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 5030301/10002

Тугай В.В.

Преподаватель:

к.т.н., доц. Плетнев А.А.

Санкт-Петербург

2023

# Содержание

1. Физическая постановка задачи
2. Математическая постановка задачи
3. Метод решения
4. Тестовый расчет
5. Результаты решения задачи
6. Выводы
7. Приложение

# Физическая постановка задачи

Плоская неограниченная пластина из бронзы толщиной  60 см испытывает конвективный теплообмен с окружающей средой (с обеих сторон пластины интенсивность конвективного теплообмена одинакова). В начальный момент времени температура пластины постоянна во всем сечении и равна  500 °С. Температура окружающей среды  130 °С. Найти распределение температуры пластины в зависимости от координаты и времени для трех значений коэффициента конвективной теплоотдачи:

α1 = 35 Вт/(м²⋅К);   α2= 400 Вт/(м²⋅К);   α3= 25000 Вт/(м²⋅К).

*T*– температура пластины, K

*Tw*– температура на границе пластины, K

*Te*– температура окружающей среды, K

*T0*– начальная температура пластины, K

*τ* – время, с

*x*– координата, м

*δ* – толщина пластины, м

*q*– плотность теплового потока, Вт/м2

a – коэффициент конвективной теплоотдачи, Вт/(м2×К)

Материал пластины: бронза. Физические свойства материала и значения физический величин (Вариант-7):

*p*(1)=8600 кг/м2

*c*(2)=380 Дж/(К⋅кг)

*λ*=110 Вт/(м⋅К)

*δ*=0.6 м

*Te*=130 С˚

*T0*=500 С˚

(1) http://ru.solverbook.com/spravochnik/ximiya/plotnost/plotnost-bronzy/

(2) http://thermalinfo.ru/eto-interesno/tablitsy-udelnoj-teploemkosti-veshhestv

# Математическая постановка задачи

Diagram, engineering drawing

Description automatically generatedНачало координат – в центре пластины.  
Искомая функция – температура, которая зависит от двух переменных: координаты и времени

Плотность теплового потока (закон Фурье) : , где — коэффициент теплопроводности, Вт/(м⋅К)

Начальное условие (НУ):  
 – однородный профиль температуры

Граничные условия (ГУ):  
 – ГУ симметрии

– ГУ 3-го рода

Переход к безразмерным величинам:

— безразмерная координата ( X∈[0,1] )

— безразмерная избыточная температура( θ∈[0,1] )

— критерий Фурье (безразмерное время)

— число Био (безразмерный коэффициент теплоотдачи)

Уравнение теплопроводности в безразмерных величинах:

НУ и ГУ в безразмерных величинах:

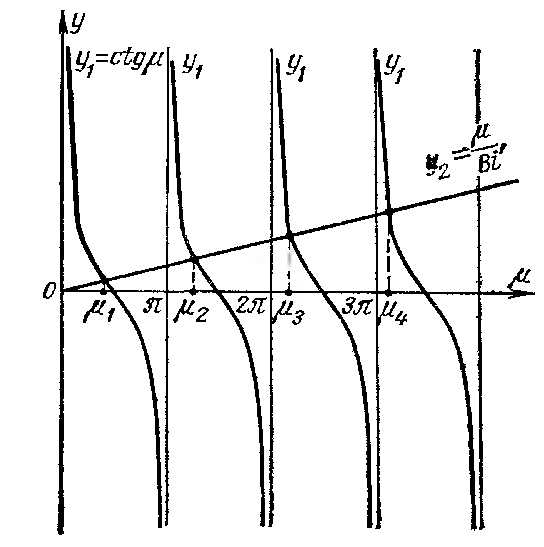
# Метод решения

Решение дифференциального уравнения будем искать в виде произведения двух функции, одна из которых является функцией только переменной Fo, а другая – только X: θ=f(Fo,X)=ϕ(Fo)ψ(X) .

После подстановки в уравнение теплопроводности: . Уравнение можно записать следующим образом: . Имеем: *.* Решив эти Д.У., получаем:

Используя граничное условие при Х=0 => А3=0. Если А1∙А2=А, то искомая функция θ=f (Fo, X ) имеет вид: .

