Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический
университет имени Н.Э. Баумана»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»
КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа № 4

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

Студент Сиденко А.Г.
Группа ИУ7-63Б
Оценка (баллы)
Преподаватель Градов В.М.

Москва, 2020 г.

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

1. Дано:

$$k(T) = a_1(b_1 + c_1 T^{m_1})$$
$$c(T) = a_2 + b_2 T^{m_2} - \frac{c_2}{T^2}$$
$$a_1 = 0.0134$$
$$b_1 = 1$$
$$c_1 = 4.35 \cdot 10^{-4}$$
$$m_1 = 1$$
$$a_2 = 2.049$$
$$b_2 = 0.563 \cdot 10^{-3}$$
$$c_2 = 0.528 \cdot 10^{5}$$
$$m_2 = 1$$
$$\alpha(x) = \frac{c}{x-d}$$
$$\alpha_0 = 0.05$$
$$\alpha_N = 0.01$$
$$l = 10$$
$$T_0 = 300$$
$$R = 0.5$$
$$F(t) = 50$$

2. Уравнение

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

3. Краевые условия

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

Функция $\alpha(x)$ задана:

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

, где

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

4. Разностная схема

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

5. Краевые условия

Обозначим:

$$\begin{aligned}
F &= -k(T)\frac{\partial T}{\partial x} \\
p(x) &= \frac{2}{R}\alpha(x) \\
f(x) &= \frac{2T_0}{R}\alpha(x)
\end{aligned} \tag{3}$$

Разностные аналоги краевых условий при x=0:

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

Взято из лекции.

Разностные аналоги краевых условий при x=l, вывод:

Проинтегрируем уравнение 1 с учетом замен 3 на отрезке $[x_{N-\frac{1}{2}}, x_N]$

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

Вычисляем интегралы методом правых прямоугольников.

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

Вычисляем интегралы методом трапеций, а первый справа – методом правых прямоугольников.

$$\Big(\overset{\frown}{c}_N (\overset{\frown}{y}_N - y_N) + \overset{\frown}{c}_{N-\frac{1}{2}} (\overset{\frown}{y}_{N-\frac{1}{2}} - y_{N-\frac{1}{2}}) \Big) \frac{h}{4} = -\tau (\overset{\frown}{F}_N - \overset{\frown}{F}_{N-\frac{1}{2}}) - \\ - \big(p_N\, \overset{\frown}{y}_N + p_{N-\frac{1}{2}}\, \overset{\frown}{y}_{N-\frac{1}{2}} \big) \tau \frac{h}{4} + (\overset{\frown}{f}_N + \overset{\frown}{f}_{N-\frac{1}{2}}) \tau \frac{h}{4}$$

Зная

$$\begin{aligned} \overset{\frown}{F}_N &= \alpha_N (\overset{\frown}{y}_N - T_0) \\ \overset{\frown}{F}_{N-\frac{1}{2}} &= \overset{\frown}{\chi}_{N-\frac{1}{2}} \frac{\overset{\frown}{y}_{N-1} - \overset{\frown}{y}_N}{h} \end{aligned} \tag{5}$$

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

$$\overset{\frown}{y}_{N-1} \left(\overset{\frown}{c}_{N-\frac{1}{2}} \frac{h}{8} - \frac{\tau\, \overset{\frown}{\chi}_{N-\frac{1}{2}}}{h} + p_{N-\frac{1}{2}} \tau \frac{h}{8} \right) + \\ + \overset{\frown}{y}_N \left(\overset{\frown}{c}_N \frac{h}{4} + \overset{\frown}{c}_{N-\frac{1}{2}} \frac{h}{8} + \tau \alpha_N + \frac{\tau\, \overset{\frown}{\chi}_{N-\frac{1}{2}}}{h} + p_N \tau \frac{h}{4} + p_{N-\frac{1}{2}} \tau \frac{h}{8} \right) = \\ = \overset{\frown}{c}_{N-\frac{1}{2}} \left(y_N \frac{h}{8} + y_{N-1} \frac{h}{8} \right) + \overset{\frown}{c}_N y_N \frac{h}{4} + \tau \alpha_N T_0 + (\overset{\frown}{f}_N + \overset{\frown}{f}_{N-\frac{1}{2}}) \tau \frac{h}{4} \tag{6}$$

