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

Теоретическая часть

Задана математическая модель.
Уравнение для функции $T(x,t)$

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

Краевые условия

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

В обозначениях уравнения (14.1) лекции №14

$$p(x) = \frac{2}{R}\alpha(x)$$

$$f(x) = \frac{2T_0}{R}\alpha(x)$$

Функция $\alpha(x)$ задана уравнением:

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

Константы c и d из условий $\alpha(0) = a_0, \alpha(l) = a_N$,

$$c = -\alpha_0 d = \frac{\alpha_0\alpha_N l}{\alpha_0 - \alpha_N}$$

$$d = \frac{\alpha_N l}{\alpha_N - \alpha_0}$$

Разностная схема

$$\begin{aligned} &\widehat{A}_n \widehat{y}_{n-1} - \widehat{B}_n \widehat{y}_n + \widehat{D}_n \widehat{y}_{n+1} = -\widehat{F}_n, 1 \le n \le N-1 \\ &\widehat{A}_n = \widehat{\chi}_{n-\frac{1}{2}} \frac{\tau}{h}, \\ &\widehat{B}_n = \widehat{A}_n + \widehat{D}_n + \widehat{c}_n h + p_n h\tau, \\ &\widehat{D}_n = \widehat{\chi}_{n+\frac{1}{2}} \frac{\tau}{h}, \\ &\widehat{F}_n = f_n h\tau + \widehat{c}_n y_n h \end{aligned} \quad (2)$$

Разностные аналоги краевых условий при $x = 0$ (получены в Лекции №14)

$$\left(\frac{h}{8}\widehat{c}_{\frac{1}{2}} + \frac{h}{4}\widehat{c}_0 + \widehat{\chi}_{\frac{1}{2}}\frac{\tau}{h} + \frac{\tau h}{8}p_{\frac{1}{2}} + \frac{\tau h}{4}p_0\right)\widehat{y}_0 + \left(\frac{h}{8}\widehat{c}_{\frac{1}{2}} - \widehat{\chi}_{\frac{1}{2}}\frac{\tau}{h} + \frac{\tau h}{8}p_{\frac{1}{2}}\right)\widehat{y}_1 = \\ = \frac{h}{8}\widehat{c}_{\frac{1}{2}}(y_0 + y_1) + \frac{h}{4}\widehat{c}_0 y_0 + \widehat{F}\tau + \frac{\tau h}{4}(\widehat{f}_{\frac{1}{2}} + \widehat{c}_0) \quad (3)$$

Получим интегро-интерполяционным методом разностный аналог краевого условия при $x = l$. Для этого обозначим $F = -k(T)\frac{\partial T}{\partial x}$ и учтем то, что поток

$$\widehat{F}_N = \alpha_N(\widehat{y}_N - T_0), \widehat{F}_{N-\frac{1}{2}} = \widehat{\chi}_{N-\frac{1}{2}}\frac{\widehat{y}_{N-1} - \widehat{y}_N}{h}$$

Проинтегрируем уравнение (1) на отрезке $[x_{N-\frac{1}{2}}, x_N]$

$$\int_{x_{N-\frac{1}{2}}}^{x_N} dx = -\int_{t_m}^{t_{m+1}} dt \int_{x_{N-\frac{1}{2}}}^{x_N} \frac{\partial F}{\partial x} dx - \int_{x_{N-\frac{1}{2}}}^{x_N} dx \int_{t_m}^{t_{m+1}} p(x) T dt +$$
$$+ \int_{x_{N-\frac{1}{2}}}^{x_N} dx \int_{t_m}^{t_{m+1}} f(T) dt$$

$$\int_{x_{N-\frac{1}{2}}}^{x_N} \widehat{c}\ (\widehat{T} - T) dx = -\int_{t_m}^{t_{m+1}} (F_N - F_{N-\frac{1}{2}}) dt - \int_{x_{N-\frac{1}{2}}}^{x_N} p\ \widehat{T}\ \tau dx + \int_{x_{N-\frac{1}{2}}}^{x_N} \widehat{f}\ \tau dx$$

