МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

Физико-механический институт

Высшая школа прикладной математики и вычислительной физики

Отчет по курсовому проекту  
по курсу:

ЧИСЛЕННОЕ РЕШЕНИЕ КРАЕВЫХ ЗАДАЧ

на тему:

Исследование нестационарного поля температур в плоской неограниченной пластине с использованием метода Фурье

Работу выполнил:

студент группы 5030301/20001

Медведев Кирилл Максимович

Преподаватель:

к.т.н., доц. Плетнев А. А.

Санкт-Петербург

2024

**Оглавление**

[1 Постановка задачи 3](#_Toc164888865)

[1.1 Физическая постановка задачи 3](#_Toc164888866)

[1.2 Математическая постановка задачи 3](#_Toc164888867)

[3 Метод решения №1. Фурье 5](#_Toc164888868)

[4 Тестовый расчёт 7](#_Toc164888869)

[5 Результаты решения задачи 7](#_Toc164888870)

[5.1 Зависимости температуры от координаты 7](#_Toc164888871)

[5.2 Зависимости температуры от времени 9](#_Toc164888872)

[6 Выводы 10](#_Toc164888873)

[Приложения 11](#_Toc164888874)

# 1 Постановка задачи

## 1.1 Физическая постановка задачи

Дана стальная неограниченная пластина со следующими характеристиками [1].

Находящаяся в следующих условиях:

Для следующих коэффициентов теплоотдачи требуется провести исследование нестационарного температурного поля, используя метод Фурье

## 1.2 Математическая постановка задачи

|  |  |
| --- | --- |
| Обозначения | Рисунок |
| α – коэффициент конвективной теплоотдачи,  δ – толщина пластины,  τ – время,  𝑇 – температура пластины,  𝑇𝑒 – температура окружающей среды,  𝑇𝑤 – температура на границе пластины,  𝑞 – плотность теплового потока  𝑥 – координата, м |  |

Уравнение теплопроводности в размерных переменных имеет вид:

Начальные и граничные условия в размерных переменных

Приведём его к безразмерному виду, которое и будем в дальнейшем решать, с помощью следующей замены переменных:

Безразмерное уравнение теплопроводности имеет вид:

Начальные и граничные условия в безразмерных переменных:

# 3 Метод решения №1. Фурье

Решение уравнения (1) ищется в виде произведения функции зависящих отдельно от времени и координаты.

Таким образом, получаем два ОДУ с разделяющимися переменными, решая которые получим:

И используя граничное условие (3), находим, что A3 = 0. Введём A = A1\*A2

Используя граничное условие (4), получим

Изображение выглядит как диаграмма

Автоматически созданное описаниеИз графического анализа этого уравнения следует что оно имеет бесконечно много корней при каждом Био. Таким образом, для каждого получаем частное решение уравнения теплопроводности:

Рис 1. Графики функций y = ctg(u), y = u/Bi

Общее решение будет представляться суммой частных решений:

Используя теперь начальное условие (2) получаем выражение для An

Расчет зависимости = f(Fo, X) будем вести до достижения теплового равновесия пластины с окружающей средой. Будем считать, что тепловое равновесие наступает тогда, когда безразмерная температура в центре пластины достигла значения 0,1. Соответствующий момент времени обозначим за Fomax, который определяется исходя из значения Bi следующим образом:

* Bi <1,25 Fomax=
* 1,25 ≤ Bi ≤20 Fomax=2,76
* Bi>20 Fomax=1,1

Для определения нужно найти корни из решения уравнения (5). Для этого используется метод половинного деления. Его суть заключается в выделении отрезка, где непрерывная функция на концах имеет разные знаки и половинным делением отрезка находить корень, где функция пересекает оси абсцисс. Как видно из рисунка 1, каждое принадлежит отрезку

Блок-схема алгоритма вычисления (Fo, X) методом Фурье имеет вид:

Изображение выглядит как диаграмма

Автоматически созданное описание

# 4 Тестовый расчёт

Тестовый расчёт проведём для Bi = 0,7; Fo = 3,4; X = 0

По номограмме[2](приложение 1): = 0,16; Данные из программы = 0,1608

Совпадение данных говорит о корректности работы программы

# 5 Результаты решения задачи

Для различных вариантов коэффициента теплоотдачи c помощью написанной программы(приложение 2) получим результаты расчёта, которые представим в виде графиков, используя пакет SciDavis

