Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Самарский национальный исследовательский университет
имени академика С.П. Королева»

Институт информатики и кибернетики

Кафедра технической кибернетики

Отчет по курсовой работе
Дисциплина: «Уравнения математической физики»

Тема: **«АНАЛИТИЧЕСКОЕ РЕШЕНИЕ КРАЕВЫХ ЗАДАЧ МАТЕМАТИЧЕСКОЙ ФИЗИКИ»**

Вариант № 30

Выполнили студенты: Лякин М.Я., Минеев В.А., Прокопович Д.И.

Группа 6308-010302D

Преподаватель Дегтярев А.А.

2 0 2 2

Исходные данные и задание к курсовой работе

Тема: Аналитическое решение краевых задач математической физики.

Исходные данные

Вариант работы №30, значения физических и геометрических параметров, заданные преподавателем:

- $l = 12$;
- $D = 0{,}06$;
- $u_c = 0$;
- $H = 0$;
- $T = 150$;
- $\psi(x) = \begin{cases} 0, x \notin \left[\frac{l}{4}; \frac{3l}{4}\right]; \\ 1, x \in \left[\frac{l}{4}; \frac{3l}{4}\right]. \end{cases}$

Задание к курсовой работе

1. Осуществить математическую постановку краевой задачи для физического процесса, описанного в предложенном варианте курсовой работы.

2. Используя метод разделения переменных (метод Фурье) получить решение краевой задачи в виде разложения в ряд Фурье, соответствующим краевым условиям задачи.

3. Исследовать сходимость ряда Фурье, получить оценку остатка ряда.

4. Разработать компьютерную программу расчета функции-решения краевой задачи (суммирования ряда Фурье) с требуемой точностью. При расчете коэффициентов ряда использовать метод численного интегрирования, если это необходимо. Обеспечить контроль погрешности численного интегрирования. Если необходимо, то разработать специальный программный модуль для вычисления используемых собственных чисел оператора Лапласа. Обеспечить контроль погрешности вычисления собственных чисел. Компьютерная программа должна обеспечивать возможность диалогового режима ввода физических, геометрических

параметров задачи, числа суммируемых элементов ряда, графическую визуализацию рассчитанного решения задачи.

5. Используя разработанную программу, провести экспериментальное исследование качества полученной аналитической оценки остатка ряда Фурье.

6. Оформить отчет о выполненной курсовой работе в соответствии с требованиями, изложенными в подразделе 3.2 настоящих методических указаний.

Вариант задачи 30

В тонком цилиндре конечной длины l находится диффундирующее вещество, концентрация частиц которого в момент времени $t = 0$ описывается функцией

$$u|_{t=0} = \psi(x), 0 \leq x \leq l,$$

где x - числовая ось, направленная вдоль оси цилиндра.

Коэффициент диффузии вещества является постоянным и равен D.

Один из концов цилиндра закрыт полунепроницаемой мембраной, через которую происходит диффузия вещества. Другой конец закрыт непроницаемой пробкой. Предполагается, что поток вещества через мембрану пропорционален разности концентраций вещества во внешней среде и в цилиндре, т.е.

$$\left[\frac{\partial u}{\partial x} + H(u - u_c)\right]_{x=l} = 0,$$

где u_c – концентрация вещества во внешней по отношению к цилиндру среде, H – мембранный коэффициент диффузии.

Боковая поверхность цилиндра является непроницаемой.

Для получения аналитической формы решения описанной задачи математической физики применить метод разделения переменных.

Разработать программу расчёта динамического поля концентрации вещества в цилиндре на временном промежутке $0 < t \leq T$.

Для проведения расчетов использовать представление решения задачи в виде ряда Фурье по собственным функциям оператора Лапласа, удовлетворяющим соответствующим краевым условиям [1].

РЕФЕРАТ

Отчет по курсовой работе: 24 с., 2 рисунка, 2 таблицы, 2 источника, 1 приложение.

УРАВНЕНИЯ МАТЕМАТИЧЕСКОЙ ФИЗИКИ, КРАЕВАЯ ЗАДАЧА, МЕТОД РАЗДЕЛЕНИЯ ПЕРЕМЕННЫХ, РЯД ФУРЬЕ, ОЦЕНКА ОСТАТКА РЯДА.

Целью курсовой работы является получение решения краевой задачи диффузии для цилиндра в виде разложения в ряд Фурье по собственным функциям оператора Лапласа и создание компьютерной программы для расчета функции-решения с заданной точностью.

