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 № 3

Тема: Программно-алгоритмическая реализация моделей на основе ОДУ второго порядка с краевыми условиями II и III род

Студент: Лаврова А. А.

Группа: ИУ7-65Б

Оценка (баллы) _______________

Преподаватель: Градов В.М.

Москва.
2020 г.

Цель работы: Получение навыков разработки алгоритмов решения краевой задачи при реализации моделей, построенных на ОДУ второго порядка.

Исходные данные:

1. Задана математическая модель.

   Уравнение для функции $T(x)$

$$\frac{d}{dx}\left(k(x)\frac{dT}{dx}\right)-\frac{2}{R}\alpha(x)T+\frac{2T_0}{R}\alpha(x)=0$$

Краевые условия

$$\begin{cases} x=0, \quad -k(0)\dfrac{dT}{dx}=F_0, \\ x=l, \quad -k(l)\dfrac{dT}{dx}=\alpha_N\,(T(l)-T_0) \end{cases}$$

2. Функции $k(x),\alpha(x)$ заданы своими константами

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

Константы a,b следует найти из условий $k(0)=k_0,\ k(l)=k_N$, а константы c,d из условий $\alpha(0)=\alpha_0,\ \alpha(l)=\alpha_N$. Величины $k_0,\ k_N,\alpha_0,\alpha_N$ задает пользователь, их надо вынести в интерфейс.

3. Разностная схема с разностным краевым условием при $x=0$. Получено в лекции №7, и может быть использовано в данной работе. Самостоятельно надо получить интегро-интерполяционным методом разностный аналог краевого условия при $x=l$, точно так же, как это было сделано применительно к краевому условию при $x=0$ в лекции №7. Для этого надо проинтегрировать на отрезке [$x_{N-1/2}$, x_N] уравнение $\frac{d}{dx}\left(k(x)\frac{dT}{dx}\right)-\frac{2}{R}\alpha(x)T+\frac{2T_0}{R}\alpha(x)=0$ и учесть, что поток $F_N=\alpha_N\,(y_N-T_0)$, а

$$F_{N-1/2}=\chi_{N-1/2}\frac{y_{N-1}-y_N}{h}.$$

4. Значения параметров для отладки

$k_0 = 0.4$ Вт/см К,

$k_N = 0.1$ Вт/см К,

$\alpha_0 = 0.05$ Вт/см^2 К,

$\alpha_N = 0.01$ Вт/см^2 К,

$l =$ 10 см,

$T_0 = 300$К,

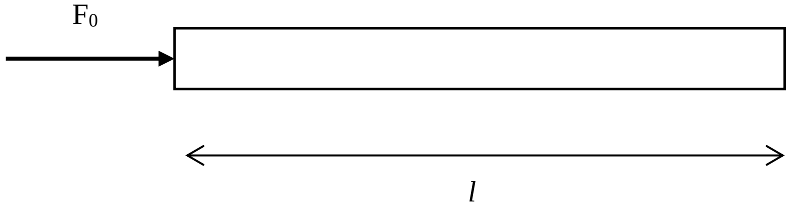
$R = 0.5$ см,

$F_0 = 50$ Вт/см^2.

Физическое содержание задачи:

Сформулированная математическая модель описывает температурное поле $T(x)$ вдоль цилиндрического стержня радиуса R и длиной l, причем $R << l$ и температуру можно принять постоянной по радиусу цилиндра. Ось x направлена вдоль оси цилиндра и начало координат совпадает с левым торцем стержня. Слева при $x = 0$ цилиндр нагружается тепловым потоком F_0. Стержень обдувается воздухом, температура которого равна T_0. В результате происходит съем тепла с цилиндрической поверхности и поверхности правого торца при $x = l$. Функции $k(x), \alpha(x)$ являются, соответственно, коэффициентами теплопроводности материала стержня и теплоотдачи при обдуве.

