iМИНИСТЕРСТВО 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 неявного метода конечных разностей

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](#_Toc166011056)

[1.1 Физическая постановка задачи 3](#_Toc166011057)

[1.2 Математическая постановка задачи 3](#_Toc166011058)

[3 Метод решения №3. Неявный метод конечных разностей 5](#_Toc166011059)

[4 Тестовый расчёт 11](#_Toc166011060)

[5 Результаты решения задачи 11](#_Toc166011061)

[5.1 Зависимости температуры от координаты 11](#_Toc166011062)

[5.2 Зависимости температуры от времени 13](#_Toc166011063)

[5.3 Влияние величины шага по времени на точность решения 14](#_Toc166011064)

[5.4 Сравнение с явным методом 14](#_Toc166011065)

[6 Выводы 15](#_Toc166011066)

[Приложения 16](#_Toc166011067)

# 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.

## 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 №3. Неявный метод конечных разностей

Основные идеи метода заключаются в следующем:

**1. Дискретизация.**

Область непрерывного изменения аргумента (координата X) разбивается на конечное число интервалов (или ячеек), в пределах каждого интервала размещается точка (узел), в которой задается значение искомой функции (температуры) для этого интервала. Совокупность узлов с упорядоченной нумерацией называется конечно-разностной сеткой.

Для безразмерных координат:

Аналогичная дискретизация выполняется для оси времени. В результате искомая непрерывная функция становится сеточной.

Где i – номер узла по координате; n – номер узла по времени

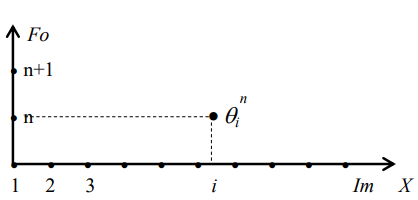


Рис 1. Расположение и нумерация узлов конечно разностной сетки.

Будем использовать равномерную сетку. Её параметры следующие:

**2. Аппроксимация**

Производные в безразмерном уравнении теплопроводности заменяются следующими конечно-разностными аналогами.

Уравнение теплопроводности в безразмерных переменных (4) будет иметь теперь вид.

Заметим, что в отличии от явного метода, производная по координате аппроксимируется на старшем временном слое, что позволяет снять ограничение на шаг по времени

Выражения для левой и правой границы получается из соответствующих ГУ (3) и (4) следующим образом. Запишем безразмерное уравнение теплопроводности, где за **q** обозначим тепловой поток

Построим конечно-разностную сетку и рассмотрим сеточную ячейку, примыкающую к левой границе расчетной области (на рис. показана штриховкой).

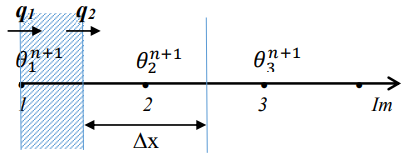


Рис 1. Расположение узлов и расчетных ячеек вблизи левой границы области.

Из рисунка видно, что ширина первой сеточной ячейки равна .

**q1** = 0 из ГУ (3)

**q2** =

Так как **q** = **q1** – **q2**, то после подстановки в уравнение (6) получим следующее выражение:

Аналогичные соображения для правой границы с использованием ГУ (4) приводят к выражению

**3. Формирование СЛАУ**

Сеточное уравнение (5) представим в виде

Для внутренних узлов (i = 2…Im-1)

Для левой границы (i = 1) из (7)

Для правой границы (i = Im) из (8)

Поскольку в МКР сетка имеет сквозную упорядоченную нумерацию узлов (в том же порядке будут располагаться и уравнения СЛАУ), то все ненулевые элементы матрицы коэффициентов [K], будут расположены на трех соседних диагоналях вдоль главной диагонали так, как показано на рисунке.