Соответственно из этих краевых условий (формулы 4, 6) можем найти коэффициенты $K_0,\ K_N,\ M_0,\ M_N,\ P_0,\ P_N$.

$$\begin{cases} \overset{\frown}{K}_0 \overset{\frown}{y}_0 + \overset{\frown}{M}_0 \overset{\frown}{y}_1 = \overset{\frown}{P}_0 \\ \overset{\frown}{A}_n \overset{\frown}{y}_{n-1} - \overset{\frown}{B}_n \overset{\frown}{y}_n + \overset{\frown}{D}_n \overset{\frown}{y}_{n+1} = -\overset{\frown}{F}_n, 1 \le n \le N-1 \\ \overset{\frown}{K}_N \overset{\frown}{y}_N + \overset{\frown}{M}_{N-1} \overset{\frown}{y}_{N-1} = \overset{\frown}{P}_N \end{cases} \tag{7}$$

Система 7 решается методом простых итераций.

6. Метод простых итераций

Обозначим текущую итерацию s, а предыдущую (s-1) , тогда итерационный процесс организуется по схеме

$$A_n^{s-1} y_{n+1}^s - B_n^{s-1} y_n^s + D_n^{s-1} y_{n-1}^s = -F_n^{s-1} \tag{8}$$

Решение которой осуществляется методом прогонки. Итерации прекращаются при условии

$$\max \left| \frac{y_n^s - y_n^{s-1}}{y_n^s} \right| \le \varepsilon, \text{для всех } n = \overline{0...N}$$

Приведем листинг.

Листинг 1: Заданные параметры

```
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 = 0.01
t = 1
```

Листинг 2: Коэффициенты теплопроводности материала стержня и теплоотдачи при обдуве; теплоемкость

```
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)
```

Листинг 3: Выполненные замены

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

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

Листинг 4: Метод средних для вычисления значения функций

```
def func_plus_1_2(x, step, func):
    return (func(x) + func(x + step)) / 2

def func_minus_1_2(x, step, func):
    return (func(x) + func(x - step)) / 2
```

Листинг 5: Функции для нахождения параметров разностной схемы

```
def A(T):
    return func_minus_1_2(T, t, k) * t / h

def D(T):
    return func_plus_1_2(T, t, k) * t / h

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
```

Листинг 6: Метод прогонки

```
def run_through(prevT):
```

```
    K0 = h / 8 * func_plus_1_2(prevT[0], t, c) + \
                h / 4 * c(prevT[0]) + \
                func_plus_1_2(prevT[0], t, k) * t / h + \
                t * h / 8 * p(h / 2) + t * h / 4 * p(0)

    M0 = h / 8 * func_plus_1_2(prevT[0], t, c) - \
                func_plus_1_2(prevT[0], t, k) * t / h + \
                t * h * p(h / 2) / 8

    P0 = h / 8 * func_plus_1_2(prevT[0], t, c) * (prevT[0]+ \
                prevT[1]) + \
                h / 4 * c(prevT[0]) * prevT[0] + \
                F0 * t + t * h / 8 * (3 * f(0) + f(h))

    KN = h / 8 * func_minus_1_2(prevT[-1], t, c) + \
                h / 4 * c(prevT[-1]) + \
                func_minus_1_2(prevT[-1], t, k) * t / h + \
                t * alphaN + \
                t * h / 8 * p(l - h / 2) + t * h / 4 * p(l)

    MN = h / 8 * func_minus_1_2(prevT[-1], t, c) - \
                func_minus_1_2(prevT[-1], t, k) * t / h + \
                t * h * p(l - h / 2) / 8

    PN = h / 8 * func_minus_1_2(prevT[-1], t, c) * \
                (prevT[-1] + prevT[-2]) + \
                h / 4 * c(prevT[-1]) * prevT[-1] + \
                t * alphaN * T0 + \
                t * h / 4 * (f(l) + f(l - h / 2))

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

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

