Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»
Кафедра информатики

Отчет по лабораторной работе №3
Метод сеток решения одномерного нестационарного уравнения теплопроводности

Выполнил:
студент гр. 953505
Басенко К. А.

Руководитель:
доцент
Анисимов В. Я.

Минск 2021

Цель работы:

- изучить метод разностных аппроксимаций для уравнения теплопроводности;
- составить алгоритмы решения уравнения теплопроводности методом сеток, применимыми для организации вычислений на ПЭВМ;
- составить программы решения уравнения теплопроводности по разработанным алгоритмам;
- выполнить тестовые примеры и проверить правильность работы программ;
- получить численное решение заданного уравнения теплопроводности.

Краткие теоретические сведения

Требуется найти непрерывную на замкнутом прямоугольнике $\overline{D}$ функцию $u(x, t)$, которая на D' удовлетворяет уравнению теплопроводности

$$Lu \equiv \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = f(x,t), \qquad (2.10)$$

которое при $t = 0$ удовлетворяет начальному условию

$$u(x, 0) = s(x), \qquad (2.11)$$

а при $x = 0$ и $x = 1$ подчиняется краевым условиям

$$u(0, t) = p(t),\ \ u(1, t) = q(t), \qquad (2.12)$$

где $f(x, t)$, $s(x)$, $p(t)$, $q(t)$ − заданные достаточно гладкие функции, причем $s(0) = p(0),\ \ s(l) = q(l)$.

Задача (2.10) − (2. 12) называется *смешанной задачей*, поскольку она содержит как *начальные условия*, так и *краевые условия*. Известно, [11] что у поставленной задачи существует единственное решение $u(x, t)$. Мы будем предполагать, что это решение имеет на замкнутом прямоугольнике $\overline{D}$ непрерывные частные производные $\partial u/\partial t$, $\partial^2 u/\partial t^2$, $\partial^2 u/\partial x^2$, $\partial^4 u/\partial x^4$.

Сетки и нормы.

Пусть $h = 1/N$, $\tau = T/M$ − шаги по x и t, где N, M – натуральные числа, а $x_k = kh$, $t_\nu = \nu\tau$, $u_k^\nu = u(x_k, t_\nu)$. Построим сетки (рис. 2.1)

$\omega_h = \{(x_k, t_\nu):\ k = 0, 1, \ldots, N,\ \nu = 0, 1, \ldots, M\}$,

$\omega_h' = \{(x_k, t_\nu):\ k = 1, 2, \ldots, N\text{-}1,\ \nu = 1, 2, \ldots, M\}$,

$\omega_h^* = \omega_h \setminus \omega_h'$.

Сетка ω_h^* состоит из узлов сетки ω_h, обозначенных на рис. 2.1 крестиками. Эти узлы расположены на трех сторонах прямоугольника $\overline{D}$, на которых заданы начальное и краевые условия. Сетка ω_h' состоит из остальных узлов сетки ω_h. Зададим для сеточных функции определенных на ω_h или на ω_h', следующие нормы:

$$\|y\|_h = \max_{\omega_h} |y_k^\nu|, \quad \|y\|_h' = \max_{\omega_h'} |y_k^\nu|. \tag{2.13}$$

Введем разностный оператор Λ:

$$\Lambda y_k^\nu = -\frac{y_{k-1}^\nu - 2y_k^\nu + y_{k+1}^\nu}{h^2}. \tag{2.14}$$

Здесь под выражением Λy_k^ν подразумевается значение сеточной функции Λy в точке с координатами (x_k, t_ν), т. е. $(\Lambda y)_k^\nu$.

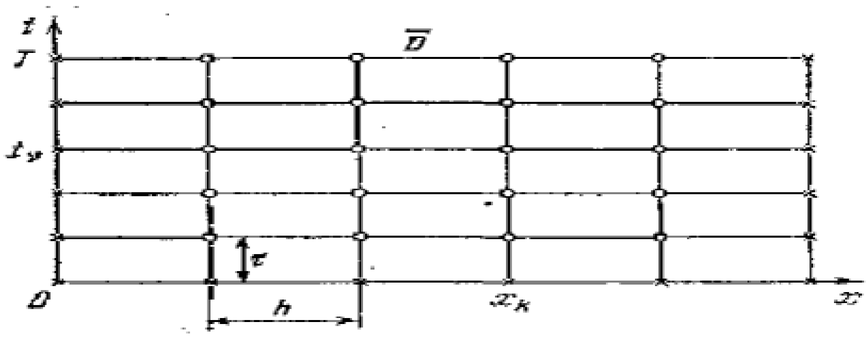


Рис. 2.1

Скобки в выражении (2.14) опущены для упрощения записи. Аналогичные упрощения в записи будем допускать и при введении других операторов. Зададим на сетке ω_h^* тождественный оператор

$$l^h y \equiv y \qquad (2.15)$$

и сеточную функцию

$$g = \begin{cases} s(x_k), & x = x_k, \quad t = 0, \quad k = 1, \ 2, \ldots, N-1 \\ p(t_\nu), & x = 0, \quad t = t_\nu, \quad \nu = 0, \ 1, \ldots, M, \\ q(t_\nu), & x = 1, \quad t = t_\nu, \quad \nu = 0, \ 1, \ldots, M. \end{cases} \qquad (2.16)$$

Рассмотрим две разностные схемы:

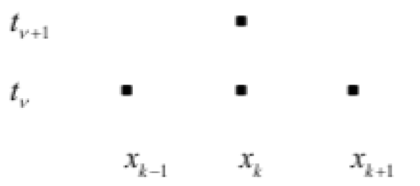
$$L_1^h y_k^\nu \equiv \frac{y_k^\nu - y_k^{\nu-1}}{\tau} + \Lambda y_k^{\nu-1} = f_k^{\nu-1}, \qquad (2.17)$$

$$l^h y = g, \qquad (2.18)$$

$$L_2^h y_k^\nu \equiv \frac{y_k^\nu - y_k^{\nu-1}}{\tau} + \Lambda y_k^\nu = f_k^\nu, \qquad (2.19)$$

$$l^h y = g. \qquad (2.20)$$

Здесь и далее индекс k изменяется от 1 до N-1, $\nu = 1, 2, \ldots, M$. Шаблоны разностных уравнений (2.17) и (2.19) представлены соответственно на рис. (2.2) и (2.3). Обе разностные



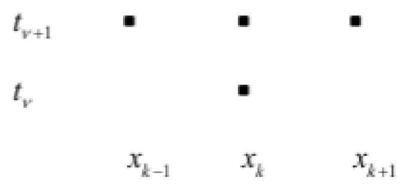


Рис. 2.2 Рис. 2.3

схемы (2.17)–(2.18) и (2.19)– (2.20) называются *двухслойными*, так как шаблоны разностных уравнений (2.17) и (2.19) содержат узлы, лежащие только на двух временных *слоях* − подмножествах сетки ω_h, отвечающих значениям времени $t = t_{\nu-1}$ и $t = t_\nu$. Слой, находящийся на горизонтальной прямой $t = t_{\nu-1}$, называется *нижним*, а слой, находящийся на горизонтальной прямой $t = t_\nu$, − *верхним*. Разностные схемы (2.17), (2.18) и (2.19), (2.20) отличаются тем, что в уравнении (2.17) оператор Λ действует на нижнем слое, а в уравнении (2.19) оператор Λ вынесен на верхний слой, и, кроме того, значения правой части $f_k^{\nu-1} = f(x_k, t_{\nu-1})$ и $f_k^\nu = f(x_k, t_\nu)$ берутся на разных слоях. Ограничимся пока предложенным формальным описанием двух разностных схем. Их качественное различие будет описано далее в данной лабораторной работе.