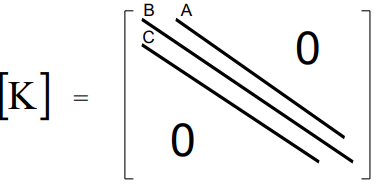


Рис 2. Схема ленточной трёхдиагональной матрицы коэффициентов СЛАУ

То есть матрица СЛАУ является трехдиагональной и для её решения целесообразно использовать метод прогонки, вычислительные затраты которого составляют

**4. Метод прогонки aka Thomas algorithm**

Решение системы уравнений

Будем искать в виде

После подстановки и перегруппировки находим следующие рекуррентные соотношения

Вспоминая, что на левой границе , а на правой границе получаем следующие соотношения соответственно:

Также следует учесть, что в нашем случае, когда все физические свойства постоянны, коэффициенты A, B и C зависят только от шагов сетки, поэтому их, в отличие от вектор-столбца D, достаточно рассчитать только один раз перед началом цикла по времени.

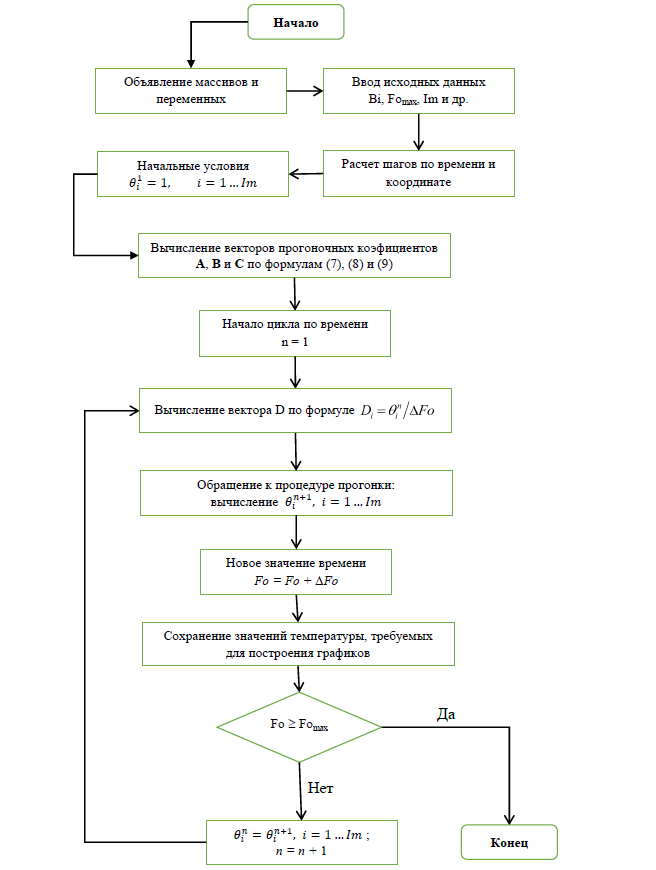


Рис 3. Блок схема вычислительного алгоритма

# 4 Тестовый расчёт

Тестовый расчёт проведём для третьего варианта (Bi = 90,9); в точке x=δ/4, Fo=0.28; полученный результат сравним с результатом метода Фурье.   
Параметры сетки: ,

|  |  |
| --- | --- |
| полученная методом Фурье | полученная методом конечных разностей |
| 307.5049 | 307.4118 |

Метод конечных разностей написан корректно. Точность численного решения составляет 3 значащие цифры.

# 5 Результаты решения задачи

Для различных вариантов коэффициента теплоотдачи c помощью написанной программы (приложение 1) получим результаты расчёта, которые представим в виде графиков, используя пакет SciDavis

Параметры используемой сетки:

## 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.

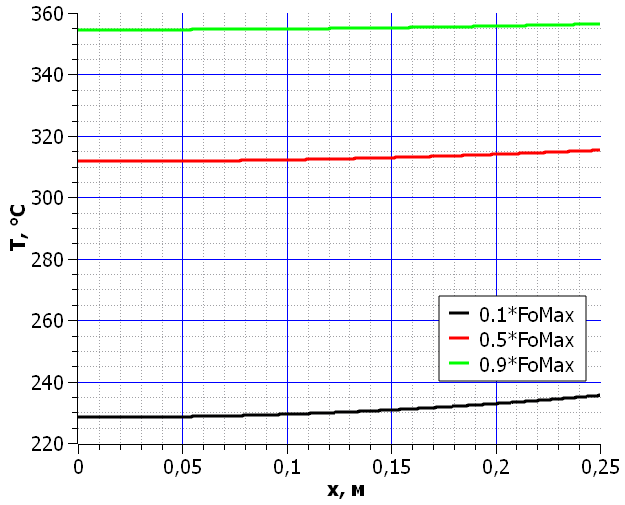


Рис 4. Зависимость температуры от координаты при Bi = 0,091; FoMax = 25,66

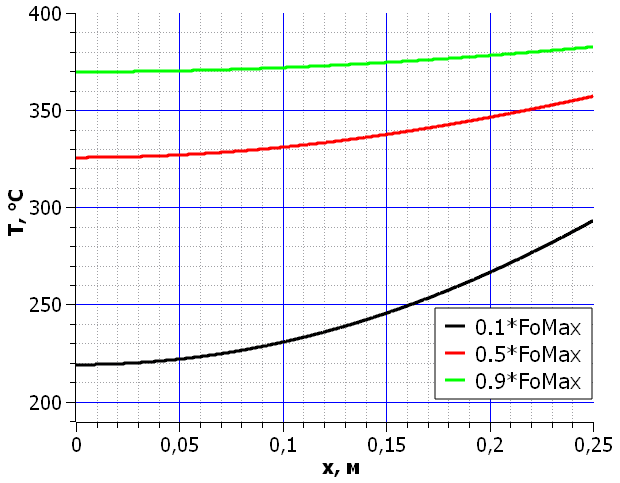


Рис 5. Зависимость температуры от координаты при Bi = 1,36; FoMax = 2,507

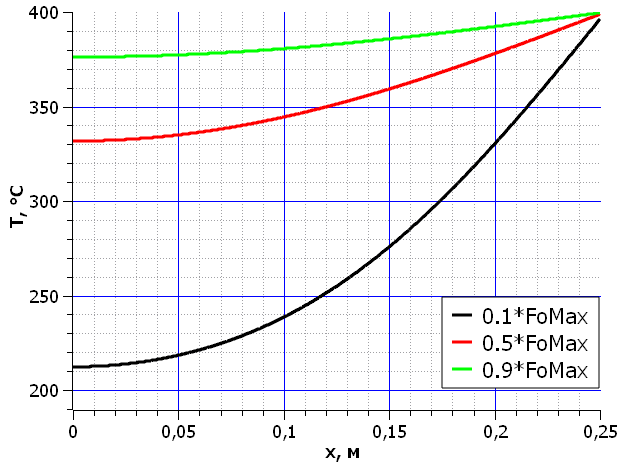


Рис 6. Зависимость температуры от координаты при Bi = 90,9; FoMax = 1,10

## 5.2 Зависимости температуры от времени

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

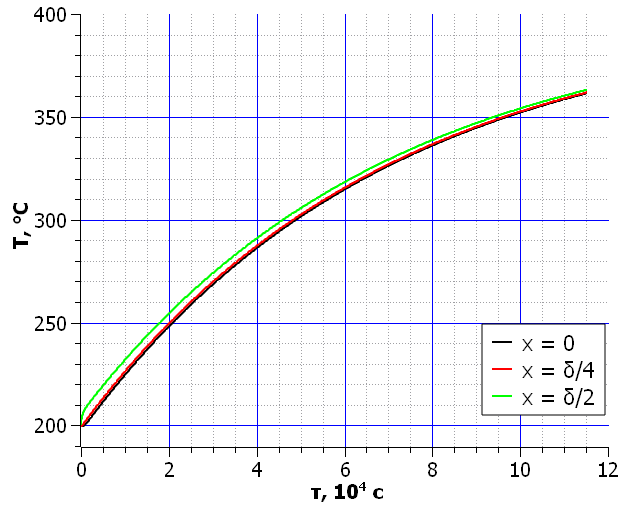


Рис 7. Зависимость температуры от времени в трех сечениях пластины при Bi = 0,091

