|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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 рода  **Студент** Белоусова Ю.С.  **Группа** ИУ7-61Б  **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 

 (1)

Краевые условия



2. Функции заданы своими константами



Константы  следует найти из условий , а константы  из условий . Величины  задает пользователь, их надо вынести в интерфейс.

3. Разностная схема с разностным краевым условием при . Получено в Лекции №7 (7.14), (7.15), и может быть использовано в данной работе. Самостоятельно надо получить интегро -интерполяционным методом разностный аналог краевого условия при , точно так же, как это было сделано применительно к краевому условию при  в Лекции №7 (формула (7.15)). Для этого надо проинтегрировать на отрезке [xN-1/2, xN] выписанное выше уравнение (1) с учетом (7.9) из Лекции №7 и учесть, что поток , а .

4. Значения параметров для отладки (все размерности согласованы)

0.4 Вт/см К,

0.1 Вт/см К,

0.05 Вт/см2 К,

0.01 Вт/см2 К,

 10 см,

300К,

0.5 см,

50 Вт/см2.

**Физическое содержание задачи** (для понимания получаемых результатов при отладке программы)

Сформулированная математическая модель описывает температурное поле  вдоль цилиндрического стержня радиуса  и длиной , причем  и температуру можно принять постоянной по радиусу цилиндра. Ось  направлена вдоль оси цилиндра и начало координат совпадает с левым торцом стержня. Слева при  цилиндр нагружается тепловым потоком . Стержень обдувается воздухом, температура которого равна . В результате происходит съем тепла с цилиндрической поверхности и поверхности правого торца при . Функции являются, соответственно, коэффициентами теплопроводности материала стержня и теплоотдачи при обдуве.

**Результаты работы**

1. Представить разностный аналог краевого условия при  и его краткий вывод интегро -интерполяционным методом.

Учитывая, что , проинтегрируем уравнение (1) на отрезке [xN-1/2, xN]:

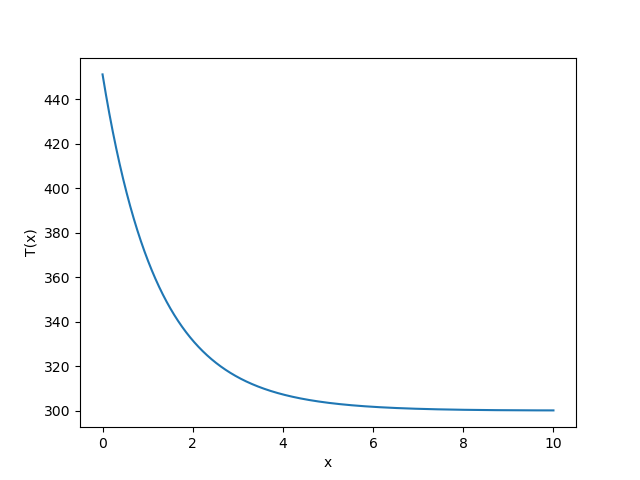
Получаем

Учитывая, что , , ,

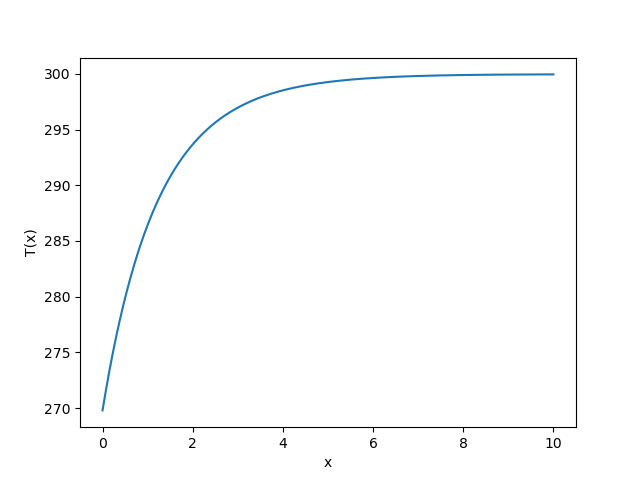
получим

Раскрыв скобки и приведя подобные слагаемые получаем разностный аналог:

2. График зависимости температуры  от координаты  при заданных выше параметрах.

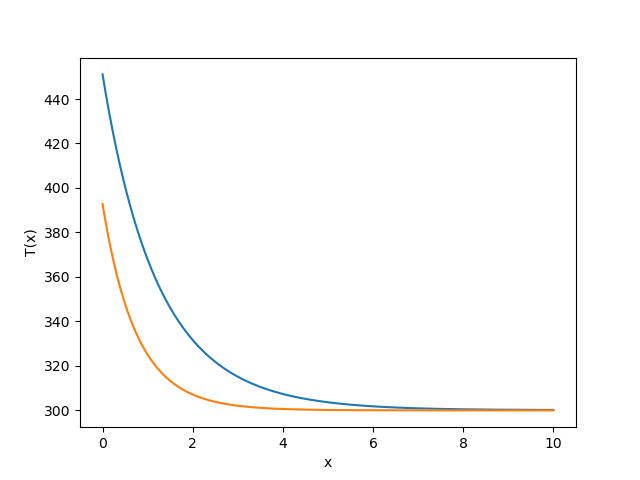


3. График зависимости  при -10 Вт/см2.



В соответствии со справкой – слева идет съем тепла, производная  положительна.

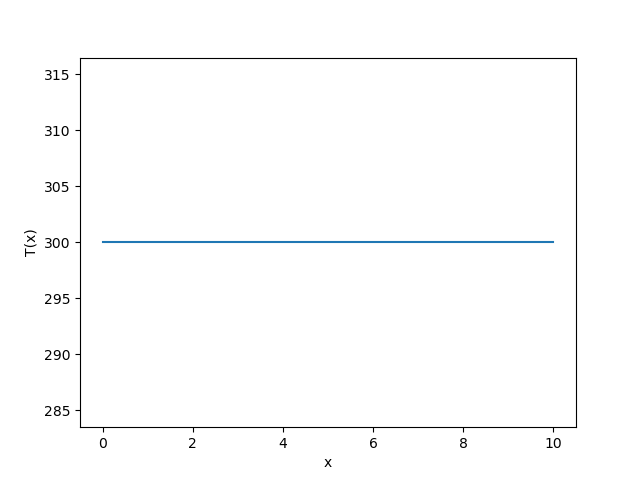
4. График зависимости  при увеличенных значениях ( в 3 раза) - оранжевый. Сравнить с п.2. – синий.



В соответствии со справкой - уровень температур  снижается, а градиент увеличивается.

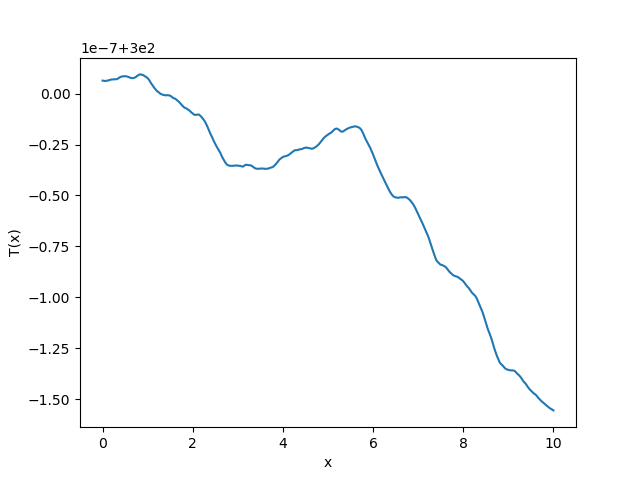
5. График зависимости  при 0.

В соответствии со справкой – нагрева нет, температура стержня постоянна и равна температуре окружающей среды. Тогда идеальный график выглядит так:



Но из-за приближенного характера вычислений появляется погрешность, график получается такой:

|  |  |
| --- | --- |
|  |  |



**Текст программы**

*Листинг 1. Файл с расчетами.*

l = 10  
T0 = 300  
R = 0.5  
F0 = 50  
def get\_consts():  
 k0 = 0.4  
 kN = 0.1  
 alpha0 = 0.05  
 alphaN = 0.01

'''  
 k0 = float(input("k0 = "))  
 kN = float(input("kN = "))  
 alpha0 = float(input("alpha0 = "))  
 alphaN = float(input("alphaN = "))  
 '''  
 a = (-k0\*kN\*l)/(kN-k0)  
 b = (kN\*l)/(kN-k0)  
 c = (-alpha0\*alphaN\*l)/(alphaN-alpha0)  
 d = (alphaN\*l)/(alphaN-alpha0)  
 return a, b, c, d  
  
def k(x):  
 return a/(x-b)  
def alpha(x):  
 return c/(x-d)  
  
def hi(i,h):  
 k1 = k((i-0.5)\*h)  
 k2 = k((i+0.5)\*h)  
 return (2\*k1\*k2)/(k1+k2)  
  
def progonka(h, N):  
 znam = hi(0.5,h) + (h\*\*2 \* (alpha(0)+alpha(h))) / (8\*R) + (h\*\*2 \* alpha(0)) / (4\*R)  
 ksi1 = (hi(0.5,h) - (h\*\*2 \* (alpha(0)+alpha(h))) / (8\*R)) / znam  
 etta1 = (h\*F0 + (h\*\*2\*T0\*(3\*alpha(0)+alpha(h)) / (4\*R))) / znam  
 ksi = [ksi1]  
 etta = [etta1]  
 for n in range(1,N):  
 A = hi(n+0.5,h)/h  
 C = hi(n-0.5,h)/h  
 B = A+C + 2\*alpha(n\*h)\*h/R  
 D = 2\*T0\*alpha(n\*h)\*h/R  
 znam = B-A\*ksi[n-1]  
 ksi.append(C/(znam))  
 etta.append((D+A\*etta[n-1])/znam)  
 return ksi, etta  
  
def go\_back(ksi, etta, N, h):  
 yarr = [0]\*(N+1)  
 xarr = [0]\*(N+1)  
 xarr[N] = l  
 znam = (h\*\*2 \* (alpha(xarr[N]) + alpha(xarr[N]-h)) / (8\*R)) - hi(N-0.5,h)  
 m1 = (h\*alpha(xarr[N])+hi(N-0.5,h)+h\*\*2\*alpha(xarr[N])/2/R + h\*\*2\*(alpha(xarr[N])+alpha(xarr[N]-h))/(8\*R)) / znam  
 m2 = (h\*alpha(xarr[N])\*T0 + T0\*h\*\*2\*(3\*alpha(xarr[N])+alpha(xarr[N]-h))/4/R) / znam  
 yarr[N] = (m2 - etta[N-1])/(ksi[N-1] + m1)  
 for n in range(N,0,-1):  
 xarr[n-1] = xarr[n]-h  
 yarr[n-1] = yarr[n]\*ksi[n-1] + etta[n-1]  
 return xarr, yarr  
  
def work(h):  
 global a, b, c, d  
 a, b, c, d = get\_consts()  
 N = int(l/h + 1)  
 ksi, etta = progonka(h, N)  
 xarr, yarr = go\_back(ksi, etta, N, h)  
 return xarr, yarr

*Листинг 2. Main, вывод графика*

from calc import work  
import matplotlib.pyplot as plt  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 h = 1e-4  
 xarr, yarr = work(h)  
 plt.plot(xarr, yarr)  
 plt.xlabel("x")  
 plt.ylabel("T(x)")  
 plt.show()

**Ответы на вопросы**

1. Какие способы тестирования программы можно предложить?

Тестирование нужно проводить, ориентируясь на физический смысл задачи. Необходимо проверить ситуации, описанные представленными выше графиками, а именно если значение теплового потока отрицательно, то производная T(x) должна быть положительной; если коэффициент теплоотдачи увеличивается, то уровень температур уменьшается, а градиент увеличивается; если нагрева нет, то температура постоянна по всей длине стержня и равна T0.

2. Получите простейший разностный аналог нелинейного краевого условия при 

,

где - заданная функция.

Производную аппроксимируйте односторонней разностью.

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

Каноническая форма СЛАУ с трехдиагональной матрицей имеет вид:

В рамках нашей задачи коэффициенты можно найти по доказанным в Лекции 7 формулам.

Основная прогоночная формула:

Прямой ход: находим все прогоночные коэффициенты от 1 до N.

Используя разностную схему с разностным краевым условием при x=0, полученную в Лекции 7, в соответствие с нашей задачей (уравнением (1)) и сопоставляя ее с основной прогоночной формулой, получим первые значения прогоночных коэфиициентов:

Далее находим все и :

Обратный ход: находим все y от N до 0:

- основная формула

Подставим в разностный аналог нелинейного краевого условия при , найденного в п.2:

Приведем подобные слагаемые:

Функция задана, полученное уравнение можно решить методом половинного деления.

Нашли , по основной формуле обратного хода найдем все до .

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

Правая прогонка: (начальные значения и определяются из левого краевого условия, остальные значения и – по рекуррентным формулам, как описано в п.3)

Тогда

Левая прогонка: (начальные значения и определяются из правого краевого условия, остальные значения и – по рекуррентным формулам)

Тогда

Получили, что использование встречной прогонки сводит задачу нахождения к системе:

Прогоночные коэффициенты найдены – значение определено. ­­­­