Аппроксимация.

Сопоставляя, с одной стороны, дифференциальное уравнение (2.10), а, с другой стороны, разностные уравнения (2.17) и (2.19), видим, что частной производной u_t' отвечает разностная производная $\dfrac{y_k^\nu - y_k^{\nu-1}}{\tau}$, а частной производной $-u_{xx}''$ соответствует разностная производная второго порядка в направлении *x*, образуемая с противоположным знаком с помощью оператора Λ (2.14).

Пусть $u(x,t)$ − решение задачи (2.10) − (2.12). Поскольку частные производные $\partial^2 u/\partial t^2$ и $\partial^4 u/\partial x^4$ по предположению непрерывны и, следовательно, ограничены на замкнутом прямоугольнике $\overline{D}$, то согласно (2.14),

$$\Lambda y_k^{\nu-1} = -u_{xx}''(x_k, t_{\nu-1}) + r_k^\nu, \tag{2.21}$$

$$\frac{u_k^\nu - u_k^{\nu-1}}{\tau} = u_t'(x_k, t_{\nu-1}) + \rho_k^\nu, \tag{2.22}$$

где $k = 1, 2, \ldots, M\text{-}1$, $\nu = 1, 2, \ldots, N$,

$$\left|r_k^\nu\right| \le c_1 h^2, \ \left|\rho_k^\nu\right| \le c_2 \tau, \tag{2.23}$$

а c_1, c_2 − некоторые постоянные, не зависящие от h, τ, k, ν. В силу непрерывности частных производных u_t' и u_{xx}'', на $\overline{D}$ решение задачи (2.10) − (2.12) удовлетворяет уравнению (2.10) на замкнутом прямоугольнике $\overline{D}$.

Следовательно, выполняется равенство

$$u_t'(x_k,t_{\nu-1}) - u_{xx}''(x_k,t_{\nu-1}) = f_k^{\nu-1} \tag{2.24}$$

для $k = 1, 2, \ldots, N-1$, $\nu = 1, 2, \ldots, M$, т. е., в частности, и для $t_{\nu-1} = 0$.

Согласно (2.21), (2.22), (2.23) невязка ψ_1 решения u задачи (2.10) — (2.12) для разностного уравнения (2.17) имеет следующее выражение:

$$\psi_{1k}^{\nu} = L_1^h y_k^{\nu} - f_k^{\nu-1} = \frac{u_k^{\nu} - y_k^{\nu-1}}{\tau} + \Lambda y_k^{\nu-1} - f_k^{\nu-1} =$$

$$= u_t'(x_k,t_{\nu-1}) + \rho_k^{\nu} - u_{xx}''(x_k,t_{\nu-1}) + r_k^{\nu} - f_k^{\nu-1} = r_k^{\nu} + \rho_k^{\nu}.$$

Отсюда с учетом (2. 23) получаем

$$\|\Psi_1\|_h' = \max_{\omega_h'} |\Psi_{1k}^{\nu}| = \max_{1\le\nu\le M} \max_{1\le k\le N-1} |r_k^{\nu} + \rho_k^{\nu}| = O(h^2 + \tau). \tag{2.25}$$

Аналогично находим

$$\|\Psi_2\|_h' = O(h^2 + \tau), \tag{2.26}$$

где Ψ_2 − невязка решения u задачи (2.10) − (2.12) для разностного уравнения (2.19).

Таким образом, оба разностных уравнения (2.17) и (2.19) аппроксимируют дифференциальное уравнение (2.10) на решении u задачи (2.10) — (2.12) со вторым порядком по h и с первым порядком по τ.

Дополнительные условия, т. е. начальное условие (2.11) и краевые условия (2.12), аппроксимируются на сетке ω_h^* с помощью тождественного оператора l^h условием (2.18) или соответственно условием (2.20) точно, т. е. невязка решения u задачи (2.10) − (2.12) для условий (2.18) и (2.20) равна нулю на сетке ω_h^*.

Итак, обе разностные схемы (2.17), (2.18) и (2.19), (2.20), с точки зрения аппроксимации задачи (2.10) − (2.12), обладают одинаковой по порядку относительно h и τ гарантируемой точностью.

Вычислительные алгоритмы

Разрешив разностное уравнение (2.17) относительно y_k^{ν}, получим

$$y_k^{\nu} = \frac{\tau}{h^2} y_{k-1}^{\nu-1} + \left(1 - \frac{2\tau}{h^2}\right) y_k^{\nu-1} + \frac{\tau}{h^2} y_{k+1}^{\nu-1} + \tau f_k^{\nu-1}. \tag{2.27}$$

Поскольку y_k^0, y_0^ν, y_N^ν, k = 1, 2, . . . , N–l, ν = 0, 1, . . . , M известны (они задаются на ω_h^* условием (2.18)), решение разностной схемы (2.17), (2.18) находится по формуле (2.27) явно, слой за слоем. Поэтому разностная схема (2.17), (2.18) называется *явной*.

Разностное уравнение (2.19) с учетом (2.14) может быть записано в виде

$$\frac{\tau}{h^2} y_{k-1}^\nu - (1 + \frac{2\tau}{h^2}) y_k^\nu + \frac{\tau}{h^2} y_{k+1}^\nu = -y_k^{\nu-1} - \tau f_k^\nu \,. \tag{2.28}$$

Согласно (2.15), (2.16), (2.20) имеем также

$$y_0^\nu = \rho_\nu \,, \quad y_N^\nu = q_\nu \tag{2.29}$$

Таким образом, если $y_k^{\nu-1}$, k = 1, 2, . . . , N–l, известны (в частности, y_k^0, k = 1, 2, . . . , N–l, заданы условием (2.20)), то для нахождения решения разностной схемы (2.19), (2.20) на следующем ν-м слое нужно решить трехточечное разностное уравнение (2.28) с краевыми условиями первого рода (2.29), т. е. разностную краевую задачу. Поэтому разностная схема (2.19), (2.20) называется *неявной*.

