УДК 519.6 **Технические науки**

Ильичев Владимир Юрьевич, *к.т.н., доцент кафедр «Тепловые двигатели и гидромашины» и «Мехатроника и робототехнические системы» Калужский филиал ФГОУ ВО «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет), г. Калуга, Россия*

Юрик Елена Алексеевна, *к.т.н., доцент кафедры «Тепловые двигатели и гидромашины», Калужский филиал ФГОУ ВО «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет), г. Калуга, Россия*

Медов Данила Сергеевич, *студент кафедры «Тепловые двигатели и гидромашины», Калужский филиал ФГОУ ВО «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет), г. Калуга, Россия*

РЕШЕНИЕ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ В ЧАСТНЫХ ПРОИЗВОДНЫХ С ИСПОЛЬЗОВАНИЕМ ФУНКЦИЙ ЯЗЫКА JULIA

Аннотация: Статья посвящена разработке методики решения, часто применяемых в научных исследованиях дифференциальных уравнений в частных производных. Обзор подходящих для этого программных продуктов позволил при проведении работы остановиться на оптимальном по всем параметрам (бесплатность, скорость вычислений, предоставляемые возможности), специально предназначенном для решения математических задач языке программирования Julia, в последнее время стремительно развивающемся. Визуализацию результатов решения дифференциальных уравнений было решено производить с помощью интеграции Julia с графическими библиотеками популярнейшего языка Python. Создан достаточно простой по структуре код программы, апробированный при рассмотрении

задачи теплопроводности однородного стержня, результаты решения которой позволили наглядно продемонстрировать влияние начальных параметров на протекание исследуемого процесса. По результатам работы сформулированы выводы.

Ключевые слова: дифференциальные уравнения, частные производные, теплопроводность, язык Julia, библиотеки Python.

Annotation: The article is devoted to the development of a methodology for solving partial differential equations often used in scientific research. A review of suitable software products made it possible to focus on the optimal in all parameters (free of charge, speed of calculations, opportunities provided), specially designed to solve mathematical problems of the Julia programming language, which has recently been rapidly developing. It was decided to visualize the results of solving differential equations using the integration of Julia with the graphic libraries of the most popular Python language. A fairly simple program code was created, tested when considering the thermal conductivity problem of a homogeneous rod, the results of which made it possible to clearly demonstrate the influence of the initial parameters on the course of the investigated process. Based on the results of the work, conclusions are formulated.

Keywords: differential equations, partial derivatives, thermal conductivity, Julia language, Python libraries.

Введение

Многие сложные физические процессы в современной науке описываются в виде дифференциальных уравнений в частных производных [9]. К таким процессам относятся, например, распространение тепла в окружающей среде или эпидемий в популяциях, протекание химической диффузии, движение элементов сложных робототехнических систем [3] и многие другие. Существует также множество направлений математики, использующих названные выше системы уравнений для моделирования в особых областях

исследований. Некоторыми из таких наиболее актуальных направлений являются топология [14], комплексный и функциональный анализ.

Описав процесс уравнениями указанного класса, можно создать модель изменения его параметров с течением времени. Сложность такого моделирования состоит в том, что дифференциальные уравнения в частных производных поддаются точному решению только в простейших случаях, однако к настоящему времени разработаны «продвинутые» методы их численного решения с использованием компьютерной техники (называемые методами математической физики) [1].

Существуют программные продукты, имеющие встроенные средства решения вышеупомянутых задач (MathCAD [10], MatLab [7]), но они являются коммерческими (и обладают достаточно высокой стоимостью), имеют весьма ограниченные возможности и характеризуются малой скоростью работы.

Однако, для решения весьма сложных математических задач к настоящему времени создан свободно распространяемый высокоуровневый и высокопроизводительный язык программирования Julia [8]. Описание задачи с его помощью отличается простотой, из-за чего данный язык называют «научным калькулятором». Кроме того, язык Julia стремительно развивается, в нём реализуются новые математические методы, и он может взаимодействовать с другим очень распространённым и мощным языком Python [4] (использовать его библиотеки функций, например, предназначенные для работы с массивами или для визуализации результатов).