## 5.1 Зависимости температуры от координаты

Для разных коэффициентов теплоотдачи (чисел Био) построим три зависимости температуры от координаты в три различных момента безразмерного времени: 0.1\*FoMax, 0.5\*FoMax, 0.9\*FoMax

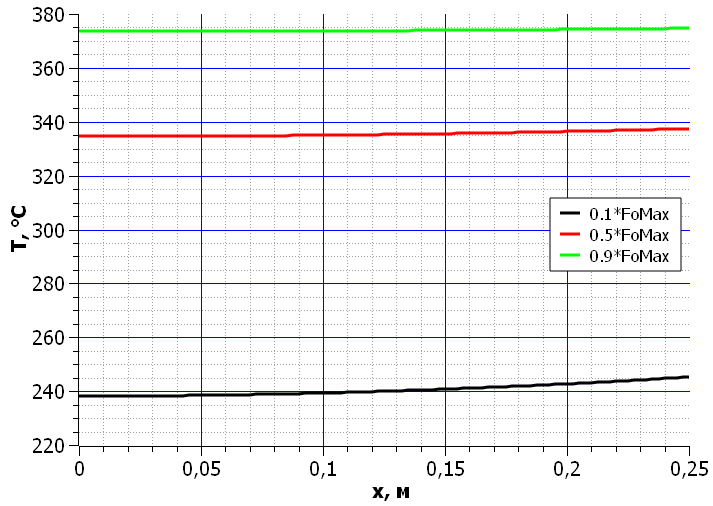


Рис 2. Зависимость температуры от координаты при Bi = 0,091; FoMax = 25,66

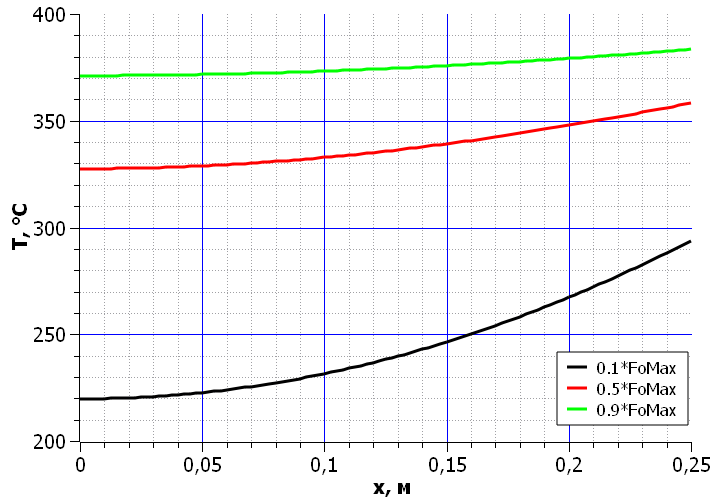


Рис 3. Зависимость температуры от координаты при Bi = 1,36; FoMax = 2,507

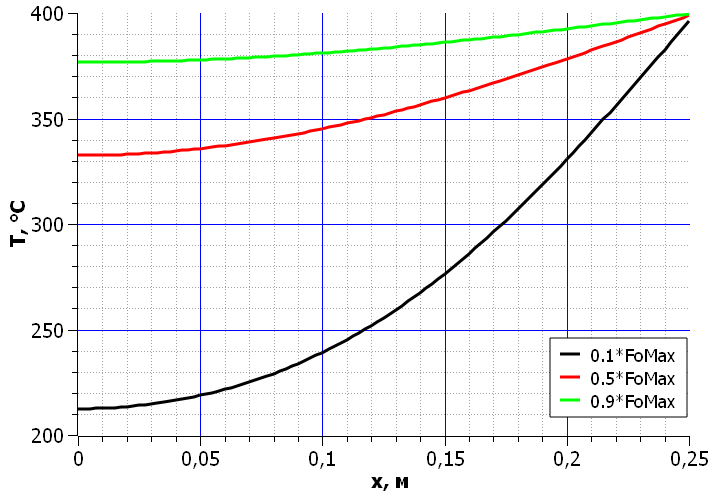


Рис 4. Зависимость температуры от координаты при Bi = 90,9; FoMax = 1,10

## 5.2 Зависимости температуры от времени

Для двух крайних значений числа Био построим зависимости температуры от времени в трёх сечениях пластины: x= 0, x= d/4, x = d/2.