Используя граничное условие при Х=1 получаем:

−θ′XX=1=Ae−μ2Fosin(μX)| X=1=Biθ=BiAe−μ2Focos(μX)|X=1,   
Из графического анализа уравнения следует, что при каждом значении Bi существует бесконечное множество корней.

Тогда при Bi →∞μn =+π(n −1) , а при Bi →0 μn =π(n −1), т.е. μn∈(π(n−1),π(n)).

Для каждого μn имеем частное решение вида: θ=Ae−μn2Focos(μX)   
Общее решение θ=f (Fo, X ) есть сумма всех частных решений, представляющая собой быстро сходящийся ряд:

θ=−μn2Focos(μX) (1)

где An=– может быть определено из начального условия. Тогда полученное решение можно записать так:

### Блок-схема вычислительной программы (внешний цикл по Fo)



Начало

Описание переменных

Ввод исходных данных, вычисление FoMax и dFo



Начало цикла по времени Fo=0



n = 0, θ = 0

Начало цикла суммирования

n = n + 1

Вычисление 𝜇𝑛

Вычисление 𝐴𝑛

Вычисление 𝜃𝑛

Нет

|𝜃 | < 𝜀

𝑛

Да

Конец

Да

Fo FoMax

Нет

𝜃 = 𝜃 + 𝜃𝑛

Сохранение требуемого решения

Fo=Fo+dFo

# Тестовый расчет

# Для значений X=0, Fo=6, Bi=0.45, урезанная программа для тестирования (в ней осуществляется лишь расчет θ при заданных значениях X, Fo, Bi) выдает значение θ=0.102; для X=0, Fo=3, Bi=0.5: θ=0.297; для X=0, Fo=8, Bi=0.6: θ=0.02; данные значения θ соответствуют монограмме (см. Приложение 2). Отсюда делаем вывод, что программа работает исправно.

# Результат решения задачи

# На рисунках 1 и 2 изображены зависимости температуры от времени для трех разных сечений пластины (х=0; х=δ/4; х=δ/2) при Bi=0.0955 и Bi=68.2. На рисунках видно, что с увеличением значения параметра Bi уменьшается время, необходимое для достижения теплового равновесия. График зависимости температуры от времени для x=δ/2 испытывает резкий спад, а для х=δ/4 и х=0 графики асимптотически приближаемся к температуре, соответствующей состоянию теплового равновесия. На рисунках 3, 4 и 5 изображены зависимости температуры от координаты для трех разных временных точек (t=0.1⋅Fomax; t=0.5⋅Fomax; t=0.9⋅Fomax) при Bi=0.0955, Bi=1.091 и Bi=68.2. На рисунках видно, что с увеличением значения параметра Bi разница температуры между центром пластины и ее краем возрастает, и при приближении к точке времени Fomax температура в точках пластины приближается к значениям температуры в момент теплового равновесия.

# Chart Description automatically generated with medium confidenceChart, histogram Description automatically generated

Рисунок 1 Рисунок 2

Зависимость температуры от Зависимость температуры от

времени при Bi=0.095 времени при Bi=68.2

Chart

Description automatically generated with medium confidenceChart, line chart

Description automatically generated

Рисунок 3 Рисунок 4

Зависимость температуры от Зависимость температуры от

координаты при Bi=0.095 координаты при Bi=1.09 Fomax = 55752 с Fomax = 7750 с

Chart, line chart

Description automatically generated

# Рисунок 5

Зависимость температуры от

координаты при Bi=68.2

Fomax = 2941 с

# Выводы

# В ходе решения поставленной задачи был изучен метод Фурье. Решение задачи данным методом было реализовано на языке программирования Fortran (см. Приложение 1), результаты программы представлены в виде графиков с использованием пакета Matlab.

# При больших значениях Bi (Bi>10) температура быстрее стремится к положению равновесия, нежели в сравнении с малыми (Bi<0.1) значениями (см. Рисунок 1 и 2). Так же при малых значениях данного параметра распределение температуры по пластине можно назвать равномерным, в отличии от больших значений (см. Рисунок 3 и 5).

