DOI: 10.15514/ISPRAS-2023-35(5)-16

Применение физически-обоснованной нейронной сети на примере моделирования гидродинамических процессов, допускающих аналитическое решение

[1] *К.Б. Кошелев,* ORCID: 0000-0002-7124-3945 <*k.koshelev@ispras.ru*>
[2] *С.В. Стрижак,* ORCID: 0000-0001-5525-5180 <*s.strijhak@ispras.ru*>

[1] *Институт водных и экологических проблем СО РАН,*
656038, Алтайский край, г. Барнаул, ул. Молодежная, д.1.
[2] *Институт системного программирования им. В.П. Иванникова РАН,*
109004, Россия, г. Москва, ул. А. Солженицына, д. 25.

Аннотация. Рассматривается актуальный подход для разработки физически-обоснованной нейронной сети для решения модельных задач для течения Коважного, геофизического течения Бельтрами, течения на участке реки по теории мелкой воды. Физически-обоснованные нейронные сети (PINN) позволяют существенно сокращать время расчета по сравнению с обычными вычислениями. Для каждого модельного течения существует свое аналитическое решение. Обсуждается архитектура программной библиотеки DeepXDE, ее состав по модулям, приводятся фрагменты программного кода на языке программирования Python. Модель PINN протестирована на тестовой выборке. Оценка предсказания выполнена с помощью метрики MSE. Полносвязанная нейронной сеть может содержать в себе 4, 7,10 скрытых слоев с количеством нейронов 50, 50, 100 соответственно. Обсуждается влияние гиперпараметров нейронной сети на величину ошибки предсказания. Расчеты, выполненные на сервере с графической картой Nvidia GeForce RTX 3070, позволяют существенно сократить время обучения для PINN.

Ключевые слова: нейронная сеть; слои; нейроны; течение Коважного; течение Бельтрами; теория мелкой воды; сетка; канал; аналитическое решение; область; точки; обучение; ошибка.

Для цитирования: Кошелев К.Б., Стрижак С.В. Применение физически-обоснованной нейронной сети на примере моделирования гидродинамических процессов, допускающих аналитическое решение. Труды ИСП РАН, том 35, вып. 5, 2023 г., стр. 245–258. DOI: 10.15514/ISPRAS–2023–35(5)–16

Application of Physics-Informed Neural Network on the Example of Modeling Hydrodynamic Processes That Allow an Analytical Solution

[1] *K.B. Koshelev*, ORCID: 0000-0002-7124-3945 <*k.koshelev@ispras.ru*>
[2] *S.V. Strijhak,* ORCID: 0000-0001-5525-5180 <*s.strijhak@ispras.ru*>

[1] *Institute for water and environmental problems SB RAS,*
1, Molodezhnaya str., Altai region, Barnaul, 656038.
[2] *Ivannikov Institute for System Programming of the Russian Academy of Sciences,*
25, Alexander Solzhenitsyn st., Moscow, 109004, Russia.

Abstract. We consider an actual approach to develop a physically based neural network for solving model problems for the Kovazhny flow, for the geophysical Beltrami flow, and for the flow in a section of the river by the shallow water theory. Physics-informed neural networks (PINN) allow to significantly reduce the

computation time compared to conventional computations. There is a different analytical solution for each model flow. The architecture of the DeepXDE software library, its composition by modules, and fragments of program code in the Python programming language are discussed. The PINN model is tested on a test sample. The prediction is evaluated using the MSE metric. The fully connected neural network can contain 4, 7, 10 hidden layers with the number of neurons 50, 50, 100 respectively. The influence of hyperparameters of the neural network on the magnitude of the prediction error is discussed. The calculations performed on a server with Nvidia GeForce RTX 3070 card can significantly reduce the training time for PINN.

Keywords: neural network; layers; neurons; Kovasznay flow; Beltrami flow; Swallow Water Equations; grid; canal; analytical solution; domain; points; training; error.

For citation: Koshelev K.B., Strijhak S.V. Application of physics-informed neural network on the example of modeling hydrodynamic processes that allow an analytical solution. *Trudy ISP RAN/Proc. ISP RAS*, vol. 35, issue 5, 2023. pp. 245-258 (in Russian). DOI: 10.15514/ISPRAS-2023-35(5)-16.

1. Введение

В настоящее время существует большое количество различных вычислительных методов для решения задач гидродинамики, разработанных на протяжении последних пятидесяти лет, которые работают достаточно эффективно, если известны основные уравнения и все масштабы турбулентности разрешены.

За последние пять лет, было предпринято ряд попыток создать модели на базе нейронных сетей (NN) для решения задач несжимаемой жидкости в форме уравнений Навье-Стокса. Самый распространенный подход для изучения турбулентных течений заключался в построении моделей замыкания турбулентности с управляемыми данными.

Другое перспективное направление связано с развитием физически-обоснованных нейронных сетей, так называемых Physics-Informed Neural Network - PINN, для решения систем уравнений Эйлера и Навье-Стока.

Нейронные сети являются универсальными аппроксиматорами непрерывных функций. Универсальная теорема аппроксимации утверждает, что нейронные сети могут быть использованы для аппроксимации любой непрерывной функции с произвольной точностью, если не накладывать ограничений на ширину и глубину скрытых слоев. Теорема, доказанная математиком Джорджем Цыбенко в 1989 году, утверждает, что полносвязанная нейронная сеть с одним скрытым слоем может точно аппроксимировать любой нелинейный непрерывный оператор. Эта универсальная теорема аппроксимации наводит на мысль о потенциальном применении нейронных сетей для обучения нелинейных операторов на основе доступных данных. Однако теорема гарантирует лишь малую ошибку аппроксимации для достаточно большой сети, и не учитывает важные ошибки оптимизации и обобщения.

В настоящее время имеется необходимость в разработке, обосновании и практической реализации новых архитектур нейронных сетей для аппроксимации непрерывных функций.