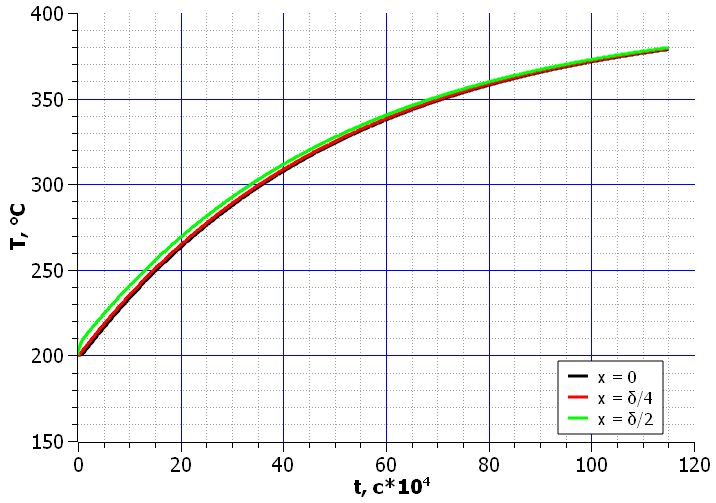


Рис 5. Зависимость температуры от времени в трех сечениях пластины при Bi = 0,091

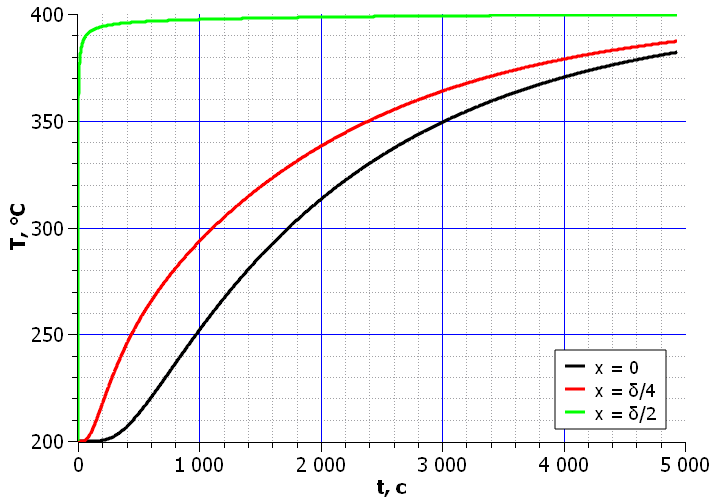


Рис 6. Зависимость температуры от времени в трех сечениях пластины при Bi = 90,9

# 6 Выводы

1. При малых значениях Био распределение температуры в пластине является практически равномерным по координате.

При больших значениях Био распределение не является равномерным. Температура на краях пластины отличается от температуры в ее центральной части, как видно на рисунке 4. Более того, температура на краях пластины почти мгновенно приближается к температуре окружающей среды (рисунки 4 и 6).

1. При малом значение числа Фурье (~10-2) для достижения заданной точности ε=10-7 требуется учитывать 12–14 членов ряда. Однако при увеличении этого числа, количество членов падает до 1–4
2. При малом Bi = 0.096 для установления теплового равновесия требуется 115159 c (25.66 Fo). При большом Bi = 90.96 требуется уже значительно меньше времени – 4937 с (1.1 Fo)

Ссылки на литературу:

1. <http://thermalinfo.ru/Sets/cache/supercache/thermalinfo.ru/svojstva-materialov/metally-i-splavy/teploemkost-stali/index.html>
2. Е.А. Краснощеков, А.С. Сукомел. Задачник по теплопередаче. М.: Энергия, 1984. Стр. 38-39.

# Приложения

Приложение 1

Номограмма

Изображение выглядит как диаграмма

Автоматически созданное описание

Приложение 2

Код программы

1 Program lab1Furie

program Lab1Furie

use methods

implicit none

integer :: i, n, m(2), k

real(8) :: l, d, T0, Te, eps, p, c

real(8) :: a(3), Bi(3), FoMax(3), dFo, X, Fof(3,3), Fo, Xof(3), dX

l = 55d0

d = 0.5d0

a(1) = 0.02d0

a(2) = 300d0

a(3) = 20000d0

T0 = 200d0

Te = 400d0

dFo = 0.01d0

dX = 0.01d0

p = 7900d0

c = 500d0

n = dint(1d0/dX)

!Вычисление чисел Bio

Bi(1) = atoBi(a(1), d, l, 1)

write(\*,\*) Bi(1)

Bi(2) = atoBi(a(2), d, l, 1)

write(\*,\*) Bi(2)

Bi(3) = atoBi(a(3), d, l, 1)

write(\*,\*) Bi(3)

!Вычисление FoMax и Fof

FoMax(1) = getFoMax(Bi(1))

FoMax(2) = getFoMax(Bi(2))

FoMax(3) = getFoMax(Bi(3))

Fof(:,1) = 0.1d0\*FoMax(:)

Fof(:,2) = 0.5d0\*FoMax(:)

Fof(:,3) = 0.9d0\*FoMax(:)

Xof(1) = xtoX(0d0, d, 1)

Xof(2) = xtoX(d/4d0, d, 1)

Xof(3) = xtoX(d/2d0, d, 1)

eps = 1e-7!Выбор точности вычисления

10 format(f18.9, 3(a,f18.9))

20 format(f18.9,3(a,f18.9))

open(1, file = "T1(x).csv")

open(2, file = "T2(x).csv")

open(3, file = "T3(x).csv")

write(1,\*) "X", ";", "T(a1, Fo11)", ";", "T(a1, Fo21)", ";", "T(a1, Fo31)"

write(2,\*) "X", ";", "T(a2, Fo12)", ";", "T(a2, Fo22)", ";", "T(a2, Fo32)"

write(3,\*) "X", ";", "T(a3, Fo13)", ";", "T(a3, Fo23)", ";", "T(a3, Fo33)"

do i = 0, n

X = dX\*real(i,8)

write(1,10) xtoX(X, d, -1), ";", TX(Fof(1,1), X, Bi(1), eps, T0, Te), ";", TX(Fof(1,2), X, Bi(1), eps, T0, Te), ";", TX(Fof(1,3), X, Bi(1), eps, T0, Te)

write(2,10) xtoX(X, d, -1), ";", TX(Fof(2,1), X, Bi(2), eps, T0, Te), ";", TX(Fof(2,2), X, Bi(2), eps, T0, Te), ";", TX(Fof(2,3), X, Bi(2), eps, T0, Te)

write(3,10) xtoX(X, d, -1), ";", TX(Fof(3,1), X, Bi(3), eps, T0, Te), ";", TX(Fof(3,2), X, Bi(3), eps, T0, Te), ";", TX(Fof(3,3), X, Bi(3), eps, T0, Te)

end do

close(1)

close(2)

close(3)

open(4, file="Ta1(tau).csv")

open(5, file="Ta3(tau).csv")

write(4,\*) "tau", ";", "Ta1(X1)", ";", "Ta1(X2)", ";", "Ta1(X3)"

write(5,\*) "tau", ";", "Ta3(X1)", ";", "Ta3(X2)", ";", "Ta3(X3)"

dFo = 0.0001d0

m(1) = dint(FoMax(1)/dFo) + 1

m(2) = dint(FoMax(3)/dFo) + 1

write(4, 20) tautoFo(0d0, l, d, c, p, -1), ";", TtoTheta(1d0, T0+273.15d0, Te+273.15d0, -1), ";", TtoTheta(1d0, T0+273.15d0, Te+273.15d0, -1), ";", TtoTheta(1d0, T0+273.15d0, Te+273.15d0, -1)

do i = 1, m(1)