Для нахождения разностного решения на ν-м слое может быть применен метод прогонки, [2] поскольку для задачи (2.28), (2.29) достаточные условия выполнены (проверьте, положив $k = j$, $y_k^\nu = z_j$, $y_{k\pm1}^\nu = z_{j\pm1}$, $-y_k^{\nu-1} - \tau f_k^\nu = F_j$). При этом число выполняемых арифметических действии для нахождения разностного решения на одном слое имеет величину порядка $O(N)$, т. е. по порядку относительно N не больше, чем при применении явной формулы (2.27) для схемы (2.17), (2.18).

Устойчивость и сходимость

Так как дополнительные условия (2.11), (2.12) аппроксимируются в разностных схемах (2.17), (2.18) и (2.19), (2.20) на сетке ω_h^* точно, то нам будет достаточно исследовать устойчивость только по правой части. Остановимся сначала на разностной схеме (2.17), (2.18). Для исследования ее устойчивости по правой части нужно рассмотреть решение z вспомогательной разностной задачи:

$$L_1^h z_k^\nu \equiv \frac{z_k^\nu - z_k^{\nu-1}}{\tau} + \Lambda z_k^{\nu-1} = \xi_k^\nu, \tag{2.30}$$

$$l^h z = 0, \tag{2.31}$$

где ξ − произвольная заданная на ω_h' сеточная функция.

Разрешив разностное уравнение (2.30) относительно z_k^ν, аналогично (2.27) получаем

$$z_k^\nu = \frac{\tau}{h^2} z_{k-1}^{\nu-1} + (1 - \frac{2\tau}{h^2}) z_k^{\nu-1} + \frac{\tau}{h^2} z_{k+1}^{\nu-1} + \tau \xi_k^\nu, \tag{2.32}$$

где $K = 1, 2, \ldots, N-1$, $\nu = 1, 2, \ldots, M$.

Кроме того, в соответствии с (2.22) имеем

$$\begin{gathered} z_k^0 = 0,\ k = 1, 2, \ldots, N-1; \\ z_0^\nu = z_N^\nu = 0,\ \nu = 0, 1, \ldots, M. \end{gathered} \tag{2.33}$$

Предположим, что τ и h удовлетворяют следующему условию:

$$\frac{\tau}{h^2} \le \frac{1}{2}. \tag{2.34}$$

Тогда очевидно, что

$$\frac{\tau}{h^2} + \left|1 - \frac{2\tau}{h^2}\right| + \frac{\tau}{h^2} = 1.$$

Отсюда и из (2.32), (2.33) вытекает неравенство

$$\max_{0 \le k \le N} \left|z_k^\nu\right| \le \max_{0 \le k \le N} \left|z_k^{\nu-1}\right| + \tau \max_{1 \le k \le N-1} \left|\xi_k^\nu\right|, \tag{2.35}$$

и поскольку $\max\limits_{0 \le k \le N} \left|z_k^0\right| = 0$, то

$$\max_{0 \le k \le N} \left|z_k^\nu\right| \le \nu\tau \left\|\xi\right\|_h' .$$

Следовательно,

$$\left\|z\right\|_h = \max_{0 \le \nu \le M} \max_{0 \le k \le N} \left|z_k^\nu\right| \le M\tau \left\|\xi\right\|_h' = T \left\|\xi\right\|_h'$$

или, окончательно,

$$\|z\|_h \le T\,\|\xi\|'_h. \tag{2.36}$$

Полученное неравенство (2.36) для решения задачи (2.30), (2.31), в котором постоянная T не зависит от h и τ, а также от функции ξ, означает устойчивость разностной схемы (2.17), (2.18) по правой части при условии (2.34). Можно доказать, что нарушение условия (2.34) может привести к нарушению устойчивости разностной схемы (2.17), (2.18). В частности, если $h \to 0$, $\tau \to 0$, а $\tau / h^2 \ge$ const > 1/2, то разностная схема (2.17), (2.18) будет неустойчива.

Для исследования устойчивости разностной схемы (2.19), (2.20) зададим на ω'_h произвольную сеточную функцию ξ и рассмотрим разностную задачу

$$L_2^h z_k^\nu \equiv \frac{z_k^\nu - z_k^{\nu-1}}{\tau} + \Lambda z_k^\nu = \xi_k^\nu, \tag{2.37}$$

$$l^h z = 0, \tag{2.38}$$

не накладывая никаких ограничений на соотношение шагов τ и h. Задачу (2.37), (2.38) можно аналогично (2.28), (2.29) записать в следующем виде:

$$\frac{\tau}{h^2} z_{k-1}^\nu - \left(1 + \frac{2\tau}{h^2}\right) z_k^\nu + \frac{\tau}{h^2} z_{k-1}^\nu = -\, z_k^{\nu-1} - \tau \xi_k^\nu, \tag{2.39}$$

$$z_0^\nu = 0,\ z_N^\nu = 0. \tag{2.40}$$

Если $z_k^{\nu-1}$, k = 1, 2, . . . , $N-1$, известны (в частности, по условию (2.38) $z_k^0 = 0$, k = 0, 1, . . . , N), то, как отмечалось ранее, для разностной задачи (2.39), (2.40), где ν фиксировано, выполнены условия (2.12). Следовательно, по лемме эта задача однозначно разрешима на ν-м слое.

Очевидно, имеется такое k', $0<k'<N$, что

$$\left|z_{k'}^\nu\right| = \max_{0 \le k \le N} \left|z_k^\nu\right|. \tag{2.41}$$

Так как $\left|z_{k'-1}^\nu\right| \le \left|z_{k'}^\nu\right|$, $\left|z_{k'+1}^\nu\right| \le \left|z_{k'}^\nu\right|$, то

$$\left|z_{k'}^{\nu}\right| \le \left|z_{k'}^{\nu}(1+\frac{2\tau}{h^2}) - \frac{\tau}{h^2}\left(z_{k'-1}^{\nu} + z_{k'+1}^{\nu}\right)\right|$$

и, следовательно, согласно (2.39)

$$\left|z_{k'}^{\nu}\right| \le \left|z_{k'}^{\nu-1}\right| + \tau\left|\xi_{k'}^{\nu}\right|.$$

Из полученного неравенства с учетом (2.41) вытекает неравенство (2.35) и, в конечном счете, оценка (2.36), что и означает устойчивость по правой части разностной схемы (2.19), (2.20) при любом соотношении шагов τ и h.