Для получения аналитического решения краевой задачи использован метод разделения переменных (метод Фурье). Решение задачи получено в виде бесконечного ряда Фурье. Получена оценка остатка ряда.

Разработана компьютерная программа, обеспечивающая расчет и графическую визуализацию количества вещества в цилиндре на временном промежутке $0 \leq t \leq T$. Для контроля погрешности усечения бесконечного ряда использована оценка остатка ряда.

Приведены графические результаты численного решения задачи диффузии, а также результаты экспериментального исследования практической пригодности полученной оценки остатка ряда.

Программа написана на языке Python в среде разработки PyCharm Community Edition 2021.3.2, операционная система Windows 10 Home.

СОДЕРЖАНИЕ

Введение 7
1 Математическая постановка задачи 8
2 Получение решения в виде ряда Фурье (Минеев В.А.) 9
3 Оценка остатка ряда (Лякин М.Я.) 14
4 Графическая демонстрация динамики физических процессов (Прокпович Д.И.) 16
Заключение 18
Список использованной литературы 19
Приложение А Программная реализация аналитического решения 20

ВВЕДЕНИЕ

Метод разделения переменных относится к классу аналитических методов решения краевых задач математической физики. Характеризуя этот метод необходимо выделить его достоинства и недостатки в сравнении с другими методами.

К достоинствам метода разделения переменных следует отнести возможность получения точного решения краевой задачи в виде ряда Фурье. Такая форма решения задачи часто и весьма успешно используется для теоретического исследования свойств этого решения. В случае достаточно быстрой сходимости ряда Фурье она может с успехом использоваться для численного моделирования физического процесса (явления).

К числу недостатков метода следует отнести его невысокую универсальность. Этот метод весьма проблематично использовать для решения нелинейных уравнений математической физики, уравнений с переменными операторными коэффициентами, а также для решения краевых задач в областях со сложными границами.

Суть метода разделения переменных состоит в факторизации по каждой независимой переменной функции, определяющей решение уравнения математической физики. Далее осуществляется переход к так называемой задаче Штурма-Лиувилля, решение которой приводит к получению собственных функций и соответствующих им собственных чисел оператора Лапласа. Затем решение исходной задачи ищется в виде ряда Фурье по этим собственным функциям.

В настоящей работе метод разделения переменных применен для получения аналитической формы расчета процесса диффузии. На основе этого результата разработан алгоритм и компьютерная программа численного моделирования процесса диффузии в цилиндре.

1 Математическая постановка задачи

В общем виде уравнение диффузии выглядит следующим образом [2]:

$$u_t = Du_{xx}. \tag{1.1}$$

Оно будет действовать на всей длине трубки, так как в условии задачи сказано, что концентрацию следует считать одинаковой во всех точках поперечного сечения цилиндра в любой момент времени. Тогда уравнение действует при $0 < x < l, 0 < t \leq T$.

По условию левый конец трубки закрыт непроницаемой пробкой. Значит диффузия через этот конец идти не будет, получим условие:

$$u_x(0,t) = 0, 0 \leq t \leq T. \tag{1.2}$$

Так как правый конец цилиндра закрыт полунепроницаемой мембраной, диффузия через которую идет по закону, описанному в условии, то можем говорить, что изменение концентрации вещества на правом конце трубки изменяется по закону:

$$u_x(l,t) = -H(u - u_c), 0 \leq t \leq T. \tag{1.3}$$

В условии дан закон, по которому можно определить концентрацию вещества в трубке в начальный момент времени:

$$u(x,0) = \psi(x), 0 \leq x \leq l. \tag{1.4}$$

Таким образом, имеем следующую систему:

$$\begin{cases} u_t = Du_{xx}, 0 < x < l, & 0 < t \leq T; \\ u_x(0,t) = 0, & 0 \leq t \leq T; \\ u_x(l,t) = -H(u - u_c), & 0 \leq t \leq T; \\ u(x,0) = \psi(x), & 0 \leq x \leq l. \end{cases} \tag{1.5}$$

2 Построение решения в виде ряда Фурье (Минеев В.А.)