Авторы из Brown University, США пошли путем, используя свойство универсальной аппроксимации нейронных сетей, которое вместе с автоматическим дифференцированием функции в рамках фреймворка для машинного обучения (TensorFlow, PyTorch) позволяет разрабатывать «решатели» для систем уравнений Эйлера, Навье-Стокса, которые не требуют генерации расчетной сетки.

В частности, в работах [1-2] авторы впервые ввели концепцию физически-обоснованных нейронных сетей (PINN) для решения прямых и обратных задач, включающих несколько различных типов уравнений в частных производных и обыкновенных дифференциальных уравнений, с использованием бессеточного метода.

2. Математическая модель

Уравнения, описывающие двумерные стационарные несжимаемые ламинарные течения, имеют следующий вид:

$$\nabla \cdot u = 0$$

$$\frac{\partial u}{\partial t} + \nabla(u^T u) - \frac{1}{Re}\Delta u = -\nabla p/\rho$$

Здесь u – векторная функция, представляющая скорость жидкости, p – скалярная функция давления жидкости, ρ- плотность, Re– число Рейнольдса.

Из курса гидродинамики известно, что существует ряд модельных краевых задач, для которых существуют аналитические решения. Это течение Коважного, течение Куэтта, вихрь Тейлора-Грина, геофизическое течение Бельтрами, ячейка Хеле-Шоу, всплытие пузырька в вязкой жидкости, перемешивание жидкости между двумя эксцентрично вращающимися круговыми цилиндрами [3-4].

3. Модель физически-обоснованной нейронной сети

В рамках данной работы рассматривается архитектура полносвязанной нейронной сети для построения физически-обоснованной нейронной сети для моделирования различных простейших течений. Для нейронной сети вводятся понятия: входной слой с нейронами с заданными на входе признаками в форме координат точек и дискретных значений времени, несколько скрытых слоев с нейронами, выходной слой с нейронами. Также рассматривается задание начальных и граничных условий, расчетной области с заданным облаком точек, и функции потерь, которая связана с уравнениями неразрывности, движения, с заданными граничными и начальными условиями. Дополнительно для нейронной сети выбирается функция активации нейронов и метод оптимизации для функции потерь.

Подобная физически-обоснованная нейронная сеть состоит из трех основных блоков (рис.1). Первая часть включает в себя модуль для вычисления остаточных слагаемых для дифференциальных уравнений в частных производных или относительную погрешность решения в норме L2, а также погрешности для начальных и граничных условий. Параметры для полносвязанной нейронной сети определяются путем нахождения минимума для общей функции потерь. Входы для нейронной сети преобразуются в соответствующие выходы. Вторая часть это полносвязанная нейронная сеть с физическими данными, которая берет выходные поля скоростей и вычисляет их производные, используя исходные уравнения для движения и неразрывности при решении задач механики жидкости. Также оцениваются граничные и начальные условия, данные наблюдений из эксперимента. Последним шагом является механизм формирования обратной связи, который минимизирует функцию потерь, используя заданный оптимизатор (Adam, L-BFGS-B), в соответствии с некоторой скоростью обучения, чтобы получить оптимальные параметры для нейронной сети [5].

Для вычисления невязок при решении системы уравнений Навье-Стокса для скорости и давления, частные дифференциальные операторы вычисляются с помощью процедуры автоматического дифференцировании (AutoDiff - AD), которое может быть непосредственно сформулировано в нейронных сетях глубокого обучения, например, используя оператор "tf.gradients()" в библиотеке TensorFlow [6]. Пример архитектуры физически-обоснованной нейронной сети для системы уравнений Навье-Стокса приведен на рис. 2.

4. Библиотека DeepXDE

Для разработки тематической PINN можно использовать открытую библиотеку DeepXDE. В библиотеку DeepXDE входят различные модули (рис. 3).

Для программной реализации нейронной сети и решения задачи течения Коважного, Бельтрами, в прямоугольном канале была использована открытая библиотека DeepXDE, разработанная на языке программирования Python, с использованием библиотек для машинного обучения TensorFlow, PyTorch [7-8]. Библиотека DeepXDE поддерживает следующий функционал:

- 5 типов граничных условий (1-го рода, 2-го рода, 3-го рода, другие);
- построение простейшей геометрии.

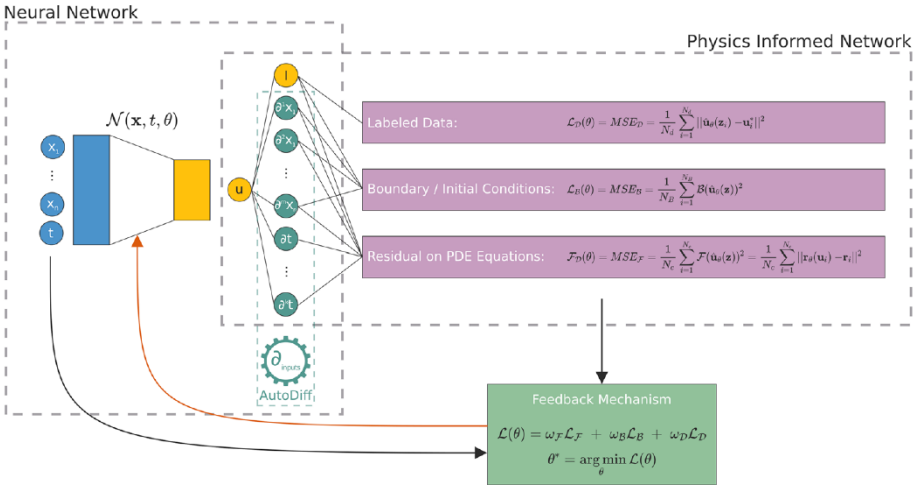

Рис.1 Общая схема для построения физически-обоснованной нейронной сети [5].
Fig. 1 The typical scheme for PINN architecture [5].

