Министерство науки и высшего образования Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
Физико-механический институт
Высшая школа теоретической механики и математической физики

Работа допущена к защите
Директор ВШТМиМФ,
д.ф – м.н., чл.-корр. РАН
_____ А.М. Кривцов
«___» _____2024 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

магистерская диссертация

ИССЛЕДОВАНИЕ ПОДХОДОВ ГИБРИДНОГО МОДЕЛИРОВАНИЯ В ЗАДАЧАХ НЕФТЯНОЙ ИНДУСТРИИ

по направлению подготовки (специальности)
01.04.03 Механика и математическое моделирование
Направленность (профиль)
01.04.03_04 Математическое моделирование процессов нефтегазодобычи

Выполнил
студент гр. 5040103/20401 Н.А. Зырянов
Руководитель
Доцент ВШТМиМФ, к.ф.-м.н. Е.В. Юдин
Консультант
Владелец продукта ООО “Недра” Н.С. Марков

Санкт-Петербург
2024

САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

Физико-механический институт

Высшая школа теоретической механики и математической физики

УТВЕРЖДАЮ

Директор ВШТМиМФ

А.М.Кривцов

«__»_____20__г.

ЗАДАНИЕ

на выполнение выпускной квалификационной работы

студенту Зырянову Никите Андреевичу, гр. 5040103/20401

1. Тема работы: Исследование подходов гибридного моделирования в задачах нефтяной индустрии

2. Срок сдачи студентом законченной работы: 30.05.2024

3. Исходные данные по работе: Актуальные научные публикации по теме работы и смежным темам, постановка задачи, дифференциальные уравнения из нефтяной индустрии, уравнение Баклея – Леверетта, уравнение пьезопроводности, уравнение GORM.

4. Содержание работы (перечень подлежащих разработке вопросов): Исследование точности и возможностей классических методов решения задач нефтяной индустрии с методами гибридного моделирования. Валидация методов гибридного моделирования, дополнение текущих архитектур. Исследование применимости нейронных операторов, PINN моделей и валидация гиперпараметров нейронных сетей.

5. Перечень графического материала (с указанием обязательных чертежей): не предусмотрено

6. Консультанты по работе: Н.С. Марков – Владелец продукта ООО “Недра”.

7. Дата выдачи задания: 26.02.2024

Руководитель ВКР _____Е.В. Юдин – доцент ВШТМиМФ, к.ф.-м.н.

Задание принял к исполнению 26.02.2024

Студент _____Н.А. Зырянов

РЕФЕРАТ

На 52 с., 25 рисунков, 4 таблицы, 0 приложений

КЛЮЧЕВЫЕ СЛОВА: ГИБРИДНОЕ МОДЕЛИРОВАНИЕ, ФИЗИЧЕСКИ – ИНФОРМИРОВАННОЕ МАШИННОЕ ОБУЧЕНИЕ, НЕЙРОННЫЕ СЕТИ, НЕЙРОННЫЕ ОПЕРАТОРЫ, МОДЕЛИРОВАНИЕ ПЛАСТА.

В данной работе рассматриваются подходы гибридного моделирования, а именно физически – информированных нейронных сетей PINN и нейронных операторов PINO для описания процессов, происходящих в пласте. Описана методология построения таких моделей. Приведены примеры использования и ключевые достоинства и недостатки.

ABSTRACT

52 pages, 25 pictures, 4 tables, 0 applications

KEYWORDS: HYBRID MODELLING, PHYSICAL - INFORMED MACHINE LEARNING, NEURAL NETWORKS, NEURAL OPERATORS, RESERVOIR MODELLING, PINN, PINO.

This work discusses hybrid modelling approaches, namely physically-informed neural networks PINN and neural operators PINO to describe the processes occurring in the reservoir. The methodology of building such models is described. Examples of use and key advantages and disadvantages are given.

Содержание

ВВЕДЕНИЕ

Математическое моделирование физических процессов за последние десятилетия стало неотъемлемой частью работы со сложными системами. Кроме того, с ростом сложности моделей, увеличивалось и требование к входным данным, их количеству и точности. Использование математических моделей особенно важно для описания процессов, которые сложно наблюдать в лабораторных условиях. Например, в нефтегазовой отрасли основные процессы происходят на глубине нескольких километров. Моделирование пласта с необходимыми параметрами в лабораторных условиях - трудоемкая задача, которая даже в случае успешной реализации не дает полного понимания происходящих процессов. Однако чаще всего при анализе сложных физических, биологических или инженерных систем, стоимость сбора данных оказывается непомерно высокой, и мы неизбежно сталкиваемся с необходимостью делать выводы и принимать решения на основе неполной информации. В условиях такого малого объема данных подавляющее большинство современных методов машинного обучения (например, глубокие/конволюционные/рекуррентные нейронные сети) не обладают достаточной устойчивостью и не дают никаких гарантий сходимости.

Несмотря на огромные эмпирические перспективы и несомненный успех в решении целого ряда задач, большинство подходов машинного обучения на сегодняшний день не способны извлекать интерпретируемую информацию из большого потока данных. Более того, модели, основанные исключительно на данных, могут точно соответствовать наблюдениям, но давать физически противоречивые и неправдоподобные предсказания (например, из-за ошибок наблюдения), что приводит к низкой эффективности использования таких методов и данных.

В этом контексте машинное обучение на основе физики (Physics-Informed Machine Learning, PIML) представляет собой мощный инструмент для моделирования сложных процессов. PIML позволяет интегрировать физические законы в

нейросетевые модели, что повышает точность моделирования и позволяет работать с ограниченными данными.

Также такие модели называются “Гибридными”, так как в их основе лежать методы ИИ, использующие знания физики процессов. До недавнего времени методы машинного обучения основанные только на данных хотя и бурно развивались во многих областях, тем не менее показывали достаточно посредственные результаты в наукоемких задачах.

Тем временем классические методы, также пришли к стадии развития, когда не смотря на высокую точность, вычислительная трудоемкость численных подходов уже не позволяет оперативно рассчитывать сложные системы, проверять множественные гипотезы для задач управления, адаптироваться к различным конфигурациям систем, без полного переформатирования расчетной схемы и т.д.

В данной работе рассматривается подход, связывающий две фундаментальные концепции, современных численных решателей и искусственного интеллекта, для получения новой методологии обладающей точностью сравнительной с математическими пакетами, а также высокой скоростью решения, как прямых так и обратных задач. При этом также данный подход является гораздо более гибким и адаптивным в каждой конкретной математической и физической постановке.

Глава 1. ФИЗИЧЕСКИ – ИНФОРМИРОВАННЫЕ НЕЙРОННЫЕ СЕТИ

Нейронные сети с физическими данными (PINN) - это класс алгоритмов машинного обучения, в которых физические законы и ограничения интегрированы в нейронные сети. В традиционных нейронных сетях информация о входных и выходных данных усваивается без предварительного знания физических законов, управляющих моделируемой системой. В отличие от них, PINN используют уравнения, описывающие физическую систему, в качестве ограничений, накладываемых на нейронную сеть в процессе обучения.

Рассмотрим общую форму дифференциального уравнения (PDE), заданного на ограниченной пространственно-временной области:

$$D[u(x,t)] = f(x,t),\ x \in \Omega,\ t \in [0,T] \quad (1)$$

$$u(x,t) = g(x,t), \qquad x \in \partial\Omega,\ t \in [0,T] \quad (2)$$

$$(x,0) = h(x), \qquad x \in \Omega \quad (3)$$

где D – дифференциальный оператор, $\partial\Omega$ - граница области Ω и $f(\cdot)$, $g(\cdot)$ и $h(\cdot)$ определяемые правая часть дифференциального уравнения, граничные и начальные условия соответственно.

PINN позволяет найти приближенное решение уравнения (1) – (3) путем минимизации L^2 ошибки. Нейронная сеть минимизирует общую ошибку, которая состоит из:

- Ошибки дифференциального уравнения
- Ошибки начальных условий (IC)
- Ошибки граничных условий (BC)
- Ошибки относительно измеренных данных (опционально)

$$L_{all} = \alpha L_{ic}^2 + \beta L_{bc}^2 + \gamma L_{pde}^2 \quad (4)$$

где L_{ic}^2 – Ошибка по начальным условиям (IC); L_{bc}^2 – Ошибка по граничным условиям (на границе); L_{pde}^2 – Ошибка дифференциального уравнения; α, β, γ – весовые коэффициенты.

Использование физических законов позволяет повысить точность модели и сделать ее более интерпретируемой, поскольку обученная модель согласуется с известными физическими законами. Кроме того, PINN могут обрабатывать недостающие или зашумленные данные, используя уравнение, описывающее физическую систему, в качестве регуляризирующего члена, который действует как сглаживающее ограничение для модели [1].

Далее будет описано применение нейросетевого подхода на основе физики для решения задач нефтегазового инжиниринга.

1.2Автоматическое дифференцирование

Производные, в основном в виде градиентов и гессианов, повсеместно используются в машинном обучении. На данный момент самым популярным методом расчета градиентов, производных и т.д. при его высокой эффективности является автоматическое дифференцирование (АД), или просто «автодифференцирование». Данный набор методов используется для вычислительно эффективного расчета частных производных в компьютерных алгоритмах. Ранее автодифференцирование и машинное обучение не были связаны между собой, но с появлением алгоритмов обратного распространения ошибки, которые заместили генетические алгоритмы в обучении, АД стал наиболее используемым методом расчетов. Идеи, лежащие в основе АД, возникли еще в 1950-х годах [2, 3]. АД как общий метод оценки частных производных был, по сути, открыт Венгертом (1964). После этого наступил период относительно низкой активности, пока интерес к этой области не возродился в 1980-х годах в основном благодаря работе Гриванка (1989), чему также способствовали усовершенствования в современных языках программирования и возможность создания эффективного обратного режима АД. Таким образом обратный режим АД и обратное распространение имеют переплетенную историю. Работа Линнаинмаа [9]

часто упоминается как первое опубликованное описание обратного режима. Speelpenning [10] впоследствии представил обратный режим АД в том виде, в котором мы его знаем, в том смысле, что он дал первую реализацию, которая была фактически автоматической, принимая спецификацию вычислительного процесса, написанную на языке программирования общего назначения и автоматически выполняющую преобразование обратного режима. Несмотря на свою актуальность, АД общего назначения отсутствовали в инструментарии машинного обучения, и эта ситуация постепенно меняется с их появлением под названиями «динамические вычислительные графы» и «дифференцируемое программирование».