$$\left(\widehat{c}_N\ (\widehat{y}_N - y_N) + \widehat{c}_{N-\frac{1}{2}}\ (\widehat{y}_{N-\frac{1}{2}} - y_{N-\frac{1}{2}}) \right) \frac{h}{4} = -\tau (\widehat{F}_N - \widehat{F}_{N-\frac{1}{2}}) -$$
$$- \left(p_N\ \widehat{y}_N + p_{N-\frac{1}{2}}\ \widehat{y}_{N-\frac{1}{2}} \right) \tau \frac{h}{4} + (\widehat{f}_N + \widehat{f}_{N-\frac{1}{2}}) \tau \frac{h}{4}$$

$$\frac{h}{4}\ \widehat{c}_N\, \widehat{y}_N - \frac{h}{4}\ \widehat{c}_N\, y_N + \frac{h}{8}\ \widehat{c}_{N-\frac{1}{2}}\, \widehat{y}_N + \frac{h}{8}\ \widehat{c}_{N-\frac{1}{2}}\, \widehat{y}_{N-1} - \frac{h}{8}\ \widehat{c}_{N-\frac{1}{2}}\, y_N$$
$$- \frac{h}{8}\ \widehat{c}_{N-\frac{1}{2}}\, y_{N-1} = -\tau \alpha_N\ \widehat{y}_N + \tau \alpha_N T_0 + \tau\ \widehat{\chi}_{N-\frac{1}{2}}\ \frac{\widehat{y}_{N-1}}{h} - \tau\ \widehat{\chi}_{N-\frac{1}{2}}\ \frac{\widehat{y}_N}{h} -$$
$$- p_N\ \widehat{y}_N\ \tau \frac{h}{4} - p_{N-\frac{1}{2}}\ \widehat{y}_N\ \tau \frac{h}{8} - p_{N-\frac{1}{2}}\ \widehat{y}_{N-1}\ \tau \frac{h}{8} + (\widehat{f}_N + \widehat{f}_{N-\frac{1}{2}}) \tau \frac{h}{4}$$

$$\widehat{y}_{N-1} \left(\widehat{c}_{N-\frac{1}{2}}\ \frac{h}{8} - \frac{\tau\ \widehat{\chi}_{N-\frac{1}{2}}}{h} + p_{N-\frac{1}{2}} \tau \frac{h}{8} \right) +$$
$$+ \ \widehat{y}_N \left(\widehat{c}_N\ \frac{h}{4} + \widehat{c}_{N-\frac{1}{2}}\ \frac{h}{8} + \tau \alpha_N + \frac{\tau\ \widehat{\chi}_{N-\frac{1}{2}}}{h} + p_N \tau \frac{h}{4} + p_{N-\frac{1}{2}} \tau \frac{h}{8} \right) =$$
$$= \widehat{c}_{N-\frac{1}{2}} \left(y_N \frac{h}{8} + y_{N-1} \frac{h}{8} \right) + \widehat{c}_N\ y_N \frac{h}{4} + \tau \alpha_N T_0 + (\widehat{f}_N + \widehat{f}_{N-\frac{1}{2}}) \tau \frac{h}{4} \quad (4)$$

Из полученных краевых условий находим коэффициенты $K_0,\ K_N,\ M_0,\ M_N,\ P_0,\ P_N$.

$$\begin{cases} \widehat{K}_0\, \widehat{y}_0 + \widehat{M}_0\, \widehat{y}_1 = \widehat{P}_0 \\ \widehat{A}_n\, \widehat{y}_{n-1} - \widehat{B}_n\, \widehat{y}_n + \widehat{D}_n\, \widehat{y}_{n+1} = -\ \widehat{F}_n, 1 \le n \le N-1 \\ \widehat{K}_N\, \widehat{y}_N + \widehat{M}_{N-1}\, \widehat{y}_{N-1} = \widehat{P}_N \end{cases} \quad (5)$$

Данная система решается методом итераций. Обозначим текущую итерацию s, а предыдущую $s-1$

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

Значения параметров

$k(T) = a_1(b_1 + c_1 T^{m_1}), \frac{\text{Вт}}{\text{см К}}$,
$c(T) = a_2 + b_2 T^{m_2} - \frac{c_2}{T^2}, \frac{\text{Дж}}{\text{см}^3\text{К}}$,
$a_1 = 0.0134$,
$b_1 = 1$,
$c_1 = 4.35 \cdot 10^{-4}$,
$m_1 = 1$,
$a_2 = 2.049$,
$b_2 = 0.563 \cdot 10^{-3}$,
$c_2 = 0.528 \cdot 10^5$,