# Число членов ряда (1) (см. Метод решения) также зависит и от значения параметра Bi – при малых Bi число членов ряда меньше, при больших же – больше. При точности ε=10-12 для расчета температуры при малых Bi и при малых, средних и больших числах Фурье требуется 2, 1 и 1 члена ряда соответственно, при больших же значениях Bi – 5, 2 и 2 члена ряда соответственно (см. Приложение 3).

# Будем считать, что тепловое равновесие – момент времени tравн , в котором безразмерная температура в центре пластины достигает значения 0,1 (см. Приложение 4 и 5). Тогда:

# При Bi = 0.0955: tравн = 67008 с

# При Bi = 68.2: tравн = 2838 с

# Приложение

# Основная программа

module functions

contains

real\*16 Function f(Mu, Bi) result(res)

real\*16 :: Mu, Bi

res=(cos(Mu))/(sin(Mu))-(Mu/Bi) !function for bisection

end function

real\*16 Function A\_f(Mu) result(res)

real\*16 :: Mu

res=(2\*sin(Mu))/(Mu+sin(Mu)\*cos(Mu)) !function for A

end function

real\*16 Function O\_f(Mu, A, Fo, X) result(res)

real\*16 :: Mu, A, Fo, X

res=A\*exp( (-1)\*Fo\*(Mu\*\*2))\*cos(Mu\*X) !function for theta

end function

end module

module methods

contains

real\*16 Function bis(n, E, Bi) result(res) !bisection

use functions

real\*16 :: E, a\_Mu, b\_Mu, c\_Mu, f1, f2, check, pi=3.1415926535897932, Bi, delt, n\_r

integer :: n, i

n\_r=real(n, 16)

a\_Mu=pi\*(n\_r-1)

b\_Mu=pi\*(n\_r-0.5)

f1=f(b\_Mu, Bi)

do i=1,1000

c\_Mu=(a\_Mu+b\_Mu)/2.0

f2=f(c\_Mu, Bi)

check=f1\*f2

if (check>0) then

b\_Mu=c\_Mu

f1=f2

elseif (check<0) then

a\_Mu=c\_Mu

elseif (check==0) then

a\_Mu=c\_Mu

b\_Mu=c\_Mu

end if

delt=abs(a\_Mu-b\_Mu)

if (delt<E) exit

end do

res=c\_Mu !saving result

end function

character(len=20) function str(k) !function for translating integer to string

integer, intent(in) :: k

write (str, \*) k

str = adjustl(str)

end function

subroutine Fourier(*check1*, *Te*, *T0*, *delta*, *alpha*, *lambda*, *cp*, *p*, *E*)

use functions

implicit none

real\*16 Te, T0, delta, lambda, cp, p, O, Bi, Mu, A, Fo, X, Fo\_max, d, check, E, en, param, st, time\_01

real\*4 Bi\_wr

integer :: i, j, n, check1, k, u, h, alpha, start, Fo\_points

real\*16, allocatable, dimension (:) :: O\_n, T, tau, xx,tau\_points

Bi=(alpha\*delta)/(2\*lambda)

write(\*,'(F7.4)') Bi

if (Bi<0.5) then !Fo\_max based on Bi

Bi\_wr=anint(Bi\*10000)/10000

elseif (Bi>10) then

Bi\_wr=anint(Bi\*10)/10

else

Bi\_wr=anint(Bi\*1000)/1000

end if

if (Bi<1.25) then

Fo\_max=3.11\*Bi\*\*(-0.81)

elseif (Bi>20) then

Fo\_max=1.1

else

Fo\_max=2.76\*Bi\*\*(-0.31)

end if

if (check1==0) then

d=Fo\_max/100

en=Fo\_max

write(13,\*) alpha, Bi\_wr

else

d=1.0/99

en=1.0

write(14,\*) alpha, Bi\_wr

end if

if (check1==0) then

allocate(tau\_points(100))