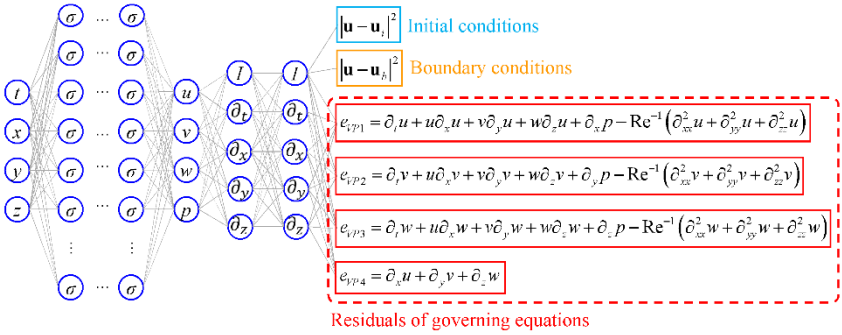

Рис. 2. Архитектура нейронной сети для системы уравнений Навье-Стокса [12].
Fig. 2. The architecture of neural network for Navier-Stokes system equations [12].

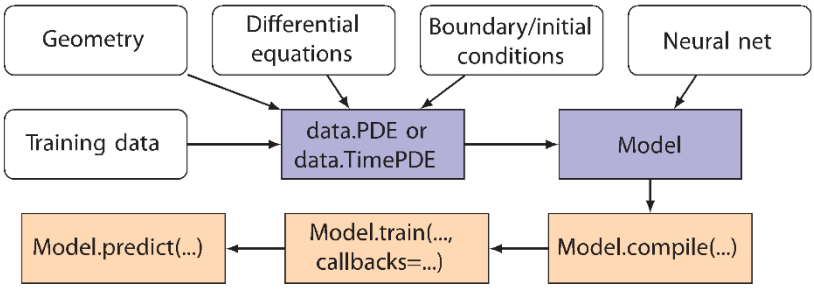

Рис.3 Состав библиотеки DeepXDE [7].
Fig.3 Structure of DeepXDE library [7].

Для этого имеется реализация работы с типовыми объектами: 1D: random_points, random_boundary_points, periodic_point, background_points, background_points_left, background_points_right; 2D: "Disk", "Polygon", "Rectangle", "Triangle"; 3D: Cuboid (Hypercube), Sphere (Hypersphere). Имеется поддержка булевых операций.

В DeepXDE реализованы различные методы для решателей NN: Fractional PDE решатель, IDE решатель, ODE решатель, time-independent PDE решатель. Имеется набор тестовых примеров (pinn_forward; pinn_inverse, другие). Функция потерь определена через задание величины ошибки MAE. Имеется поддержка работы с различными фреймворками для машинного обучения TensorFlow, PyTorch, Paddle, JAX.
Библиотека DeepXDE может быть настроена под Linux ОС для работы с графической картой Nvidia GPU с использованием дополнительных библиотек на языке CUDA.

5. Модельные течения, допускающие аналитические решения

5.1 Двумерное течение Коважного

Двумерное течение Коважного является стационарным течением за решеткой (сеткой) и имеет точное решение вида для функции скорости u(x,y), v(x,y), давления p(x) [9].

$$u(x,y) = 1 - e^{(-\lambda x)}cos(2\pi y)$$
$$v(x,y) = -\frac{\lambda}{2\pi}e^{(-\lambda x)}sin(2\pi y)$$
$$p(x) = -\frac{1}{2}e^{-2\lambda x}$$

Здесь параметр λ определяется соотношением:

$$\lambda = \sqrt{\frac{Re^2}{4} + 4\pi^2} - \frac{Re}{2}$$

Данная задача является хорошим тестом для проверки устойчивости численного решения и точного решения, чтобы продемонстрировать пространственную экспоненциальную скорость сходимости выбранного алгоритма [9-11].

5.1.1 Программная реализация

Приведем фрагмент программного кода на языке программирования Python.
Задание исходных величин:

```
Re = 20
nu = 1 / Re
l = 1 / (2 * nu) - np.sqrt(1 / (4 * nu ** 2) + 4 * np.pi ** 2)
```

Функция на Python для расчета уравнений для количества движения, неразрывности [8]:

```
def pde(x, u):
    u_vel, v_vel, p = u[:, 0:1], u[:, 1:2], u[:, 2:]
    u_vel_x = dde.grad.jacobian(u, x, i=0, j=0)
    u_vel_y = dde.grad.jacobian(u, x, i=0, j=1)
    u_vel_xx = dde.grad.hessian(u, x, component=0, i=0, j=0)
    u_vel_yy = dde.grad.hessian(u, x, component=0, i=1, j=1)
    v_vel_x = dde.grad.jacobian(u, x, i=1, j=0)
    v_vel_y = dde.grad.jacobian(u, x, i=1, j=1)
    v_vel_xx = dde.grad.hessian(u, x, component=1, i=0, j=0)
    v_vel_yy = dde.grad.hessian(u, x, component=1, i=1, j=1)
    p_x = dde.grad.jacobian(u, x, i=2, j=0)
    p_y = dde.grad.jacobian(u, x, i=2, j=1)
    momentum_x = (
        u_vel * u_vel_x + v_vel * u_vel_y + p_x - 1 / Re * (u_vel_xx + u_vel_yy)
    )
    momentum_y = (
        u_vel * v_vel_x + v_vel * v_vel_y + p_y - 1 / Re * (v_vel_xx + v_vel_yy)
    )
    continuity = u_vel_x + v_vel_y
```

```
    return [momentum_x, momentum_y, continuity]
```

Функция для задания аналитического решения для u (x,y):

```
def u_func(x):
    return 1 - np.exp(l * x[:, 0:1]) * np.cos(2 * np.pi * x[:, 1:2])
```