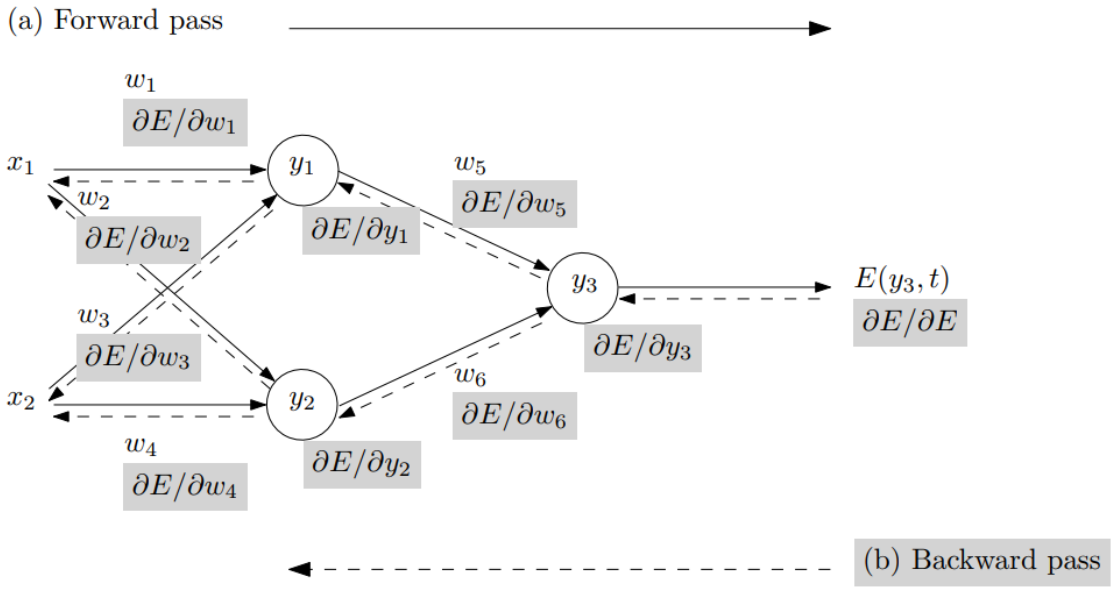


Рис.1 Схема автодифференцирования. Обучающий данные x_i создают соответствующие активации y_i. $E -$ ошибка между выходом последнего слоя и целевым значением t. Для обратного прохода в процессе обучения используется градиент ошибки по обучаемым весам w_i.

В общем виде автодифференцирование не является символьным или численным методом взятия частных производных и нахождения градиентов. Традиционно многие методы машинного обучения требовали оценки производных, а большинство традиционных алгоритмов обучения опирались на вычисление градиентов и гессианов целевой функции [11]. При внедрении новых моделей

исследователи машинного обучения тратили значительные усилия на ручное выведение аналитических производных для последующего использования их в стандартных процедурах оптимизации таких как L-BFGS [12] или стохастический градиентный спуск [13]. Ручное дифференцирование отнимает много времени и чревато ошибками. Из других альтернатив численное дифференцирование просто в реализации, но может быть очень неточным из-за ошибок округления и ошибок усечения; что еще более важно, оно плохо масштабируется для градиентов, что делает его непригодным для машинного обучения, где обычно требуются градиенты по миллионам параметров. Символьное дифференцирование устраняет недостатки как ручного, так и численного методов, но часто приводит к сложным формулам. Мы рассматриваем мощную четвертую технику - автоматическое дифференцирование (АД). АД выполняет нестандартную интерпретацию заданной компьютерной программы, заменяя область переменных, чтобы включить в нее значения производных, и переопределяя семантику операторов, чтобы распространять производные по цепному правилу дифференциального исчисления.

АД к определенному семейству методов, которые вычисляют производные путем накопления значений во время выполнения кода для создания численных оценок производных, а не выражений производных. Это позволяет точно оценивать производные с машинной точностью лишь с небольшим постоянным коэффициентом накладных расходов и идеальной асимптотической эффективностью. В основных библиотеках глубокого обучения: Tensorflow, Tensorflow 2.0, Pytorch – эффективно имплементированы методы автодифференцирования с использованием вычислительных графов. В данной работе используется библиотека Pytorch для реализации автодифференцирования и обучения нейронных сетей. Особенности Pytorch – это динамическое построение графов вычислений, что позволяет использовать любые функции основного языка, что отличает этот фреймворк от статических как Tensorflow, а также мгновенно выполняет тензорные операции, без необходимости, не создавая лишние вычислительные графы.

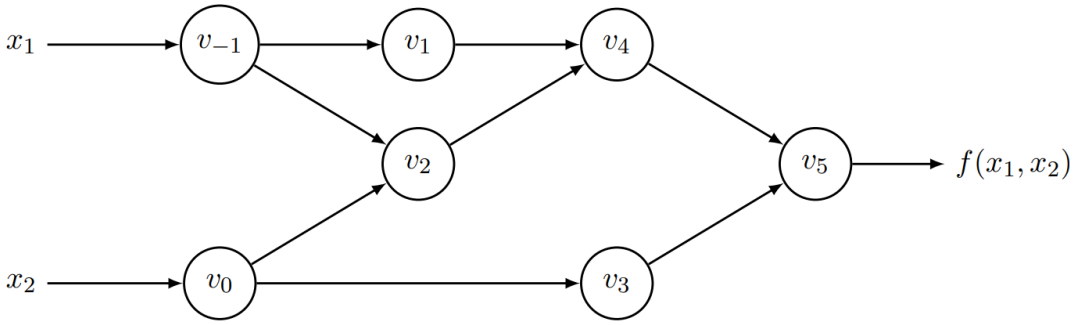


Рис.2. Упрощенный пример для графа вычислений функции

$$f(x_1, x_2) = \ln(x_1) + x_1 x_2 - \sin(x_2)$$

AD в режиме прямого накопления является концептуально наиболее простым типом. Рассмотрим след оценки функции $f(x_1, x_2) = \ln(x_1) + x_1 x_2 - \sin(x_2)$ приведенный в левой части в таблице 2 и в виде графика на рисунке 4.

Для вычисления производной f относительно x_1, мы начинаем с того, что связываем с каждой промежуточной переменной v_i производную. Применяя правило цепочки к каждой элементарной операции в прямом первоначальном следе, мы генерируем соответствующий касательный (производный) след, приведенный в правой части таблицы 2. Самый естественный способ расчета цепочного правила - это вычисление Якобиана отображения $f: R^n \to R^m$ с n независимыми входными переменными x_i и m зависимых (выходных) переменных. Каждый прямой проход АД инициализируется установкой только одной переменной $x_i = 1$, а остальные переменные устанавливаются нулевыми значениями.

$$\dot{y}_j = \frac{\partial y_i}{\partial x_i}|_{x=a}, j = 1, \dots, m.$$

Прямой ход первообразной			Прямой ход производной		
v_{-1}	$= x_1$	$= 2$	$\dot{v}_{-1}$	$= \dot{x}_1$	$= 1$
v_0	$= x_2$	$= 5$	$\dot{v}_0$	$= \dot{x}_2$	$= 0$
v_1	$= \ln v_{-1}$	$= \ln 2$	$\dot{v}_1$	$= \dot{v}_{-1}/v_{-1}$	$= 1/2$
v_2	$= v_{-1} \times v_0$	$= 2 \times 5$	$\dot{v}_2$	$= \dot{v}_{-1} \times v_0 + \dot{v}_0 \times v_{-1}$	$= 1 \times 5 + 0 \times 2$
v_3	$= \sin v_0$	$= \sin 5$	$\dot{v}_3$	$= \dot{v}_0 \times \cos v_0$	$= 0 \times \cos 5$
v_4	$= v_1 + v_2$	$= 0.693 + 10$	$\dot{v}_4$	$= \dot{v}_1 + \dot{v}_2$	$= 0.5 + 5$
v_5	$= v_4 - v_3$	$= 10.693 + 0.959$	$\dot{v}_5$	$= \dot{v}_4 - \dot{v}_3$	$= 5.5 - 0$
y	$= v_5$	$= 11.652$	$\dot{y}$	$= \dot{v}_5$	$= 5.5$

Таблица 1. Пример Автодифференцирования в прямом режиме. Исходная прямая оценка первообразных слева дополняется касательными операциями справа, где каждая строка дополняет оригинал, расположенный непосредственно слева от нее.

Получается столбец матрицы Якобиана.

$$J_f = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} \tag{4}$$

Математически, прямой проход АД является методом разложения функции в ряд Тейлора до первого порядка малости, с использованием чисел абелевой группы:

$$v + \dot{v}\epsilon \tag{5}$$

где $v, \dot{v} \in \mathbb{R}$ и ϵ это нильпотентное (обобщение абелевой группы) число, $\epsilon^2 = 0$ и $\epsilon \neq 0$. Рассмотрим для примера:

$$(v + \dot{v}\epsilon) + (u + \dot{u}\epsilon) = v + u + (\dot{v} + \dot{u})\epsilon \tag{6}$$

$$(v + \dot{v}\epsilon) + (u + \dot{u}\epsilon) = vu + (u\dot{v} + v\dot{u})\epsilon \tag{7}$$

Основной смысл такой интерпретации в том, что можно удобно производить дифференцирование произвольной функции:

$$f(v + \dot{v}\epsilon) = f(v) + f'(v)\dot{v}\epsilon \tag{8}$$

Тогда дифференцирование сложной функции или цепное правило:

$$f(g(v + \dot{v}\epsilon)) = f(g(v) + g'(v)\dot{v}\epsilon) = f(g(v)) + f'\big(g(v)\big)g'(v)\dot{v} \qquad (9)$$

Таким образом автоматически инвариант 5 распространяется на все возможные функциональные представления и комбинации. Данная методология позволяет одновременно как значения функции в точке при прямом проходе, так и соответствующие производные элементарных функций, на которые разбивается входная функция. В любом языке программирования в практической реализации функция со значением в точке является входным параметром для инструмента автодифференцирования, далее вычисляются и записываются в память необходимые значения и производные или градиенты. Данный подход считает производные выходов по произвольному направлению, но такое направление единственное, что накладывает ограничение на эффективность в некоторых случаях.

Поэтому существует метод обратного прохода при составлении графа вычислений. Это делается путем дополнения каждой промежуточной переменной v_i производной по этой переменной v_i :

$$\overline{v_i} = \frac{\partial y_j}{\partial v_i} \qquad (10)$$

В случае обратного распространения y_j будет скаляром, соответствующим ошибке E. Вычисление таким образом выполняется в два этапа, прямой проход и заполнением в память значений декомпозированных функций из исходной, на втором этапе производные вычисляются путем распространения производных, соответствующих переменным $\overline{v_i}$ в обратном направлении, от выходов к входам. Пример $f(x_1, x_2) = \ln(x_1) + x_1 x_2 - \sin(x_2)$, в таблице 2 мы видим производне выражения в правой части, соответствующие каждой исходной элементарной операции в левой части. Вычисление вклада $\overline{v_i} = \frac{\partial y}{\partial v_i}$ изменение переменной v_i к изменению выхода y.

Соответствующие производные к v_0:

$$\frac{\partial y}{\partial v_0} = \frac{\partial y}{\partial v_2}\frac{\partial v_2}{\partial v_0} + \frac{\partial y}{\partial v_3}\frac{\partial v_3}{\partial v_0} \quad \text{или} \quad v_0 = v_2\frac{\partial v_2}{\partial v_0} + v_3\frac{\partial v_3}{\partial v_0} \tag{11}$$

Данный шаг вычисляется последовательно по цепному правилу (производной сложной функции).

$$\overline{v_0} = \overline{v_3}\frac{\partial v_3}{\partial v_0} \qquad \text{и} \qquad \overline{v_0} = \overline{v_0} + \overline{v_2}\frac{\partial v_2}{\partial v_0} \tag{12}$$

После прямого выполненного прямого хода, вычисляем производные из обратного хода.