$m_2 = 1$
$\alpha(x) = \frac{c}{x-d}$,
$\alpha_0 = 0.05\ \frac{\text{Вт}}{\text{см}^2\text{К}}$,
$\alpha_N = 0.01\ \frac{\text{Вт}}{\text{см}^2\text{К}}$,
$l = 10$ см,
$T_0 = 300$ К,
$R = 0.5$ см
$F(t) = 50\frac{\text{Вт}}{\text{см}^2}$.

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 х и меняющееся во времени.

2. Свойства материала стержня привязаны к температуре, т.е. теплоемкость и коэффициент теплопроводности $c(T), k(T)$ зависят от T 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 оте математическая модель описывает нестационарное температурное поле $T(x,t)$. Это поле в дальнейшем с течением времени меняться не будет и должно совпасть с температурным распределением $T(x)$ математическая модель описывает нестационарное температурное поле $T(x)$, получаемым в лаб. работе №3, если все параметры задач совпадают, в частности, вместо $k(T)$ надо использовать $k(x)$ из лаб. работы №3. 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.

Практическая часть

В листинге 1 представлен код программы:

Листинг 1: Листинг программы

```
import matplotlib.pyplot as plt
import numpy as np
from math import fabs


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


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 * alpha(x) / R


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


def func_plus_half(x, step, func):
    return (func(x) + func(x + step)) / 2


def func_minus_half(x, step, func):
    return (func(x) + func(x - step)) / 2


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


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


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


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


def left_boundary_condition(T_prev):
    K0 = h / 8 * func_plus_half(T_prev[0], t, c) + h / 4 * c(T_prev[0]) + \
          func_plus_half(T_prev[0], t, k) * t / h + \
          t * h / 8 * p(h / 2) + t * h / 4 * p(0)

    M0 = h / 8 * func_plus_half(T_prev[0], t, c) - \
          func_plus_half(T_prev[0], t, k) * t / h + \
          t * h * p(h / 2) / 8

```

```
    P0 = h / 8 * func_plus_half(T_prev[0], t, c) * (T_prev[0] + T_prev[1]) + \
          h / 4 * c(T_prev[0]) * T_prev[0] + \
          F0 * t + t * h / 8 * (3 * f(0) + f(h))

    return K0, M0, P0


def right_boundary_condition(T_prev):
    KN = h / 8 * func_minus_half(T_prev[-1], t, c) + h / 4 * c(T_prev[-1]) + \
          func_minus_half(T_prev[-1], t, k) * t / h + t * alphaN + \
          t * h / 8 * p(l - h / 2) + t * h / 4 * p(l)

    MN = h / 8 * func_minus_half(T_prev[-1], t, c) - \
          func_minus_half(T_prev[-1], t, k) * t / h + \
          t * h * p(l - h / 2) / 8

    PN = h / 8 * func_minus_half(T_prev[-1], t, c) * (T_prev[-1] + T_prev[-2]) + \
          h / 4 * c(T_prev[-1]) * T_prev[-1] + t * alphaN * T0 + \
          t * h / 4 * (f(l) + f(l - h / 2))

    return KN, MN, PN


def run(T_prev, K0, M0, P0, KN, MN, PN):
    eps = [0, -M0 / K0]
    eta = [0, P0 / K0]

    x = h
    n = 1
    while (x + h < l):
        eps.append(D(T_prev[n]) / (B(x, T_prev[n]) - A(T_prev[n]) * eps[n]))
        eta.append((F(x, T_prev[n]) + A(T_prev[n]) * eta[n]) / (B(x, T_prev[n]) - A(T_prev[n]) * eps[n]))
        n += 1
        x += h

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

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

    return y


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

    while True:
        prev = T
        while True:
```

```
            K0, M0, P0 = left_boundary_condition(prev)
            KN, MN, PN = right_boundary_condition(prev)
            T_new = run(prev, K0, M0, P0, KN, MN, PN)

            max = fabs((T[0] - T_new[0]) / T_new[0])
            for step2, j in zip(T, T_new):
                d = fabs(step2 - j) / j
                if d > max:
                    max = d
            if max < 1:
                break

            prev = T_new
        res.append(T_new)
        ti += t

        check_eps = 0
        for i, j in zip(T, T_new):
            if fabs((i - j) / j) > 1e-2:
                check_eps = 1
        if check_eps == 0:
            break
        T = T_new

    x = [i for i in np.arange(0, l, h)]
    te = [i for i in range(0, ti, t)]

    step1 = 0
    for i in res:
        if (step1 % 2 == 0):
            plt.plot(x, i[:-1])
        step1 += 1
    plt.plot(x, res[-1][:-1])
    plt.xlabel("x, cm")
    plt.ylabel("T, K")
    plt.grid()
    plt.show()

    step2 = 0
    while (step2 < l / 3):
        point = [j[int(step2 / h)] for j in res]
        plt.plot(te, point[:-1])
        step2 += 0.1
    plt.xlabel("t, sec")
    plt.ylabel("T, K")
    plt.grid()
    plt.show()


if __name__ == "__main__":
    main()
```