```
        x += h

    # Обратный ход
    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
```

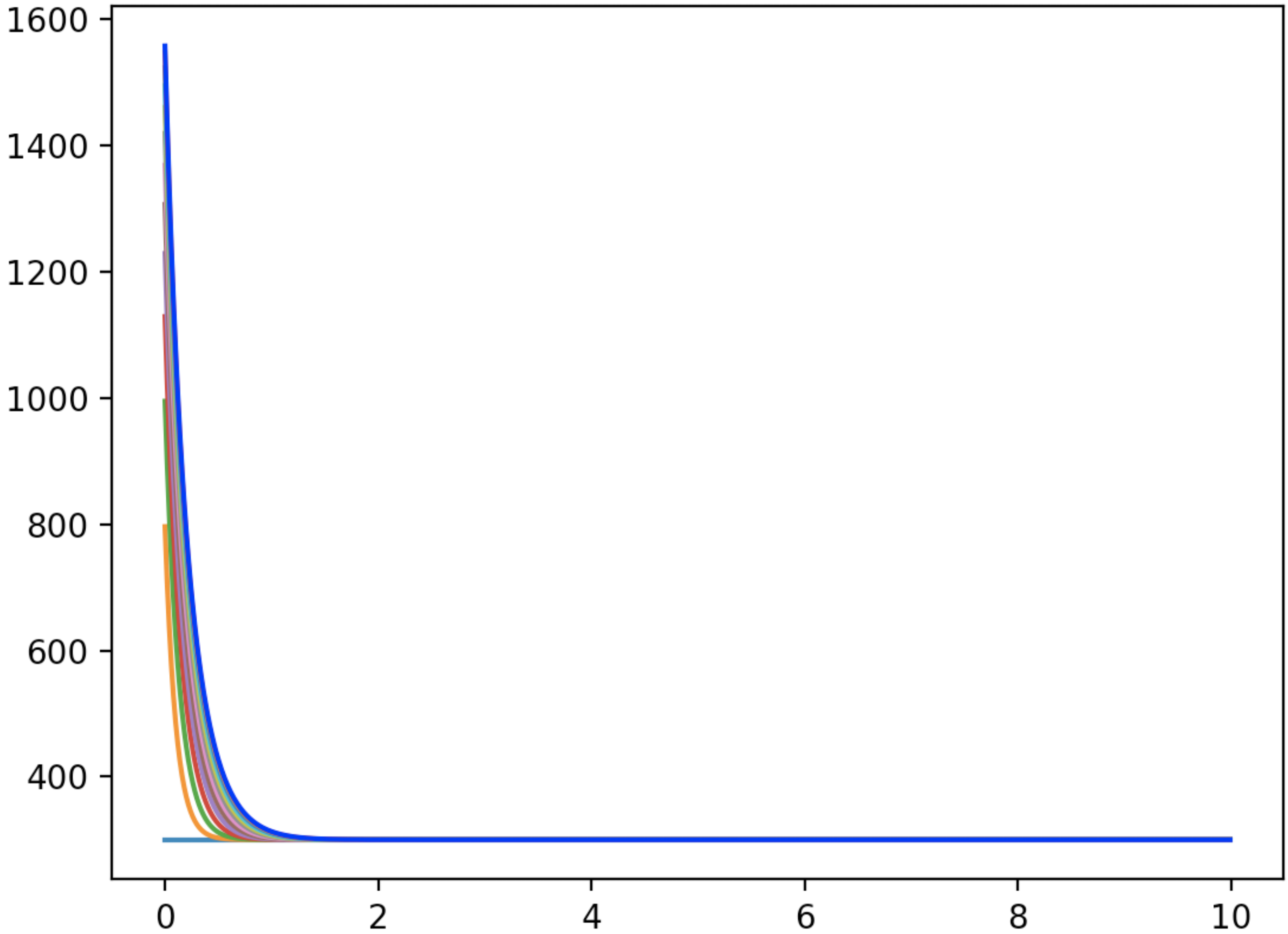
Листинг 7: Метод простых итераций