open(15, file='bin/res/time/fo\_points\_alpha'//trim(str(alpha))//'.txt')

do Fo\_points=1,100

if (Fo\_points>Fo\_max) exit

tau\_points(Fo\_points)=(Fo\_points\*cp\*p\*(delta\*\*2))/(4\*lambda)

write(15,\*) Fo\_points, tau\_points(Fo\_points), 400

end do

deallocate(tau\_points)

end if

do k=1,3 !do for 3 points (x=0, x=d/4 and x=d/2 or Fo=0.1\*Fo\_max, Fo=0.5\*Fo\_max or Fo=0.9\*Fo\_max)

allocate(O\_n(100))

allocate(T(100))

T(1)=(T0-Te)+Te

if (check1==0) then !opening files for results

allocate(tau(100))

tau(1)=0

open(10, file='bin/res/time/alpha'//trim(str(alpha))//'\_point'//trim(str(k))//'\_temp.txt')

open(11, file='bin/res/time/alpha'//trim(str(alpha))//'\_point'//trim(str(k))//'\_time.txt')

write(10,\*) T(1)

write(11,\*) tau(1)

start=2

else

allocate(xx(100))

open(10, file='bin/res/coord/alpha'//trim(str(alpha))//'\_point'//trim(str(k))//'\_temp.txt')

open(12, file='bin/res/coord/alpha'//trim(str(alpha))//'\_point'//trim(str(k))//'\_coord.txt')

start=1

end if

if (k==1)then !param=static time or coordinate

if (check1==0) then

param=(2\*0)/delta

write(\*,'(a40,$)')'for point (x=0) iterations='

else

param=Fo\_max\*0.1

write(\*,'(a40,$)')'for point (0.1\*Fo\_max) iterations='

end if

elseif (k==2) then

if (check1==0) then

param=(2\*(delta/4))/delta

write(\*,'(a40,$)')'for point (x=delta/4) iterations='

else

param=Fo\_max\*0.5

write(\*,'(a40,$)')'for point (0.5\*Fo\_max) iterations='

end if

elseif (k==3) then

if (check1==0) then

param=(2\*(delta/2))/delta

write(\*,'(a40,$)')'for point (x=delta/2) iterations='

else

param=Fo\_max\*0.9

write(\*,'(a40,$)')'for point (0.9\*Fo\_max) iterations='

end if

end if

st=0

if (check1==1) then

st=-d

end if

do j=start,100

st=st+d !st=dynamic time or coordinate

n=0

O=0

O\_n(j)=0

check=1

do h=1,1000

n=n+1

Mu=bis(n, E, Bi)

A=A\_f(Mu)

if (check1==0) then

O=O\_f(Mu, A, st, param)

else

O=O\_f(Mu, A, param, st)

end if

O\_n(j)=O\_n(j)+O

check=abs(O)

if (check<E) then

n=n-1

exit

end if

end do

T(j)=O\_n(j)\*(T0-Te)+Te

if (check1==0) then

tau(j)=(st\*cp\*p\*(delta\*\*2))/(4\*lambda) !real time

write(10,\*) T(j)

write(11,\*) tau(j)

else

xx(j)=(st\*delta)/2.0 !real coordinate

write(10,\*) T(j)

write(12,\*) xx(j)

end if

if (st>en) exit

end do

close(10)

if (check1==0) then

close(11)

else

close(12)

end if

deallocate(O\_n)

deallocate(T)

if (check1==0) then

deallocate(tau)

else

deallocate(xx)

end if

write(\*,'(i2)') n !number of iterations for |theta|<eps

end do

end subroutine

end module

program main

use functions

use methods

implicit none

real\*16 Te, T0, delta, lambda, cp, p, E

integer, allocatable, dimension (:) :: alpha\_n

integer :: check1, n, i, alpha

Te=130.0+273.15

T0=500.0+273.15

delta=0.6

lambda=110.0

cp=380.0

p=8600.0

E=0.000000000001

n=3

allocate(alpha\_n(n))

alpha\_n(1)=35

alpha\_n(2)=400

alpha\_n(3)=25000

open(13, file='bin/res/time/bi\_alpha.txt')