Следует отметить, что для решения дифференциальных уравнений в частных производных необходимо задать функцию исследуемой величины (скорости, температуры, давления или другой) от переменных, которыми обычно являются координаты (одна, две или три) и время. Также задаются начальные условия, с которых начинается расчёт, и граничные условия, накладывающие ограничения на область допустимых решений (определяющие область пространства, в которой протекает исследуемый процесс). В результате составленные уравнения содержат частные производные исследуемой

величины от названных переменных. Это объясняется тем, что в каждой локальной точке пространства с течением времени значение искомой величины изменяется.

Целью данной работы являлась разработка методики последовательного выполнения указанных выше операций с помощью программы на языке Julia. Для апробации методики необходимо рассмотреть решение одной из задач математической физики, для описания которой используется дифференциальное уравнение в частных производных.

Материал и методы исследования

Пакет для использования языка программирования Julia можно бесплатно скачать с сайта [13]. Существуют версии для 32-х и 64-х битных платформ различных операционных систем: Windows, macOS, Linux, FreeBSD. Установив языковой пакет, необходимо дополнительно скачать некоторые библиотеки языка Python: PyCall [12] – для взаимодействия Julia с установленным пакетом Python, PyPlot [5] и Plots – для использования функций создания графических изображений.

Теперь рассмотрим функции Julia, используемые в разработанной методике решения систем дифференциальных уравнений в частных производных в порядке их применения в программном продукте:

1. Подключаются перечисленные выше графические библиотеки Python с помощью функции *using*.

2. Определяются начальные (функция *startcond*) и граничные (функция *bordrcond*) условия. В описываемой программе начальными условиями является описание вычисляемой функции до начала исследуемого процесса, а граничными условиями – пространственные координаты, за которые процесс распространяться не может.

3. Задаются значения постоянных коэффициентов, используемых при описании исследуемого процесса.

4. С помощью команды *function* создаётся блок программы, описывающий основную функцию, позволяющей найти значение искомой

физической величины в заданной точке пространства в заданный момент времени. В созданном коде программы основная функция представляет из себя математическую запись разностной аппроксимации уравнения Эйлера, позволяющей при каждой последующей итерации найти значение основной функции, используя её значение, полученное при предыдущей итерации.

Так как реализуется итерационная схема решения, в описываемый блок программы помещаются начальные условия и определяются значения окончания вычислений по времени и координатам, которые разбиваются на заданное количество интервалов. Количество интервалов равно числу циклических повторений вычислений основной функции (числу итераций).

В данном блоке программы учитываются также и граничные условия (значение исследуемой величины на границах рассматриваемого пространства).

5. Выполняется созданный в п. 4 блок программы со следующими заданными для него аргументами: число итераций по осям координат (в данном случае используется только одна координата x) и по времени, а также время окончания исследуемого процесса. Значение полученной для каждой итерации по координате и по времени функции записывается в массив.

7. С использованием подключённых к программе графических библиотек Python результаты вычисления искомой функции визуализируются.

Созданный код программы подходит для решения различных одномерных задач, описываемых дифференциальными уравнениями в частных производных.

Пример расчёта

В качестве примера рассмотрим решение уравнения теплопроводности параболического типа (по одной координате) [2]. С его помощью можно описать распространение тепла в однородном тонком стержне (рис. 1) с течением времени. Данное уравнение содержит частные производные первого порядка по времени t и второго – по координате x.

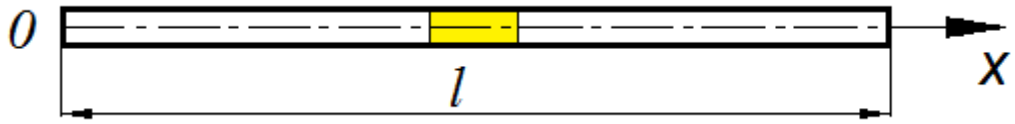

Рис. 1. Модель стержня, теплопроводность которого исследуется

Начало координат совпадает с началом стержня, ось x совпадает с осью стержня, а длина стержня l принята равной 1. Также сделано предположение, что стержень является теплоизолированным (отсутствуют потери тепла в окружающую среду). Начальным условием является нагревание стержня на участке x=0,45…0,55 (обозначен жёлтым цветом на рис. 1), во всех остальных участках температура равна нулю. После нагревания источник тепла отключается, после чего необходимо проследить распространение тепла вдоль стержня (по x) в разные моменты времени t.