1. График зависимости температуры $T(x, t_m)$ от координаты x при нескольких фиксированных значениях времени t_m при заданных выше параметрах.

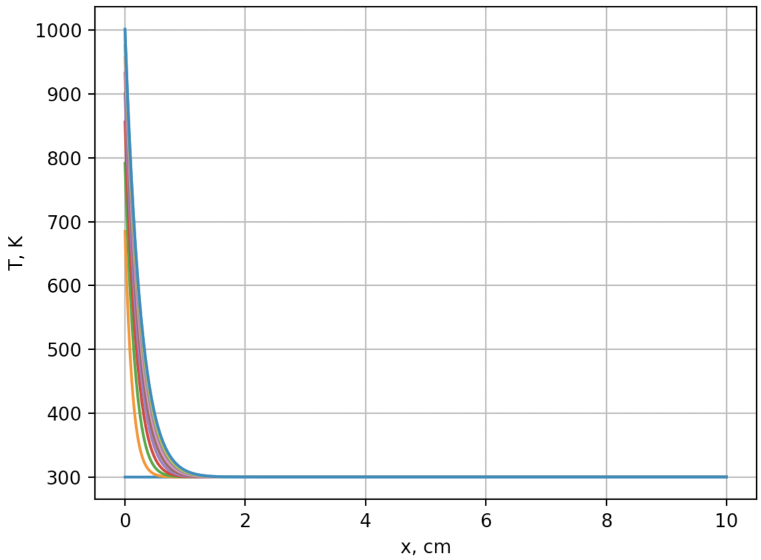

Рис. 1: График зависимости температуры $T(x, t_m)$ от координаты x

2. График зависимости $T(x_n, t)$ при нескольких фиксированных значениях координаты x_n.

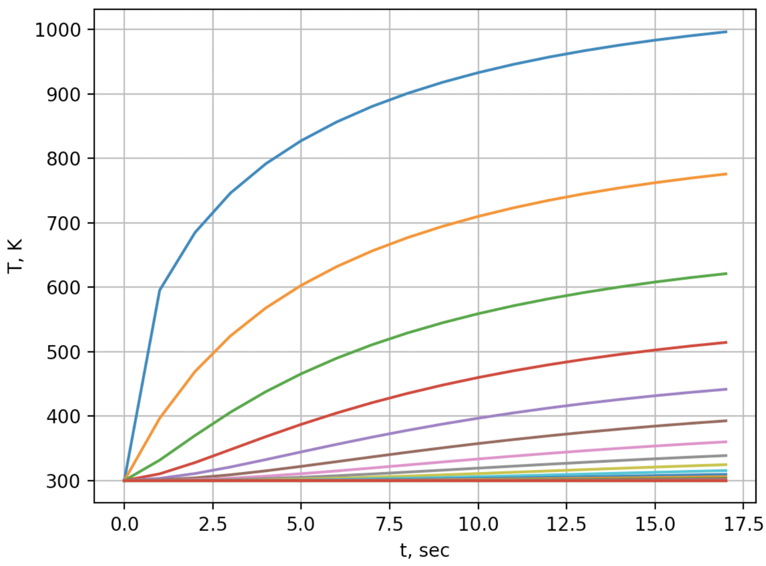

Рис. 2: График зависимости $T(x_n, t)$ при нескольких фиксированных значениях координаты x_n

Верхний график соответствует случаю $x = 0$, нижний случаю $x = l$.