В итоге получаются производные $\frac{\partial y}{\partial x_1} = \bar{x}_1$ и $\frac{\partial y}{\partial x_2} = \bar{x}_2$ за один обратный проход. В сравнении с прямым режимом накопления, обратный режим существенно менее интуитивен в первую очередь из - за многократного взятия производных сложных функций. Важным преимуществом обратного режима является то, что он значительно менее затратен (в смысле количества операций), чем прямой режим для функций с большим количеством входов. В предельном случае $f: R^n \to R$, только одно применение обратного распространения достаточно для вычисления полного градиента:

$$\nabla f = \left(\frac{\partial y}{\partial x_1}, \ldots, \frac{\partial y}{\partial x_n}\right) \tag{13}$$

по сравнению с n проходами прямого режима, необходимыми для их расчета. Поскольку в практике машинного обучения в основном используется градиент скалярно - значного целевого значения по большому числу параметров, это позволяет считать обратный режим, в отличие от прямого, основной техникой в виде алгоритма обратного распространения.

Прямой ход			Обратный ход		
$v_{-1} = x_1$		$= 2$	$\bar{x}_1 = \bar{v}_{-1}$		$= \mathbf{5.5}$
$v_0 = x_2$		$= 5$	$\bar{x}_2 = \bar{v}_0$		$= \mathbf{1.716}$
$v_1 = \ln v_{-1}$		$= \ln 2$	$\bar{v}_{-1} = \bar{v}_{-1} + \bar{v}_1 \frac{\partial v_1}{\partial v_{-1}}$	$= \bar{v}_{-1} + \bar{v}_1 / v_{-1}$	$= 5.5$
$v_2 = v_{-1} \times v_0$		$= 2 \times 5$	$\bar{v}_0 = \bar{v}_0 + \bar{v}_2 \frac{\partial v_2}{\partial v_0}$	$= \bar{v}_0 + \bar{v}_2 \times v_{-1}$	$= 1.716$
			$\bar{v}_{-1} = \bar{v}_2 \frac{\partial v_2}{\partial v_{-1}}$	$= \bar{v}_2 \times v_0$	$= 5$
$v_3 = \sin v_0$		$= \sin 5$	$\bar{v}_0 = \bar{v}_3 \frac{\partial v_3}{\partial v_0}$	$= \bar{v}_3 \times \cos v_0$	$= -0.284$
$v_4 = v_1 + v_2$		$= 0.693 + 10$	$\bar{v}_2 = \bar{v}_4 \frac{\partial v_4}{\partial v_2}$	$= \bar{v}_4 \times 1$	$= 1$
			$\bar{v}_1 = \bar{v}_4 \frac{\partial v_4}{\partial v_1}$	$= \bar{v}_4 \times 1$	$= 1$
$v_5 = v_4 - v_3$		$= 10.693 + 0.959$	$\bar{v}_3 = \bar{v}_5 \frac{\partial v_5}{\partial v_3}$	$= \bar{v}_5 \times (-1)$	$= -1$
			$\bar{v}_4 = \bar{v}_5 \frac{\partial v_5}{\partial v_4}$	$= \bar{v}_5 \times 1$	$= 1$
$y = v_5$		$= 11.652$	$\bar{v}_5 = \bar{y}$	$= 1$	

Таблица 2. Обратный проход при автодифференцировании

В общем случае для функций $f: R^n \to R^m$, обратный режим автодифференцирования работает лучше, когда m ≪ n. Аналогично безматричному вычислению якобиан-векторных произведений с помощью прямого режима, обратный режим может быть использован для вычисления транспонированного якобиан-векторного произведения.

$$J^T r = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} \begin{bmatrix} r_1 \\ \vdots \\ r_m \end{bmatrix} \tag{14}$$

Однако за преимущества обратного режима AD приходится платить увеличением требований к хранению данных. Требования к хранению, растущие (в худшем случае) пропорционально количеству операций в оцениваемой функции. АД как инструмент на данный момент является наиболее распространенным алгоритмом для обучения нейронных сетей, тем не менее в случае с применением машинного обучения в научных задачах, он может быть либо не столь эффективен, или в отличие от конечных разностей, не передавать знания о пространственно – временых зависимостях в модель. Это также является предметом текущих исследований.

1.3 Дизайн экспериментов

В предыдущей главе был описан механизм того, каким образом по нейронной сети распространяются градиенты, в процессе обучения. Отличие и специфика PINN нейронных сетей, а именно встраивание дифференциального уравнения в потери нейронной сети с помощью автоматического дифференцирования, и эти потери оцениваются в наборе разрозненных пространственно-временных точек (называемых остаточными точками). Данный механизм является преимуществом нейронных сетей над классическими численными схемами (метод конечных разностей, элементов, объемов и т.д.), в которых зачастую пространственно-временные точки распределены равномерно. Расположение и распределение этих остаточных точек очень важны для работы PINN. Таким образом методы сэмплирования (выборки точек) можно разделить на 2 типа по использованию априорных знаний о системе:

1. Неадаптивная равномерная выборка
2. Адаптивная неравномерная выборка

Для задач в большой области декомпозиция пространственно-временной области ускоряет обучение PINN и повышает их точность [17, 18, 19]. Особенно важна методика выбора точек для решений, имеющих неоднородную пространственную структуру, такие как уравнения с источниками/стоками, высокочастотные начальные условия, “жесткие” дифференциальные уравнения. В дополнение к этим общим методам были разработаны и другие методы, ориентированные на конкретные задачи. PINN в основном оптимизируются с учетом потерь PDE, что гарантирует, что обученная сеть соответствует решаемому PDE. Потери PDE оцениваются на множестве рассеянных остаточных точек. Интуитивно понятно, что влияние остаточных точек на PINN аналогично влиянию точек сетки на FEM, и поэтому расположение и распределение точек сэмплирования должно быть очень важным для производительности PINN.

Рассмотрим PDE:

$$N[u(x,t)] = f(x,t),\ x \in \Omega,\ t \in [0,T] \tag{15}$$

$$u(x,t) = g(x,t), \qquad x \in \partial\Omega, \ t \in [0,T] \tag{16}$$

$$u(x,0) = h(x), \qquad x \in \Omega \tag{17}$$

Физически – информированная функция потерь:

$$L(\theta) = L_r(\theta) + L_b(\theta) + L_i(\theta) \tag{18}$$

Случайная выборка точек из Ω:

$$L_r(\theta) = \frac{1}{N_r}\sum_{n=1}^{N_r} |N[u_\theta(x_r^n, t_r^n)] - f(x_r^n, t_r^n)|^2, \{x_r^n, t_r^n\}_{n=1}^{N_r} \tag{19}$$

Точки на границе:

$$L_b(\theta) = \frac{1}{N_b}\sum_{n=1}^{N_b} |u_\theta(x_b^n, t_b^n) - g(x_b^n, t_b^n)|^2, \{x_b^n, t_b^n\}_{n=1}^{N_b} \tag{20}$$

Точки начального условия:

$$L_i(\theta) = \frac{1}{N_i}\sum_{n=1}^{N_i} |u_\theta(x_i^n, 0) - h(x_i^n)|^2, \{x_i^n, t_i^n\}_{n=1}^{N_b} \tag{2}$$

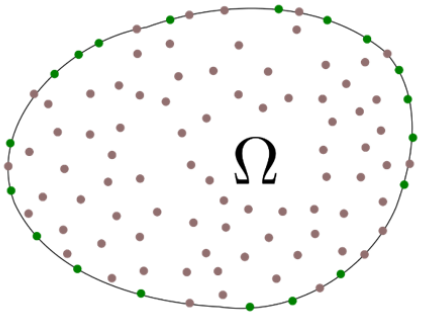


Рис.2. Область решения дифференциального уравнения

В подавляющем большинстве работ точки для расчета остатка по PDE выбираются или равномерно или с помощью LHS. Другие методы выборки, последовательность Соболя [23], также используются в некоторых исследованиях [1, 24, 25]. Последовательность Соболя - это один из видов квазислучайных последовательностей с низким расхождением.

Последовательности с низким расхождением – это последовательности, которые позволяют получить бесконечное количество точек из пространства любой размерности, которые уменьшают вероятность расхождения при этом такие точки покрывают все пространство. Фундаментальное преимущество таких последовательностей заключается в том, что если итоговые расхождения на основании конечного количества членов слишком велики, то последовательность может быть расширена без отбрасывания всех предыдущих вычисленных точек. Такие виды последовательностей можно разбить на группы по методу построения их базисных параметров:

1. Рациональные дроби: Кронекер, Рихтмайер, Рэмшоу
2. Взаимно простые числа: Ван дер Корпут, Холтон, Форе
3. Несводимые полиномы: Нидеррайтер
4. Примитивные многочлены: Соболь

Далее будут рассматриваться основные последовательности с низким расхождением (малой погрешностью), такие как методы Холтона [26] и Хаммерсли [27].

Ниже перечислены шесть методов равномерной выборки, примеры 400 точек, сгенерированных в $[0, 1]^2$ с использованием различных методов, показаны на рис. 4.

1. **Равноудаленная равномерная сетка (Grid):** Классический метод выбора равноудаленных точек.
2. **Равномерно случайная выборка (Random):** Точки выбираются случайным образом, в программном коде реализовывается с помощью псевдо - случайных методов.
3. **Латинский Гиперкуб (LHS):** Стратифицированная выборка Монте-Карло. Метод, который генерирует случайные выборки, встречающиеся в интервалах на основе равной вероятности и с нормальным распределением для каждого интервала.

4. Квази-случайные последовательности с малым количеством отклонений:

(a) **Последовательность Холтона (Halton):** Выборка Гальтона генерируются в соответствии с обратным преобразованием оснований чисел с помощью простых чисел.

(b) **Последовательность Хаммерсли (Hammersley):** Последовательность Хаммерсли это та же последовательность Гальтона, за исключением первого измерения, где точки расположены равноудаленно друг от друга.

(c) **Последовательность Соболя (Sobol):** Последовательность с основанием 2, из которой выбираются точки упакованные в пространстве наиболее равномерно из всех рассмотренных методов.