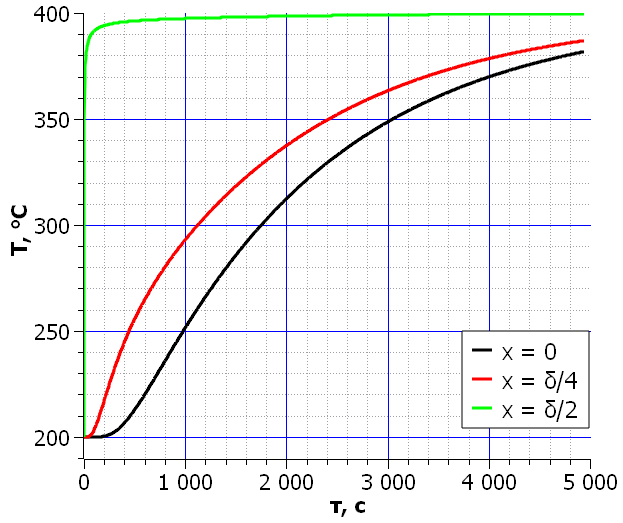


Рис 8. Зависимость температуры от времени в трех сечениях пластины при Bi = 90,9

## 5.3 Влияние величины шага по времени на точность решения

Проведём расчёт для третьего варианта (Bi = 90,9) в x= δ/4 при числе шагов по координате Im = 21, с разной величиной шага по времени (ΔFo) в фиксированный момент времени Fo = 0.50

Момент времени подобран так, чтобы он был кратен максимальной величине шага (100dFo = 0.125) и примерно равен 0.5\*Fomax = 0.55. Момент времени Fo был постоянен для всех расчётов с точностью не менее трёх значащих чисел.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Величина шага ΔFo | в методе Фурье | в неявном МКР | Ошибка | Относительная ошибка |
| 1 dFo | 323,849 | 323,704 | 0.145 | 0.04% |
| 5 dFo | 323,154 | 0.696 | 0.21% |
| 20 dFo | 321,143 | 2.706 | 0.84% |
| 100 dFo | 311,774 | 12.075 | 3.73% |

## 5.4 Сравнение с явным методом

Для третьего варианта (Bi = 90.9) проведём расчёт явным, неявным методами и сравним решения в x= δ/4, Fo = 0.5

Параметры сетки:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Число узлов | в методе Фурье | в явном МКР | в неявном МКР | Относительная ошибка явного МКР | Относительная ошибка неявного МКР |
| 11 | 351.868 | 356.445 | 351.456 | 1.301% | 0.117% |
| 21 | 354.139 | 351.765 | 0.645% | 0.029% |
| 41 | 353.002 | 351.842 | 0.322% | 0.007% |
| 81 | 352.435 | 351.859 | 0.161% | 0.003% |

# 6 Выводы

1. При фиксированном шаге по координате увеличение шага по времени в 4 раза ведёт к увеличению ошибки примерно в 4 раза.

Значит и неявный метод, как и явный, имеет первый асимптотический порядок точности для шага по времени

1. При числе Im = 11 точность решения неявным методом конечных разностей уже составляет 3 значащие цифры. При уменьшении шага по координате в 2 раза относительная ошибка падает в 4 раза, в то время как у явной версии метода ошибка падает в 2 раза – неявный метод оказывается точнее явного. Учитывая первое заключение, приходим также к выводу, что ошибка аппроксимации для неявного метода составляет:

Ссылки на литературу:

1. <http://thermalinfo.ru/Sets/cache/supercache/thermalinfo.ru/svojstva-materialov/metally-i-splavy/teploemkost-stali/index.html>

# Приложения

Приложение 1

Код программы

1 Program Main

program FDM\_implicit

use methods

implicit none

integer :: i,j, N, m(2), k, Xof(3)

