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:

$$\alpha_1 = 35 \text{ Вт/(м}^2\cdot\text{К)}; \quad \alpha_2 = 400 \text{ Вт/(м}^2\cdot\text{К)}; \quad \alpha_3 = 25000 \text{ Вт/(м}^2\cdot\text{К)}.$$

T– температура пластины, К

T_w– температура на границе пластины, К

T_e– температура окружающей среды, К

T_0– начальная температура пластины, К

τ – время, с

x– координата, м

δ – толщина пластины, м

q– плотность теплового потока, Вт/м2

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

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

$p^{(1)}$=8600 кг/м^2

$c^{(2)}$=380 Дж/(К·кг)

λ=110 Вт/(м·К)

δ=0.6 м

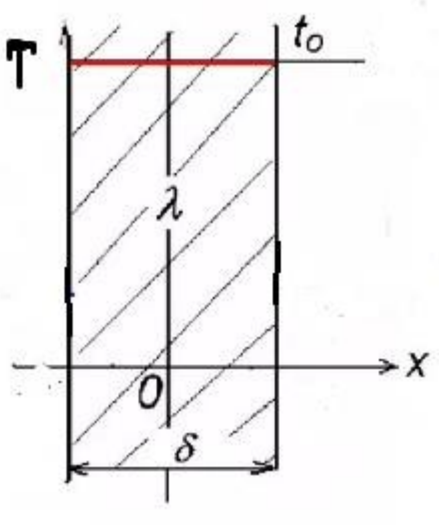
T_e=130 C°

T_0=500 C°

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

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

Математическая постановка задачи



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

$$T = T(\tau, \mathrm{x})$$

Плотность теплового потока (закон Фурье) : $q = -\lambda \frac{\partial T}{\partial n}$, или $q = -\lambda\,\mathrm{grad}\mathrm{T} = -\lambda\nabla\mathrm{T}$, где λ — коэффициент теплопроводности, Вт/(м·К)

Начальное условие (НУ):
$T(0, \mathrm{x}) = T_0$ – однородный профиль температуры

Граничные условия (ГУ):
$\frac{\partial T}{\partial x}(\tau, 0) = 0$ – ГУ симметрии

$$q\big|_{x=\frac{\delta}{2}} = -\lambda \frac{\partial T}{\partial x}\Big|_{x=\frac{\delta}{2}} = \alpha(T_w - T_e) \text{ – ГУ 3-го рода}$$

Переход к безразмерным величинам:

$X = \frac{2x}{\delta}$ — безразмерная координата ($X \in [0,1]$)

$\theta = \frac{T - T_e}{T_0 - T_e}$ — безразмерная избыточная температура($\theta \in [0,1]$)

$Fo = \frac{\tau}{\left(\frac{\delta}{2}\right)^2} \frac{\lambda}{cp}$ — критерий Фурье (безразмерное время)

$Bi = \frac{\alpha\delta}{2\lambda}$ — число Био (безразмерный коэффициент теплоотдачи)

Уравнение теплопроводности в безразмерных величинах:

$$\frac{\partial\theta}{\partial Fo} = \frac{\partial^2\theta}{\partial X^2}$$

НУ и ГУ в безразмерных величинах: $\begin{cases} \theta(0, X) = 1 \\ \frac{\partial\theta}{\partial X}(Fo; 0) = 0 \\ -\frac{\partial\theta}{\partial X}(Fo; 1) = Bi\theta \end{cases}$

Метод решения

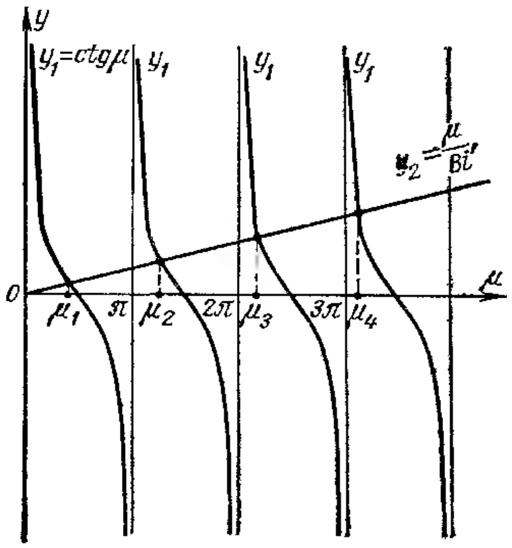
Решение дифференциального уравнения $\frac{\partial\theta}{\partial Fo} = \frac{\partial^2\theta}{\partial X^2}$ будем искать в виде произведения двух функции, одна из которых является функцией только переменной Fo, а другая – только X: $\theta=f(Fo,X)=\varphi(Fo)\psi(X)$.