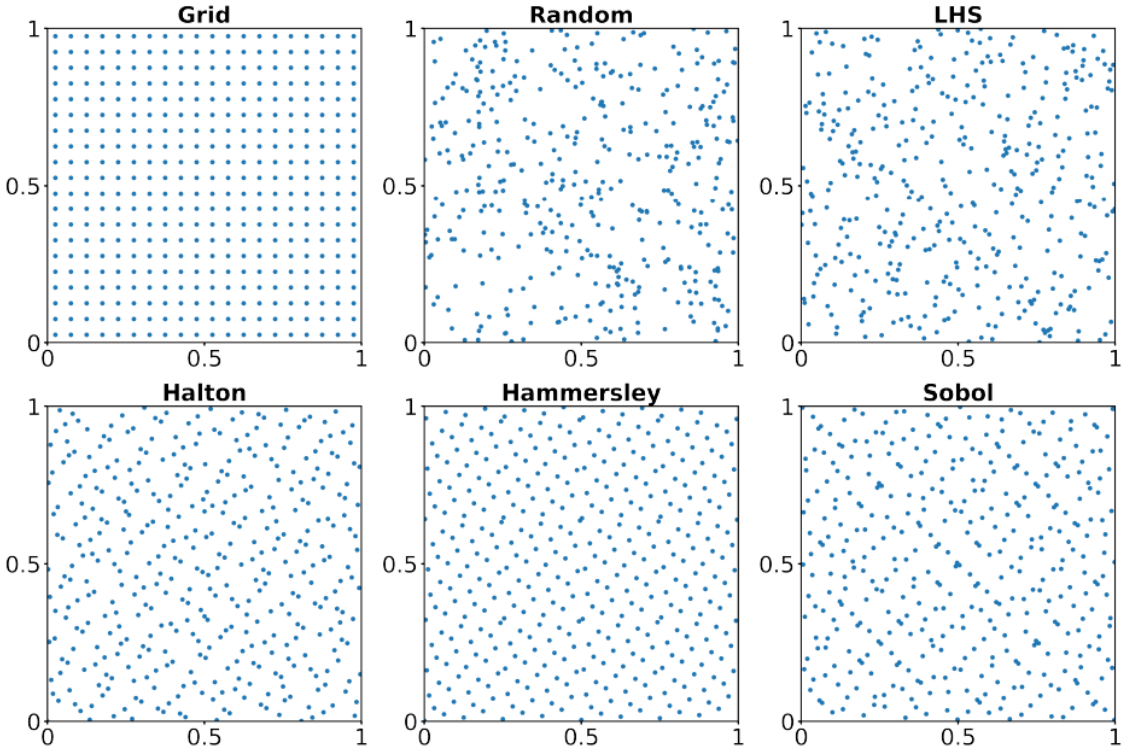


Рис.3. Сравнение различных сэмплирований 400 точек из случайного распределения различными методами

Весомым преимуществом PINN, является возможность использовать любые наборы точек в пространстве решения, вместо того чтобы использовать фиксированные остаточные точки во время обучения, возможно также выбирать новый набор остаточных точек в каждой определенной итерации оптимизации. Конкретный метод выборки точек каждый раз может быть выбран из предыдущего списка методов.

Кроме статичных методов сэмплирования рассматриваются адаптивные:

1. **Метод повторной выборки (Random-R):** Данный метод аналогичен методу Random, за исключением того, что остаточные точки подвергаются повторной выборке каждые N итераций.
2. **Метод адаптивной выборки (RAR-G):** Первым методом адаптивного обучения являлся RAR, предложенный в работе [28]. RAR направлен на улучшение распределения остаточных точек в процессе обучения путем выборки большего количества точек в местах, где остаток PDE велик, то есть этот алгоритм добавляет точки после каждой определенной итерации в места, где остаток PDE является наибольшим. Описание алгоритмов на псевдокоде ниже.

Алгоритм RAR-G [28]:

1. Произведите выборку исходных остаточных точек T, используя один из методов, из списка;
2. Обучить PINN за определенное число итераций;
3. **Повторение** в цикле
   a. Выборка набора плотных точек S0 с помощью одного из методов из списка;
   b. Вычислить остатки PDE для точек в S0;
   c. $S \leftarrow$ m точек с наибольшими остатками в S0;
   d. $T \leftarrow T \cup S$;

е. Обучение PINN в течение определенного количества итераций;

4. **Пока** общее количество итераций или общее количество остаточных точек не достигнет заданной границы;

RAR-G значительно улучшает производительность PINN при решении некоторых PDE с решениями с крутыми градиентами. Тем не менее, RAR-G фокусируется в основном на месте, где остаток PDE остаток наибольший и игнорирует местоположение меньших остатков. Другая стратегия выборки была разработанная позже в работе [29], в которой все остатки, оставшиеся от ПДЭ, сосредотачиваются на месте наибольшего остатка и игнорируют места меньших остатков. [29], где все остаточные точки повторно дискретизируются в соответствии с функцией плотности вероятности плотности вероятности (PDF) $p(x)$, пропорциональной остатку PDE.

В частности, для любой точки x мы сначала вычисляем остаток PDE $\varepsilon(x) = |f(x;\ \hat{u}(x))|$, а затем вычисляем вероятность как

$$p(x) \propto \varepsilon(x) \tag{22}$$

$$p(x) = \frac{\varepsilon(x)}{A} \tag{23}$$

где $A = \int_{\Omega} \varepsilon(x)dx$ нормализующая константа. Затем все остаточные точки отбираются в соответствии с вероятностью $p(x)$. Таким образом можно получить достаточно гибкий способ получения точек, с учетом апостериорного знания о виде уравнения, сложности геометрии и т.д.

$$\mathrm{p}(x) \propto \frac{\varepsilon^k(x)}{\mathrm{E}[\varepsilon^k(x)]} + c \tag{24}$$

где $k \geq 0$ и $c \geq 0$ два гиперпараметра. $\mathrm{E}[\varepsilon^k(x)]$ может быть аппроксимирован путем численного интегрирования методом Монте Карло.

С одной стороны данный метод может быть гораздо более эффективным чем предыдущие описанные, но при этом добавляется сложность в подборе гиперпараметров, на данный момент нет автоматического способа их найти.

Описание алгоритма RAD на псевдокоде:

Алгоритм 2: RAD

1. Произведите выборку начальных остаточных точек T, используя один из методов, описанных ранее;

2. Обучить PINN за определенное количество итераций;

3. **повторять** в цикле

4. $T \leftarrow$ Новый набор точек, отобранных случайным образом в соответствии с PDF уравнения (2);

5. Обучить PINN за определенное количество итераций;

6. **пока** общее количество итераций не достигнет предела;

В RAD нам необходимо произвести выборку множества точек из вероятностного пространства с $p(x)$, основным эффективным на практике подходом является такой:

1. Сделайте выборку множества точек S_0, используя один из методов, описанных ранее;

2. Вычислите $p(x)$ для точек в S_0;

3. Определите массовую функцию вероятности $\tilde{p}(x) = \frac{p(x)}{A}$ с нормирующей константой A = $\sum_{x \in S_0} p(x)$;

4. Произвести выборку подмножества точек из S_0 в соответствии с $\tilde{p}(x)$.

Этот метод прост, легко реализуем и достаточен для решения многих задач PDE.

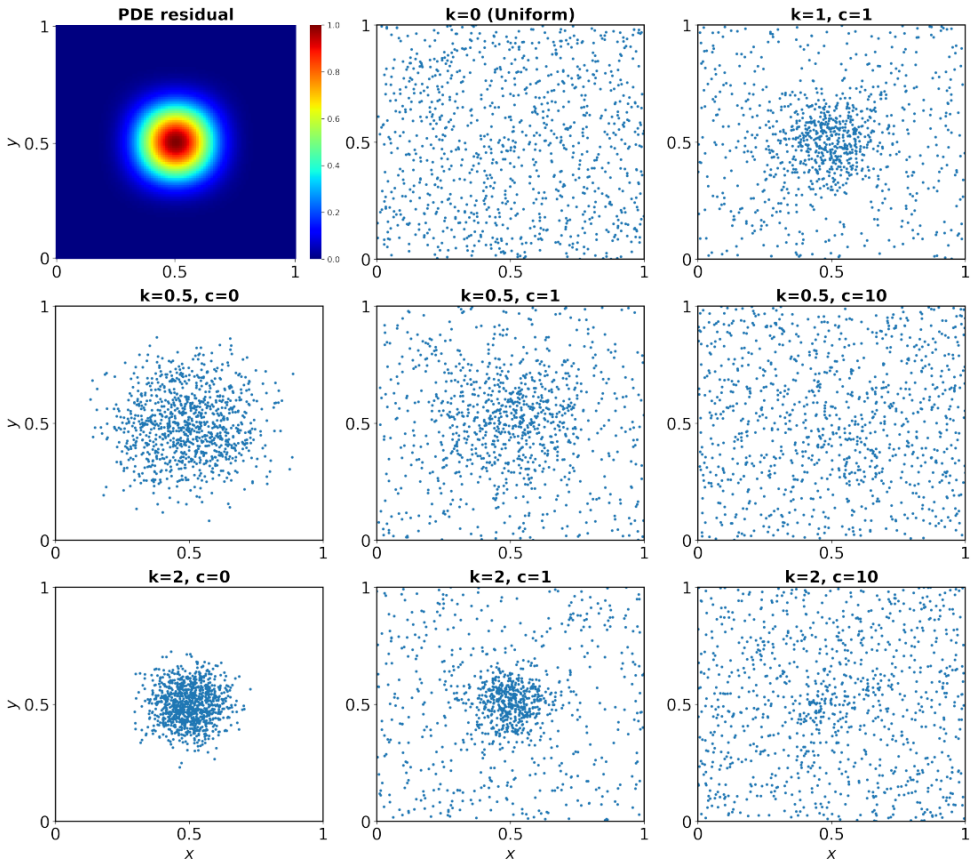


Рис.4. Примеры 1000 точек по PDE, отобранных методом RAD с различными значениями k и c для остатка PDE $\varepsilon(x, y)$.

	Диффузия	Burgers	Allen-Cahn	Волновое
Количество точек	30	2000	1000	2000
Grid	0.66 ± 0.06%	13.7 ± 2.37%	93.4 ± 6.98%	81.3 ± 13.7%
Random	0.74 ± 0.17%	13.3 ± 8.35%	22.2 ± 16.9%	68.4 ± 20.1%
LHS	0.48 ± 0.24%	13.5 ± 9.05%	26.6 ± 15.8%	75.9 ± 33.1%
Halton	0.24 ± 0.17%	4.51 ± 3.93%	0.29 ± 0.14%	60.2 ± 10.0%
Hammersley	0.17 ± 0.07%	3.02 ± 2.98%	**0.14 ± 0.14%**	58.9 ± 8.52%
Sobol	0.19 ± 0.07%	3.38 ± 3.21%	0.35 ± 0.24%	57.5 ± 14.7%
Random-R	**0.12 ± 0.06%**	1.69 ± 1.67%	0.55 ± 0.34%	**0.72 ± 0.90%**
RAR-G	0.20 ± 0.07%	**0.12 ± 0.04%**	0.53 ± 0.19%	0.81 ± 0.11%
RAD	**0.11 ± 0.07%**	**0.02 ± 0.00%**	**0.08 ± 0.06%**	**0.09 ± 0.04%**
RAR-D	**0.14 ± 0.11%**	**0.03 ± 0.01%**	**0.09 ± 0.03%**	**0.29 ± 0.04%**

Таблица 3. Сравнение относительной ошибки обычного PINN с разными методами сэмплирования.

Ниже приведены примеры математических постановок

Уравнение Диффузии

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + e^{-t}(-\sin(\pi x) + \pi^2 \sin(\pi x)), x \in [-1,1], \quad t \in [0,1], \tag{25}$$

$$u(x,0) = \sin(\pi x), \tag{26}$$

$$u(-1,t) = u(1,t) = 0, \tag{27}$$

где u - концентрация вещества.