Ответы на вопросы

1. Результаты тестирования программы.
   Если из полученных разностных аналогов краевых условий обнулить $c(u)$, принять τ равным 1 и заменить зависимость коэффициента теплопроводности от T на зависимость от координаты на стержне x, то можно получить график (Рис. 3), соответствующий полученному графику в 3 лабораторной работе (Рис. 4).

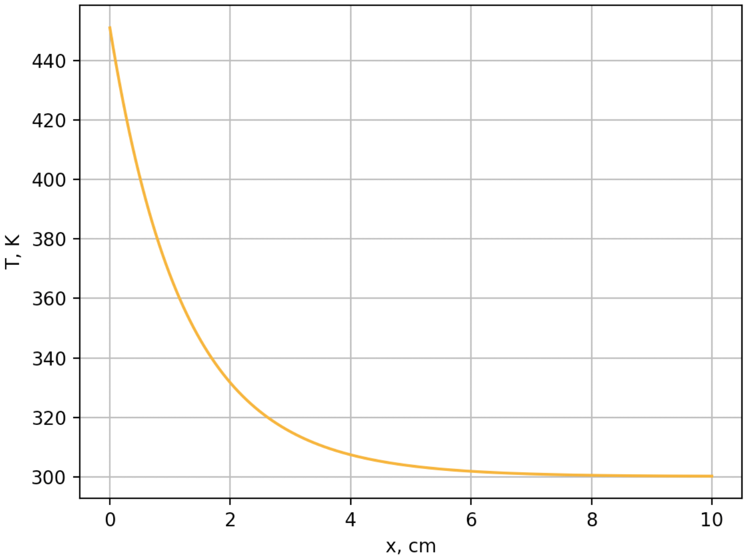

Рис. 3: График из 4 лабораторной работы

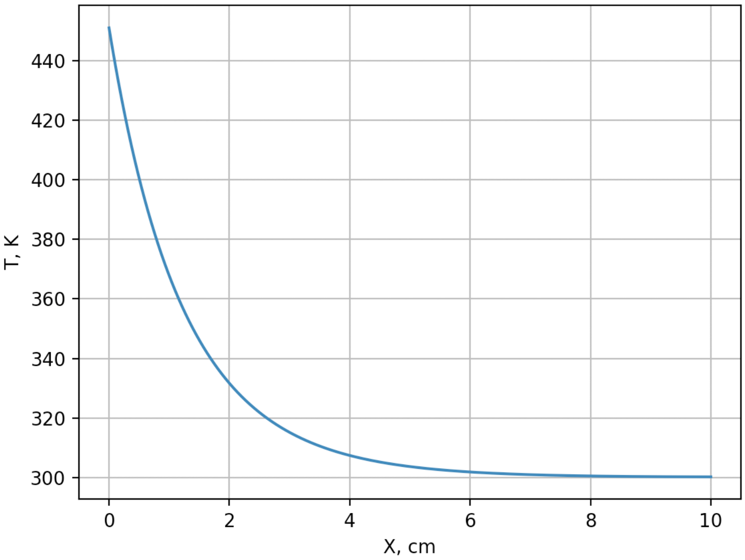

Рис. 4: График из 3 лабораторной работы

Если обнулить поток $F_0(T)$, то на выходе должны получить график температуры, установившейся в соответствии с температурой окружающей среды. (Рис. 5)