Уравнение теплопроводности для рассматриваемого случая выглядит следующим образом:

$$\frac{\partial T(x,t)}{\partial t} = \frac{k}{c\rho}\frac{\partial^2 T(x,t)}{\partial x^2},$$

где $T(x,t)$ - искомая функция зависимости температуры от координаты и времени;

k, c, ρ – коэффициент теплопроводности, удельная теплоёмкость и плотность материала стержня. Отношение $\frac{k}{c\rho}$ называют коэффициентом тепловой диффузии.

Граничными условиями являются: $T(x,t)\big|_{x=0} = 0$ и $T(x,t)\big|_{x=l} = 0$.

Начальное условие: $T(x,t)\big|_{x=0.45...0.55,\,t=0} = 1$.

Подставим в код программы сформулированные условия, зададимся коэффициентом тепловой диффузии, равным 0,5 и максимальным рассматриваемым временем, равным 0,007 с. Примем также, что данный промежуток времени t и длина стержня l разбиваются на 50 интервалов

(следовательно, программа должна выполнить 50^2=2500 циклов расчёта основной функции). После этого запустим расчёт, результат которого выводится в графическом виде двумя способами и представлен на рис. 2.

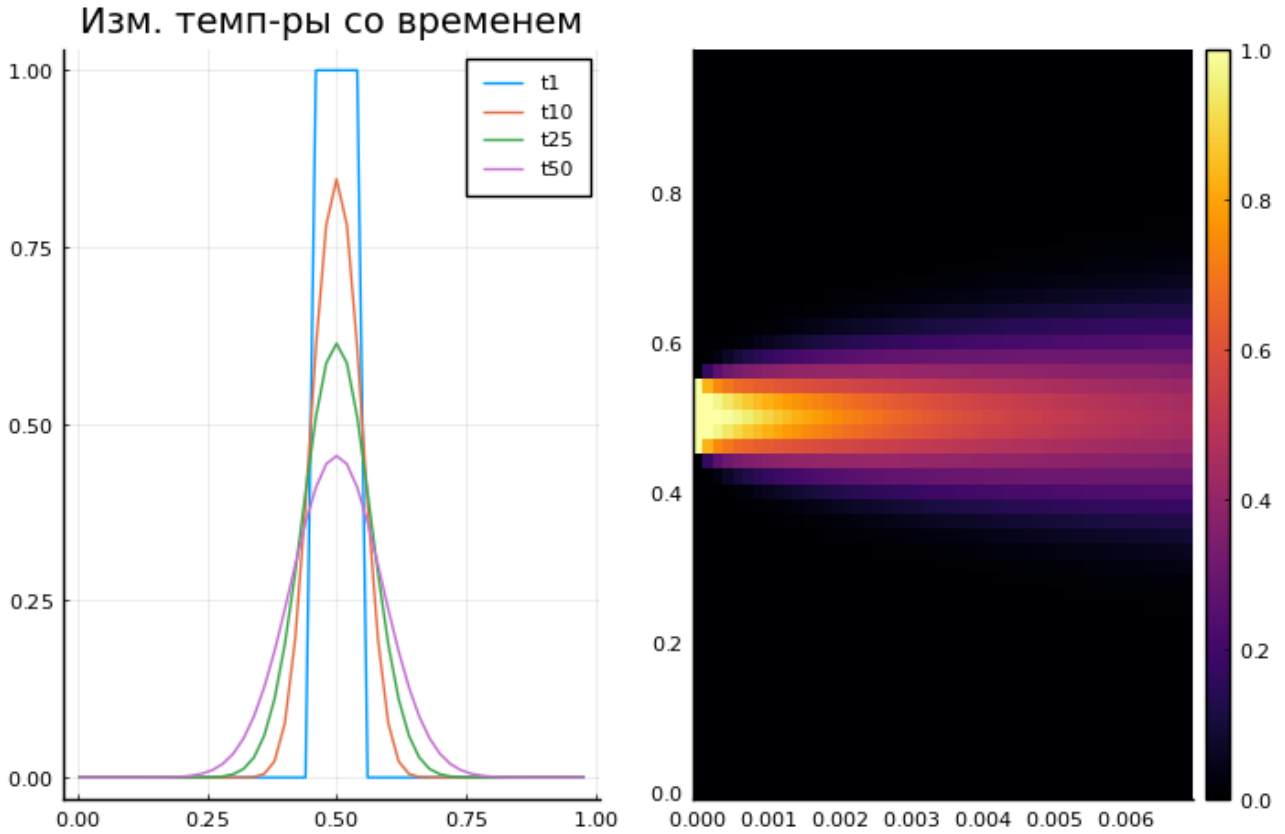

Рис. 2. Результат расчёта задачи теплопроводности стержня при коэффициенте тепловой диффузии, равном 0,5.