После подстановки в уравнение теплопроводности: $\varphi'(\mathrm{Fo})\psi(\mathrm{X}) = \psi''(\mathrm{X})\,\varphi(\mathrm{Fo})$. Уравнение можно записать следующим образом: $\frac{\psi''(\mathrm{X})}{\psi(\mathrm{X})} = \frac{\varphi'(\mathrm{Fo})}{\varphi(\mathrm{Fo})} = const = -\mu^2$. Имеем: $\varphi' + \mu^2\varphi = 0$ и $\psi'' + \mu^2\psi = 0$. Решив эти Д.У., получаем: $\varphi(\mathrm{Fo}) = \mathrm{A}_1\mathrm{e}^{-\mu^2 Fo}$, $\psi(\mathrm{X}) = A_2\cos(\mu X) + A_3\sin(\mu X)$

Используя граничное условие при X=0 => $A_3=0$. Если $A_1\cdot A_2=A$, то искомая функция $\theta = f(Fo, X)$ имеет вид: $\theta = \mathrm{A}\mathrm{e}^{-\mu^2 Fo}\cos(\mu X)$.

Используя граничное условие при X=1 получаем:

$-\theta'_X|_{X=1} = Ae^{-\mu^2 Fo}\sin(\mu X)|_{X=1} = Bi\theta = BiAe^{-\mu^2 Fo}\cos(\mu X)|_{X=1}$, $ctg(\mu) = \frac{\mu}{Bi}$



Из графического анализа уравнения следует, что при каждом значении Bi существует бесконечное множество корней.

Тогда при $Bi \to \infty$ $\mu_n = \frac{\pi}{2} + \pi(n-1)$, а при $Bi \to 0$ $\mu_n = \pi(n-1)$, т.е. $\mu_n \in (\pi(n-1), \pi(n-\frac{1}{2}))$.

Для каждого μ_n имеем частное решение вида: $\theta = Ae^{-\mu_n^2 Fo}\cos(\mu X)$