Функция для задания аналитического решения для v(x,y):

```
def v_func(x):
    return l / (2 * np.pi) * np.exp(l * x[:, 0:1]) * np.sin(2 * np.pi * x[:, 1:2])
```

Функция для задания аналитического решения для p(x):

```
def p_func(x):
    return 1 / 2 * (1 - np.exp(2 * l * x[:, 0:1]))
```

Функция для задания граничного условия для выходного условия:

```
def boundary_outflow(x, on_boundary):
    return on_boundary and np.isclose(x[0], 1)
```

Задание расчетной области и граничных условий:

```
spatial_domain = dde.geometry.Rectangle(xmin=[-0.5, -0.5], xmax=[1, 1.5])
boundary_condition_u = dde.icbc.DirichletBC(
    spatial_domain, u_func, lambda _, on_boundary: on_boundary, component=0
)
boundary_condition_v = dde.icbc.DirichletBC(
    spatial_domain, v_func, lambda _, on_boundary: on_boundary, component=1
)
boundary_condition_right_p = dde.icbc.DirichletBC(
    spatial_domain, p_func, boundary_outflow, component=2
)
```

Определение исходных данных для системы уравнений:

```
data = dde.data.TimePDE(
    spatial_domain,
    pde,
    [boundary_condition_u, boundary_condition_v, boundary_condition_right_p],
    num_domain=2601,
    num_boundary=400,
    num_test=100000,
)
```

Формирование модели для нейронной сети:

```
net = dde.nn.FNN([2] + 4 * [50] + [3], "tanh", "Glorot normal")
model = dde.Model(data, net)
```

В состав архитектуры нейронной сети вход один входной слой с двумя входами (координаты точек и время), 4 скрытых слоя, в каждом задано по 50 нейронов, выходной слой с 3 выходами (две компоненты скорости и давление).

5.1.2 Результаты решения задачи течения Коважного

Задача о течении Коважного, связанная с движением потока через решетку с заданными параметрами в аэродинамической трубе, решалась при Re=20 (рис. 4). Для данного течения не существует начального условия.

Рассматривалась расчетная область с размерами [-0.5; 1:0] x [-0.5; 1.5] и временной областью [0; 1]. На верхней и нижней границах задавалось условие симметрии. На левой границе задавалось условие Дирихле для значения скорости, основанное на точном решении.

На правой границе задавалось специальное граничное условие для выхода потока, которое было получено в работе [9]. Начальное значение для скорости полагалось равным 0.

На каждой границе прямоугольной области задавалась 101 точка с фиксированной пространственной координатой, т.е. Nb = 4 x 101. Для вычисления функции потерь в нейронной сети, случайным образом внутри домена выбиралась 2 601 точка.

В этой работе использовалась нейронная сеть с 4-мя скрытыми слоями и 50-ю нейронами на слой, с 7-ю скрытыми слоями и 50-ю нейронами на слой, с 10-ю скрытыми слоями и 100 нейронами на слой. Также применялся оптимизатор Адама для обеспечения наилучшего набора начальных обучаемых переменных для нейронной сети. Затем L-BFGS-B использовался для тонкой настройки нейронных сетей для достижения более высокой точности. Динамические веса обновлялись каждые 100 эпох обучения.

Анализировались градиенты функции потерь в зависимости от гиперпараметров нейронной сети. Относительные ошибки L2 могли достигать порядка 10^{-5}.

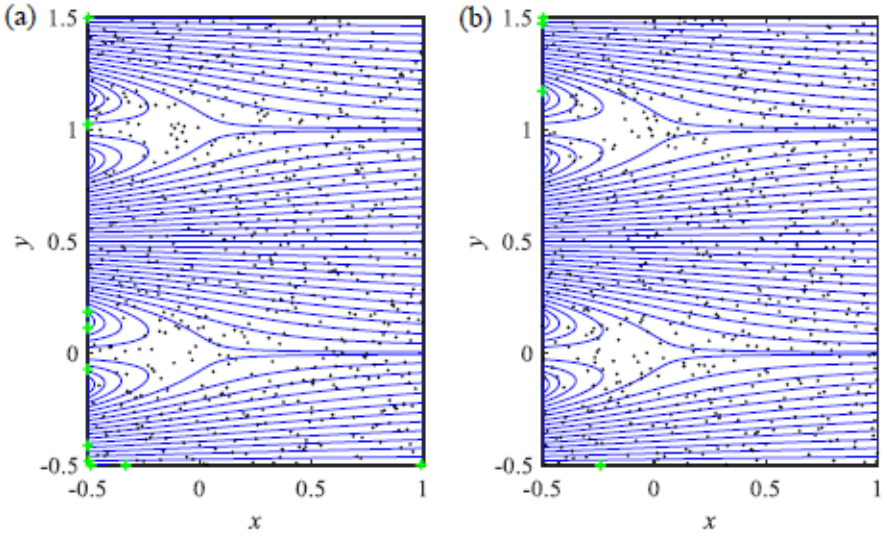

Рис. 4. Расчетная область для течения Коважного [12].
Fig. 4. Numerical domain for Kovasznay flow [12].

Обучение нейронной сети и процесс предсказания параметров течения выполнялся на ноутбуке с 32 GB RAM. Время обучения нейронной сети занимало от 30 минут до 4 часов.

Результатом выполнения программы является расчет среднеквадратичной ошибки и L2 относительной ошибки для двух компонент скорости и давления:

Mean residual: 0.0005657403
L2 relative error in u: 0.00019108094
L2 relative error in v: 0.0009619565
L2 relative error in p: 0.00042990936

5.2 Течение Бельтрами

Рассматривалась задача о нестационарном трехмерном геофизическом течении Бельтрами, для которого существует известное аналитическое решение в литературе [12].