open(14, file='bin/res/coord/bi\_alpha.txt')

check1=1 !static Fo

write(\*,\*) 'T(x)'

write(\*,\*)

do i=1,n

alpha=alpha\_n(i)

write(\*,'(A7,$)') 'Bi='

call Fourier(check1, Te, T0, delta, alpha, lambda, cp, p, E)

write(\*,\*)

end do

check1=0 !static X

write(\*,\*) 'T(t)'

write(\*,\*)

alpha=alpha\_n(1)

write(\*,'(A7,$)') 'Bi='

call Fourier(check1, Te, T0, delta, alpha, lambda, cp, p, E)

write(\*,\*)

alpha=alpha\_n(n)

write(\*,'(A7,$)') 'Bi='

call Fourier(check1, Te, T0, delta, alpha, lambda, cp, p, E)

close(13)

close(14)

end program

# Урезанная программа для тестирования

module functions

contains

real\*16 Function f(Mu, Bi) result(res)

real\*16 :: Mu, Bi

res=(cos(Mu))/(sin(Mu))-(Mu/Bi)

end function

real\*16 Function A\_f(Mu) result(res)

real\*16 :: Mu

res=(2\*sin(Mu))/(Mu+sin(Mu)\*cos(Mu))

end function

real\*16 Function O\_f(Mu, A, Fo, X) result(res)

real\*16 :: Mu, A, Fo, X

res=A\*exp( (-1)\*Fo\*(Mu\*\*2))\*cos(Mu\*X)

end function

end module

module methods

contains

real\*16 Function bis(n, E, Bi) result(res)

use functions

real\*16 :: E, a\_Mu, b\_Mu, c\_Mu, f1, f2, check, pi=3.1415926535897932, Bi, delt, n\_r

integer :: n, i

n\_r=real(n, 16)

a\_Mu=pi\*(n\_r-1)

b\_Mu=pi\*(n\_r-0.5)

f1=f(b\_Mu, Bi)

do i=1,1000

c\_Mu=(a\_Mu+b\_Mu)/2.0

f2=f(c\_Mu, Bi)

check=f1\*f2

if (check>0) then

b\_Mu=c\_Mu

f1=f2

elseif (check<0) then

a\_Mu=c\_Mu

elseif (check==0) then

a\_Mu=c\_Mu

b\_Mu=c\_Mu

end if

delt=abs(a\_Mu-b\_Mu)

if (delt<E) exit

end do

res=c\_Mu

end function

subroutine Fourier(*X*, *Fo*, *Bi*, *E*)

use functions

implicit none

real\*16 O, Bi, Mu, A, Fo, X, E, O\_n, check

integer :: i, n

n=0

O=0

O\_n=0

check=1

do i=1,1000

n=n+1

Mu=bis(n, E, Bi)

A=A\_f(Mu)

O\_n=O\_f(Mu, A, Fo, X)

O=O+O\_n

check=abs(O)

if (check<E) exit

end do

write(\*,\*)'O=', O

end subroutine

end module

program main

use functions

use methods

implicit none

real\*16 X, Fo, Bi, E

write(\*,'(A9,$)')'Input X: '

read(\*,\*) X

write(\*,'(A10,$)')'Input Fo: '

read(\*,\*) Fo

write(\*,'(A10,$)')'Input Bi: '

read(\*,\*) Bi

E=0.000000000001

call Fourier(X, Fo, Bi, E)

end program

# Результат основной программы (подсчет количества итераций для схождения ряда)

T(x)