Итак, поскольку дополнительные условия (2.11), (2.12) аппроксимируются на $\overset{*}{\omega}_h$ точно, то из аппроксимации (см. (2.25), (2.26)) и установленной устойчивости по правой части в силу основной теоремы теории разностных схем вытекает сходимость решений разностных схем (2.17), (2.18) и (2.19), (2.20) к решению задачи (2.10) − (2.12) со вторым порядком по h и с первым порядком по τ, т. е.

$$\|u - y\|_h = O(h^2 + \tau). \qquad (2.42)$$

При этом в случае явной схемы (2.17), (2.18) предполагается выполнение условия (2.34).

Определение. Разностная схема, устойчивая при любом соотношении шагов τ и h, называется *абсолютно устойчивой*, а устойчивая при ограничениях на τ и h − *условно устойчивой*.

Недостатком разностной схемы (2.17), (2.18) является ее условная устойчивость (ограничение (2.34) является жестким для шага τ по времени). Преимущество − простота счета по явной формуле (2.27) и возможность распространения на задачу Коши (когда условие (2.11) задано на всей оси x, а краевые условия (2.12) отсутствуют). В случае смешанной задачи (2.10) − (2.12) предпочтение отдают неявной абсолютно устойчивой разностной схеме (2.19), (2.20). Разностная краевая задача (2.28), (2.29) при переходе на каждый следующий слой решается методом прогонки весьма эффективно.

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ №13

Задача 1. Промоделировать стационарные процессы теплопроводности стержня в зависимости от входных данных задачи:

$$\begin{cases} -\dfrac{d}{dx}\left(K(x)\dfrac{du}{dx}\right) = f, \\ u(a) = U_A, \quad u(b) = U_B. \end{cases}$$

Программная реализация:

Записываем:

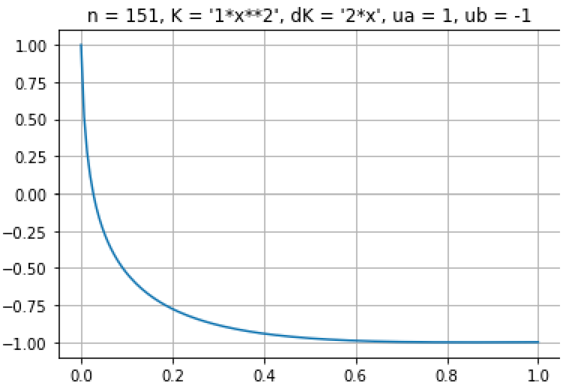
```
def f(x):
    return x*x - x

a = 0
b = 1
ua = 1
ub = -1

data = [
    # [K(x, c), Ua, Ub]
    ["1*x**2", ua, ub],
    ["2*x**2", ua, ub],
    ["0.1*x**2", ua, ub],
    ["1/x**2", ua, ub],
    ["1*x**2", -ua, ub],
    ["1*x**2", ua, -ub],
    ["1*x**2", -ua, -ub]
]

params = [
    # [K(x, c), dK(x,c)/dx, Ua, Ub]
    [k, str(diff(k, 'x')), ua, ub] for k, ua, ub in data
]
```

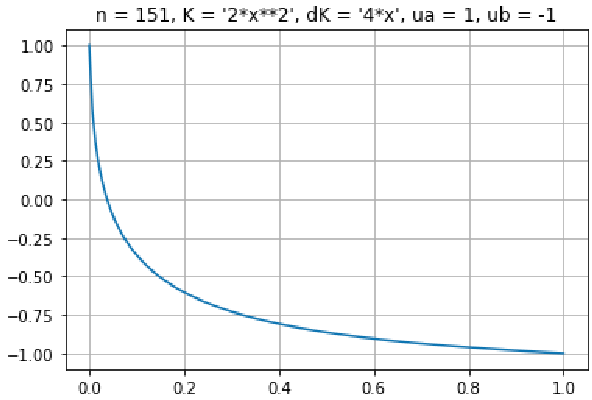
Дописываем:

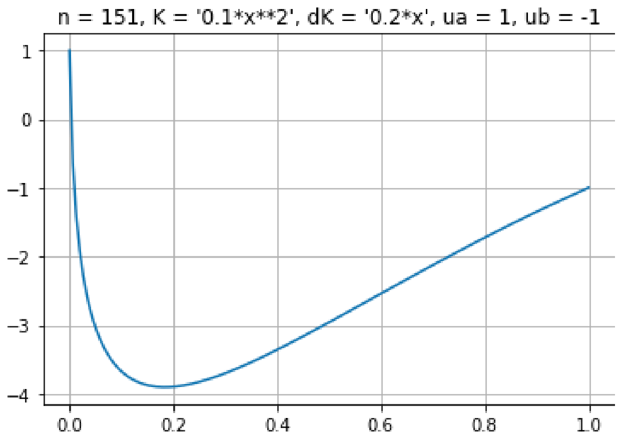
```python
def решить(n, K_str, dK_str, ua, ub):
    def K(x):
        x = x
        res = eval(K_str)
        return res

    def dK(x):
        x = x
        res = eval(dK_str)
        return res

    xs = np.linspace(a, b, n)

    h = (b - a) / (n - 1)

    diags = [
        [2*K(x) for x in xs],  # k-1
        [-h*dK(x) - 4*K(x) for x in xs],  # k
        [h*dK(x) + 2*K(x) for x in xs]   # k+1
    ]

    syst = np.zeros((n, n))
    vec = [-2*h*h*f(x) for x in xs]

    for i in range(1, n-1):
        syst[i][i-1] = diags[0][i]  # k-1
        syst[i][i] = diags[1][i]  #k
        syst[i][i+1] = diags[2][i]  # k+1

    syst[0][0] = 1
    syst[-1][-1] = 1

    vec[0] = ua
    vec[-1] = ub

    ys = np.linalg.solve(syst, vec)

    return xs, ys
```

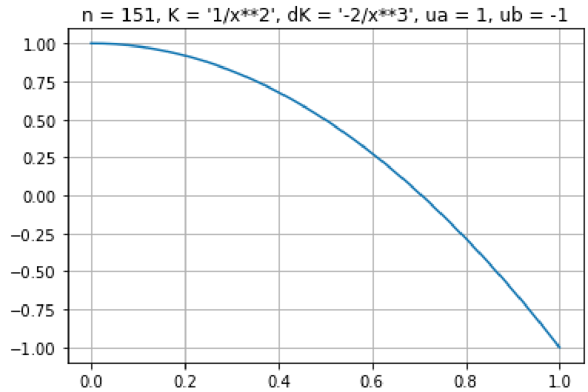
```python
n = 151
for (K, dK, ua, ub) in params:
    xs, ys = решить(n, K, dK, ua, ub)

    plt.plot(xs, ys)
    plt.title(f'{n = }, {K = }, {dK = }, {ua = }, {ub = }')
    plt.grid()
    plt.show()
```