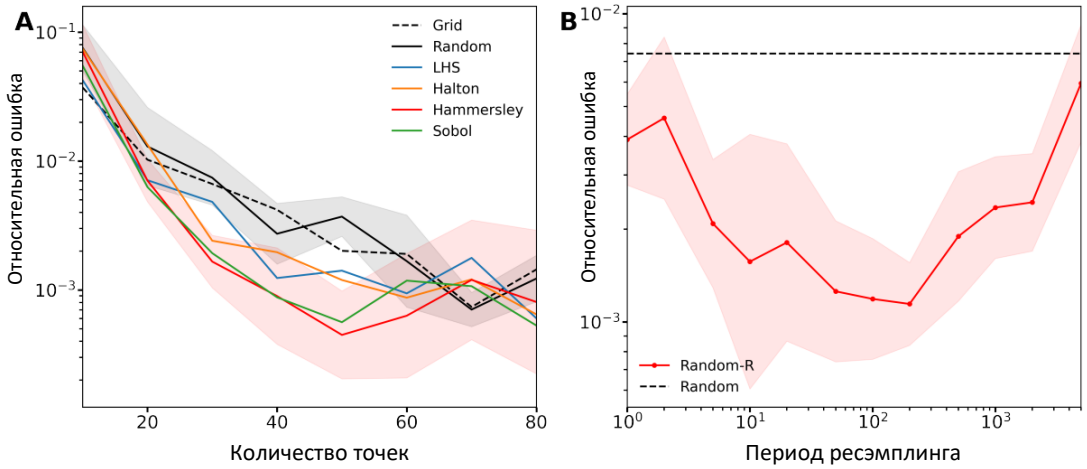


Рис.5. Сравнение методов сэмплирования для уравнения диффузии

Уравнение Burgers

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2}, \quad x \in [-1,1], t \in [0,1], \tag{28}$$

$$u(x,0) = -\sin(\pi x), \tag{29}$$

$$u(-1,t) = u(1,t) = 0, \tag{30}$$

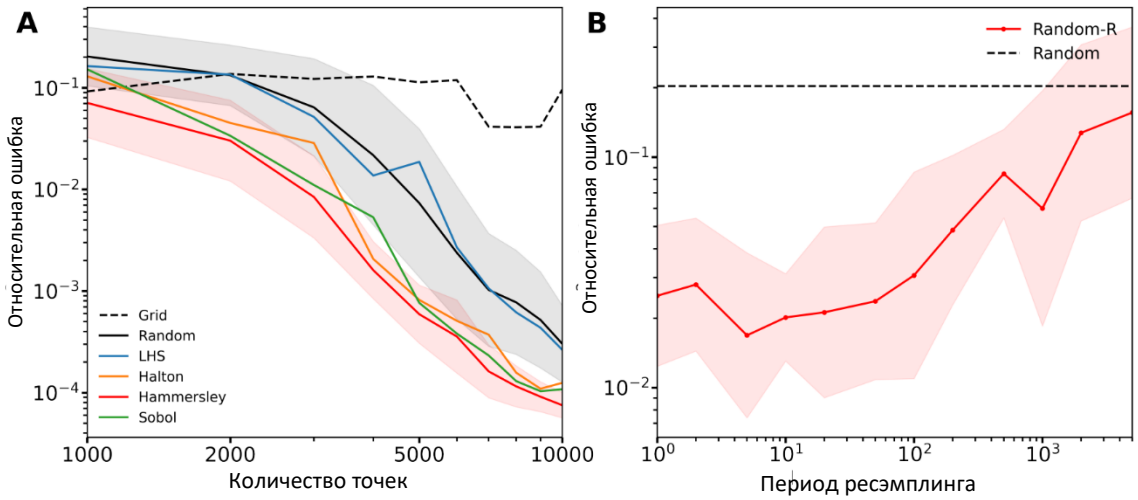


Рис.6. Сравнение методов сэмплирования для уравнения Burgers

Уравнение Allen-Cahn

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} + 5(u - u^3), \qquad x\,\epsilon[-1,1], t\,\epsilon[0,1], \tag{31}$$

$$u(x,0) = x^2\,cos(\pi x)\,, \tag{32}$$

$$u(-1,t) = u(1,t) = -1, \tag{33}$$

где $D = 0.001$

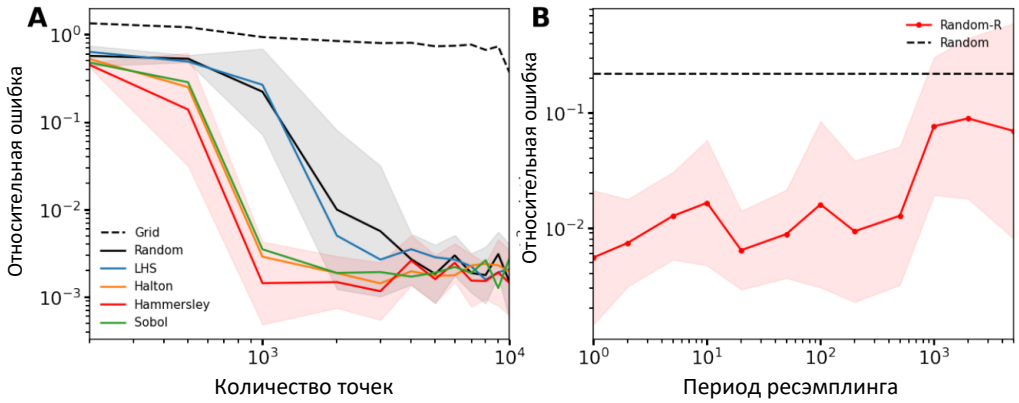


Рис.7. Сравнение методов сэмплирования для уравнения Allen-Cahn

Волновое уравнение

$$\frac{\partial^2 u}{\partial t^2} - 4\frac{\partial^2 u}{\partial x^2} = 0, \qquad x\,\epsilon[-1,1], t\,\epsilon[0,1], \tag{34}$$

$$u(x,0) = \sin(\pi x) + \frac{1}{2}\sin(4\pi x)\,, x \in [0,1], \tag{35}$$

$$\frac{\partial u}{\partial t}(x,0) = 0\,, \qquad x \in [0,1], \tag{36}$$

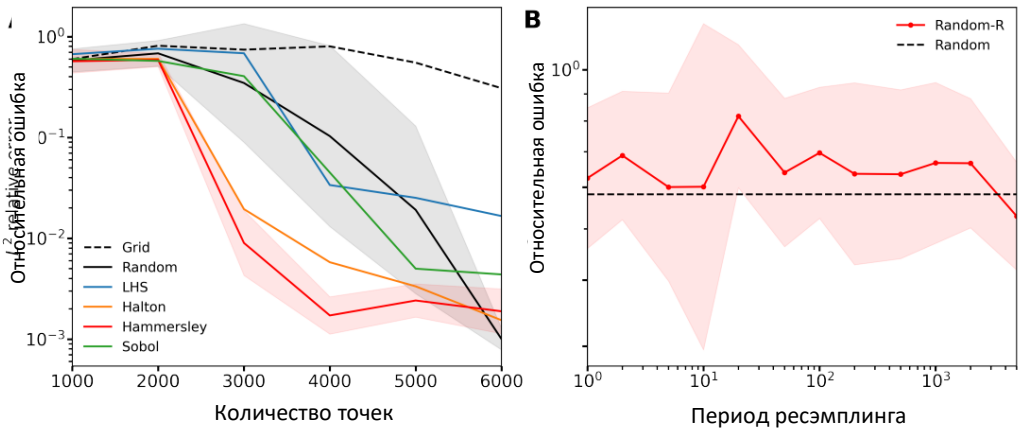


Рис.8. Сравнение методов сэмплирования для волнового уравнения

Из таблицы 3 можно сделать вывод, что в большинстве из рассматриваемых задач метод RAD показывает наилучшую точность. Он в дальнейшем и будет использован.

Глава 2. НЕЙРОННЫЕ ОПЕРАТОРЫ

Классическое развитие нейронных сетей в основном было направлено на обучение отображений между конечно-размерными евклидовыми пространствами, так как в задачах, компьютерного зрения (CV), обработки и генерации текста (NLP), а также других методов главного фарватера развития машинного и глубокого обучения, входные данные дискретизированы по времени/пространству. В случае с задачами научного машинного обучения (SciML), основным объектом изучения являются непрерывные отображения между входными (начальными и граничными условиям) и выходными данными (решениями дифференциальных уравнений). Для этой цели были созданы нейронные операторы, которые непосредственно учат отображение от любой функциональной параметрической зависимости к решению. Таким образом, они изучают целое семейство PDE, в отличие от классических методов, которые решают один экземпляр уравнения. В данной главе описан нейронный оператор, который параметризует интегральное ядро непосредственно в пространстве Фурье, что позволяет создать эффективную с вычислительной точки зрения архитектуру. Кроме того, он достигает более высокой точности по сравнению с предыдущими решателями, основанными на обучении при фиксированном разрешении. Традиционные решатели, такие как методы конечных элементов (МКЭ) и методы конечных разностей (МКР), решают уравнение путем дискретизации пространства. Поэтому они накладывают компромисс на разрешение: крупные сетки быстры, но менее точны; мелкие сетки точные, но медленные. Сложные системы PDE, как описано выше, обычно требуют очень тонкой дискретизации, а значит, очень сложны и трудоемки для традиционных решателей. С другой стороны, методы, основанные на данных, могут непосредственно изучать оператор семейства уравнений на основе данных. В результате метод, основанный на обучении, может быть на порядки быстрее традиционных решателей.

Методы машинного обучения могут стать ключом к революции в научных дисциплинах, предоставляя быстрые решатели, которые приближают или улучшают традиционные [1,30,31].

Ограничением нейронных сетей, при суррогатном обучении является фиксированная дискретизация данных, на которых она обучалась. Данная проблема - это также одна из причин почему требовалась разработка нейронных сетей, инвариантных к сетке. Кроме того, нейронный оператор нужно обучать только один раз. Получение решения для нового экземпляра параметра требует только прямого прохода сети, что снимает основные вычислительные проблемы, возникающие в методах нейронного МКЭ. Самым важным достоинством нейронного оператора (NO) является способность обучения дифференциальному оператору в неявном виде на основе данных, что значительно повышает гибкость данного метода. Благодаря быстрому преобразованию Фурье создается эффективная архитектура с точки зрения как высокой точности, так и вычислительной скорости.

Рассмотрим нейронный оператор как отображение между двумя пространствами бесконечной размерности на основе конечного набора наблюдаемых пар вход-выход.

Пусть $D \subset R^d$ ограниченно на открытом множестве и $\mathcal{A} = \mathcal{A}(D; R^{d_a})$ и $\mathcal{U} = \mathcal{U}(D; R^{d_u})$ разные банаховы пространства функций, принимающие значения в R^{d_a} и R^{d_u} соответственно. Определим $\mathrm{G}^\dagger: \mathcal{A} \to \mathcal{U}$ нелинейное отображение. Главной целью является рассмотрение операторов, которые возникают при решении дифференциальных уравнений в частных производных. Предположим, что есть наблюдения $\{a_j, u_j\}_{j=1}^N$ где $a_j \sim \mu$ независимые одинаково распределенные случайные величины из вероятностной меры μ на $\mathcal{A}$ и $u_j = \mathrm{G}^\dagger(a_j)$. Построение аппроксимации $\mathrm{G}^\dagger$ путем построения параметрической карты.

$$G: \mathcal{A} \times \theta \to \mathcal{U} \text{ или } G: \mathcal{A} \to \mathcal{U}, \theta \in \Theta \qquad (2.1)$$

Для некоторого конечно - мерного пространства параметров Θ выбрав $\theta^\dagger \in \Theta$.