Fo = dFo\*real(i,8)

write(4, 20) tautoFo(Fo, l, d, c, p, -1), ";", TX(Fo, Xof(1), Bi(1), eps, T0, Te), ";", TX(Fo, Xof(2), Bi(1), eps, T0, Te), ";", TX(Fo, Xof(3), Bi(1), eps, T0, Te)

end do

write(5, 20) tautoFo(0d0, l, d, c, p, -1), ";", TtoTheta(1d0, T0+273.15d0, Te+273.15d0, -1), ";", TtoTheta(1d0, T0+273.15d0, Te+273.15d0, -1), ";", TtoTheta(1d0, T0+273.15d0, Te+273.15d0, -1)

do i = 1, m(2)

Fo = dFo\*real(i,8)

write(5, 20) tautoFo(Fo, l, d, c, p, -1), ";", TX(Fo, Xof(1), Bi(3), eps, T0, Te), ";", TX(Fo, Xof(2), Bi(3), eps, T0, Te), ";", TX(Fo, Xof(3), Bi(3), eps, T0, Te)

end do

close(4)

close(5)

pause

end program Lab1Furie2 Module Methods

module methods

implicit none

contains

function TX(Fo, X, Bi, eps, T0, Te) result(Theta)

real(8), intent(in) :: Fo, X, Bi, eps, T0, Te

real(8) :: Thetan, Theta, mu

integer :: n

n = 1

Theta = 0d0

Thetan = 1d0

do while(dabs(Thetan) >= eps)

mu = MPD(n,Bi, eps)

Thetan = (2d0\*dsin(mu)\*dcos(mu\*X)/(mu + dsin(mu)\*dcos(mu)))\*dexp(-Fo\*(mu)\*\*2)

Theta = Theta + Thetan

n = n + 1

end do

!write(\*,"(f18.13, f18.13, f18.8, i9)") Fo, X, Bi, n

Theta = TtoTheta(Theta, T0+273.15d0, Te+273.15d0, -1)

end function

function MPD(n, Bi, eps) result(y)

integer, intent(in) :: n

real(8), parameter :: pi = 4d0\*datan(1d0)

real(8) :: a, b, c, y, Bi, eps

a = pi\*real((n-1), 8)

b = pi\*real(2\*n - 1,8)/2d0

do while (dabs(a-b) >= eps)

c = (a+b)/2d0

if(f(b, Bi)\*f(c,Bi) < 0d0) then

a = c

else

b = c

end if

end do

y = (a+b)/2d0

end function

function f(x, Bi) result(y)

real(8), intent(in) :: x, Bi

real(8) :: y

y = dcotan(x) - x/Bi

end function

function getFoMax(Bi) result(FoMax)

real(8), intent(in) :: Bi

real(8) :: FoMax

if(Bi < 1.25d0) then

FoMax = 3.11d0\*((Bi)\*\*(-0.88d0))

else if (Bi > 20d0) then

FoMax = 1.10d0

else

FoMax = 2.76d0\*((Bi)\*\*(-0.31d0))

end if

end function

function xtoX(x, d, m) result(X1)

real(8), intent(in) :: x, d

integer, intent(in) :: m

real(8) :: X1

if(m == 1) then

X1 = 2d0\*x/d

else

X1 = d\*x/2d0

end if

end function

function TtoTheta(T, T0, Te, m) result(Theta)

real(8), intent(in) :: T, T0, Te

integer, intent(in) :: m

real(8) :: Theta

if(m == 1) then

Theta = (T - Te)/(T0 - Te)

else

Theta = T\*(T0 - Te) + Te - 273.15d0

end if

end function

function tautoFo(tau, l, d, c, p, m) result(Fo1)

real(8), intent(in) :: tau, l, d, c, p

integer, intent(in) :: m

real(8) :: Fo1

if(m == 1) then

Fo1 = tau\*l\*4d0/(c\*p\*(d\*\*2))

else

Fo1 = c\*p\*(d\*\*2)\*tau/(4d0\*l)

end if

end function

function atoBi(a,d,l,m) result(Bi)

real(8), intent(in) :: a, d, l

integer, intent(in) :: m

real(8) :: Bi

if(m == 1) then

Bi = a\*d/(2d0\*l)

else

Bi = 2d0\*l\*a/d

end if

end function

end module