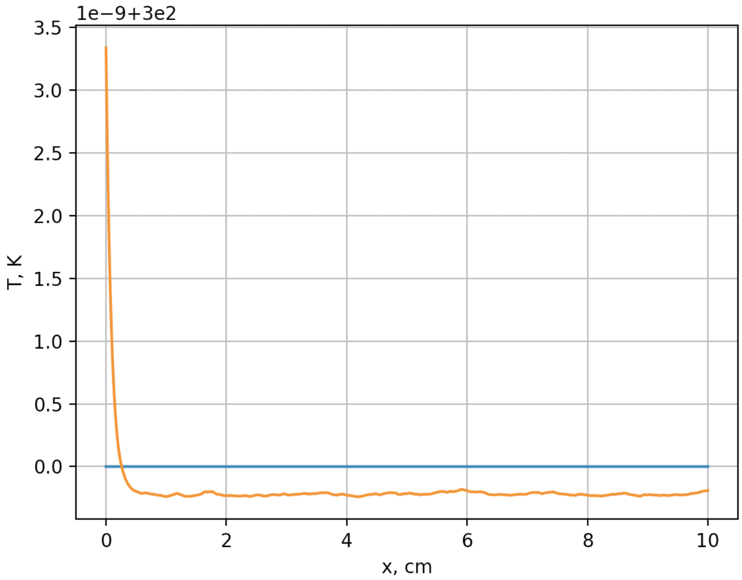

Рис. 5: График при нулевом потоке

2. Выполните линеаризацию уравнения (14.8) по Ньютону, полагая для простоты, что все коэффициенты зависят только от одной переменной $\widehat{y}_n$.

Уравнение, которое необходимо линеаризовать

$$\begin{cases} \widehat{K}_0 \widehat{y}_0 + \widehat{M}_0 \widehat{y}_1 = \widehat{P}_0 \\ \widehat{A}_n \widehat{y}_{n-1} - \widehat{B}_n \widehat{y}_n + \widehat{D}_n \widehat{y}_{n+1} = -\widehat{F}_n, 1 \le n \le N-1 \\ \widehat{K}_N \widehat{y}_N + \widehat{M}_{N-1} \widehat{y}_{N-1} = \widehat{P}_N \end{cases} \tag{6}$$

Выполним линеаризацию методом Ньютона по переменным $\widehat{y}_{n-1}$, $\widehat{y}_n$, $\widehat{y}_{n+1}$
Обозначим текущую итерацию s, а предыдущую итерацию за $(s-1)$.
Начальное распределение задается произвольно $\widehat{y}_0$.
Будем полагать, что значения предыдущей итерации известны.

Зная, что коэффициенты системы зависят только от $\widehat{y}_n$, получим

$$\left(\widehat{A}_n \widehat{y}_{n-1} - \widehat{B}_n \widehat{y}_n + \widehat{D}_n \widehat{y}_{n+1} + \widehat{F}_n \right)\Bigg|_{(s-1)} + \widehat{A}_n^{(s-1)} \Delta \widehat{y}_{n-1}^{(s)} + \\ + \left(\frac{\partial \widehat{A}_n}{\partial \widehat{y}_n} \widehat{y}_{n-1} - \frac{\partial \widehat{B}_n}{\partial \widehat{y}_n} \widehat{y}_n - \widehat{B}_n + \frac{\partial \widehat{D}_n}{\partial \widehat{y}_n} \widehat{y}_{n+1} + \frac{\partial \widehat{F}_n}{\partial \widehat{y}_n} \right)\Bigg|_{(s-1)} \cdot \Delta \widehat{y}_n^{(s)} + \\ + \widehat{D}_n^{(s-1)} \Delta \widehat{y}_{n+1}^{(s)} = 0$$

Введем обозначения

$$A_n = \overset{\frown}{A}{}_n^{(s-1)}$$
$$B_n = \left(-\frac{\partial \overset{\frown}{A}_n}{\partial \overset{\frown}{y}_n} \overset{\frown}{y}_{n-1} + \frac{\partial \overset{\frown}{B}_n}{\partial \overset{\frown}{y}_n} \overset{\frown}{y}_n + \overset{\frown}{B}_n - \frac{\partial \overset{\frown}{D}_n}{\partial \overset{\frown}{y}_n} \overset{\frown}{y}_{n+1} + \frac{\partial \overset{\frown}{F}_n}{\partial \overset{\frown}{y}_n} \right)\Bigg|_{(s-1)}$$
$$D_n = \overset{\frown}{D}{}_n^{(s-1)}$$
$$F_n = \left(\overset{\frown}{A}_n \overset{\frown}{y}_{n-1} - \overset{\frown}{B}_n \overset{\frown}{y}_n + \overset{\frown}{D}_n \overset{\frown}{y}_{n+1} + \overset{\frown}{F}_n \right)\Bigg|_{(s-1)}$$