```
def check_eps(T, newT):
    for i, j in zip(T, newT):
        if fabs((i - j) / j) > 1e-2:
            return True
    return False

def check_iter(T, newT):
    max = fabs((T[0] - newT[0]) / newT[0])
    for i, j in zip(T, newT):
        d = fabs(i - j) / j
        if d > max:
            max = d
    return max < 1e-2

def iter_method():
    result = []
    n = int(l / h)
    T = [T0] * (n + 1)
    newT = [0] * n
    ti = 0

    result.append(T)

    while (True):
        buf = T
        while True:
            newT = run_through(buf)
            if check_iter(buf, newT):
                break
            buf = newT

```

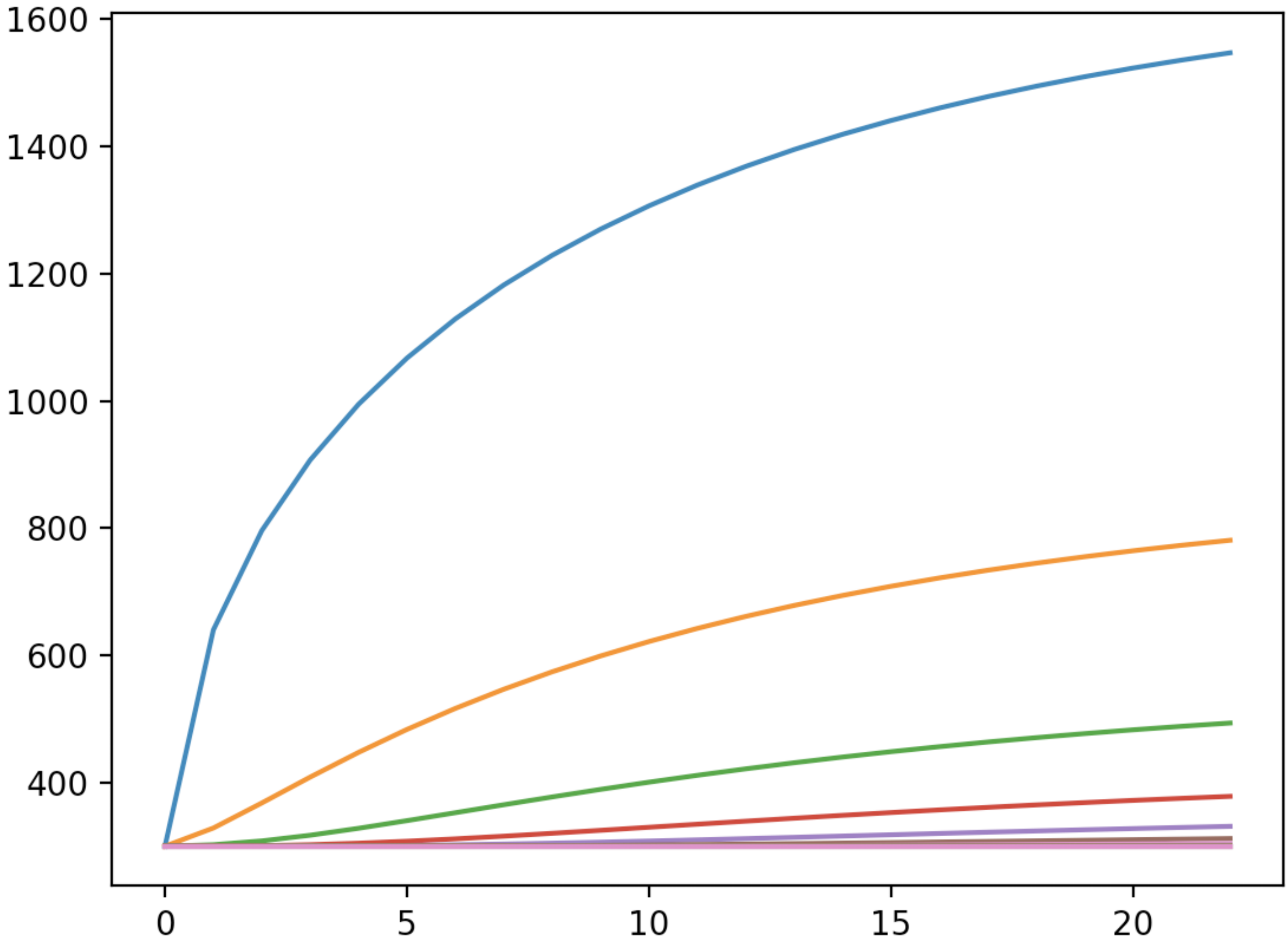
```
        result.append(newT)