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

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

real(8) :: time1, time2, treads

type(point), dimension(:), allocatable :: p\_c, p\_f, data1, data2, data3

real(8), dimension(:), allocatable :: A1, B1, C1

call CPU\_TIME(time1)

l = 55d0

d = 0.5d0

a(1) = 20d0

a(2) = 300d0

a(3) = 20000d0

T0 = 200d0 + 273.15d0

Te = 400d0 + 273.15d0

N = 80 !Im - 1

dX = 1d0/(N+0d0)

dFo = (dX\*\*2)\*0.5d0

p = 7900d0

c = 500d0

!Вычисление чисел Bio

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

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

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

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

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

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

write(\*,\*) FoMax(3)

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

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

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

Xof(1) = dint(xtoX(0d0, d, 1)/dX) + 1

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

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

!Нужно будет хранить один временной срез

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")

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

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

write(1,\*) "X", ";", "T(a1, Fo11)", ";", "T(a1, Fo12)", ";", "T(a1, Fo13)"

write(2,\*) "X", ";", "T(a2, Fo21)", ";", "T(a2, Fo22)", ";", "T(a2, Fo23)"

write(3,\*) "X", ";", "T(a3, Fo31)", ";", "T(a3, Fo32)", ";", "T(a3, Fo33)"

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

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

allocate(p\_c(N+1), p\_f(N+1), data1(N+1), data2(N+1), data3(N+1), A1(N+1), B1(N+1), C1(N+1))

call set\_ics(dX, dFo, 0d0, 1d0, p\_c, N, A1, B1, C1, Bi(1))

do while(p\_c(1) % Fo <= FoMax(1) + dFo)

call FDM\_imp(dFo, dX, p\_c, p\_f, Bi(1), N, A1, B1, C1) !Расчитали будущий на основе текущего

call add\_data(data1, data2, data3, p\_c, N, Fof(1,:), dFo, T0, Te, d)

write(4, 20) tautoFo(p\_c(1) % Fo, l, d, c, p, -1), ';', TtoTheta(p\_c(Xof(1)) % Theta, T0, Te, -1), ';',TtoTheta(p\_c(Xof(2)) % Theta, T0, Te, -1), ';',TtoTheta(p\_c(Xof(3)) % Theta, T0, Te, -1)

p\_c = p\_f

end do

do i = 1, N+1

write(1, 10) data1(i) % X,';', data1(i) % Theta,';', data2(i) % Theta,';', data3(i) % Theta

end do

call set\_ics(dX, dFo, 0d0, 1d0, p\_c, N, A1, B1, C1, Bi(2))

do while(p\_c(1) % Fo <= FoMax(2) + dFo)

CALL FDM\_imp(dFo, dX, p\_c, p\_f, Bi(2), N, A1, B1, C1)

call add\_data(data1, data2, data3, p\_c, N, Fof(2,:), dFo, T0, Te, d)

p\_c = p\_f

end do

do i = 1, N+1

write(2, 10) data1(i) % X,';', data1(i) % Theta,';', data2(i) % Theta,';', data3(i) % Theta

end do

call set\_ics(dX, dFo, 0d0, 1d0, p\_c, N, A1, B1, C1, Bi(3))

do while(p\_c(1) % Fo <= FoMax(3) + dFo)

CALL FDM\_imp(dFo, dX, p\_c, p\_f, Bi(3), N, A1, B1, C1)

call add\_data(data1, data2, data3, p\_c, N, Fof(3,:), dFo, T0, Te, d)

write(5, 20) tautoFo(p\_c(1) % Fo, l, d, c, p, 0), ';', TtoTheta(p\_c(Xof(1)) % Theta, T0, Te, -1), ';',TtoTheta(p\_c(Xof(2)) % Theta, T0, Te, -1), ';',TtoTheta(p\_c(Xof(3)) % Theta, T0, Te, -1)

p\_c = p\_f

end do

do i = 1, N+1

write(3, 10) data1(i) % X,';', data1(i) % Theta,';', data2(i) % Theta,';', data3(i) % Theta

end do

close(1)

close(2)

close(3)

deallocate(p\_c, p\_f, data1, data2, data3, A1, B1, C1)

call CPU\_TIME(time2)

write(\*,\*) dabs(time1 - time2)