Сделаем замену переменных. Пусть $v = u - u_c$, $u = v + u_c$. Тогда система (1.5) примет вид:

$$\begin{cases} v_t = Dv_{xx}, & 0 < t \le T; \\ v_x(0,t) = 0, & 0 \le t \le T; \\ v_x(l,t) = -Hv, & 0 \le t \le T; \\ v(x,0) = \psi(x) - u_c, & 0 \le x \le l. \end{cases} \tag{2.1}$$

Будем искать решение в виде $v(x,t) = X(x)T(t)$. Подставим $v(x,t) = X(x)T(t)$ в уравнение из системы (2.1)

$$\frac{d(XT)}{dt} = \mathrm{D}\frac{d^2(XT)}{dx^2};$$

$$XT' = D\mathrm{TX}'';$$

$$\frac{T'}{DT} = \frac{X''}{X} = \lambda.$$

Тогда система (2.1) приобретает вид:

$$\begin{cases} T' - \lambda \mathrm{DT} = 0; \\ X'' - \lambda X = 0; \\ X'(0) = 0; \\ X'(l) = -HX(l),. \end{cases} \tag{2.2}$$

Пусть $\lambda < 0$ и $\lambda = -\rho^2$. Тогда

$$X'' + \rho^2 X = 0.$$

Характеристическое уравнение:

$$k^2 + \rho^2 = 0,$$

$$k = \pm i\rho.$$

Получаем общее решение уравнения:

$$X = C_1 \cos \rho x + C_2 \sin \rho x,$$

$$X' = \rho C_2 \cos \rho x - \rho C_1 \sin \rho x,$$

$$X'(0) = \rho C_2 = 0,$$

Так как $\rho \neq 0, C_2 = 0.$ Получим:

$$X'(l) = -\rho C_1 \sin \rho l = -HX(l),$$

$$-\rho C_1 \sin \rho l + HC_1 \cos \rho l = 0$$

Разделим полученное выражение на C_1:

$$-\rho \sin \rho l + H \cos \rho l = 0,$$

$$tg\rho l = \frac{H}{\rho}.$$

Пусть

$$\rho l = \ \mu \text{ и } Hl = p.$$

Тогда

$$tg\mu = \frac{p}{\mu} \qquad (2.3)$$

Будем считать, что $\mu_0, \mu_1, \dots, \mu_k$ – неотрицательные корни уравнения (2.3)

Получаем собственные функции:

$$X_k(x) = C_k \cos\frac{\mu_k x}{l}.$$

$$T' + \rho^2 \mathrm{DT} = 0,$$

$$T_k{}'(t) + \frac{\mathrm{D}\mu_k{}^2}{l^2} T_k(t) = 0,$$

$$T_k{}'(t) = -\frac{\mathrm{D}\mu_k{}^2}{l^2} T_k(t),$$

$$\frac{dT_k(t)}{dt} = -\frac{\mathrm{D}\mu_k{}^2}{l^2} T_k(t),$$

$$\int \frac{dT_k(t)}{T_k} = -\frac{\mathrm{D}\mu_k{}^2}{l^2} \int dt,$$

$$ln|T_k(t)| = -\frac{\mathrm{D}\mu_k{}^2}{l^2} t + C_k,$$

$$T_k(t) = e^{-\frac{\mathrm{D}\mu_k{}^2}{l^2} t} e^{C_k}.$$

Пусть

$$e^{C_k} = \mathrm{A_k}.$$

Тогда

$$T_k(t) = \mathrm{A_k} e^{-\frac{\mathrm{D}\mu_k{}^2}{l^2} t}, \text{при } k = 0,1, \dots$$

Построим ряд

$$v(x,t) = \sum_{k=0}^{\infty} a_k e^{-\frac{\mathrm{D}\mu_k{}^2}{l^2} t} \cos\frac{\mu_k x}{l}, \qquad (2.4)$$

$$a_k = C_k A_k.$$

При подстановке (2.4) в начальное условие, получаем

$$v(x,0) = \sum_{k=0}^{\infty} a_k \cos\frac{\mu_k x}{l} = \psi(x) + u_c. \qquad (2.5)$$

Умножим левую и правую часть (2.5) на $cos\frac{\mu_n x}{l}$ и проинтегрируем в пределах от 0 до l:

$$\int_0^l (\psi(x) + u_c) cos\frac{\mu_n x}{l} dx = \sum_{k=0}^{\infty} a_k \int_0^l cos\frac{\mu_n x}{l} cos\frac{\mu_k x}{l} dx.$$

Так как собственные функции ортогональны:

$$\int_0^l X_k(x) X_n(x) dx = 0\,, k \neq n.$$

Значит, сохраняются только слагаемые, где $k=n$:

$$\int_0^l (\psi(x) + u_c) cos\frac{\mu_k x}{l} dx = a_k \int_0^l cos^2\frac{\mu_k x}{l} dx\,. \qquad (2.6)$$