        ti += t
        if (check_eps(T, newT) == False):
            break

        T = newT

    return result, ti
```

Результаты работы программы.

1. График зависимости температуры от координаты при нескольких фиксированных значениях времени.



2. График зависимости температуры от времени при нескольких фиксированных значениях координаты.


1600
1400
1200
1000
800
600
400
0
5
10
15
20

Ответы на вопросы.

1. Какие способы тестирования программы можно предложить?

   3 способа тестирования:

   (a) Задаем тепловой поток отрицательным, это означает что идет съем тепла с левого торца $x = 0$, а значит температура от 0 до l будет увеличиваться. (производная положительная)

   (b) Увеличиваем коэффициент теплоотдачи (стержень при том же самом тепловом потоке отдает больше тепла). А следовательно температура снижается, а скорость снижения температуры (от длины стержня) увеличивается.

   (c) Тепловой поток устанавливаем в 0. Съема тепла не происходит, причин для нагрева нет. А значит стержень так и останется при температуре окружающей среды.

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.

Построим простейшую разностную схему методом разностной апроксимации на равномерной сетке с шагом h.

Тогда в любой точке разностный аналог:

$$\frac{dT}{dx} = \frac{T(x+h) - T(x)}{h}$$

Тогда при $x = l$:

$$\frac{dT}{dx} = \frac{T_{l+1} - T_l}{h}$$

И тогда

$$-k_l\frac{T_l - T_{l-1}}{h} = \alpha_N(T_l - T_0) + \varphi(T_l)$$

, где $T_l = T(l), k_l = k(l)$

$$-k_lT_l + k_lT_{l-1} = \alpha_N hT_l - \alpha_N hT_0 + \varphi(T_l)h$$

$$-(k_l + \alpha_N h)T_l + k_lT_{l-1} = \varphi(T_l)h - \alpha_N hT_0$$

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.

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

Изменим направление прогонки, т.е. прогоночные коэффициенты определять справа налево, а функцию - слева направо. Такая прогонка называется левой. В этом случае основная прогоночная формула записывается в виде

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

Принимая простейшую (первого порядка точности) аппроксимацию краевого условия при $x = 0$, получим его разностный аналог

$$-k_0\frac{T_1 - T_0}{h} = F_0$$

$$-k_0T_1 + k_0T_0 = F_0h$$

$$\begin{cases} \varepsilon_0 = 1 \\ \eta_0 = -\frac{F_0h}{k_0} \end{cases}$$

Аналогичная разностная аппроксимация правого краевого условия имеет вид:

$$-(k_l + \alpha_N h)T_l + k_lT_{l-1} = \varphi(T_l)h - \alpha_N hT_0$$

$$-(k_l + \alpha_N h)T_l + k_l\frac{T_l - \eta_{l-1}}{\varepsilon_{l-1}} = \varphi(T_l)h - \alpha_N hT_0$$

Откуда получаем уравнение для определения T_0.

$$\varphi(T_l)h - (k_l + \alpha_N h - \frac{k_l}{\varepsilon_{l-1}})T_l = \frac{k_l\eta_{l-1}}{\varepsilon_{l-1}} - \alpha_N hT_0$$

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

   Комбинируя левую и правую прогонки, получаем метод встречных прогонок.

Пусть $i = p$, где $0 < p < N$.

Тогда в области $0 \leq i \leq p+1$ прогоночные коэффициенты α_i, β_i (правая прогонка):

$$\alpha_{i+1} = \frac{C_i}{B_i - \alpha_i A_i}$$

$$\beta_{i+1} = \frac{A_i \beta_i + D_i}{C_i - \alpha_i A_i}$$

А в области $p \leq i \leq N$ прогоночные коэффициенты ε_i, η_i (левая прогонка):

$$\varepsilon_i = \frac{C_i}{B_i - \varepsilon_{i+1} A_i}$$

$$\eta_i = \frac{A_i \eta_{i+1} + D_i}{B_i - \varepsilon_{i+1} A_i}$$

Тогда при $i = p$:

$$y_p = \alpha_{p+1} y_{p+1} + \beta_{p+1}, \quad y_{p+1} = \varepsilon_{p+1} y_p + \eta_{p+1}$$

И тогда:

$$y_p = \frac{\beta_{p+1} + \alpha_{p+1} \eta_{p+1}}{1 - \alpha_{p+1} \varepsilon_{p+1}}$$