На левом графике рис. 2 по оси абсцисс отложена координата *x* стержня, а по оси ординат – относительная температура. По представленным кривым видно, как меняется распределение температуры по длине стержня после 1-й, 10-й, 25-й и 50-й (соответствующей *t*=0,007 с) итераций. Из данной картины можно сделать вывод, что за промежуток времени *t* средняя часть стержня остынет более, чем в 2 раза, за счёт теплопередачи в соседние участки, в результате чего на участке *x*=0,25…0,75 температура повысится.

Правый график называется тепловой картой (heatmap) [11], по оси абсцисс которой откладывается время *t*, с, а по оси ординат – координата *x* стержня. Для отображения температуры используется цветовая схема viridis

библиотеки PyPlot (представлена в правой части тепловой карты), согласно которой максимальное значение температуры обозначается ярко-жёлтым цветом, а нулевое – чёрным. Таким образом, представление данных в виде тепловой карты наглядно показывает изменение температуры в различных областях стержня с течением времени.

Для сравнения произведён также расчёт с коэффициентом тепловой диффузии, равным 1. Результаты представлены на рис. 3.

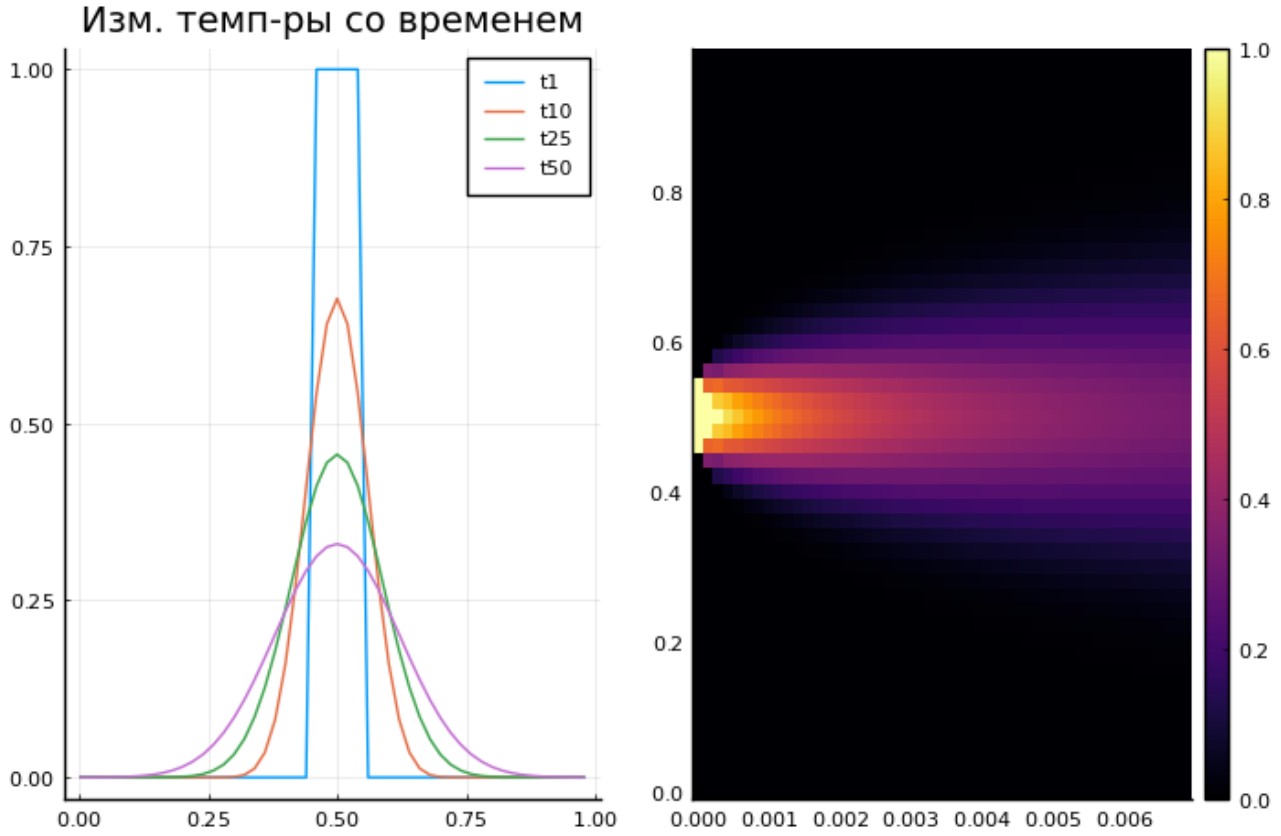

Рис. 3. Результат расчёта задачи теплопроводности стержня при коэффициенте тепловой диффузии, равном 1.