Стержень, нагружаемый тепловым потоком F_0 :



Получение разностной схемы для уравнения второго порядка:

На отрезке [$X_{N-1/2}$; X_N] проинтегрируем уравнение $\frac{d}{dx}\left(k(x)\frac{du}{dx}\right) - p(x) * u + f(x) = 0$, учитывая, что $F = -k(x)\frac{du}{dx}$, а поток $F_N = \alpha(y_n - \beta)$, $F_{N-\frac{1}{2}} = X_{N-\frac{1}{2}} * \frac{Y_{N-1} - Y_N}{h}$

$$\int_{X_{N-\frac{1}{2}}}^{X_N} \frac{dF}{dx} dx - \int_{X_{N-\frac{1}{2}}}^{X_N} p(x) * y * dx + \int_{X_{N-\frac{1}{2}}}^{X_N} f(x) dx = 0$$

где $F = -k(x) * \frac{dT}{dx}$; $\quad f(x) = \frac{2*T_0}{R}; p(x) = \frac{2}{R} * \alpha(x);\quad f_n = f(x_n); p_n = p(x_n)$

Также используем простую аппроксимацию:

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

$$f_{N-\frac{1}{2}} = \frac{f_N + f_{N-1}}{2}$$

Вычислим второй и третий интегралы методом трапеций:

$$F_{N-\frac{1}{2}} - F_N - \frac{h}{4}\left(p_N y_{N+} p_{N-\frac{1}{2}} y_{N-\frac{1}{2}}\right) + \frac{h}{4}\left(f_N + f_{N-\frac{1}{2}}\right) = 0$$

Разностная схема:

$$\begin{cases} A_n * y_{n-1} - B_n * y_n + C_n * y_{n+1} = -D_n, n \in [1; N-1] \\ K_0 * y_0 + M_0 * y_1 = P_0 \\ K_N * y_N + M_N * y_{N-1} = P_N \end{cases}$$

Для получения коэффициентов K_0, M_0, P_0, K_N, M_N, P_N необходимо взять разностную схему с краевым условием при x = 0:

$$y_0\left(x_{\frac{1}{2}} + p_{\frac{1}{2}} * \frac{h^2}{8} + p_0 * \frac{h^2}{4}\right) - y_1\left(x_{\frac{1}{2}} - p_{\frac{1}{2}} * \frac{h^2}{8}\right) = \left(\frac{h^2}{4} * \left(f_{\frac{1}{2}} + f_0\right) + h * f_0\right)$$

Подставим значения $F_{N-\frac{1}{2}}$ и F_N :

$$\frac{X_{N-\frac{1}{2}} * y_{N-1}}{h} - \frac{X_{N-\frac{1}{2}} * y_{N-1}}{h} - h * \frac{p_{N-\frac{1}{2}} * y_{N-1}}{8} - h * \frac{p_{N-\frac{1}{2}} * y_{N-1}}{8} - h * \frac{p_N * y_N}{4} + h * \frac{f_{N-\frac{1}{2}} * f_N}{4} + \alpha_N * T_N - \alpha_N * y_0 = 0$$

Для того, чтобы решить систему уравнений разностной схемы используется метод прогонки.

$$\varepsilon_1 = -\frac{M_0}{K_0}$$

$$\eta_1 = \frac{P_0}{K_0}$$

Прямой ход:

$$y_n = \frac{C_n * y_{n+1}}{B_n - A_n * \varepsilon_n} + \frac{D_n * A_n * \eta_n}{B_n - A_n * \varepsilon_n}$$

где первая дробь – ε_{n+1}, а вторая дробь – η_{n+1}

Обратный ход:

Значение функции в последней точке:

$$y_n = \frac{P_N * M_N * \eta_n}{K_N - M_N * \varepsilon_n}$$

Найдем значения неизвестных y_n:

$$y_n = \varepsilon_{n+1} * y_{n+1} + \eta_{n+1}$$

Тестирование:

График зависимости температуры $T(x)$ от координаты x при исходных параметрах:

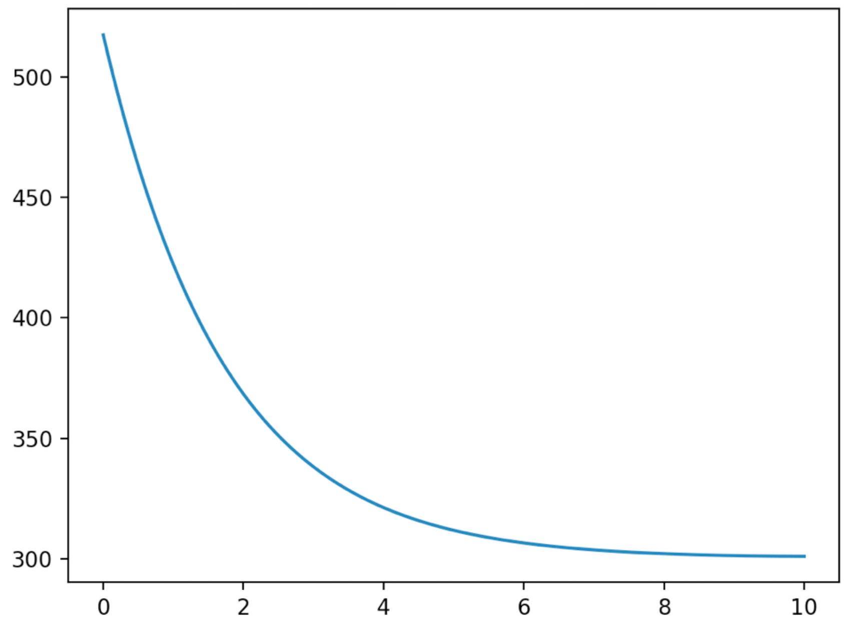


График зависимости температуры $T(x)$ от координаты x при $F_0 = 0$:

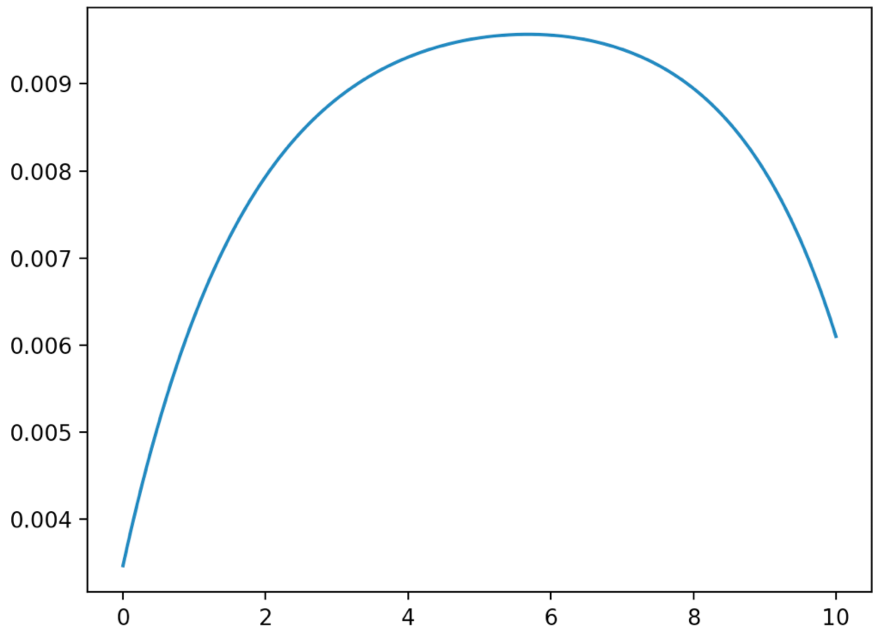


График зависимости температуры $T(x)$ от координаты x при $F_0 = -10$:

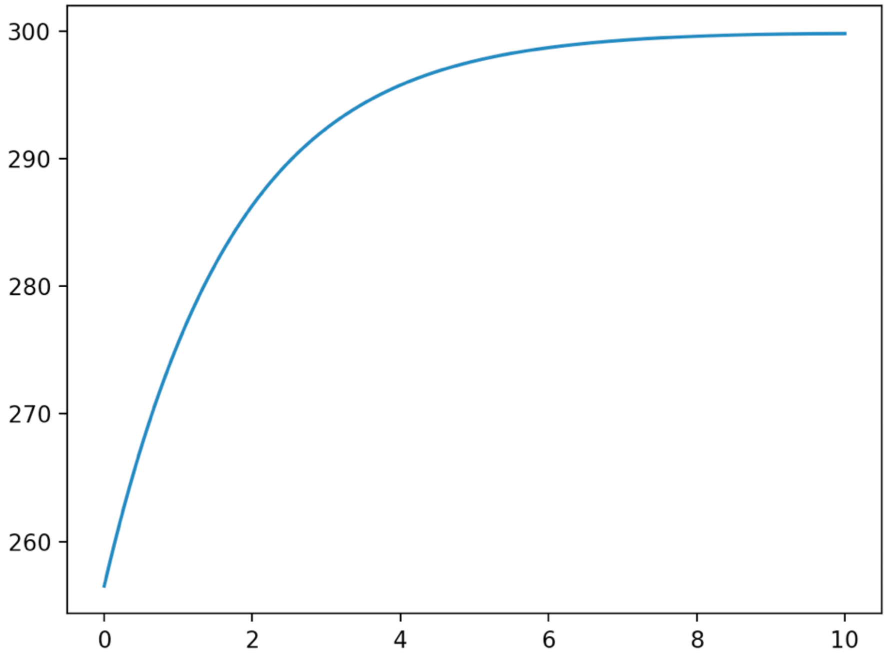
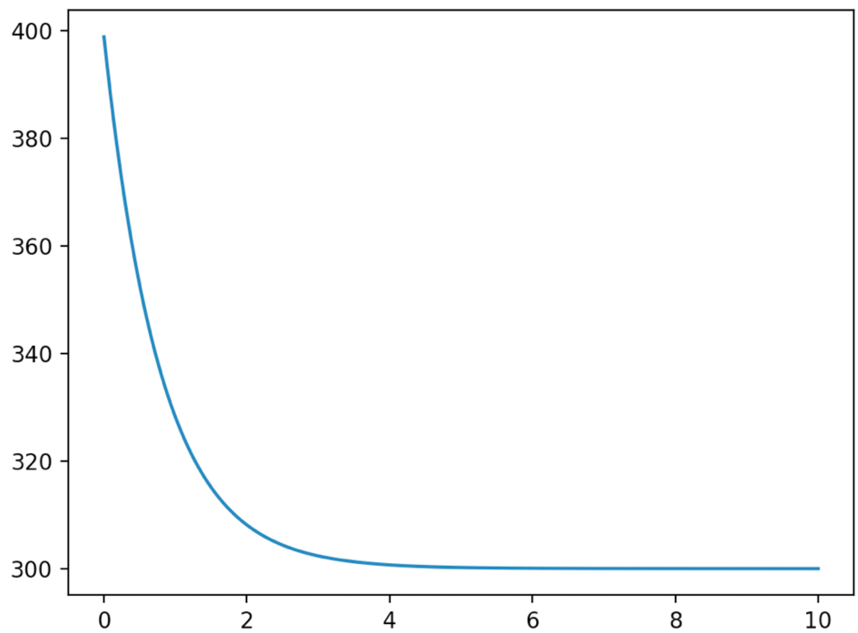