Тогда

$$A_n \Delta \overset{\frown}{y}{}_{n-1}^{(s)} - B_n \Delta \overset{\frown}{y}{}_n^{(s)} + D_n \Delta \overset{\frown}{y}{}_{n+1}^{(s)} = -F_n$$

Выполним линеаризацию краевых условий
Для $\overset{\frown}{y}_0$

$$\left(\overset{\frown}{K}_0 \overset{\frown}{y}_0 + \overset{\frown}{M}_0 \overset{\frown}{y}_1 - \overset{\frown}{P}_0 \right)\Bigg|_{(s-1)} + \overset{\frown}{K}{}_0^{(s-1)} \Delta \overset{\frown}{y}{}_0^{(s)} + \overset{\frown}{M}{}_0^{(s-1)} \Delta \overset{\frown}{y}{}_1^{(s)} = 0$$

$$K_0 = \overset{\frown}{K}{}_0^{(s-1)}$$
$$M_0 = \overset{\frown}{M}{}_0^{(s-1)}$$
$$P_0 = -\left(\overset{\frown}{K}_0 \overset{\frown}{y}_0 + \overset{\frown}{M}_0 \overset{\frown}{y}_1 - \overset{\frown}{P}_0 \right)\Bigg|_{(s-1)}$$

$$K_0 \Delta \overset{\frown}{y}{}_0^{(s)} + M_0 \Delta \overset{\frown}{y}{}_1^{(s)} = P_0$$

Для $\overset{\frown}{y}_n$

$$\left(\overset{\frown}{K}_N \overset{\frown}{y}_N + \overset{\frown}{M}_{N-1} \overset{\frown}{y}_{N-1} - \overset{\frown}{P}_N \right)\Bigg|_{(s-1)} + \overset{\frown}{K}{}_N^{(s-1)} \Delta \overset{\frown}{y}{}_N^{(s)} + \overset{\frown}{M}{}_{N-1}^{(s-1)} \Delta \overset{\frown}{y}{}_{N-1}^{(s)} = 0$$

$$K_N = \overset{\frown}{K}{}_N^{(s-1)}$$
$$M_{N-1} = \overset{\frown}{M}{}_{N-1}^{(s-1)}$$
$$P_N = -\left(\overset{\frown}{K}_N \overset{\frown}{y}_N + \overset{\frown}{M}_{N-1} \overset{\frown}{y}_{N-1} - \overset{\frown}{P}_N \right)\Bigg|_{(s-1)}$$

$$K_N \Delta \overset{\frown}{y}{}_N^{(s)} + M_{N-1} \Delta \overset{\frown}{y}{}_{N-1}^{(s)} = P_N$$

Получаем систему

$$\begin{cases} A_n \Delta \overset{\frown}{y}{}_{n-1}^{(s)} - B_n \Delta \overset{\frown}{y}{}_n^{(s)} + D_n \Delta \overset{\frown}{y}{}_{n+1}^{(s)} = -F_n \\ K_0 \Delta \overset{\frown}{y}{}_0^{(s)} + M_0 \Delta \overset{\frown}{y}{}_1^{(s)} = P_0 \\ K_N \Delta \overset{\frown}{y}{}_N^{(s)} + M_{N-1} \Delta \overset{\frown}{y}{}_{N-1}^{(s)} = P_N \end{cases} \tag{7}$$

Система решается методом прогонки. Находим все $\Delta \overset{\frown}{y}{}_n^s$. Зная приближение $s-1$, можно найти приближение s

$$\overset{\frown}{y}{}_n^s = \overset{\frown}{y}{}_n^{(s-1)} + \Delta \overset{\frown}{y}{}_n^s$$

Итерационный процесс завершается, когда достигнуто условие
$\max \left| \frac{\Delta \overset{\frown}{y}{}_n^s}{\overset{\frown}{y}{}_n^s} \right| \le \varepsilon$.

Вывод

В ходе лабораторной работы были получены навыки разработки алгоритмов решения смешанной краевой задачи при реализации моделей, построенных на квазилинейном уравнении параболического типа.