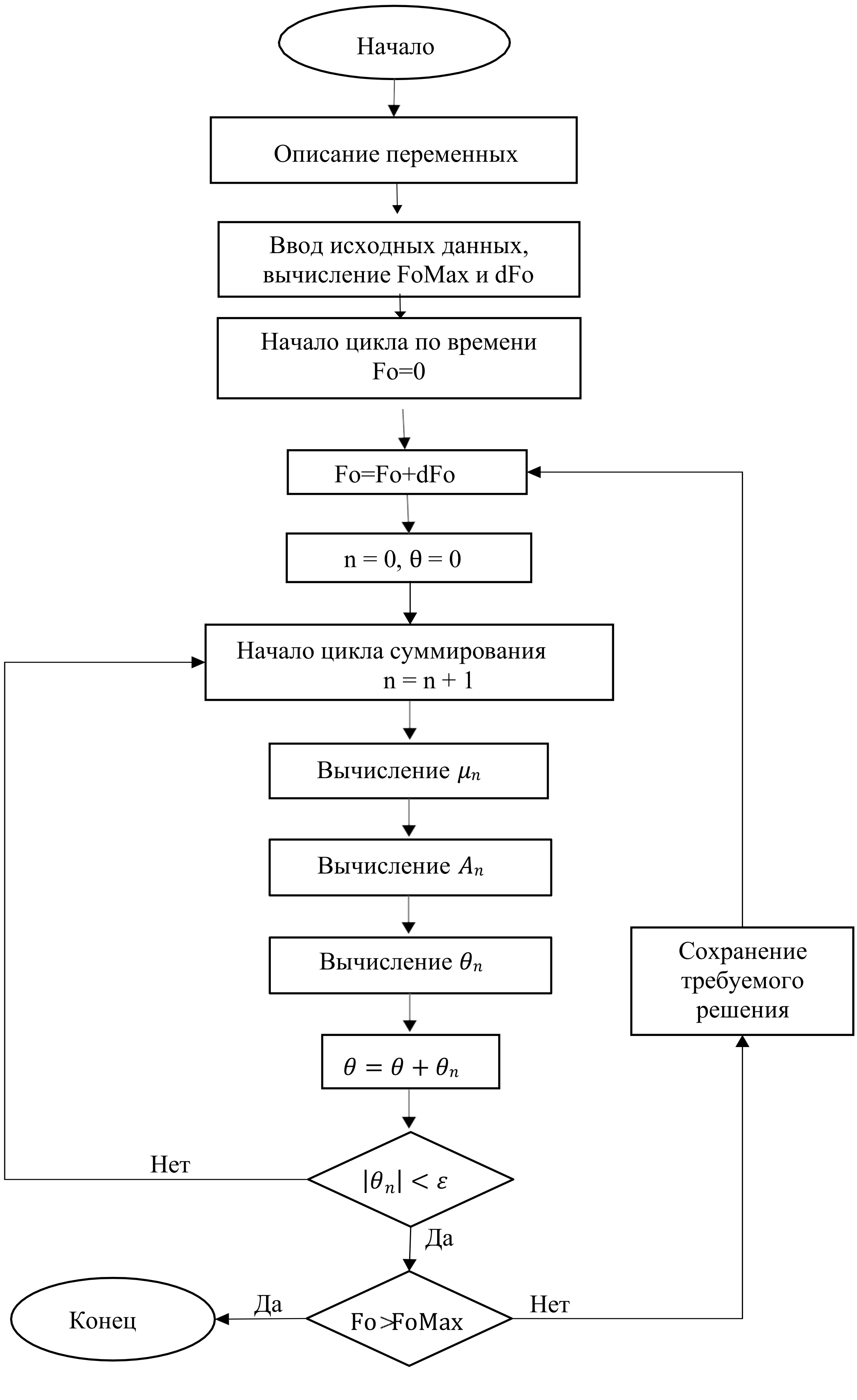
Общее решение $\theta = f(Fo, X)$ есть сумма всех частных решений, представляющая собой быстро сходящийся ряд:

$$\theta = \sum_{n=1}^{+\infty} Ae^{-\mu_n^2 Fo}\cos(\mu X) \qquad (1)$$

где $A_n = \frac{2\sin(\mu_n)}{\mu_n + \sin(\mu_n)\cos(\mu_n)}$ – может быть определено из начального условия. Тогда полученное решение можно записать так:

$$\theta = \sum_{n=1}^{+\infty}\theta_n, \quad \text{где} \quad \theta_n = A_n e^{-\mu_n Fo}\cos(\mu_n X)$$

Блок-схема вычислительной программы (внешний цикл по Fo)



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 $\theta=0.102$; для X=0, Fo=3, Bi=0.5: $\theta=0.297$; для X=0, Fo=8, Bi=0.6: $\theta=0.02$; 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 ($x=0$; $x=\delta/4$; $x=\delta/2$) 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 $x=\delta/2$ испытывает резкий спад, а для $x=\delta/4$ и $x=0$ 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 ($t=0.1\cdot Fo_{max}$; $t=0.5\cdot Fo_{max}$; $t=0.9\cdot Fo_{max}$) 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 Fo_{max} температура в точках пластины приближается к значениям температуры в момент теплового равновесия.