Bi= 0.0955

for point (0.1\*Fo\_max) iterations= 2

for point (0.5\*Fo\_max) iterations= 1

for point (0.9\*Fo\_max) iterations= 1

Bi= 1.0909

for point (0.1\*Fo\_max) iterations= 3

for point (0.5\*Fo\_max) iterations= 2

for point (0.9\*Fo\_max) iterations= 1

Bi=68.1818

for point (0.1\*Fo\_max) iterations= 5

for point (0.5\*Fo\_max) iterations= 2

for point (0.9\*Fo\_max) iterations= 2

# Урезанная программа для нахождения момента теплового равновесия

module functions

contains

real\*16 Function f(Mu, Bi) result(res)

real\*16 :: Mu, Bi

res=(cos(Mu))/(sin(Mu))-(Mu/Bi) !function for bisection

end function

real\*16 Function A\_f(Mu) result(res)

real\*16 :: Mu

res=(2\*sin(Mu))/(Mu+sin(Mu)\*cos(Mu)) !function for A

end function

real\*16 Function O\_f(Mu, A, Fo, X) result(res)

real\*16 :: Mu, A, Fo, X

res=A\*exp( (-1)\*Fo\*(Mu\*\*2))\*cos(Mu\*X) !function for theta

end function

end module

module methods

contains

real\*16 Function bis(n, E, Bi) result(res) !bisection

use functions

real\*16 :: E, a\_Mu, b\_Mu, c\_Mu, f1, f2, check, pi=3.1415926535897932, Bi, delt, n\_r

integer :: n, i

n\_r=real(n, 16)

a\_Mu=pi\*(n\_r-1)

b\_Mu=pi\*(n\_r-0.5)

f1=f(b\_Mu, Bi)

do i=1,1000

c\_Mu=(a\_Mu+b\_Mu)/2.0

f2=f(c\_Mu, Bi)

check=f1\*f2

if (check>0) then

b\_Mu=c\_Mu

f1=f2

elseif (check<0) then

a\_Mu=c\_Mu

elseif (check==0) then

a\_Mu=c\_Mu

b\_Mu=c\_Mu

end if

delt=abs(a\_Mu-b\_Mu)

if (delt<E) exit

end do

res=c\_Mu !saving result

end function

subroutine Fourier(*Te*, *T0*, *delta*, *alpha*, *lambda*, *cp*, *p*, *E*)

use functions

implicit none

real\*16 Te, T0, delta, lambda, cp, p, O, O\_n, Bi, Mu, A, Fo\_max, d, check, E, param, st, time\_01

integer :: i, j, n, k, u, h, alpha, start, Fo\_points

Bi=(alpha\*delta)/(2\*lambda)

write(\*,'(F7.4)') Bi

if (Bi<1.25) then

Fo\_max=3.11\*Bi\*\*(-0.81)

elseif (Bi>20) then

Fo\_max=1.1

end if

d=Fo\_max/10000

start=2

param=(2\*0)/delta

st=0

do j=start,1000000000

st=st+d !st=dynamic time or coordinate

n=0

O=0

O\_n=0

check=1

do h=1,1000

n=n+1

Mu=bis(n, E, Bi)

A=A\_f(Mu)

O=O\_f(Mu, A, st, param)

O\_n=O\_n+O

check=abs(O)

if (check<E) then

n=n-1

exit

end if

end do

if (O\_n<=0.1) then

time\_01=(st\*cp\*p\*(delta\*\*2))/(4\*lambda)

print\*, O\_n

exit

end if

end do

write(\*,'(a40,$)') 'time in x=0 for 0.1\*Theta : '

write(\*,\*) time\_01

end subroutine

end module

program main

use functions

use methods

implicit none

real\*16 Te, T0, delta, lambda, cp, p, E

integer :: n, i, alpha

Te=130.0+273.15

T0=500.0+273.15

delta=0.6

lambda=110.0

cp=380.0

p=8600.0

E=0.000000000001

write(\*,\*) 'T(t)'

write(\*,\*)

alpha=35

write(\*,'(A7,$)') 'Bi='

call Fourier(Te, T0, delta, alpha, lambda, cp, p, E)

write(\*,\*)

alpha=25000

write(\*,'(A7,$)') 'Bi='

call Fourier(Te, T0, delta, alpha, lambda, cp, p, E)

end program

# Результат программы для нахождения момента теплового равновесия

T(t)

Bi= 0.0955

time in x=0 for 0.1\*Theta : 67007.7981454403300436326083474926812

Bi=68.1818

time in x=0 for 0.1\*Theta : 2838.25828708207460602682615775515257