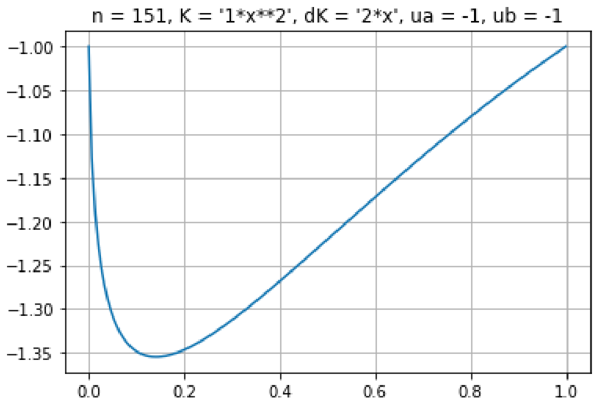
Фиксируем:

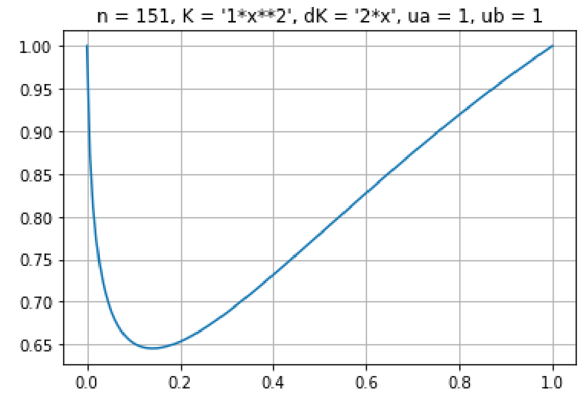

n = 151, K = '1*x**2', dK = '2*x', ua = 1, ub = -1

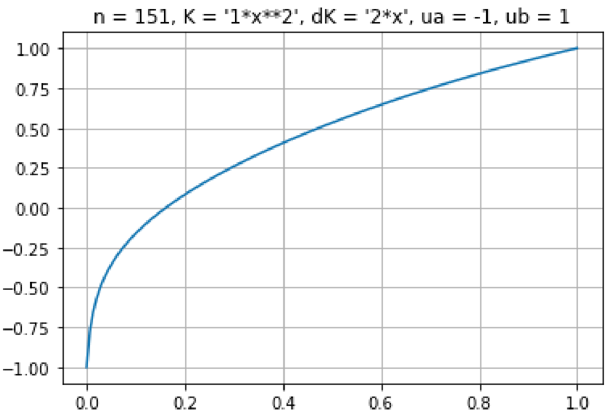

n = 151, K = '2*x**2', dK = '4*x', ua = 1, ub = -1


n = 151, K = '0.1*x**2', dK = '0.2*x', ua = 1, ub = -1


n = 151, K = '1/x**2', dK = '-2/x**3', ua = 1, ub = -1


n = 151, K = '1*x**2', dK = '2*x', ua = -1, ub = -1


n = 151, K = '1*x**2', dK = '2*x', ua = 1, ub = 1


n = 151, K = '1*x**2', dK = '2*x', ua = -1, ub = 1

Задача 2. Промоделировать стационарные процессы теплопроводности стержня в зависимости от входных данных задачи – переменного коэффициента теплопроводности *k(x)* и плотности источников тепла *f(x)*:

$$\begin{cases} -\dfrac{d}{dx}\left(k(x)\dfrac{du}{dx}\right) = f, \\ u(a) = U_A, \quad u(b) = U_B. \end{cases}$$

Записываем:

```
a = 0
b = 1
ua = 0
ub = 0

n = 151
h = (b - a) / (n-1)

xs = np.arange(a, b+h, h)

C = 10

K = 2

ks = [
    # [k1, k2, k3, c1, c2]
    [K, 2*K, 0, (b + a)/2, 2*b],  # k1 << k2
    [2*K, K, 0, (b + a)/2, 2*b],  # k1 >> k2
    [K, 2*K, 3*K, a + (b - a)/3, a + 2*(b - a)/3],  # k1 < k2 < k3
    [3*K, 2*K, K, a + (b - a)/3, a + 2*(b - a)/3],  # k1 > k2 > k3
    [K, 2*K , 3*K, a + (b - a)/3, a + 2*(b - a)/3], # k1 = k, k2 = 2*k, k3 = 3*k
    [20*K, K, 20*K, a + (b - a)/3, a + 2*(b - a)/3] # k1 = 20*k, k2 = k, k3 = 20*k
]

sources = [
    # [(x0, C)]
    [(r((b + a)/2), 10)],
    [(r((b + a)*0.2), 10), (r((b + a)*0.8), 10)],
    [(r((b + a)*0.2), 10), (r((b + a)*0.8), 20)]
]
```

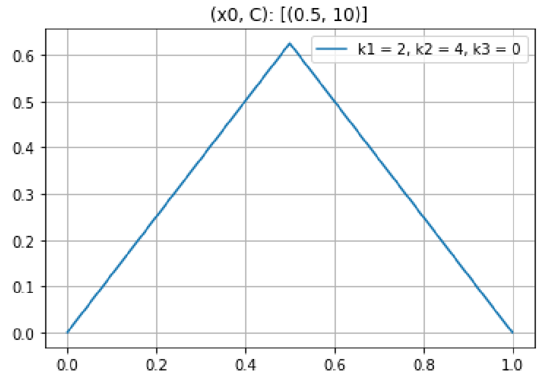
```python
def an(x, k1, k2, k3, c1, c2):
    if x < c1:
        return k1
    elif x-h < c1 < x:
        return h / ((c1 - x + h) / k1 + (x - c1) / k2)
    elif x <= c2:
        return k2
    elif x-h < c2 < x:
        return h / ((c2 - x + h) / k2 + (x - c2) / k3)
    else:
        return k3

def bn(x, k1, k2, k3, c1, c2):
    if x < c1:
        return k1
    elif x < c1 < x+h:
        return h / ((c1 - x) / k1 + (x+h - c1) / k2)
    elif x+h <= c2:
        return k2
    elif x < c2 < x+h:
        return h / ((c2 - x) / k2 + (x+h - c2) / k3)
    else:
        return k3

eps = 1e-3
def f(x, x0, C):
    if abs(x-x0) < eps:
        return C
    return 0
```

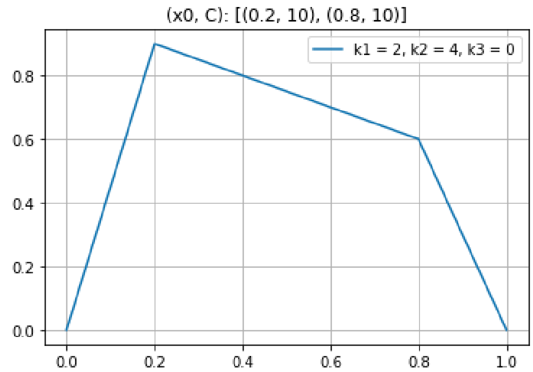
Дописываем:

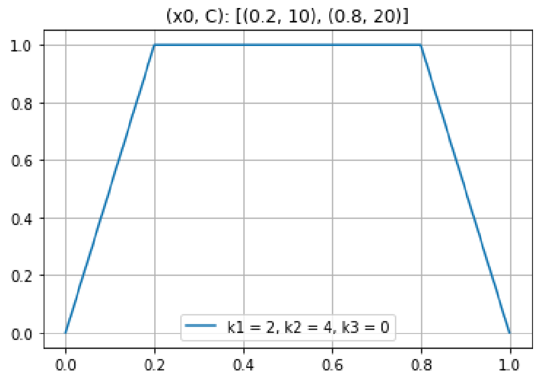
```python
def решить(k, source):
    syst = np.zeros((n, n))
    vec = [0] * n

    data = [
        [an(x, *k) for x in xs],
        [-(an(x, *k) + bn(x, *k)) for x in xs],
        [bn(x, *k) for x in xs]
    ]
    vec = [-h * sum((f(x, x0, C) for x0, C in source)) for x in xs]

    for i in range(1, n - 1):
        syst[i][i - 1] = data[0][i]
        syst[i][i] = data[1][i]
        syst[i][i + 1] = data[2][i]

    syst[0][0] = 1
    syst[-1][-1] = 1

    vec[0] = ua
    vec[-1] = ub

    ys = np.linalg.solve(syst, vec)

    return xs, ys
```

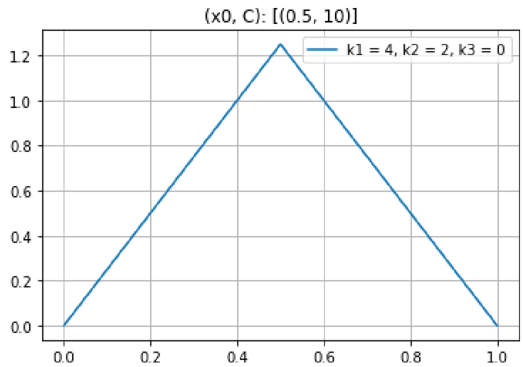
```
for k in ks:
    for source in sources:
        xs, ys = решить(k, source)
        k1 = k[0]
        k2 = k[1]
        k3 = k[2]

        plt.plot(xs, ys, label=f'{k1 = }, {k2 = }, {k3 = }')
        plt.title("(x0, C): " + str(source))
        plt.legend()
        plt.grid()
        plt.show()
```

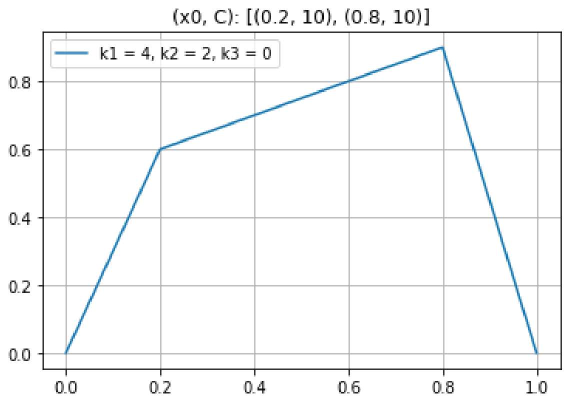
Фиксируем:

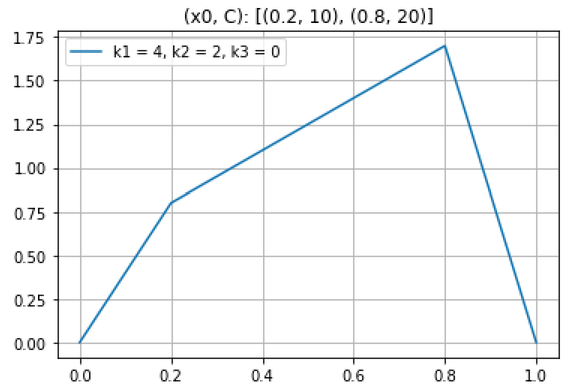


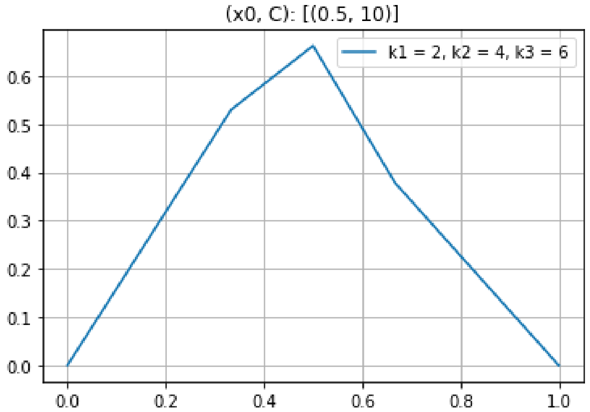


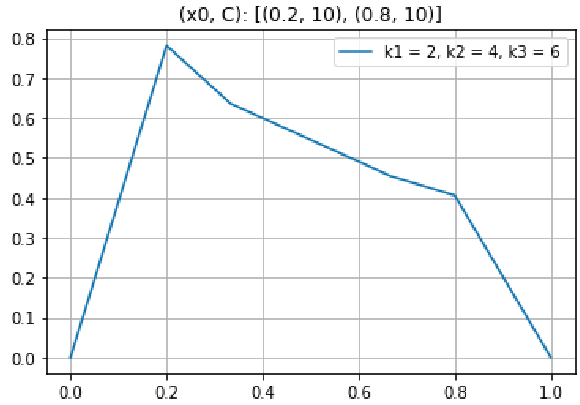


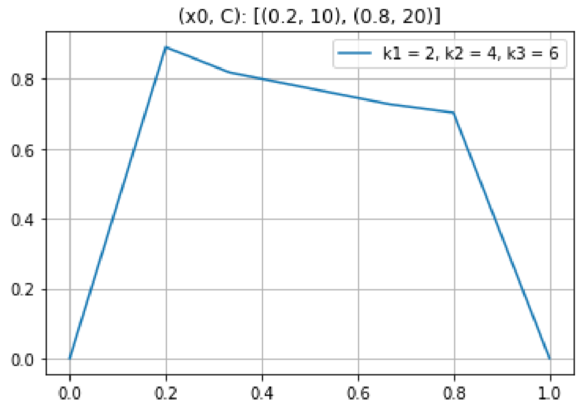


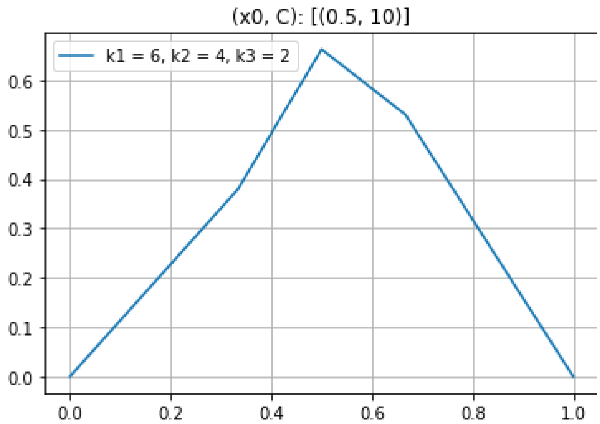


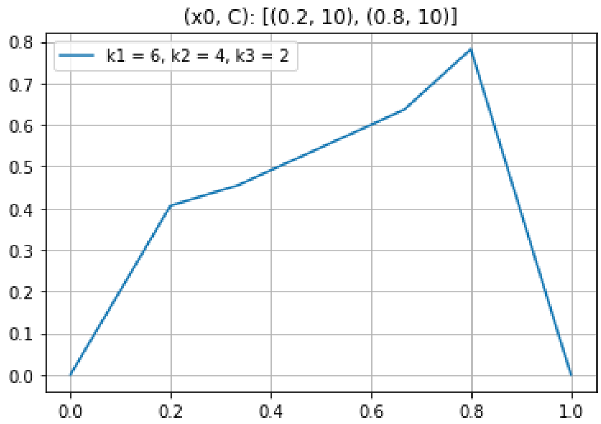


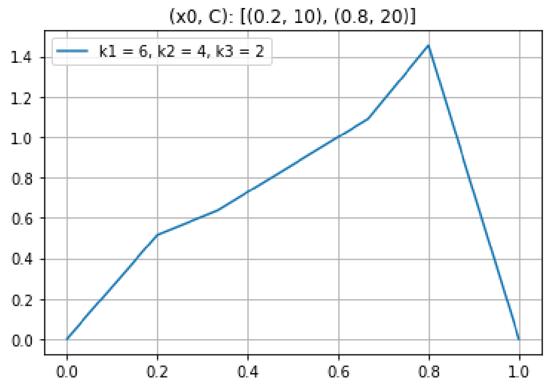

(x0, C): [(0.5, 10)]
k1 = 2, k2 = 4, k3 = 6

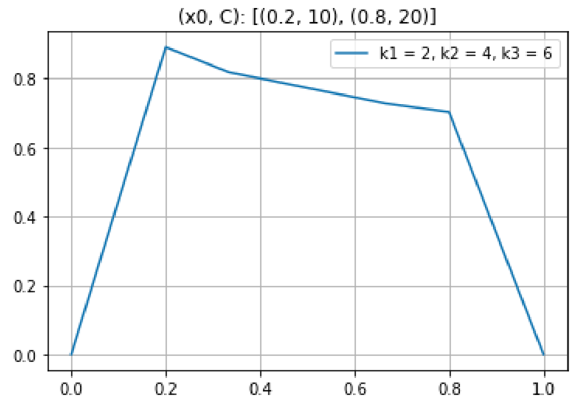

(x0, C): [(0.2, 10), (0.8, 10)]
k1 = 2, k2 = 4, k3 = 6

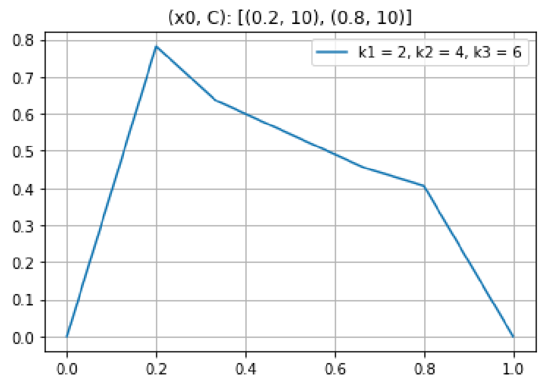

(x0, C): [(0.2, 10), (0.8, 20)]
k1 = 2, k2 = 4, k3 = 6

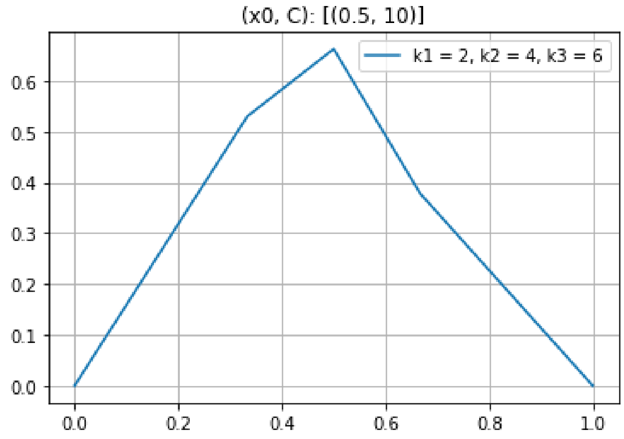

(x0, C): [(0.5, 10)]
k1 = 6, k2 = 4, k3 = 2

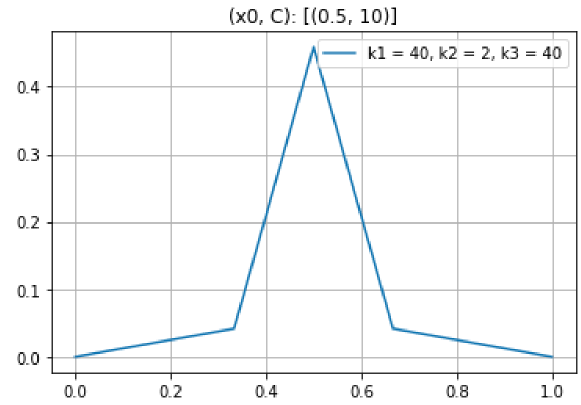

(x0, C): [(0.2, 10), (0.8, 10)]
k1 = 6, k2 = 4, k3 = 2

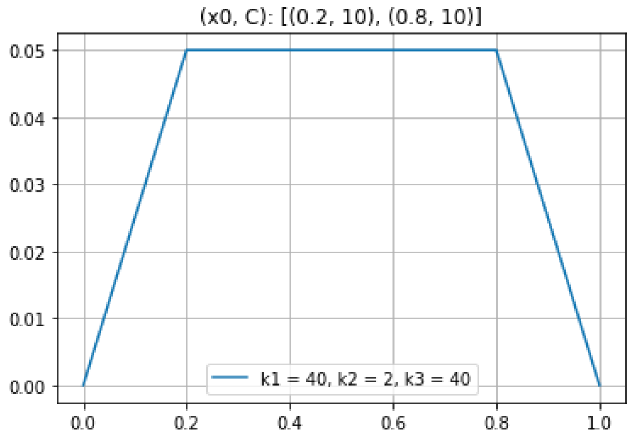

(x0, C): [(0.2, 10), (0.8, 20)]
k1 = 6, k2 = 4, k3 = 2

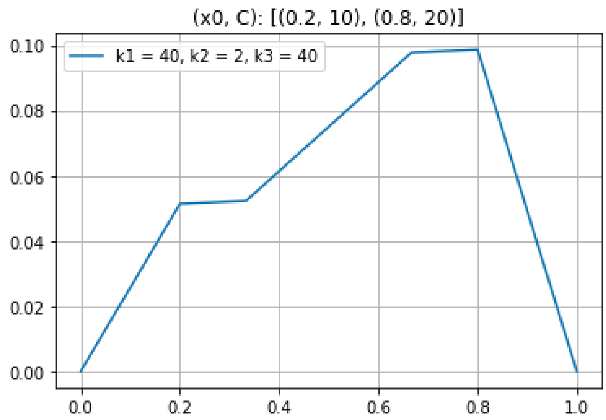

(x0, C): [(0.2, 10), (0.8, 20)]
k1 = 2, k2 = 4, k3 = 6


(x0, C): [(0.2, 10), (0.8, 10)]
k1 = 2, k2 = 4, k3 = 6









Задача 3. Промоделировать нестационарные процессы теплопроводности в зависимости от входных данных задачи – коэффициента теплопроводности и начальной температуры:

$$\begin{cases} \dfrac{\partial u}{\partial t} = \dfrac{\partial}{\partial x}\left(k(x)\dfrac{\partial u}{\partial x}\right) + f(x)(1-e^{-t}), & 0 < x < l, \quad 0 < t < T, \\ u(0,t) = U_A, \quad u(l,t) = U_B, & 0 \le t \le T, \\ u(x,0) = \phi(x), & 0 \le x \le l. \end{cases}$$

Программная реализация

Записываем:

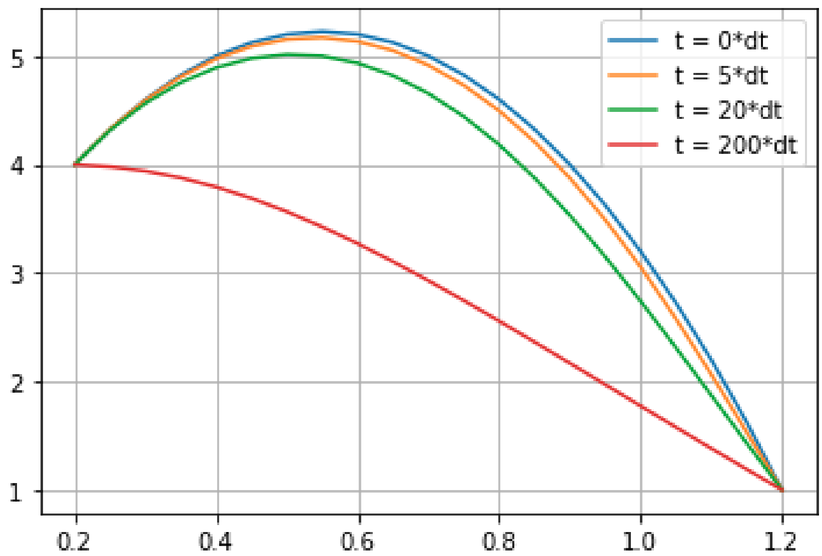
```python
a = 0.2
b = 1.2
ua = 4
ub = 1

dx = 0.05
dt = 0.001

N = round((b - a)/dx) + 1
T = 201
```

```python
def k(x):
    return x

def f(x):
    return 4*x*x*x + 6

def phi(x):
    return -10*x*x + 11*x + 2.2
```

Дописываем:

```python
def решить():
    xs = np.linspace(a, b, N)

    syst = np.zeros((T, N))
    syst[:,0] = [ua] * T
    syst[:,-1] = [ub] * T