Это специальный класс течения сплошной среды, в котором векторы скорости и завихренности коллинеараны. Для данного течения существует точное аналитическое решение, которое имеет вид:

$$u = -a[e^{ax}\sin(ay + dz) + e^{az}\sin(ax + dy)]e^{-d^2t}$$
$$v = -a[e^{ay}\sin(az + dx) + e^{ax}\sin(ay + dz)]e^{-d^2t}$$
$$w = -a[e^{az}\sin(ax + dy) + e^{ay}\sin(az + dx)]e^{-d^2t}$$

$$p = \frac{a^2}{2}\left[e^{ax} + e^{ay} + e^{az} + 2\sin(ax + dy)\cos(az + dx)e^{a(y+z)} + 2\sin(ay + dz)\cos(ax + dy)e^{a(z+x)} + 2\sin(az + dx)\cos(ay + dz)e^{a(x+y)}\right]$$

где a и d являются свободными параметрами.
Параметры уравнения: $Re = 1$, $a = 1$, $d = 1$
Для проверки работоспособности PINN использовались различные гиперпараметры (табл. 1).

Табл. 1. Гиперпараметры
Table 1. Hyperparameters

Скрытые слои	4 слоя × 100 нейронов
Первый оптимизатор	Adam, 30.000 итераций, 1e-3 скорость обучения
Второй оптимизатор	L-BFGS-B, 15.000 итераций
Точки коллокации	50000
Точки на границах	5000
Точки для начального условия	5000

5.2.1 Программная реализации

Приведем фрагмент программного кода на языке программирования Python.
Задание исходных величин:

```
a = 1
d = 1
Re = 1
```

Функция на Python для расчета уравнений неразрывности, количества движения для случая нестационарного трехмерного течения представлена ниже [8]:

```
def pde(x, u):
    u_vel, v_vel, w_vel, p = u[:, 0:1], u[:, 1:2], u[:, 2:3], u[:, 3:4]

    u_vel_x = dde.grad.jacobian(u, x, i=0, j=0)
    u_vel_y = dde.grad.jacobian(u, x, i=0, j=1)
    u_vel_z = dde.grad.jacobian(u, x, i=0, j=2)
    u_vel_t = dde.grad.jacobian(u, x, i=0, j=3)
    u_vel_xx = dde.grad.hessian(u, x, component=0, i=0, j=0)
    u_vel_yy = dde.grad.hessian(u, x, component=0, i=1, j=1)
    u_vel_zz = dde.grad.hessian(u, x, component=0, i=2, j=2)

    v_vel_x = dde.grad.jacobian(u, x, i=1, j=0)
    v_vel_y = dde.grad.jacobian(u, x, i=1, j=1)
    v_vel_z = dde.grad.jacobian(u, x, i=1, j=2)
    v_vel_t = dde.grad.jacobian(u, x, i=1, j=3)
    v_vel_xx = dde.grad.hessian(u, x, component=1, i=0, j=0)
    v_vel_yy = dde.grad.hessian(u, x, component=1, i=1, j=1)
    v_vel_zz = dde.grad.hessian(u, x, component=1, i=2, j=2)

    w_vel_x = dde.grad.jacobian(u, x, i=2, j=0)
    w_vel_y = dde.grad.jacobian(u, x, i=2, j=1)
    w_vel_z = dde.grad.jacobian(u, x, i=2, j=2)
    w_vel_t = dde.grad.jacobian(u, x, i=2, j=3)
    w_vel_xx = dde.grad.hessian(u, x, component=2, i=0, j=0)
    w_vel_yy = dde.grad.hessian(u, x, component=2, i=1, j=1)
    w_vel_zz = dde.grad.hessian(u, x, component=2, i=2, j=2)
```

```
    p_x = dde.grad.jacobian(u, x, i=3, j=0)
    p_y = dde.grad.jacobian(u, x, i=3, j=1)
    p_z = dde.grad.jacobian(u, x, i=3, j=2)

    momentum_x = (
        u_vel_t  + (u_vel * u_vel_x + v_vel * u_vel_y + w_vel * u_vel_z) + p_x
        - 1 / Re * (u_vel_xx + u_vel_yy + u_vel_zz)
    )
    momentum_y = (
        v_vel_t + (u_vel * v_vel_x + v_vel * v_vel_y + w_vel * v_vel_z) + p_y
        - 1 / Re * (v_vel_xx + v_vel_yy + v_vel_zz)
    )
    momentum_z = (
        w_vel_t  + (u_vel * w_vel_x + v_vel * w_vel_y + w_vel * w_vel_z) + p_z
        - 1 / Re * (w_vel_xx + w_vel_yy + w_vel_zz)
    )
    continuity = u_vel_x + v_vel_y + w_vel_z

    return [momentum_x, momentum_y, momentum_z, continuity]
```

Отметим, что данная программная реализация показывает, как достаточно легко записать исходную систему уравнений Навье-Стокса в коде программы на Python.

5.2.2 Результаты

Расчеты проводились для разных чисел Re=1,5,10. Полученные результаты для трех полей компонент скорости сравнивались с результатами аналитического решения (рис.5). Расчеты показали, что изменение числа Re существенно не меняет картины течения.
Процесс обучения для функции потерь от количества итераций представлен на рис. 6.