Аппроксимация оператора $\mathrm{G}^\dagger$ является более сложной задачей, чем поиск решения $u \in \mathcal{U}$ PDE для одного экземпляра. Нейронный оператор предложенный в

[32] сформулирован как итеративная архитектура $v_0 \to v_1 \to \cdots \to v_T$, где v_j – последовательность функций. Таким образом строится архитектура:

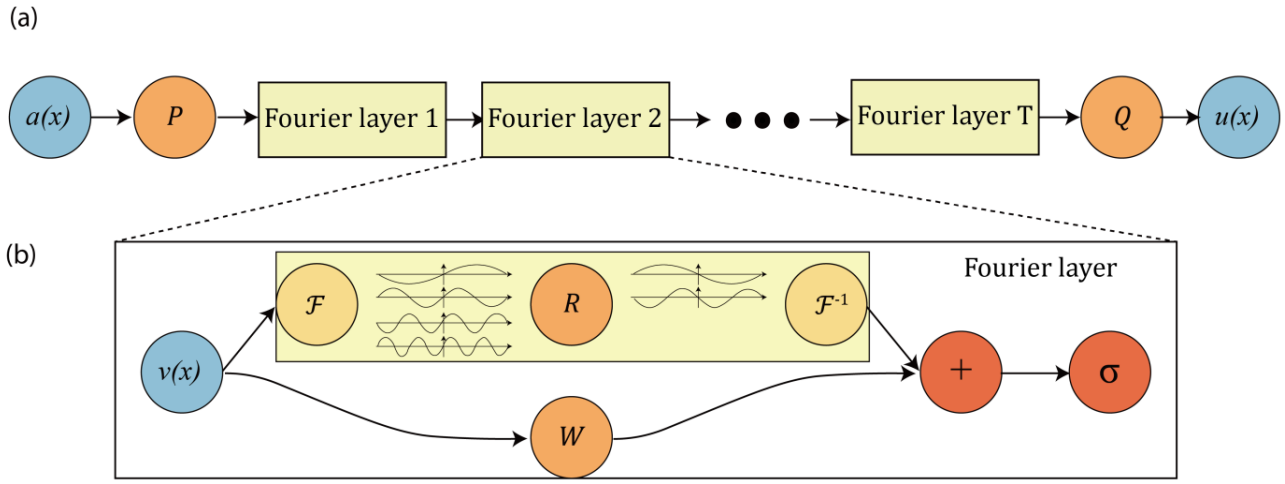


Рис.9. (a) Полная архитектура нейронного оператора Фурье. (b) Слой Фурье.

где $a(x)$ – входные параметры, $\mathcal{F}$ – быстрое преобразование Фурье, R – линейная функция (матрица свёртки) в пространстве Фурье с обучаемыми параметрами, W – смещение для учёта не периодических ГУ. σ – нелинейное преобразование, типичное в парадигме NN. P и Q – операторы отображения (NN) для увеличения и уменьшения размерности соответственно.

Определим итерацию $v_t \to v_{t+1}$ для оператора Фурье:

$$v_{t+1} := \sigma\big(W v_t(x) + (\mathcal{K}(a;\phi)v_t)(x)\big), \qquad \forall x \in D$$

где $\mathcal{K}: \mathcal{A} \times \Theta_{\mathcal{K}} \to \mathcal{L}\left(\mathcal{U}(D;R^{d_v}), \mathcal{U}(D;R^{d_v})\right)$ отображение на ограниченные линейные операторы из $\mathcal{U}(D;R^{d_v})$ и параметризация $\phi \in \Theta_{\mathcal{K}}$, $W: R^{d_v} \to R^{d_v}$ это линейное преобразование и $\sigma: R \to R$ - нелинейное преобразование, применяемое поточечно. $\mathcal{K}$ – является в данном случае ядром интегрального преобразования, параметризированного нейронной сетью.

Определим интегральное отображение оператора ядра:

$$(\mathcal{K}(a;\phi)v_t)(x) := \int_D \kappa(x,y,a(x),a(y);\phi)v_t(y)dy, \qquad \forall x \in D \qquad (2.3)$$

где $\kappa_\phi: R^{2(d+d_a)} \to R^{d_v \times d_v}$ это параметризованная нейронная сеть.

Здесь κ_ϕ играет роль функции ядра, которую мы изучаем на основе данных/физически информированной функции потерь. Можно заметить, что параметризованный таким образом оператор ядра может обучаться любым нелинейным зависимостям из данных за счет многократного использования нелинейных функций активации.

Если убрать зависимость от функции a и сделать предположение, что $\kappa_\phi(x, y) = \kappa_\phi(x - y)$, то получится, что (2.3) является оператором свертки, такое предположение вносит индуцированное смещение о инвариантности оператора, что является логичной и часто использующейся характеристикой ядра, например ядро свертки в сверточных нейронных сетях. Мы используем этот факт в следующем разделе, параметризуя κ_ϕ непосредственно в пространстве Фурье и используя быстрое преобразование Фурье (БПФ) для эффективного вычисления (3). Именно такое построение нейронного оператора является на данных момент самым эффективным с практической/инженерной точки зрения.

Заменим интегральный оператор ядра в (3) на оператор свертки, определенный в пространстве Фурье:

$$(\mathcal{F}f)_j = \int_D f_j(x) e^{-2\pi i \langle x,k \rangle} dx, \quad (\mathcal{F}^{-1}f)_j(x) = \int_D f_j(x) e^{2\pi i \langle x,k \rangle} dk \qquad (2.4)$$

Таким образом из теоремы о свертке получается:

$$(\mathcal{K}(a;\phi)v_t)(x) \coloneqq \mathcal{F}^{-1}(\mathcal{F}(\kappa_\phi) \cdot (\mathcal{F}v_t)(x), \qquad \forall x \in D \qquad (2.5)$$

$$(\mathcal{K}(a;\phi)v_t)(x) \coloneqq \mathcal{F}^{-1}(R_\phi \cdot (\mathcal{F}v_t)(x), \qquad \forall x \in D \qquad (2.6)$$

где R_ϕ это Фурье образ периодической функции $R_\phi(\kappa) \in \mathbb{C}^{d_v \times d_v}$.

Для того чтобы эффективно считать слой с интегральным ядром, вместо ассимтотической сложности $O(n^2) \to nlog(n)$, что является значительным ускорением расчетов, необходимо использовать быстрое преобразование Фурье (FFT) на равноудаленной равномерной сетке.

При равномерной дискретизации с разрешением $s_1 \times \ldots \times \mathrm{s_d} = n$ для $f \in R^{n \times \mathrm{d}_v}$,

$k = (k_1, \ldots k_d) \in \mathbb{Z}_{s_1} \times \ldots \times \mathbb{Z}_{s_d}$ и $x = (x_1, \ldots, x_d) \in D$ прямой и обратный FFT можно определить как:

$$\left(\hat{\mathcal{F}}\mathrm{f}\right)_{\mathrm{l}}(k) = \sum_{x_1=0}^{s_1-1} \ldots \sum_{x_d=0}^{s_d-1} f_l(x_1, \ldots, x_d) e^{-2i\pi \sum_{j=1}^{d} \frac{x_j k_j}{s_j}} \tag{2.7}$$

$$\left(\widehat{\mathcal{F}^{-1}}\mathrm{f}\right)_{\mathrm{l}}(k) = \sum_{x_1=0}^{s_1-1} \ldots \sum_{x_d=0}^{s_d-1} f_l(k_1, \ldots, k_d) e^{2i\pi \sum_{j=1}^{d} \frac{x_j k_j}{s_j}} \tag{2.8}$$

где k – частота в пространстве Фурье образов, фактически в дискретном случае урезается часть частот, для увеличения скорости расчетов, для высокочастотных случаев архитектура также эффективна за счет нелинейных преобразований на выходе каждого слоя. Важнейшим свойством таких преобразований является то, что Слои Фурье инвариантны к дискретизации, потому что они могут обучаться и оценивать функции, дискретизированные произвольным образом.

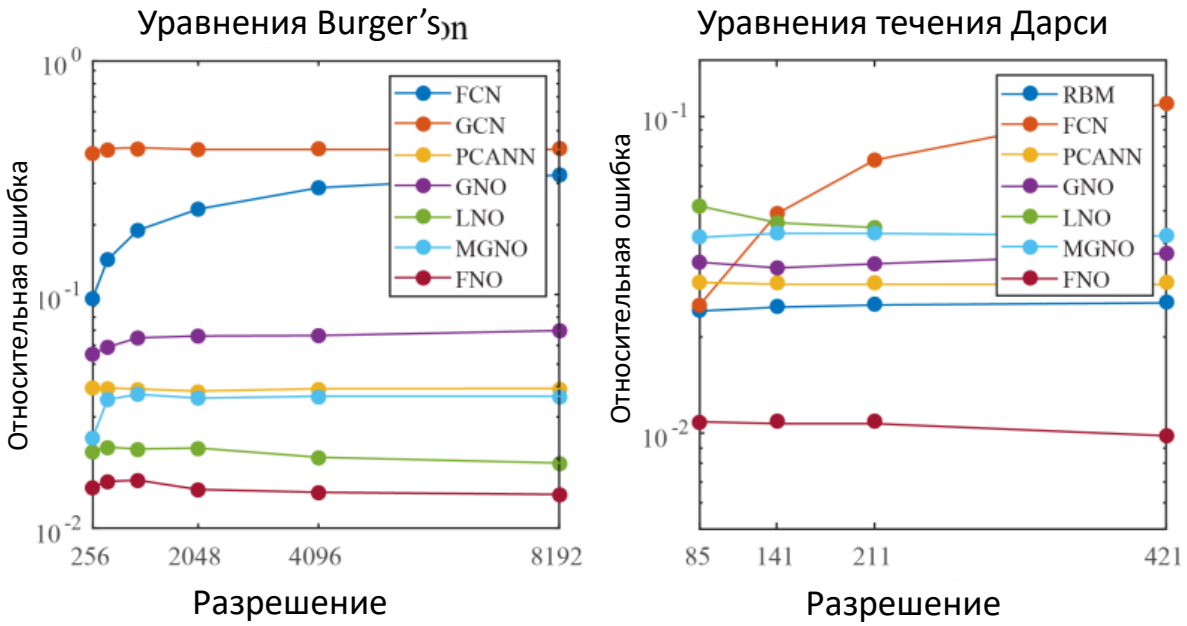


Рис.10. Пример сравнения стандартных архитектур и нейронного оператора Фурье для различных разрешений

Глава 3. ФИЗИЧЕСКИ – ИНФОРМИРОВАННЫЙ НЕЙРОННЫЙ ОПЕРАТОР

В данной главе рассматривается модель, включающая в себя подходы из прошлых глав, нейронные операторы с физическими данными (PINO), которые объединяют обучающие данные и физические ограничения для обучения оператора решения заданного семейства параметрических дифференциальных уравнений (PDE).