    syst[0,:] = [phi(x) for x in np.linspace(a, b, N)]

    for i in range(1, T):
        for j in range(1, N-1):
            x = xs[j]
            syst[i][j] = k(x + dx/2)*syst[i-1][j+1]*dt/(dx*dx) + \
                    (1 - (k(x + dx/2) + k(x - dx/2))*dt/(dx*dx)) * syst[i-1][j] + \
                    k(x - dx/2)*syst[i-1][j-1]*dt/(dx*dx) + \
                    dt*f(x)*(1 - np.exp(-i*dt))
    return xs, syst
```

```python
xs, syst = решить()

for t in [0, 5, 20, 200]:
    ys = syst[t]
    plt.plot(xs, ys, label=f't = {t}*dt')
plt.legend()
plt.grid()
plt.show()
```



Задача 4. Промоделировать нестационарные процессы теплопроводности в зависимости от входных данных задачи. Найти приближенное решение начально-краевой задачи для уравнения теплопроводности:

$$\begin{cases} \dfrac{\partial u}{\partial t} = k\dfrac{\partial^2 u}{\partial x^2} + f(x,t), \;\; a < x < b, \;\; 0 < t \le T, \\ u(a,t) = g_1(t), \;\; u(b,t) = g_2(t), \;\; 0 < t \le T, \\ u(x,0) = \varphi(x), \;\; a \le x \le b. \end{cases}$$

Записываем:

```
a = -1
b = 1
ua = 1
ub = 1
k = 0.5

dx = 0.05
dt = 0.5*(h*h/k)

N = round((b - a)/dx) + 1
T = 21
```

```
def f(x):
    return 0

def phi(x):
    return abs(x)
```

Дописываем:

```
def решить():
    xs = np.linspace(a, b, N)

    syst = np.zeros((T, N))
    syst[:,0] = [ua] * T
    syst[:,-1] = [ub] * T
    syst[0,:] = [phi(x) for x in np.linspace(a, b, N)]

    for i in range(1, T):
        for j in range(1, N-1):
            x = xs[j]
            syst[i][j] = k*syst[i-1][j+1]*dt/(dx*dx) + \
                        (1 - (k + k)*dt/(dx*dx)) * syst[i-1][j] + \
                        k*syst[i-1][j-1]*dt/(dx*dx) + \
                        dt*f(x)*(1 - np.exp(-i*dt))
    return xs, syst
```

```python
xs, syst = решить()

for t in range(0, len(syst), 2):
    ys = syst[t]
    plt.plot(xs, ys, label=f't = {t}*dt')
plt.legend()
plt.grid()
plt.show()
```

Фиксируем: