|  |  |
| --- | --- |
| 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 \_**Уласик Е. А.**\_\_\_\_\_\_\_\_\_\_\_**  **Группа \_**ИУ7-61Б**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_**Градов В. М.**\_\_\_\_** |  |

Москва.

2020 г.

# Введение

Цель работы: Получение навыков разработки алгоритмов решения краевой задачи при реализации моделей, построенных на ОДУ второго порядка.

Задача

Задана математическая модель. Уравнение для функции *T(x)*

(1)

Краевые условия

Функции *k(x),* заданы своими константами

Из условий :

Из :

Разностная схема с разностным краевым условием при *x = 0* была получена в лекции №7:

где

Получим интегро-интерполяционным методом **разностный аналог при *x=l.*** Учитывая (7.9) из лекции №7:

проинтегрируем уравнение (1) на отрезке . Учтём также:

Получим:

Второй и третий интегралы вычислим методом трапеций:

Подставим выражения потока и приведём к общему виду:

(

При этом в выражении принята простая аппроксимация:

Значения параметров для отладки

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

Сформулированная математическая модель описывает температурное поле вдоль цилиндрического стержня радиуса *R* и длиной *l*, причем *R << l* и температуру можно принять постоянной по радиусу цилиндра. Ось *x* направлена вдоль оси цилиндра и начало координат совпадает с левым торцем стержня. Слева при *x=0* цилиндр нагружается тепловым потоком . Стержень обдувается воздухом, температура которого равна . В результате происходит съем тепла с цилиндрической поверхности и поверхности правого торца при *x = l*. Функции являются, соответственно, коэффициентами теплопроводности материала стержня и теплоотдачи при обдуве.

Листинги

1. **def** right\_boundary\_coefficients():
2. x\_half = Data.xn\_minus\_half(Data.l)
4. p\_n = Data.p(Data.l)
5. p\_n\_minus\_1 = Data.p(Data.l - Data.h)
6. f\_n = Data.f(Data.l)
7. f\_n\_minus\_1 = Data.f(Data.l - Data.h)
9. kn = -x\_half - Data.alphaN \* Data.h - Data.h \* Data.h \* p\_n\_minus\_1 / 8 - Data.h \* Data.h \* p\_n / 4
10. mn = x\_half - Data.h \* Data.h \* p\_n\_minus\_1 / 8
11. pn = -Data.alphaN \* Data.t\_env \* Data.h - Data.h \* Data.h \* (f\_n\_minus\_1 + f\_n) / 4
12. **return** kn, mn, pn

15. **def** left\_boundary\_coefficients():
16. x\_half = Data.xn\_plus\_half(Data.x0)
17. p1 = Data.p(Data.x0 + Data.h)
18. f1 = Data.f(Data.x0 + Data.h)
20. p0 = Data.p(Data.x0)
21. f0 = Data.f(Data.x0)
23. p\_half = (p0 + p1) / 2
24. f\_half = (f0 + f1) / 2
26. K0 = x\_half + Data.h \* Data.h \* p\_half / 8 + Data.h \* Data.h \* p0 / 4
27. M0 = Data.h \* Data.h \* p\_half / 8 - x\_half
28. P0 = Data.h \* Data.F0 + Data.h \* Data.h \* (f\_half + f0) / 4
30. **return** K0, M0, P0

33. **def** calculate\_coefficients():
34. X = []
35. A = []
36. B = []
37. C = []
38. D = []
40. **for** x **in** np.arange(Data.x0, Data.l, Data.h):
41. An = Data.xn\_plus\_half(x) / Data.h
42. Cn = Data.xn\_minus\_half(x) / Data.h
43. Bn = An + Cn + Data.p(x) \* Data.h
44. Dn = Data.f(x) \* Data.h
46. A.append(An)
47. B.append(Bn)
48. C.append(Cn)
49. D.append(Dn)
50. X.append(x)
51. **return** A, B, C, D, X

Листинг 1. Функции вычисления параметров граничных условий и коэффициентов розничной схемы

Листинг 2. Функция прогонки

1. @staticmethod
2. **def** k(x):
3. **return** Data.ak / (x - Data.bk)
5. @staticmethod
6. **def** alpha(x):
7. **return** 20 \* Data.a\_alpha / (x - Data.b\_alpha)
9. @staticmethod
10. **def** xn\_plus\_half(x):
11. **return** 2 \* Data.k(x) \* Data.k(x + Data.h) / (Data.k(x) + Data.k(x + Data.h))
13. @staticmethod
14. **def** xn\_minus\_half(x):
15. **return** 2 \* Data.k(x) \* Data.k(x - Data.h) / (Data.k(x) + Data.k(x - Data.h))
17. @staticmethod
18. **def** p(x):
19. **return** 2 \* Data.alpha(x) / Data.R
21. @staticmethod
22. **def** f(x):
23. **return** 2 \* Data.alpha(x) / Data.R \* Data.t\_env
24. **def** run\_method(a, b, c, d, k0, m0, p0, kn, mn, pn):
25. ksi = [None, -m0 / k0]
26. eta = [None, p0 / k0]
28. **for** i **in** range(1, len(a)):
29. ksi\_n\_plus\_1 = c[i] / (b[i] - a[i] \* ksi[i])
30. nu\_n\_plus\_1 = (d[i] + a[i] \* eta[i]) / (b[i] - a[i] \* ksi[i])
32. ksi.append(ksi\_n\_plus\_1)
33. eta.append(nu\_n\_plus\_1)
35. y = [(pn - mn \* eta[-1]) / (kn + mn \* ksi[-1])]
37. **for** i **in** range(len(a) - 2, -1, -1):
38. y\_i = ksi[i + 1] \* y[0] + eta[i + 1]
39. y.insert(0, y\_i)
41. **return** y

Листинг 3. Вспомогательные функции

Результаты работы

1. График зависимости температуры *T(x)* от координаты *x* при заданных выше параметрах.



Картинка 1. График зависимости *T(x)* при отладочных входных параматрах

2. График зависимости *T(x)* при



Картинка 2. График зависимости *T(x)* при

3. График зависимости *T(x)* при увеличенных значениях .



Картинка 3. График *T(x)* при увеличении значений в 5 раз.



Картинка 4. График *T(x)* при увеличении значений в 20 раз.

4. График зависимости *T(x)* при

**

Картинка 5. График *T(x)* при

Контрольные вопросы

1. Какие способы тестирования программы можно предложить?

При увеличении длины стержня *T(x)* будет падать до значения температуры среды.

Также для тестирования можно изменять значение параметра . При отрицательных значениях должен происходить съём тепла: температура должна возрастать до температуры среды. При его обнулении ­ отсутствует нагревание, то есть температура стержня по всей его длине должна равняться температуре среды.

2. Получите простейший разностный аналог нелинейного краевого условия

,

где – заданная функция. Производную аппроксимируйте односторонней разностью.

Аппроксимируем производную:

Подставим в исходное уравнение:

Домножим на h и произведём замену , получим:

Получим уравнение относительно

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.

Для прямого хода нужно найти начальные прогоночные коэффициенты по формулам:

где коэффициенты были получены в лекции 7. Затем по формулам находим последующие прогоночные коэффициенты:

Получим , решив полученное уравнение в п.2, например, методом дихотомии. Далее по прогоночной формуле можно найти все значения неизвестных :

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

Для начала нужно вычислить начальные прогоночные коэффициенты. Для правой прогонки они вычисляются по:

Для левой прогонки по:

Далее необходимо найти остальные прогоночные коэффициенты. Для правой прогонки нужны коэффициенты с *n* от 1 до p – 1. Их найдём по формулам:

Для левой прогонки – от p до N – 1. Их найдём по формулам:

Прогоночные формулы для левой и правой прогонки:

Учитывая и составив систему уравнений, выразим :