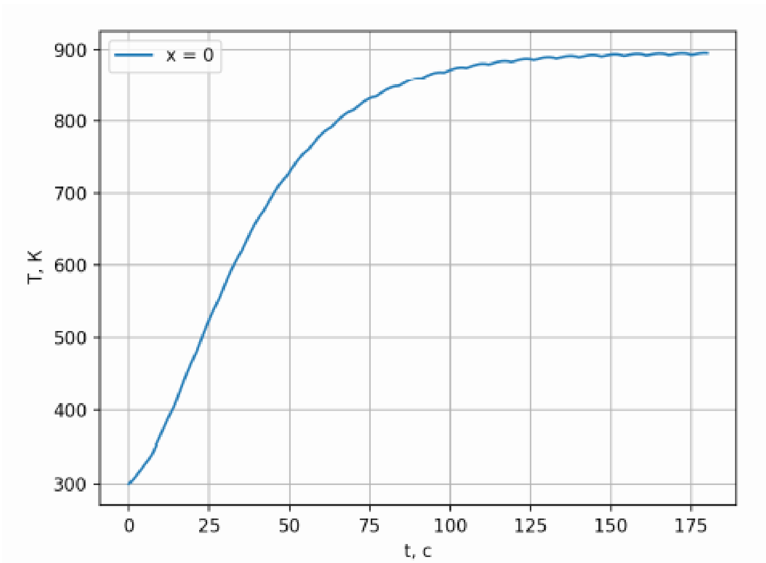


Рис. 11: График функции $T(0, t)$ при $\nu = 0.15$ 1/с. $F_{max} = 5$ Вт/см$^2$ ; $t_{max} = 20$ с.

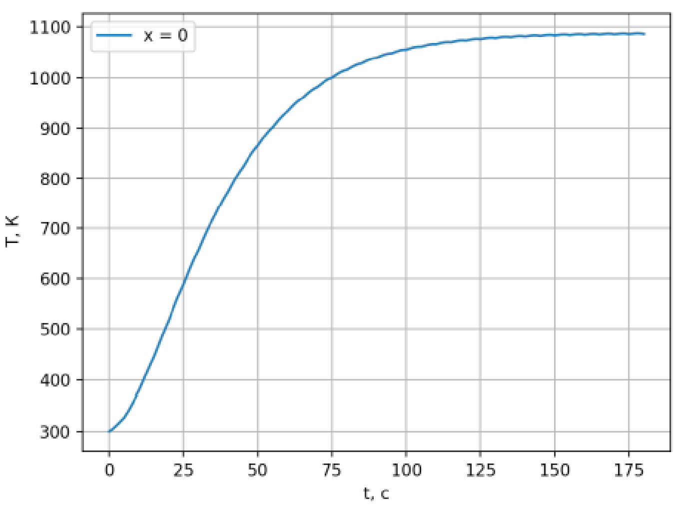


Рис. 12: График функции $T(0, t)$ при $\nu = 0.2$ 1/с. $F_{max} = 5$ Вт/см$^2$ ; $t_{max} = 20$ с.

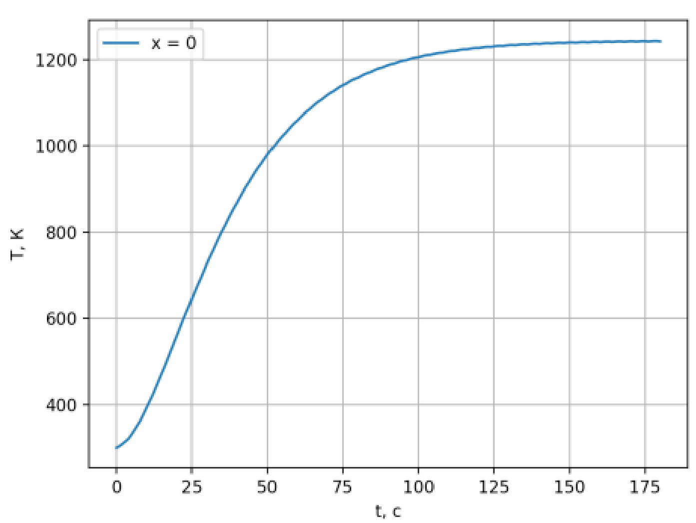


Рис. 13: График функции $T(0, t)$ при $\nu = 0.25$ 1/с. $F_{max} = 5$ Вт/см$^2$ ; $_{tmax} = 20$ с.

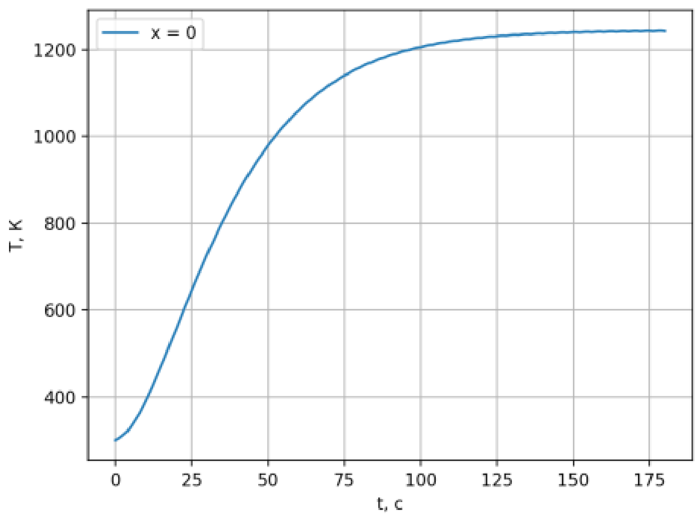


Рис. 14: График функции $T(0, t)$ при $\nu = 0.3$ 1/с. $F_{max} = 5$ Вт/см$^2$ ; $t_{max} = 20$ с.

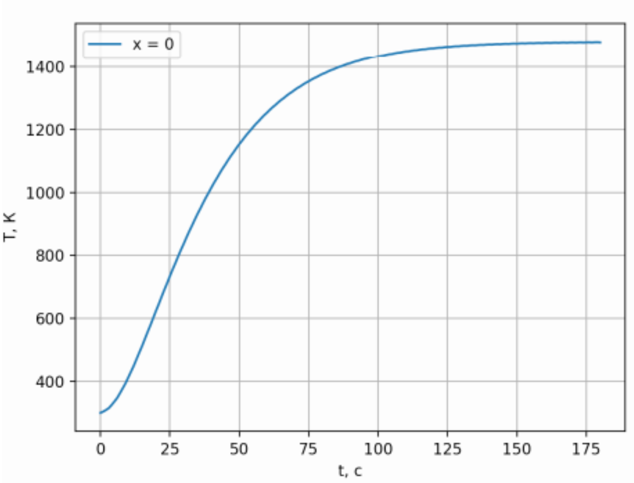


Рис. 15: График функции $T(0, t)$ при $\nu = 0.35$ 1/с. $F_{max} = 5$ Вт/см$^2$ ; $t_{max} = 20$ с.

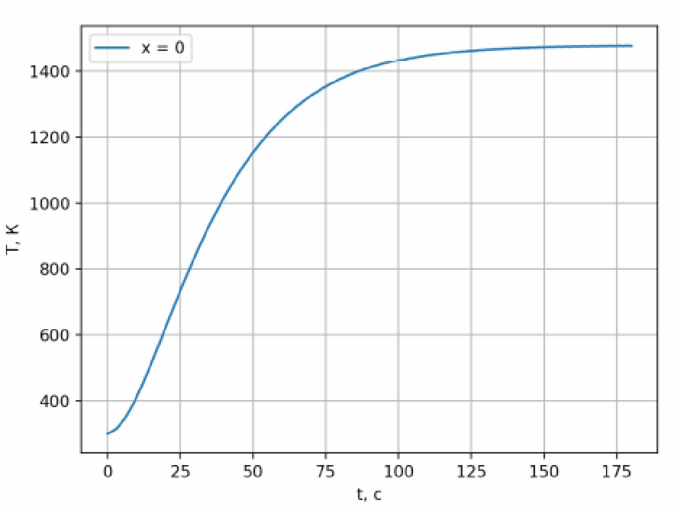


Рис. 16: График функции $T(0, t)$ при $\nu = 0.4$ 1/с. $F_{max} = 5$ Вт/см$^2$ ; $t_{max} = 20$ с.

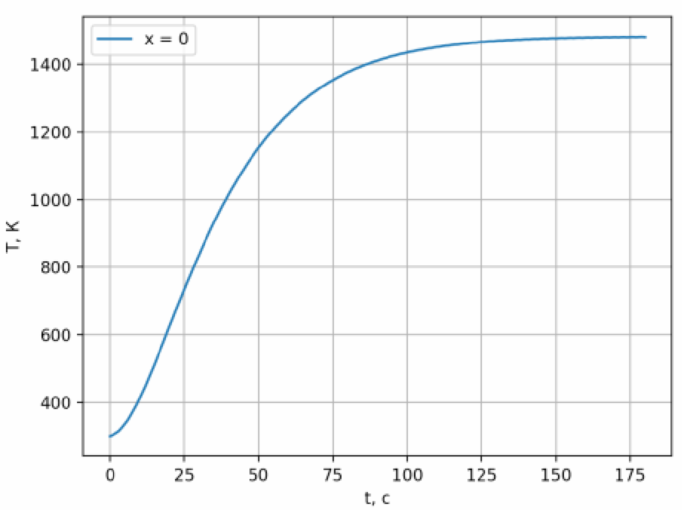


Рис. 17: График функции $T(0, t)$ при $\nu = 0.45$ 1/с. $F_{max} = 5$ Вт/см$^2$ ; $t_{max} = 20$ с.