График зависимости температуры $T(x)$ от координаты x при увеличении значений $\alpha(x)$ в 4 раза (при неизменном F_0 и увеличенном коэффициенте теплоотдачи градиент увеличивается, температура снижается):



Листинг программы:

```
import matplotlib.pyplot as plt

# функции k(x), a(x)
def alpha(x):
   return c / (x - d)

def k(x):
   return a / (x - b)

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

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

def X_left(x, h):
   return 2 * k(x) * k(x - h) / (k(x) + k(x - h))

def X_right(x, h):
   return 2 * k(x) * k(x + h) / (k(x) + k(x + h))


# прогонка
def matrix_algorithm(K0, M0, P0, KN, MN, PN, N, h):
   N+=1
   y = (N+1)*[0]
   ksi = (N+1)*[0]
   eta = (N+1)*[0]
   ksi[1] = - M0 / K0
   eta[1] = P0 / K0

   A = (N+1)*[0]
   B = (N+1)*[0]
   C = (N+1)*[0]
   F = (N+1)*[0]

   for i in range (1, N+1):
      A[i] = X_left((i-1)*h, h) / h
      C[i] = X_right((i-1)*h, h) / h
      B[i] = A[i] + C[i] + p((i-1)*h) * h
      F[i] = f((i-1)*h) * h

   for i in range(1, N):
      denom = B[i] - A[i] * ksi[i]
      ksi[i+1] = C[i] / denom
      eta[i+1] = (F[i] + A[i] * eta[i]) / denom


   y[N] = (PN - MN * eta[N]) / (KN + MN * ksi[N])
   for i in range (N - 1, 0, -1):
      y[i] = ksi[i+1] * y[i+1] + eta[i+1]

   return y

# построение графика
def graph(a, b):
   plt.plot(a, b)
   plt.show()
```

```python
# значения параметров для отладки
k0 = 0.4
kN = 0.1
a0 = 0.20
aN = 0.04
l = 10
R = 0.5
T0 = 300
F0 = 50

print ("Do you want to enter new values? 1 - yes, 0 - no")
flag = int(input())

if (flag == 1):
    k0 = float(input("Input k0 = "))
    kN = float(input("Input kN = "))
    a0 = float(input("Input a0 = "))
    aN = float(input("Input aN = "))
    l = float(input("Input l = "))
    R = float(input("Input R = "))
    T0 = float(input("Input T0 = "))
    F0 = float(input("Input F0 = "))


b = l * kN / (kN - k0)
a = -b * k0
d = l * aN / (aN - a0)
c = -d * a0

h = 1e-4
h2 = h * h

p01_2 = (p(0) + p(h))/2
pN1_2 = (p(l) + p(l - h))/2
f01_2 = (f(0) + f(h))/2
fN1_2 = (f(l) + f(l - h))/2

K0 = X_right(0, h) + h2 / 4 * p(0) + h2 / 8 * p01_2
M0 = -(X_right(0, h) - h2 / 8 * p01_2)
P0 = h * F0 + h2 / 4 * f01_2
print(K0, M0, P0)

KN = X_left(l, h) + h2 / 4 * p(l) + h2 / 8 * pN1_2 + h * aN
MN = - (X_left(l, h) - h2 / 8 * pN1_2)
PN = h * aN * T0 + h2 / 4 * fN1_2
print(KN, MN, PN)

N = int(l / h)
T = matrix_algorithm(K0, M0, P0, KN, MN, PN, N, h)
x = (N+2)*[0]
for i in range (N+2):
    x[i] = i * h

graph(x[1:], T[1:])
```

Контрольные вопросы:

1. Какие способы тестирования программы можно предложить?
   - Попробовать сделать тепловой поток T_0 отрицательным, в следствие чего производная T(x) должна быть положительной
   - С помощью увеличения длины стержня добиться стремления температуры к значению T_0
   - При значении F_0=0 стержень будет иметь значение температуры окружающей среды, то есть график T(x) будет примерно равен T_0
   - Увеличить коэффициент $\alpha(x)$, с помощью чего добиться того, что температура будет быстрее снижаться

2. Получите простейший разностный аналог нелинейного краевого условия при $x = l, -k(l)\frac{dT}{dx} = \alpha_N(T(l) - T_0) + \varphi(T)$, где $\varphi(T)$ — заданная функция. Производную аппроксимируйте односторонней разностью

   Получим разностный аналог краевого условия при $x = l$ с помощью аппроксимации:

$$-k_n * \frac{y_N - y_{N-1}}{h} = \alpha_N(y_N - T_0) + \varphi(y_N)$$

3. Опишите алгоритм применения метода прогонки, если при $x = 0$ краевое условие линейное (как в настоящей работе), а при $x = l$, как в п.2.

   Используем простую аппроксимацию и правую прогонку:

$$y_n = \varepsilon_{n+1} * y_{n+1} + \eta_{n+1}$$

$$\varepsilon_1 = 1$$

$$\eta_1 = \frac{h * F_0}{k_0}$$

   Найдем прогоночные коэффициенты:

$$\varepsilon_{n+1} = \frac{C_n}{B_n - A_n * \varepsilon_n}$$

$$\eta_{n+1} = \frac{D_n + A_n * \eta_n}{B_n - A_n * \varepsilon_n}$$

Зная, что $y_{n-1} = \varepsilon_n y_n + \eta_n$, найдем y_n:

$$y_N = \frac{k_N * \eta_N + h * \alpha * \beta - h * \varphi(y_N)}{k_N(1 - \varepsilon_n) + h * \alpha}$$

4. Опишите алгоритм определения единственного значения сеточной функции y_p в одной заданной точке p. Использовать встречную прогонку, т.е. комбинацию правой и левой прогонок (лекция №8). Краевые условия линейные.

Левая прогонка:

$$0 \le n \le p$$

$$y_n = \alpha_{n+1} * y_{n+1} + \beta_{n+1} - \text{прогонка}$$

$$\alpha_{n-1} = \frac{C_n}{B_n - A_n * \alpha_n} - \text{прогоночные коэффициенты}$$

$$\beta_{n-1} = \frac{D_n + A_n * \beta_n}{B_n - A_n * \alpha_n} - \text{прогоночные коэффициенты}$$

Правая прогонка:

$$p \le n \le N$$

$$y_n = \varepsilon_{n+1} * y_{n+1} + \eta_{n+1} - \text{прогонка}$$

$$\varepsilon_{n+1} = \frac{C_n}{B_n - A_n * \varepsilon_n} - \text{прогоночные коэффициенты}$$

$$\eta_{n+1} = \frac{D_n + A_n * \eta_n}{B_n - A_n * \varepsilon_n} - \text{прогоночные коэффициенты}$$

Получим систему, где $n = p$:

$$\begin{cases} y_p = \varepsilon_{p+1} * y_{p+1} + \eta_{p+1} \\ y_{p+1} = \alpha_p * y_p + \beta_p \end{cases}$$

Тогда:

$$y_n = \frac{\varepsilon_{p+1} * \beta_p + \eta_{p+1}}{1 - \varepsilon_{p+1} * \alpha_p}$$