Вычислим интеграл правой части (2.6).

$$\int_0^l cos^2\frac{\mu_k x}{l} dx = \frac{1}{2}\int_0^l 1 + cos\frac{2\mu_k x}{l} dx = \frac{1}{2} l + \frac{1}{2}\frac{l}{2\mu_k}\int_0^{2\mu_k} sin(y)\, dy =$$

$$= \frac{1}{2} l - \frac{1}{2}\frac{l}{2\mu_k}(\cos(2\mu_k) - \cos(0)) = \frac{l}{2},$$

$$a_k = \frac{\int_0^l (\psi(x) + u_c) cos\frac{\mu_k x}{l} dx}{\int_0^l cos^2\frac{\mu_k x}{l} dx} = \frac{2}{l}\int_0^l (\psi(x) + u_c) cos\frac{\mu_k x}{l} dx, k = 0,1,2,\ldots$$

Вычислим интеграл левой части (2.6):

$$\int_0^l (\psi(x) + u_c) cos\frac{\mu_k x}{l} dx = \int_0^{\frac{l}{4}} (0+0) cos\frac{\mu_k x}{l} dx + \int_{\frac{l}{4}}^{\frac{3l}{4}} (1+0) cos\frac{\mu_k x}{l} dx +$$

$$+ \int_{\frac{3l}{4}}^{l} (0+0) cos\frac{\mu_k x}{l} dx = \int_{\frac{l}{4}}^{\frac{3l}{4}} cos\frac{\mu_k x}{l} dx = \frac{l}{\mu_k}\left(sin\frac{3\mu_k}{4} - sin\frac{\mu_k}{4}\right) =$$

$$= \frac{2l}{\mu_k} \sin\left(\frac{\mu_k}{4}\right) \cos\left(\frac{\mu_k}{2}\right).$$

Подставим посчитанный интеграл в ряд

$$v(x,t) = \frac{2}{l}\sum_{k=0}^{\infty} e^{\frac{-\mathrm{D}\mu_k{}^2}{l^2}t}\cos\left(\frac{\mu_k x}{l}\right)\frac{2l}{\mu_k}\sin\left(\frac{\mu_k}{4}\right)\cos\left(\frac{\mu_k}{2}\right),$$

$$u(x,t) = v(x,t) + u_c = 4\sum_{k=0}^{\infty}\frac{1}{\mu_k}e^{\frac{-\mathrm{D}\mu_k{}^2}{l^2}t}\cos\left(\frac{\mu_k x}{l}\right)\sin\left(\frac{\mu_k}{4}\right)\cos\left(\frac{\mu_k}{2}\right).$$

Получаем ряд:

$$u(x,t) = 4\sum_{k=0}^{\infty}\frac{1}{\mu_k}e^{\frac{-\mathrm{D}\mu_k{}^2}{l^2}t}\cos\left(\frac{\mu_k x}{l}\right)\sin\left(\frac{\mu_k}{4}\right)\cos\left(\frac{\mu_k}{2}\right).$$

При $k = 0$ получаем неопределенность вида $\frac{0}{0}$. Обозначим первый элемент A_0 и вычислим его. При $t = 0$ справедливо:

$$\psi(x) = A_0 + \ 4\sum_{k=1}^{\infty}\frac{1}{\mu_k}\cos\left(\frac{\mu_k x}{l}\right)\sin\left(\frac{\mu_k}{4}\right)\cos\left(\frac{\mu_k}{2}\right),$$

$$\int_0^L \psi(x)dx = \int_0^L A_0 dx + \int_0^L 4\sum_{k=1}^{\infty}\frac{1}{\mu_k}\cos\left(\frac{\mu_k x}{l}\right)\sin\left(\frac{\mu_k}{4}\right)\cos\left(\frac{\mu_k}{2}\right)dx,$$

$$\int_0^L \psi(x)dx = \int_0^L A_0 dx + 4\sum_{k=1}^{\infty}\frac{1}{\mu_k}\sin\left(\frac{\mu_k}{4}\right)\cos\left(\frac{\mu_k}{2}\right)\int_0^L \cos\left(\frac{\mu_k x}{l}\right)dx,$$

Для решения интеграла под суммой введем замену переменных $z = \frac{\mu_k x}{l}$.

$$\int_0^L \psi(x)dx = \int_0^L A_0 dx + 4\sum_{k=1}^{\infty}\frac{1}{\mu_k}\sin\left(\frac{\mu_k}{4}\right)\cos\left(\frac{\mu_k}{2}\right)\frac{L}{\mu_k}\sin(\mu_k).$$

В силу того, что $\mu_k = \pi k$, k = 1, 2, … сумма будет равна 0.

$$\int_0^L \psi(x)dx = \int_0^L A_0 dx\,,$$

$$\int_{\frac{L}{4}}^{\frac{3L}{4}} 1dx = A_0\int_0^L 1dx\,,$$

$$\frac{3L}{4} - \frac{L}{4} = A_0 L,$$

$$A_0 = \frac{1}{2}.$$

Таким образом, итоговая формула будет иметь вид

$$u(x,t) = \frac{1}{2} + 4\sum_{k=1}^{N} \frac{1}{\mu_k} e^{\frac{-\mathrm{D}{\mu_k}^2}{l^2}t} \cos\left(\frac{\mu_k x}{l}\right) \sin\left(\frac{\mu_k}{4}\right) \cos\left(\frac{\mu_k}{2}\right).$$

3 Оценка остатка ряда (Лякин М.Я.)

Для нашего ряда Фурье запишем частичную сумму:

$$u(x,t) = 4\sum_{k=0}^{N} \frac{1}{\mu_k} e^{\frac{-\mathrm{D}\mu_k{}^2}{l^2}t} \cos\left(\frac{\mu_k x}{l}\right) \sin\left(\frac{\mu_k}{4}\right) \cos\left(\frac{\mu_k}{2}\right).$$

Пусть существует ε – погрешность ($\varepsilon > 0$).

Запишем оценки:

$$\cos\left(\frac{\mu_k x}{l}\right) \sin\left(\frac{\mu_k}{4}\right) \cos\left(\frac{\mu_k}{2}\right) \le 1,$$

Запишем остаток ряда с учетом оценок:

$$|R_N| \le \sum_{n=N+1}^{\infty} \left|\frac{4}{\mu_n} e^{\frac{-D\mu_n^2 t}{l^2}}\right| = \sum_{n=N+1}^{\infty} \left|\frac{4}{n\pi} e^{\frac{-D\mu_n^2 t}{l^2}}\right| = \left[m = \frac{tD\pi^2}{l^2}\right] =$$

$$= \sum_{n=N+1}^{\infty} \left|\frac{4}{n\pi} e^{-mn^2}\right|.$$

Оценим остаток ряда:

$$|R_N| \le \sum_{n=N+1}^{\infty} \left|\frac{4}{n\pi} e^{-mn^2}\right| \le \frac{4}{\pi}\int_N^{\infty} y \frac{dy}{y^2} e^{-my^2} < \frac{4}{\pi N^2}\int_N^{\infty} y \frac{dy}{1} e^{-my^2} =$$

$$= \frac{4}{\pi N^2}\int_N^{\infty} y\, e^{-my^2} dy = \left| \begin{matrix} u = -my^2,\ du = -2my \\ dy = -\dfrac{1}{2my} du \end{matrix} \right| = \frac{4}{\pi N^2}\left(-\frac{e^{-my^2}}{2m}\Bigg|_N^{\infty}\right) =$$

$$= \frac{4}{\pi N^2}\frac{e^{-mN^2}}{2m}.$$

Получим функцию мажоранта и обозначим ее $\Phi(N)$:

$$\Phi(N) = \frac{4}{\pi N^2}\frac{e^{-mN^2}}{2m} = \frac{4}{\pi N^2}\frac{e^{\frac{-D\pi^2 t N^2}{l^2}} l^2}{2D\pi^2 t} = \frac{2e^{\frac{-D\pi^2 t N^2}{l^2}} l^2}{N^2 D\pi^3 t},$$

$$\lim_{n\to\infty} \Phi(N) = 0\ ,\ \Phi(N) \le \varepsilon.$$

С помощью полученной функции можем сравнить количество слагаемых N_{min}, необходимое для достижения заданной точности ε, которое получено с помощью оценки остатка ряда, и действительное их количество, полученное экспериментально N_{exp}, для заданного t.

Для получения N_{exp} будем уменьшать N_{min}, найденное в предыдущей части, пока выполняется неравенство:

$$u_{N_{min}}(x,t) - u_{N_{min}-1}(x,t) < \varepsilon. \quad (4.1)$$

Результаты представлены в таблице 1.

Таблица 1 – Нахождение N_{min} и N_{exp} для заданного времени $t = 1$

ε	0.1	0.01	0.001	0.0001	10^{-5}	10^{-6}	10^{-7}
N_{min}	19	28	35	41	47	52	57
N_{exp}	6	18	30	34	42	46	54

Из данной таблицы можем сделать вывод о высоком качестве оценки остатка ряда, причем при увеличении точности, отрыв между N_{min} и N_{exp} стремительно уменьшается.

4 Графическая демонстрация динамики физических процессов (Прокопович Д.И.)

Используя оценку остатка ряда при значении погрешности, равной 0.01 получим количество слагаемых для соответствующего времени. Результат представлен в таблице 2.

Таблица 2 – Нахождение N_{min} для заданной погрешности $\varepsilon = 0.01$

t	0.0001	50.0	100.0	150.0
N_{min}	2720	4	3	3

На рисунке 1 представлен график зависимости количества вещества $u(x,t)$ от координаты цилиндра x при фиксированном времени t.

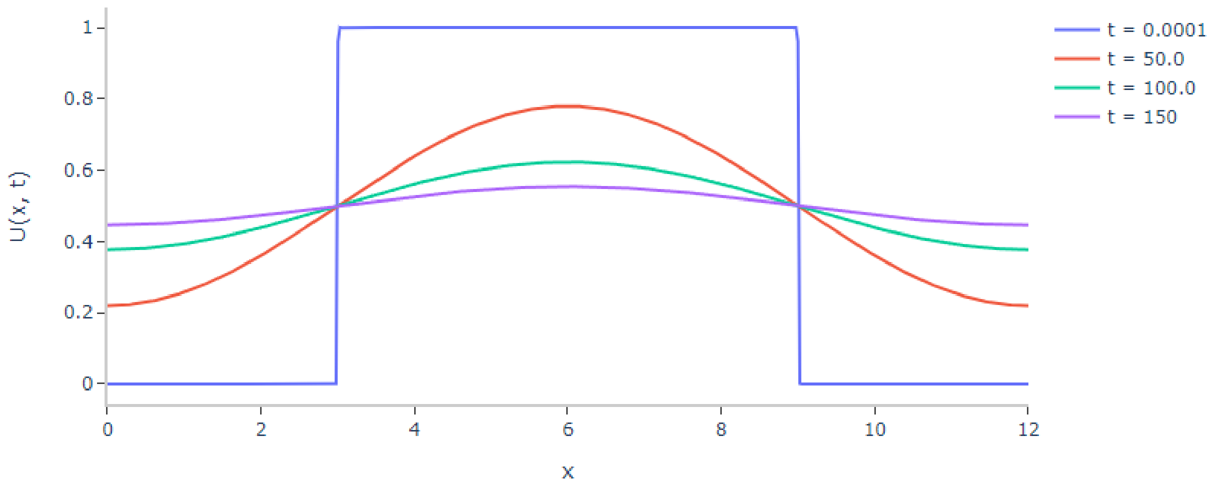

Рисунок 1 – График изменения количества вещества от координаты при $t = 0.0001, t = 50, t = 100, t = 150.$

Действительно, видно, что в начальный момент времени концентрация вещества в цилиндре соответствует функции $\psi(x)$, как и ожидалось. Затем, вещество плавно начинает распределяться по цилиндру и за его пределы, выходя через мембрану, поэтому с ростом времени график стремится принять форму прямой, что говорило бы о том, что на всем протяжении цилиндра установилась одинаковая концентрация вещества.

Получим аналогичный график, но на оси абсцисс будут значения от 0 до Т. Результат виден на рисунке 2.

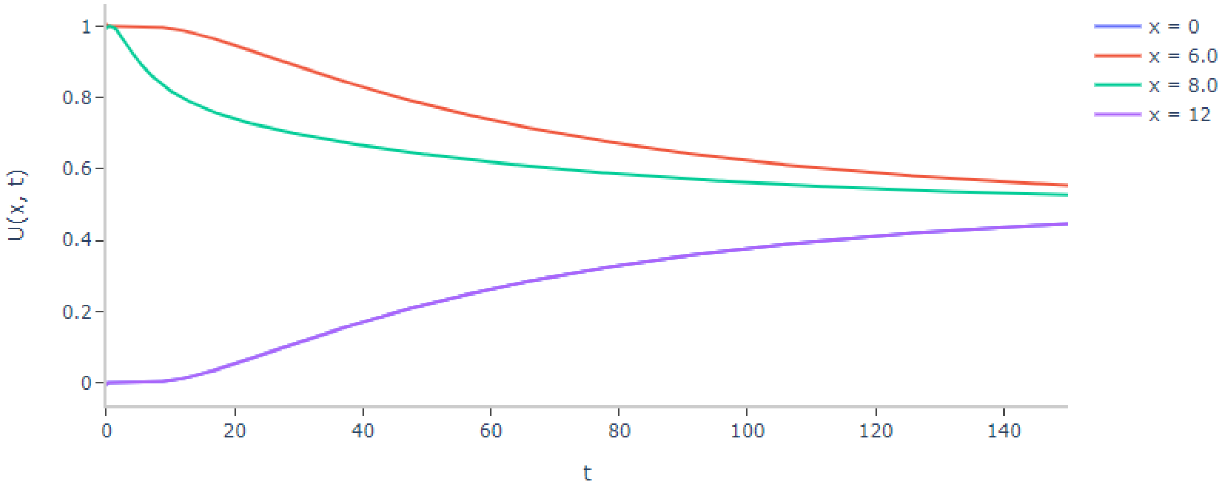

Рисунок 2 – График изменения количества вещества от времени при $x = 0, x = 6, x = 8, x = 12$.

Здесь мы не наблюдаем график при *x* = *0*. Это верно, потому что концентрация вещества в крайних точках совпадает, и графики накладываются друг на друга. При *x* = *6*, в центре цилиндра, видим, что концентрация была максимальная, а затем с течением времени начала уменьшаться. То же самое происходит и при *x* = *8*, но с большей скоростью, так как эта точка находится ближе к краю цилиндра.

Исходя из вышесказанного можем сделать вывод, что поставленная задача решена верно.

ЗАКЛЮЧЕНИЕ

В результате выполнения работы осуществлена постановка краевой задачи для уравнения диффузии, получено аналитическое решение задачи в виде бесконечного ряда Фурье.

Получена оценка сверху остатка ряда, которая была использована в компьютерной программе численного моделирования волнового процесса и позволила обеспечить контроль погрешности усечения бесконечного ряда.

В результате серии вычислительных экспериментов установлено, что использование в программе полученной оценки остатка ряда может приводить к несущественной избыточности числа суммируемых элементов ряда, оценка ряда высокого качества.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Дегтярев, А.А. Аналитическое решение краевых задач математической физики [Текст]: методические указания / А.А. Дегтярев. – Самара: Изд-во Самар. ун-та, 2020. – 60 с.

2. Араманович, И.Г. Уравнения математической физики [Текст]: учеб. пособие / И.Г. Араманович, В.И. Левин. - М.: Наука, 1969. – 288 с.