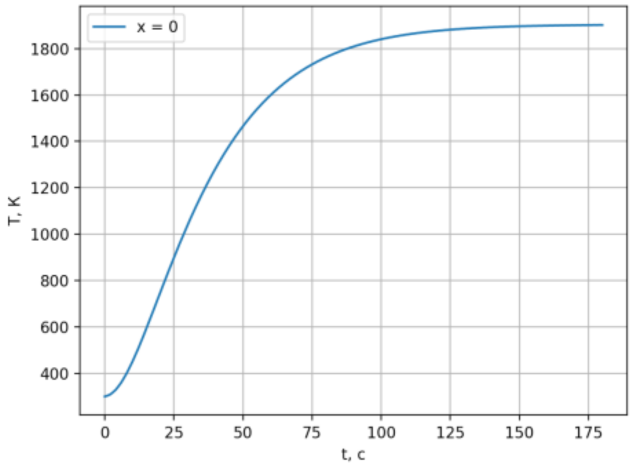


Рис. 18: График функции $T(0, t)$ при $\nu = 0.5$ 1/с. $F_{max} = 5$ Вт/см$^2$ ; $t_{max} = 20$ с.

Видно, что при $\nu = 0.5$ 1/с температурное поле в точности воспроизводиться от импульса к импульсу.

Возьмем $k(x)$ из лаб. работы №3 вместо $k(T)$. Остальные параметры оставим такими же. При $t_u = 115.0$ с. $\frac{F(t_u)}{F_{max}} \approx 0.05$

Следовательно, $F_c = 0.5 \int\limits_0^{115.0} \frac{5}{20} t\, e^{-(\frac{t}{20}-1)} dt = 132.994\, K$.

Ниже приведен график распределения $T(0, t)$ при описанных выше параметрах:

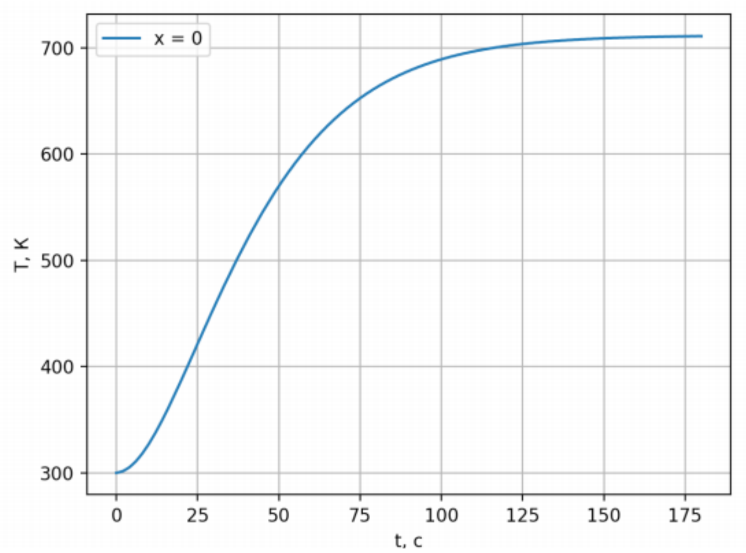


Рис. 19: График функции $T(0, t)$ при $\nu = 0.5$. $F_{max} = 5$ Вт/см$^2$ ; $t_{max} = 20$; $k(x)$ с.

## ЗАКЛЮЧЕНИЕ

Были получены навыки в исследовании математической модели на основе технологии вычислительного эксперимента. Были решены 3 типов задач, при разных температурных коэффициентах, показано при помощи графиков, что при большом количестве импульсов температурное поле начинает в точности воспроизводиться от импульса к импульсу. Также был реализован квазистационарный режим при котором в торец поступал постоянный поток $F_c = \nu \int_0^{t_{uF}} (t)\,dt$ . Также в дополнении к Заданию №3, по рис. 19, видно, что при установлении квазистационарного режима $T \approx 710$ К, это совпадает с результатом расчета $T(x)$ по программе лаб. работы №3 при $F_0 = F_c$ .

## СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ


1. Градов В.М. Методические указания: «04-05-2020-Задание_на_лаб_раб__4.doc » (дата обращения 16.05.2020)
2. Градов В.М. Методические указания: «13-05-2020-Задание_на_лаб_раб__5.doc» (дата обращения 23.05.2020)
3. Градов В.М. Компьютерные технологии в практике математического моделирования часть 2 URL: http://ebooks.bmstu.ru/secret/html/bikqxzugca/files/assets/basic-html/page-1.html (дата обращения 21.05.2020)
4. Градов В.М. Лекция №14 «04-05-2020-Лекция__14_Модели_ДУЧП_Методы_постр_разност_схем_Интегро_интерп.pdf » (дата обращения 20.05.2020)
5. Градов В.М. Лекция №13 «04-05-2020-Лекция__13_Модели_ДУЧП_Методы_постр_разност_схем_Разност_аппроксим.pdf » (дата обращения 19.05.2020)
6. Градов В.М. Лекция №15 «18-05-2020-Лекция__15_Модели_ДУЧП_Многомерные_уравнения.pdf » (дата обращения 23.05.2020)