PINO - это гибридный подход, включающий данные и ограничения PDE с разным разрешением для обучения оператора. В частности, в PINO объединяются обучающие данные с грубым разрешением с ограничениями PDE, наложенными с более высоким разрешением. В PINO используется нейронный оператор Фурье (FNO), который гарантированно является универсальным аппроксиматором для любого непрерывного оператора и сходится к дискретизации в пределе уточнения сетки. Посредством добавляя PDE-ограничения к FNO на более высоком разрешении, мы получаем высокоточную реконструкцию истинного оператора. Более того, PINO успешно работает в условиях, когда отсутствуют обучающие данные и накладываются только PDE-ограничения.

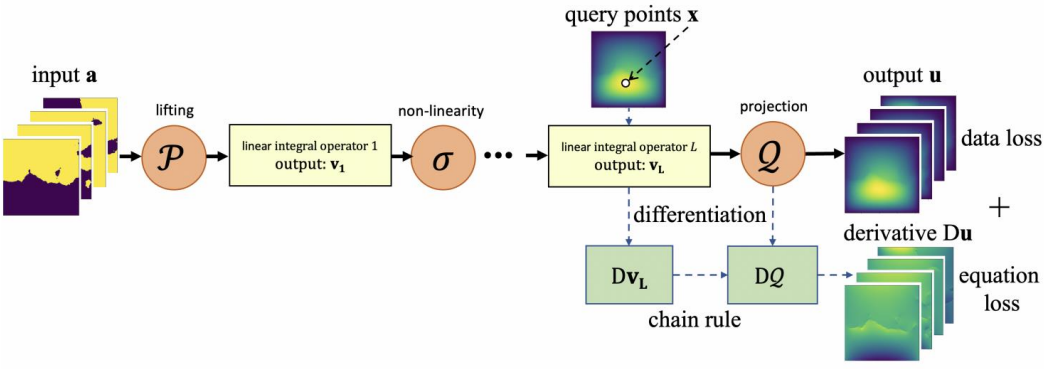


Рис.11. PINO обучает нейронный оператор, используя как обучающие данные, так и функцию потерь PDE.

На рисунке показана архитектура нейронного оператора с оператором повышения размерности входа в скрытом пространстве, который получает входную

функцию a и выдает функцию v_0 с большей размерностью. За этой операцией следуют L слоев, которые вычисляют линейные интегральные операторы, затем нелинейные функции активации, и последний слой, который выводит функцию v_L. Оператор точечной проекции проецирует v_L на выходную функцию u. И v_L, и u являются функциями, и все их производные (Dv_L, Du) могут быть вычислены в точной форме в любых точках запроса x.

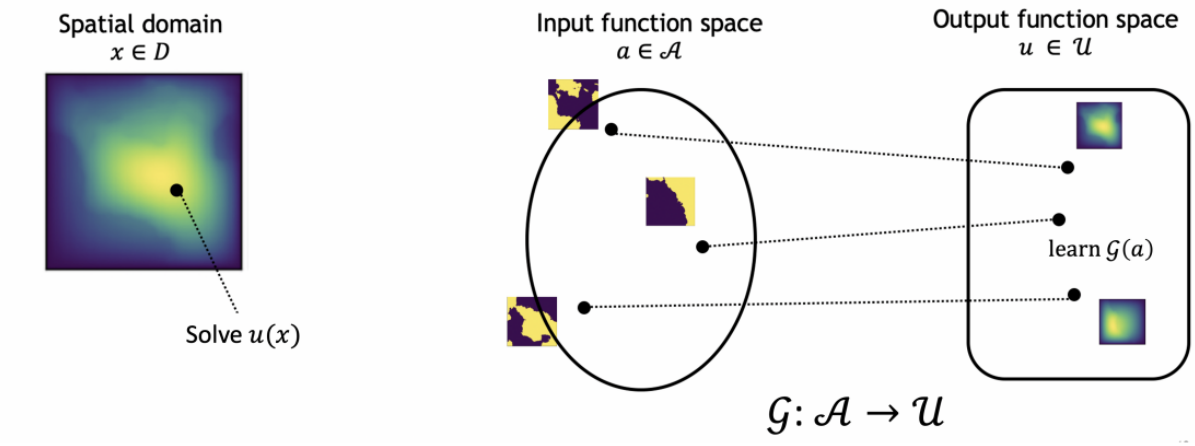
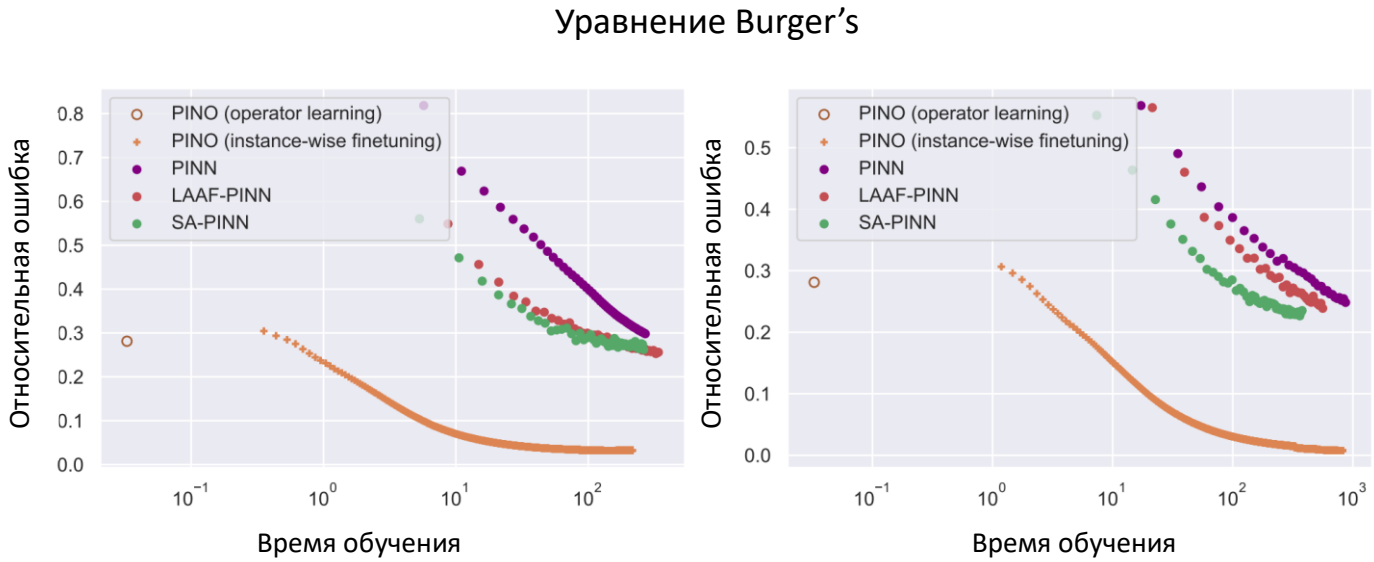


Рис.12. Решение для одного конкретного экземпляра стиха изучает весь оператор решения. Слева: численные решатели и PINN сосредоточены на решении одного конкретного случая. Справа: нейронные операторы изучают оператор решения оператора для семейства уравнений



.

Рис.13. Сравнение точности State of art методов и PINO на различных разрешениях. Уравнение Burger’s.

Глава 4. РЕЗУЛЬТАТЫ

В этой главе рассматриваются приложение методов, описанных в предыдущих главах, в задачах нефтяной индустрии. Главный фокус направлен на исследование применения методов физически – информированного машинного обучения к математическим постановкам, описывающим процессы разработки месторождения. Главным образом текущие результаты, являются начальным этапом развития концепции гибридного моделирования в задачах нефтяной индустрии.

Гибридное моделирование - инновационный метод, соединяющий машинное обучение и математические модели физических процессов. Основой этого подхода является использование как собранных реальных данных, так и знаний о физических свойствах системы при обучении искусственного интеллекта, что позволяет обеспечить более точное и эффективное управление моделируемой системы.

4.1 Задача вытеснения. PINN

Поддержание пластового давления (ППД) путем закачки воды в нефтяной пласт через нагнетательные скважины является основным методом оптимизации разработки и повышения коэффициента извлечения нефти (КИН).

В идеальной модели общий объем извлекаемой жидкости должен быть равен объему закачиваемой воды. Однако в реальных условиях это невозможно. Закачиваемая для ППД вода может либо отклоняться в неизвестном направлении, не достигая целевого пласта, либо, наоборот, прорываться к добывающим скважинам по высокопроницаемым каналам, значительно увеличивая содержание воды в добываемой нефти и оставляя неиспользованными плохо дренируемые участки пласта. В обоих вышеупомянутых случаях заводнение нельзя считать эффективным. Для обеспечения эффективности системы заводнения необходимо своевременно выявлять проблемные участки и проводить соответствующие мероприятия на нефтедобывающих и нагнетательных скважинах. Задача оценки, управления и прогнозирования эффективности различных вариантов систем заводнения значительно упрощается при наличии постоянно функционирующей геолого-

гидродинамической модели. Однако из-за высокой сложности, трудностей адаптации, необходимости постоянного обновления, а также наличия специальных знаний и опыта такие модели разрабатываются далеко не для всех девелоперских проектов. Именно поэтому внедрение упрощенных моделей, способных в кратчайшие сроки и с высокой точностью выдавать рекомендации по оптимизации, было и остается особенно актуальным.

В своей простейшей форме модель заводнения может быть представлена как одномерная модель течения несжимаемых несмешивающихся жидкостей в условиях, когда капиллярным давлением и гравитационными силами можно пренебречь. Дифференциальное уравнение, описывающее этот процесс, приведено в уравнении 4.1:

$$\frac{dS}{dt} + \frac{dF(S)}{dx} = 0 \tag{4.1}$$

где S - водонасыщенность; F(S) - функция распределения фазового потока; t - безразмерное время; x - безразмерная координата.

Решение задачи будет зависеть от заданных начальных и граничных условий, а также от кривой относительной проницаемости (RPC):

$$F(S) = \frac{f_w(S)}{f_w(S) + f_o(S) \cdot \frac{\mu_w}{\mu_o}} \tag{4.2}$$

где $f_w(S)$ - относительная проницаемость для воды; $f_o(S)$ - относительная проницаемость для нефти; μ_w - вязкость воды; μ_o - вязкость нефти.

$$f_o(S) = kro_{max} \cdot \left(\frac{1 - S - S_{or}}{1 - S_{or} - S_{wc}}\right)^{c_o} \tag{4.3}$$

где kro_{max} - максимальная относительная проницаемость для нефти; S_{or} - остаточная нефтенасыщенность; S_{wc} - связанная водонасыщенность; c_o - коэффициент Кори для нефти.

$$f_w(S) = krw_{max} \cdot \left(\frac{S - S_{wc}}{1 - S_{or} - S_{wc}}\right)^{c_w} \quad (4.4)$$

где krw_{max} - максимальная относительная проницаемость для воды; c_w - коэффициент Кори для воды.