pause

end program FDM\_implicit2 Module Methods

module methods

implicit none

type :: point

real(8) :: X, Fo, Theta

end type

contains

subroutine set\_ics(dX, dFo, Fo, Theta, p\_c, N, A, B, C, Bi)

type(point), dimension(:) :: p\_c

real(8) :: dX, dFo, Fo, Theta, Bi

real(8), dimension(:) :: A, B, C

integer :: i, N

do i = 1, N+1

p\_c(i) % X = dX\*(i-1d0)

p\_c(i) % Fo = 0d0

p\_c(i) % Theta = 1d0

end do

A(1) = 2d0/(dX\*\*2)

A(2:N) = 1d0/(dX\*\*2)

A(N+1) = 0d0

B(1:N) = -2d0/(dX\*\*2) - 1d0/dFo

B(N+1) = -2d0/(dX\*\*2) - 1d0/dFo - 2d0\*Bi/(dX)

C(1) = 0d0

C(2:N) = A(2:N)

C(N+1) = 2d0/(dX\*\*2)

end subroutine

subroutine add\_data(d1, d2, d3, p\_c, N, Fof, dFo, T0, Te, d)

type(point), dimension(:) :: p\_c, d1, d2, d3

real(8), dimension(:) :: Fof

real(8) dFo, T0, Te, d

integer :: i, N

if (dabs(p\_c(1) % Fo - Fof(1)) <= dFo) then

d1 % Fo = Fof(3)

do i = 1, N+1

d1(i) % Theta = TtoTheta(p\_c(i) % Theta, T0, Te, -1)

d1(i) % X = xtoX(p\_c(i) % X, d, -1)

end do

else if (dabs(p\_c(1) % Fo - Fof(2)) <= dFo) then

d2 % Fo = Fof(2)

do i = 1, N+1

d2(i) % Theta = TtoTheta(p\_c(i) % Theta, T0, Te, -1)

d2(i) % X = xtoX(p\_c(i) % X, d, -1)

end do

else if (dabs(p\_c(1) % Fo - Fof(3)) <= dFo) then

d3 % Fo = Fof(3)

do i = 1, N+1

d3(i) % Theta = TtoTheta(p\_c(i) % Theta, T0, Te, -1)

d3(i) % X = xtoX(p\_c(i) % X, d, -1)

end do

end if

end subroutine

subroutine FDM\_imp(dFo, dX, p\_c, p\_f, Bi, N, A, B, C)

real(8), allocatable :: D(:), F(:)

real(8), dimension(:) :: A, B, C

type(point), dimension(:) :: p\_c, p\_f

real(8) :: dFo, dX, Bi

integer :: i, N

allocate(D(N+1), F(N+1))

D = (p\_c % theta)/dFo

call Progonka(N+1, A, B, C, D, F)

p\_f % Fo = p\_c % Fo + dFo

p\_f % X = p\_c % X

p\_f % Theta = F

deallocate(D, F)

end subroutine

subroutine Progonka(Im, A, B, C, D, F) ! Метод прогонки

implicit none

integer, intent(in):: Im

real(8), dimension(1:Im), intent(in):: A, B, C, D

real(8), dimension(1:Im), intent(out):: F

real(8), dimension(1:Im):: alpha, beta

real:: k0

integer::i

!Прямой ход

alpha(1) = -A(1) / B(1)

beta(1) = -D(1) / B(1)

do i = 2, (Im-1)

k0 = (B(i) + C(i)\*alpha(i-1))

alpha(i) = -A(i) / k0

beta(i) = -(D(i) + C(i)\*beta(i-1)) / k0

end do

!Обратный ход

F = 0d0

F(Im) = -(D(Im) + C(Im)\*beta(Im-1)) / (B(Im) + C(Im)\*alpha(Im-1))

do i = (Im-1), 1, -1

F(i) = alpha(i)\*F(i+1) + beta(i)

end do

end subroutine

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 if (m == -1) then

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

else

Theta = T

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 if (m==-1) then

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

else

Fo1 = tau

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