ПРИЛОЖЕНИЕ А

Программная реализация аналитического решения

```
from math import pi, sin, cos, e

import matplotlib.pyplot as plt
import numpy
import plotly.graph_objects as go

H = 0
U_c = 0
MINIMAL_T = 0.0001  # approximate minimal T, as it can't be 0, and it should be
quite big for fast calculations
EPS_ARRAY = [0.1, 0.01, 0.001, 0.0001, 10 ** -5, 10 ** -6, 10 ** -7]
T_CHECK = 1


def build_plot(x: [float], y_array: [[float]], sections: [float], x_label: [str], y_label:
[str],
               sections_label: [str]) \
        -> plt.Figure:
    """
    Builds plot for given parameters
    :param y_array: y
    :param sections_label:
    :param y_label:
    :param x_label:
    :param x: x
    :param sections: sections
    """
    fig = go.Figure()
    fig.update_layout(
        xaxis=dict(
            showgrid=False,
            showticklabels=True,
            linecolor='rgb(204, 204, 204)',
            linewidth=2,
            ticks='outside'
        ),
        yaxis=dict(
            showgrid=False,
            zeroline=False,
            showticklabels=True,
            linecolor='rgb(204, 204, 204)',
```

```
            linewidth=2,
            ticks='outside'
        ),
        autosize=True,
        margin=dict(
            autoexpand=True,
            l=100,
            r=100,
            t=110,
        ),
        showlegend=True,
        plot_bgcolor='white'
    )
    fig.update_layout(xaxis_title=x_label, yaxis_title=y_label)
    for y, section in zip(y_array, sections):
        fig.add_trace(go.Line(x=x, y=y, mode='lines', name=sections_label +
str(section)))

    return fig


def print_matrix(matrix):
    """
    Some terrifying method from
https://stackoverflow.com/questions/13214809/pretty-print-2d-
list#:~:text=90,a%20bigger%20matrix%3A
    :param matrix: matrix to print
    :return: None
    """
    s = [[str(e) for e in row] for row in matrix]
    lens = [max(map(len, col)) for col in zip(*s)]
    fmt = ' | '.join('{{:{}}}'.format(x) for x in lens)
    table = [fmt.format(*row) for row in s]
    print('\n'.join(table))


class Model:

    def __init__(self, D, L, T, EPS):
        self.__D = D
        self.__L = L
        self.__T = T
        self.__EPS = EPS
```

```
    def set_params(self, D, L, T, EPS):
        self.__D = D
        self.__L = L
        self.__T = T
        self.__EPS = EPS

    def u(self, x: float, t: float, n: int) -> float:
        """
        Calculates Fourier sum, also known as v(x, t).
        :param n:
        :param x: x
        :param t: t
        :return:
        """
        _sum = 0
        for k in range(1, n + 1):
            _sum += \
                1 / (pi * k) * \
                e ** (-self.__D * (pi * k) ** 2 * t / self.__L ** 2) * \
                cos(pi * k * x / self.__L) * \
                sin(pi * k / 4) * \
                cos(pi * k / 2)
        return 4 * _sum + 1/2

    def f(self, n: int, t: float) -> float:
        """
        Calculates F(n).
        :param n: n
        :param t: t
        :return: F(n)
        """
        return (2 * self.__L ** 2 * e ** (-self.__D * pi ** 2 * t * n ** 2 / self.__L **
2)) / (
                self.__D * pi ** 3 * n ** 2 * t)

    def estimate_n_min_for_single_t(self, epsilon: float, t: float) -> int:
        """
        Estimates number of elements in fourier sum for single t
        :param epsilon: precision
        :param t: necessary time
        :return: numer of elements in fourier sum
        """
        i = 1
        while self.f(i, t) > epsilon:
```

```
            i += 1
        return i

    def estimate_n_min(self, epsilon: float, t_array: [float]) -> [int]:
        """
        Estimates number of elements in fourier sum for each t in t_array.
        Also known as N(eps)
        :param epsilon: precision
        :param t_array: array of necessary times
        :return: array of numer of elements in fourier sum accordingly for each t
        """
        n_array = []
        for single_t in t_array:
            n_array.append(self.estimate_n_min_for_single_t(epsilon, single_t))

        return n_array

    def estimate_experimental_n(self, epsilon: float, x_array: [float], t: float, n: int) -
> int:
        """
        Experimentally estimates number of elements of Fourier's sum needed to
satisfy given precision.
        :param n: estimated with N(eps) n
        :param t: t
        :param epsilon: precision
        :param x_array: x
        :return: array of numer of elements in fourier sum accordingly for each t
        """
        i = n
        while all([abs(self.u(x, t, i) - self.u(x, t, i-1)) < epsilon for x in x_array]):
            i -= 1

        return i

    def calculate(self):
        # estimate number of elements in fourier's sum
        t_values = [MINIMAL_T, self.__T / 3, 2 * self.__T / 3, self.__T]
        x_values = [0, self.__L / 2, 2 * self.__L / 3, self.__L]
        n_array = self.estimate_n_min(self.__EPS, t_values)

        print("With given precision: " + str(self.__EPS))
        print_matrix([["T: "] + t_values, ["N: "] + n_array])

        x = numpy.linspace(0, self.__L, 500)
```

```
    t = numpy.linspace(0, self.__T, 500)

    # check difference between n found by estimate_n_min and experimental one
    print("For t = " + str(T_CHECK))

    eps_array = ["EPS :"] + EPS_ARRAY
    n_min = ["N_min: "]
    n_exp = ["N_exp: "]
    for epsilon in EPS_ARRAY:
        current_n_min = self.estimate_n_min_for_single_t(epsilon, T_CHECK)
        n_min.append(str(current_n_min))
        n_exp.append(
            str(self.estimate_experimental_n(epsilon, x, T_CHECK,
current_n_min)))

    print_matrix([eps_array, n_min, n_exp])

    # for each t, x number of elements is reduced to satisfy given precision
    y_array = []
    for n1, _t1 in zip(n_array, t_values):
        y_array.append([self.u(_x1, _t1, n1) for _x1 in x])

    plot1 = build_plot(x, y_array, t_values, x_label="x", y_label="U(x, t)",
sections_label="t = ")

    t_array = []
    for _x2 in x_values:
        t_array.append([self.u(_x2, _t2, 100) for _t2 in t])

    plot2 = build_plot(t, t_array, x_values, x_label="t", y_label="U(x, t)",
sections_label="x = ")

    return plot1, plot2
```