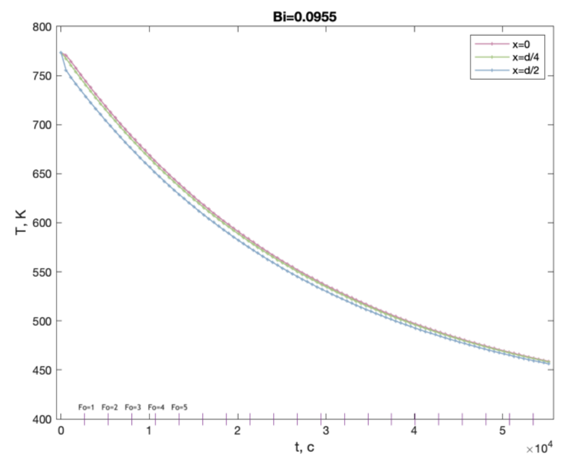


Рисунок 1
Зависимость температуры от
времени при Bi=0.095

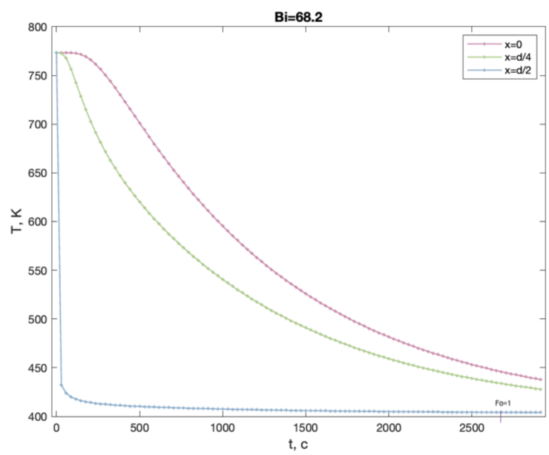


Рисунок 2
Зависимость температуры от
времени при Bi=68.2

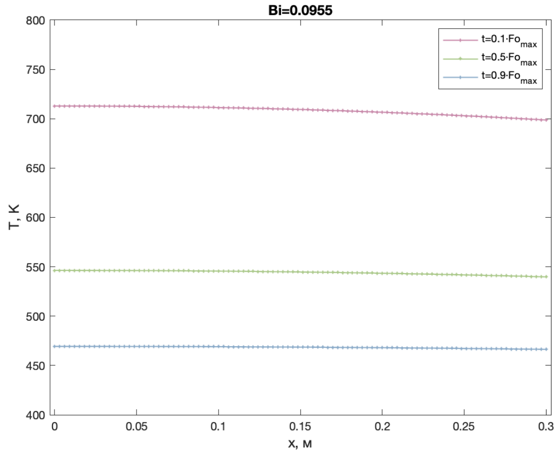


Рисунок 3
Зависимость температуры от координаты при Bi=0.095
Fo_{max} = 55752 с

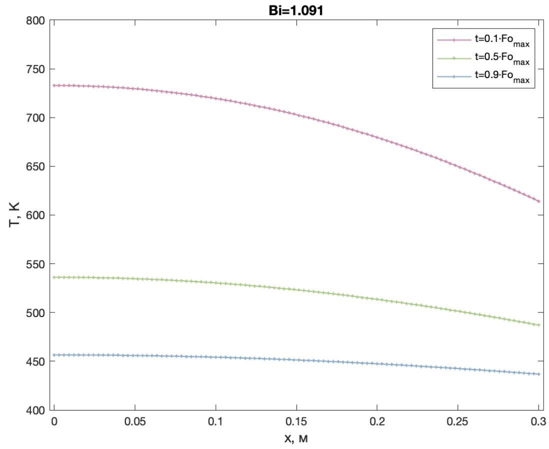


Рисунок 4
Зависимость температуры от координаты при Bi=1.09
Fo_{max} = 7750 с

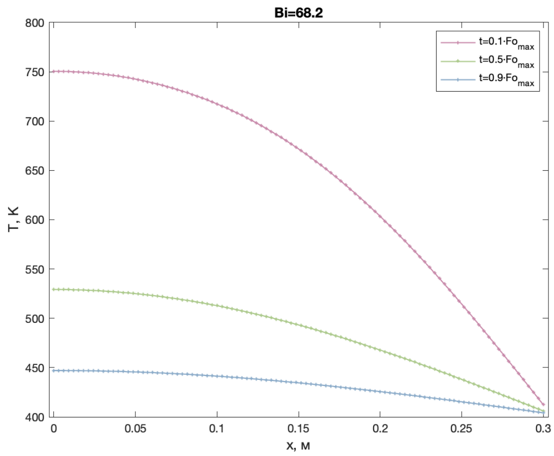


Рисунок 5
Зависимость температуры от координаты при Bi=68.2
Fo_{max} = 2941 с

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 $\varepsilon=10^{-12}$ 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 $t_{равн}$, в котором безразмерная температура в центре пластины достигает значения 0,1 (см. Приложение 4 и 5). Тогда:

При Bi = 0.0955: $t_{равн}$ = 67008 с

При Bi = 68.2: $t_{равн}$ = 2838 с

Приложение

1) 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
```

```fortran
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
```

2) 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
```

3) 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
```

4) 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
```

5) 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
```