Как видно из рисунка, по сравнению с предыдущим примером, теплопередача происходит интенсивнее, и за тот же промежуток времени средняя часть стержня успевает остыть не в 2, а примерно в 3 раза, в то время как передача тепла происходит в больших пределах по оси x, практически достигая концов стержня.

Заключение

Таким образом, цель работы достигнута: обоснован выбор языка программирования Julia для решения дифференциальных уравнений в частных производных, описан алгоритм созданной для этого программы и основных применяемых в ней команд и библиотек функций. Алгоритм программы представляет собой метод, позволяющий решать разные задачи, описываемые приведённым классом уравнений.

Приведён пример решения одной из таких задач – исследование теплопроводности однородного стержня. Представлено уравнение, характеризующее зависимость температуры в разных частях стержня от времени, а также граничные и начальные условия задачи. Процесс передачи тепла по материалу стержня рассчитан с помощью, созданной на Julia программы для двух вариантов исходных данных. Результаты визуализированы в виде графиков и диаграмм, наглядно показывающих изменение температуры в зависимости от координаты и от времени.

Код разработанной программы может являться основой для решения любых дифференциальных уравнений в частных производных, а также может являться основой для изучения основ современнейшего языка программирования Julia и организации его интеграции со средствами, предоставляемыми другим популярным языком Python [6].

Библиографический список:

1. Базанова С.В. Математическая физика как теория математических моделей физических явлений. // В сборнике: XVII Царскосельские чтения. Материалы международной научной конференции. 2013. С. 152-155.

2. Бродская Т.А. Уравнение теплопроводности. Вывод уравнения для стержня. // Ученые записки Альметьевского государственного нефтяного института. 2010. Т. 8. С. 330-332.

3. Ильичев В.Ю. Создание скриптов Python для управления роботами в симуляторе программы FreeCAD. // Заметки ученого. 2021. № 11-1. С. 181-184.

4. Ильичев В.Ю., Качурин А.В. Создание программ на языке Python для

исследования множества Мандельброта. // E-Scio. 2021. № 5 (56). С. 362-371.

5. Ильичев В.Ю., Качурин А.В. Создание программ на языке Python для исследования множества Мандельброта. // E-Scio. 2021. № 5 (56). С. 362-371.

6. Ильичев В.Ю. Использование алгоритма дифференциальной эволюции для решения оптимизационных задач. // Системный администратор. 2021. № 4 (221). С. 80-83.

7. Использование MatLab. Решение дифференциальных уравнений в частных производных. PDE Toolbox. [Электронный ресурс]. URL: http://geometry.karazin.ua/resources/documents/20140425101823_05e9c091f50f.pdf (Дата обращения 12.02.2022).

8. Кулябов Д.С., Королькова А.В. Компьютерная алгебра на Julia. // Программирование. 2021. № 2. С. 44-50.

9. Мухамбетова А.А. Исследование решений квазилинейной системы уравнений в частных производных первого порядка. // The Scientific Heritage. 2019. № 42-1 (42). С. 20-24.

10. Решатель дифференциальных уравнений в частных производных. [Электронный ресурс]. URL: http://support.ptc.com/help/mathcad/r7.0/ru/index.html#page/PTC_Mathcad_Help/pde_solver.html (Дата обращения 12.02.2022).

11. Хасенова З.Т. Создание тепловой карты для визуализации данных. // Фундаментальные и прикладные исследования в современном мире. 2017. № 20-1. С. 24-26.

12. Шадрин Ю.А., Грюмова Е.А., Гумирова А.И. Обзор языка программирования Julia. // Технологии инженерных и информационных систем. 2019. № 3. С. 43-52.

13. Download Julia. [Электронный ресурс]. URL: https://julialang.org/downloads/ (Дата обращения 12.02.2022).

14. Ilyichev V.Yu. Numerical implementation of Poincaré recurrence theory using Arnold mappings. // International Research Journal. 2021. № 6-1 (108). С. 90-94.