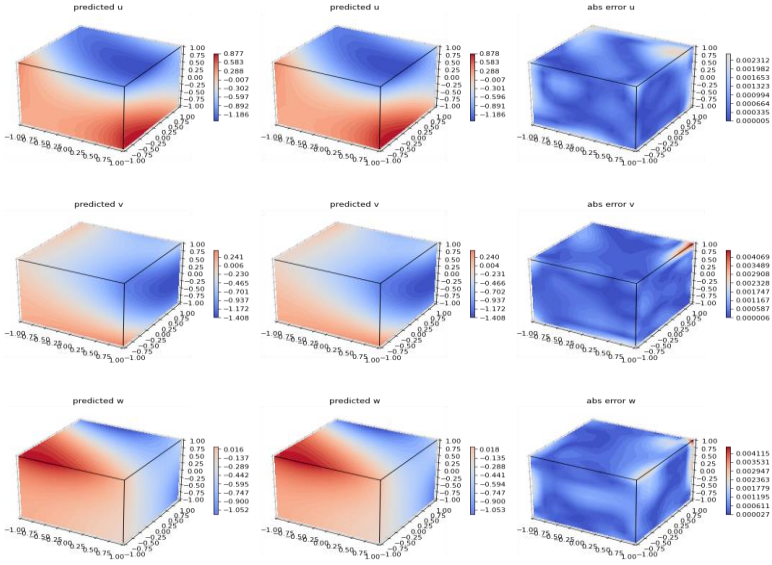

Рис.5 Поля скоростей u,v,w при t = 1.0 и значение ошибки для течения Бельтрами.
Слева на право показаны предсказанная поле скорости, аналитически-рассчитанное поле скорости, абсолютная ошибка.

Fig.5 The velocity fields u,v,w at t = 1.0 and errors for Beltrami flow.
The predicted velocity field, analytically-calculated velocity field, and absolute error are shown from left to right.

Обучение нейронных сетей и процесс предсказания параметров течения жидкости в задаче Бельтрами выполнялись на ПК с Intel Xeon, 32 GB RAM и Nvidia GPU.
Время обучения составило от 30 минут до 2 часов в зависимости от выбора гиперпараметров нейронных сетей. Динамические веса обновлялись каждые 100 эпох обучения. В качестве метрики для оценки результатов предсказания выбиралась величина MSE.
Также проводилось дополнительное исследование для случаев изменения параметра d в диапазоне от 0.75 до 1 с шагом 0.05. Данное исследование было проведено с целью отработки технологии дообучения для предсказания полей скорости и давления для промежуточных значений d и оценки времени предсказания.

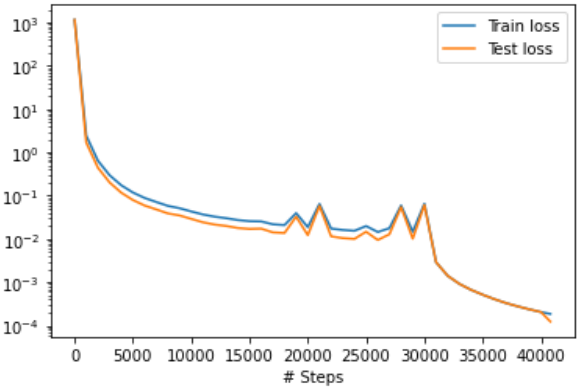

Рис. 6 Динамика изменения функции потерь при обучении нейронной сети для течения Бельтрами.
Fig.6 Dynamics of change in the loss function when training a neural network for Beltrami flow.

5.3 Течение в прямоугольном канале

Для моделирования течения на участке реки часто используются уравнения по теории мелкой воды [13, 14]. Для первоначальной оценки могут быть применены квазиодномерные уравнения Сен-Венана.
Ранее были рассмотрены архитектуры PINN для уравнений по теории мелкой воды в различной модельной постановке и программной реализации [15-17].
В данной работе рассматривалась математическая модель, включающая в себя уравнения неразрывности и движения, для безледоставного периода времени, которая имеет вид:

уравнение неразрывности

$$\frac{\partial W}{\partial t} + \frac{\partial Q}{\partial x} = q, \tag{1}$$

уравнение движения с использованием формулы Маннинга для учета трения о дно

$$\frac{\partial Q}{\partial t} + \frac{\partial}{\partial x}\left(\frac{Q^2}{W}\right) + gW\frac{\partial z}{\partial x} + \frac{gn^2 Q|Q|}{WR^{4/3}} = 0. \tag{2}$$

Здесь t – время, $x \in [0, X]$ – продольная координата вдоль русла, W – площадь живого сечения, Q – расход, q – боковая приточность на единицу длины, g – ускорение свободного падения, δ – отметка дна, n – коэффициент шероховатости по формуле Маннинга, R – гидравлический радиус, $z = \delta + H$ – уровень поверхности воды, H – глубина воды.
Полагаются известными следующие функции, определяющие морфологические параметры: $w = w(x,h)$ $h = h(x,w)$ $n = n(x)$ $\delta = \delta(x)$ $q = q(x,t)$, где w – площадь сечения, соответствующая расстоянию до отметки дна h. Начальные условия имеют вид:

$$W(x,0) = W_0(x) \quad Q(x,0) = Q_0(x). \tag{3}$$

Граничные условия для спокойного течения:

$$Q(0,t) = \hat{Q}(t) \text{ и } H(X,t) = \hat{H}(t). \quad (4)$$

Для общего случая решение поставленной задачи можно найти только численно.

Сбор необходимой для решения эмпирической информации для реального участка реки является трудоемкой задачей. Как правило, такие данные являются фрагментарными. Расстояние между имеющимися гидропостов в РФ достаточно велико. Как правило, достоверными данными на гидропостах являются только уровни поверхности воды. Данные о расходах зачастую отсутствуют или рассчитываются по суррогатным формулам. Важнейшая информация о морфологических параметрах в открытом доступе отсутствует. Значения боковой приточности могут быть определены для точечных источников, оценка же величины диффузной боковой приточности крайне приблизительна.

В этих условиях применение физически-обоснованной нейронной сети для аппроксимации набора натурных и расчетных данных позволит уточнить результаты как при решении прямой, так и обратной задач. Представляется одним из перспективных направлений получение более точной оценки распределенной (диффузной) боковой приточности.

Для оценки применимости PINN для моделирования течений в руслах и каналах представляется целесообразным начать со случаев, для которых имеются точные решения. Наиболее эффективный подход заключается в том, что для подходящего точного решения одномерных уравнений модели мелкой воды проще подобрать модельный канал, который будет полностью соответствовать выбранному точному решению. Для этой цели рассмотрим стационарное течение в прямоугольном канале шириной D с постоянным коэффициентом шероховатости n и заданными строго положительными значениями расхода $Q(x)$ и площади живого сечения $W(x)$.

Для прямоугольного канала величины глубины и гидравлического радиуса вычисляются по формулам:

$$H(x) = \frac{W(x)}{D} \text{ и } R(x) = \frac{W(x)}{D+2\cdot H(x)}. \quad (5)$$

Для выполнения уравнения неразрывности необходимо задать боковую приточность как

$$q(x) = \frac{dQ}{dx}. \quad (6)$$

Для выполнения уравнения движения надо вычислить отметку дна по формуле:

$$\delta(x) = H(0) - H(x) - \int_0^x \left(\frac{\frac{d(Q(x)^2/W(x))}{dx}}{gW(x)} + \frac{n^2 Q(x)^2}{W(x)^2 R^{4/3}} \right) dx. \quad (7)$$

Для обучения нейронной сети применяется прямая задача (1-4), в которой используются замыкающие соотношения (5), а боковая приточность и отметка дна являются предварительно вычисленными по формулам (6, 7).

5.3.1 Программная реализации

Пример реализации функции на Python для расчета уравнений неразрывности и количества движения представлен ниже.

```
def pde(x, y):
    Q, W = y[:, 0:1], y[:, 1:2]
    dQ_x = dde.grad.jacobian(y, x, i=0, j=0)
    dQ_t = dde.grad.jacobian(y, x, i=0, j=1)
    dW_t = dde.grad.jacobian(y, x, i=1, j=1)
    Q2W = Q * Q / W
    Q2W_x = dde.grad.jacobian(Q2W, x, i=0, j=0)
    H = H_func(W, x[:, 0:1])
```

```
R = R_func(W, x[:, 0:1])
R43 = tf.math.pow(R, fdt)
dz_x=dde.grad.jacobian(H, x, i=0, j=0)+ddelta_x(x)
qs = qs_func(x[:, 0:1])
return [dQ_t+Q2W_x+g*W*dz_x+gnn*Q2W/R43, dW_t+dQ_x-qs]
```

В случае применения нейронной сети для обучения на участке реальной реки текст вышеприведенной функции не меняется, но необходимо реализовать вспомогательные функции для расчета глубины H_func, гидравлического радиуса R_func и вычисления величины $\frac{\partial \delta}{\partial x}$ функцией ddelta_x.

5.3.2 Результаты решения

В качестве тестовой задачи рассматривается задача о моделировании течения в прямоугольном канале длиной 1000 м и шириной 10 м с постоянным наклоном дна, обеспечивающим постоянство по всей длине русла расхода 1 м^3/с и площади живого сечения 10 м^2. Стационарное решение достигается путем установления нестационарной задачи за период в 1 сутки.

Количество точек коллокации в расчетной области задавалось: num_domain=25001, num_boundary=500, num_initial=500, num_test=10000. При обучении в архитектуре нейронной сети использовалось 4 скрытых слоя с 50 нейронами. Функции активации для нейронов задавалась в виде tanh. Использовался оптимизатор Адама для обеспечения наилучшего набора начальных обучаемых переменных для нейронной сети. Затем L-BFGS-B использовался для тонкой настройки нейронной сети для достижения более высокой точности.

Расчет предсказания с помощью PINN величины расхода Q представлен на рис. 7.

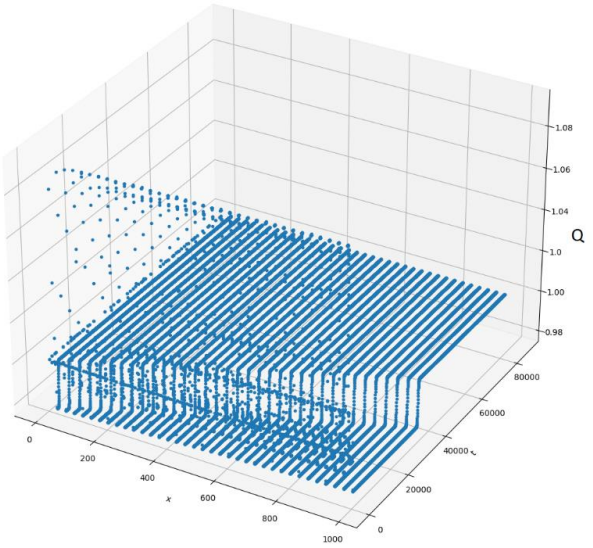

Рис. 7. Величина расхода Q.
Fig. 7. Flow rate Q.

Расчет предсказания величины площади живого сечения W представлен на рис. 8. Следует отметить, что в дальнейшем наибольший интерес представляют нестационарные решения и программная реализация создана именно для таких нестационарных процессов. Однако, пока тестирование выполнялось для стационарного решения, поэтому начальные условия задавались случайным образом. Выбросы значений расхода и площади живого сечения в начальный момент времени при обучении минимизировались и через сутки модельного

времени становились постоянными по всей длине русла канала, что свидетельствует о успехе обучения.

Обучение нейронной сети выполнялось на сервере с GPU Nvidia GeForce RTX 3070. Это позволило сократить время обучения в 15 раз по сравнению с сервером, использующим CPU с 24 потоками.

Заключение

Использование свободного программного обеспечения позволяет разрабатывать архитектуру PINN, расчетные коды для моделирования течений, допускающие аналитические решения. Для задачи с двумерным течением Коважного проведено исследование для случая Re=20.

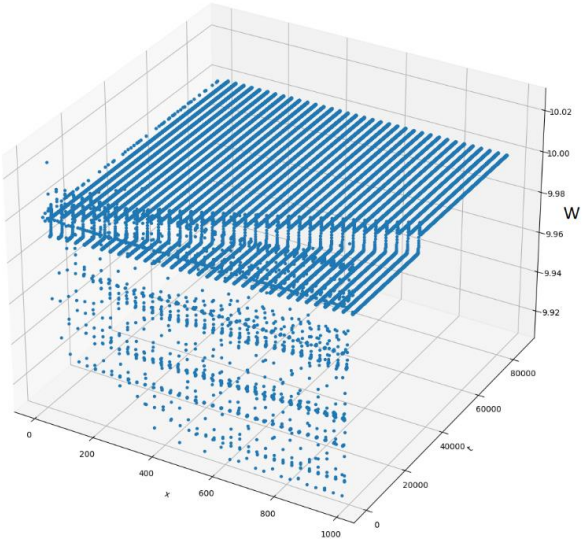

Рис. 8. Величина площади живого сечения W.
Fig. 8. Cross-sectional area value W.

Были поля скорости и давления, построены линии тока. По задаче с нестационарным трехмерным течением Бельтрами получены поля скорости и давления для заданных выбранных параметров. Выполнена оценка ошибки относительно точного аналитического решения. По задаче с течением в прямоугольном канале по теории мелкой воды в процессе обучения нейронной сети получены значения расхода и площади живого сечения, с высокой точностью совпадающие с аналитическим решением. В дальнейшем предстоит работа по применению модели для реальных участков русел рек и нестационарных течений. Дальнейшее развитие PINN для решения задач гидродинамики может быть связано с использованием различным архитектур PINN на базе глубоких операторных нейронных сетей (DeepONet) и нейронных операторов Фурье (FNO).

Список литературы / References

[1]. Raissi M., Perdikaris P., Karniadakis G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational Physics 378 (2019) 686-707.
[2]. Raissi M., Yazdani A., Karniadakis G.E. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations, Science 367 (2020) 1026-1030.
[3]. Лойцянский Л.Г. Механика жидкости и газа. Учебник для вузов. - 7-е изд., испр. - М.: Дрофа, 2003. - 840 с.
[4]. Петров А.Г. Аналитическая гидродинамика. М.: Физматлит. 2010. - 520 с.

[5]. Cuomo, Salvatore & Schiano Di Cola, Vincenzo & Giampaolo, Fabio & Rozza, Gianluigi & Raissi, Maziar & Piccialli, Francesco. (2022). Scientific Machine Learning Through Physics–Informed Neural Networks: Where we are and What's Next. Journal of Scientific Computing. 92. 10.1007/s10915-022-01939-z.
[6]. Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. Погружение в мир нейронных сетей. 2020. – 480 с.
[7]. Lu Lu, et al. DeepXDE: A Deep Learning Library for Solving Differential Equations. SIAM REVIEW. 2021. Vol. 63, No. 1, pp. 208–228.
[8]. https://github.com/lululxvi/deepxde
[9]. Kovasznay L.I.G. Laminar flow behind a two-dimensional grid. Math. Proc. Cambridge. 1948. V. 44. Is. 1. P. 58–62.
[10]. Бубенчиков А.М., Попонин В.С., Мельникова В.Н. Разработка универсальной техники реализации неоднородных граничных условий Дирихле и Неймана в методе спектральных элементов, Вестн. Томск. гос. ун-та. Матем. и мех., 2008, номер 2(3), 87–98.
[11]. Dong S., Karniadakis G.E., Cryssostomidis C. A robust and accurate outflow boundary condition for incompressible flow simulations on severely-truncated unbounded domains. J. Comput. Phys. 2014. V. 261. P. 83–105.
[12]. Jin X., Cai S., Li H., Karniadakis G.E. NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations. Journal of Computational Physics, Volume 426, 2021, 109951.
[13]. Беликов В.В., Алексюк А.И. Модели мелкой воды в задачах речной гидродинамики. Российская академия наук, Отделение наук о Земле, Институт водных проблем РАН. – Москва: Российская академия наук, 2020. – 346 с. – ISBN 978-5-907036-22-2.
[14]. Зиновьев А.Т., Кошелев К.Б., Дьяченко А.В., Коломейцев А.А. Экстремальный дождевой паводок 2014 года в бассейне Верхней Оби: причины, прогноз и натурные наблюдения // Водное хозяйство России: проблемы, технологии, управление. – 2015. – № 6. – С. 93-104.
[15]. Cedillo S. et al. Physics-Informed Neural Network water surface predictability for 1D steady-state open channel cases with different flow types and complex bed profile shapes. Adv. Model. and Simul. in Eng. Sci. (2022) 9:10.
[16]. Rosofsky S.G. et al. Applications of physics informed neural operators. Mach. Learn.: Sci. Technol. 4 (2023) 025022.
[17]. Feng D., Tan Z., He Q.Z. Physics-informed neural networks of the Saint-Venant equations for downscaling a large-scale river model. *Water Resources Research*, (2023). *59*, e2022WR033168.

Информация об авторах / Information about authors

Константин Борисович КОШЕЛЕВ – кандидат физико-математических наук, доцент, старший научный сотрудник Института водных и экологических проблем СО РАН. Сфера научных интересов: вычислительная гидродинамика, гидрология, геоинформатика.

Konstantin Borisovich KOSHELEV – Cand. Sci. (Phys.-Math.), associate professor, senior researcher at the Institute for water and environmental problems of the Siberian branch of the RAS. Research interests: computational fluid dynamics.

Сергей Владимирович СТРИЖАК – кандидат технических наук, ведущий инженер Института системного программирования им. В.П. Иванникова РАН с 2009 года. Сфера научных интересов: вычислительная гидродинамика, многофазные течения, турбулентность, ветроэнергетика, параллельные вычисления.

Sergei Vladimirovich STRIJHAK – Cand. Sci. (Tech.), leading engineer of the Ivannikov Institute for System Programming of the RAS since 2009. Research interests: computational fluid dynamics.