В данном примере граничным условием является насыщенность 0,7 на нагнетательной скважине, а начальным условием - пространственная насыщенность 0,37 (9) в начале периода времени.

$$\begin{cases} S(t, x_{source}) = 0.7 \\ S(0, x) = 0.37 \end{cases} \quad (4.5)$$

Кривая относительной фазовой проницаемости определяется следующими параметрами:

1. Вязкость воды, $\mu_w = 2$
2. Вязкость нефти, $\mu_o = 1$
3. Максимальная относительная проницаемость для нефти, $kro_{max} = 1$
4. Максимальная относительная проницаемость для воды, $krw_m = 0.7$
5. Остаточная нефтенасыщенность, $S_{or} = 0.3$
6. Связанная водонасыщенность, $S_{wc} = 0.3$
7. Коэффициент Кори для нефти, $c_o = 4$
8. Коэффициент Кори для воды, $c_w = 1.5$

Область, в которой строится решение, ограничена максимальными значениями $t_{max} = 3.18$, $x_{max} = 1.38 * 10^9$.

В процессе обучения модели PINN полученное решение имеет тенденцию к сглаживанию. Поэтому для получения фронта поршня смещения авторы работы [33] предложили следующий подход: модифицировать существующий профиль объективной функции с помощью кусочно-линейного преобразования (функция Хевисайда), формируя линейный участок до точки пересечения с исходной функцией

(рис. 1), и сохраняя исходную функцию после точки пересечения с исходной функцией (10).

$$F^*(S) = \begin{cases} \sigma S, 0 \le S \le S^* \\ F(S), S \ge S^* \end{cases} \quad (4.6)$$

где $\sigma = \frac{F(S^*)}{S^*}$, S^* - точка пересечения с исходной кривой относительной проницаемости.

Добавим к уравнению (уравнение 4.1) небольшой диффузионный член или член вязкости.

$$\frac{dS}{dt} + \frac{dF(S)}{dx} = \epsilon \cdot S_{xx} \quad (4.7)$$

где $\epsilon = 2.5 \cdot 10^{-3}$.

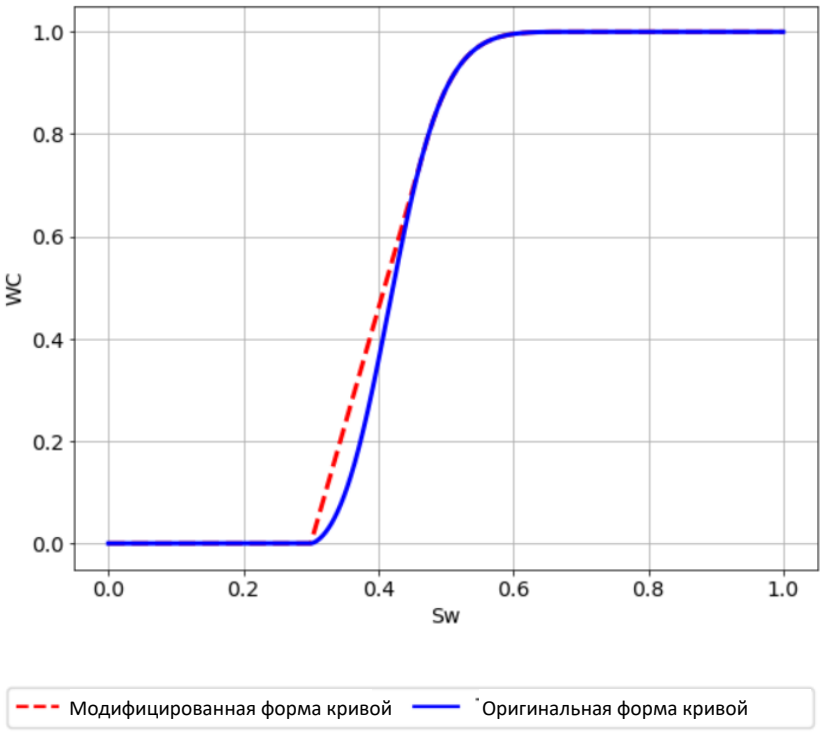


Рис.14. Кривая относительной проницаемости.

В результате моделирования мы получаем решение, которое совпадает с решением, полученным на основе численной схемы с использованием разностей

upwind (рис. 2). Средняя квадратическая ошибка (MSE) составляет $0.0285 \cdot 10e^{-3}$, а средняя абсолютная процентная ошибка (MAPE) – 1.23%.

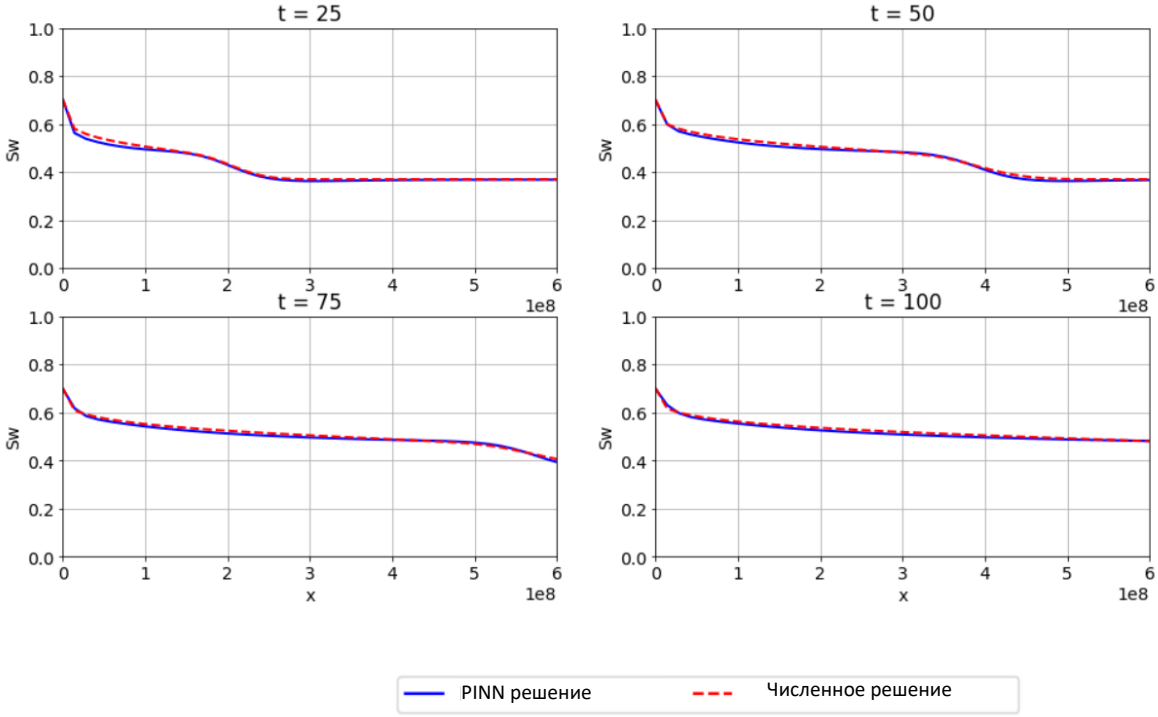


Рис.15. Сравнение работы модели PINN в различные моменты времени t.

Таким образом, по результатам расчетов мы получаем распределение насыщенности вдоль каждой линии потока, что позволяет рассчитать обводненность добывающих скважин, которые служат выходами этих линий потока. Кроме того, поскольку координаты узлов русловых линий известны, значения насыщенности легко перенести на регулярную двумерную сетку с помощью интерполяции или кригинга. Это позволяет нам получить карту текущей насыщенности пласта, которая имеет большое значение для бизнеса, так как визуализирует фронт вытеснения. На основе этой информации можно принимать решения о корректировке темпов закачки воды, бурении новых скважин, отсечении боковых ответвлений и других мерах по оптимизации. При использовании нейросетевых моделей, помимо высокого качества решения, важно также учитывать скорость получения решения. По сравнению с численными методами, обычно используемыми для решения простых

дифференциальных уравнений, нейросетевые модели часто демонстрируют значительно более медленное время решения. Это связано с тем, что такой подход и архитектура нейронных сетей могут обеспечить решение только для конкретно заданных начальных и граничных условий. Если требуются решения для других значений параметров, необходимо переобучить модель.

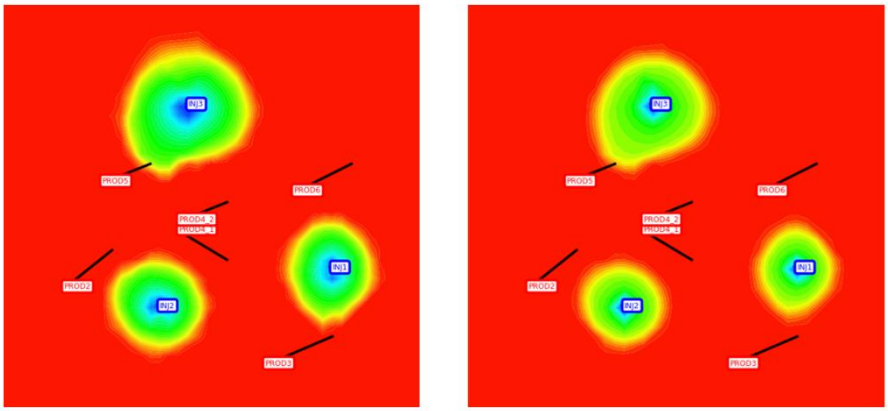


Рис.16. Карта водонасыщения (слева - решение по численной схеме с восходящими разностями ветра, справа - решение по модели PINN).

Пример построения карты водонасыщенности по произвольной системе разработки с использованием численной схемы с восходящими разностями ветра и на основе решения модели PINN. Модель PINN демонстрирует высокую точность (метрика MSE 0,0189) по сравнению с численной схемой с восходящими разностями, что позволяет использовать ее для построения решения с равномерной начальной водонасыщенностью пласта.

Общее время, необходимое для получения решения с помощью нейросетевых моделей, включает два основных этапа: время обучения и время вывода. Время обучения зависит от масштаба и сложности задачи, архитектуры модели и общего объема обучающих данных. Время вывода, с другой стороны, зависит от сложности модели и доступных вычислительных мощностей. Результаты сравнения с численными методами представлены в таблице 4.

Метод	Скорость обучения, с	Скорость инференса, с
Численное решение	-	0.069
PINN	174	0.071

Таблица.4 Сравнение метрик

Действительно, стоит отметить, что обучение нейронной сети может осуществляться с помощью графического процессора (GPU) или сети компьютеров с несколькими GPU, что приводит к распараллеливанию вычислений и ускорению процесса. В данной статье использовался базовый сценарий с одним центральным процессором (CPU) с несколькими ядрами для предоставления вычислительных ресурсов для обучения и, в некоторых случаях, GPU, ускоряющего этап обучения. Для описания многоскважинной системы разработки также важно знать значения поля пластового давления для управления параметрами закачки и добычи скважин.

4.2 Задача моделирования поля давления в пласте. PINN

Моделирование скважинной системы - это описание фильтрации флюидов в пласте. Одним из основных уравнений для расчета поля давления при межскважинном взаимодействии является уравнение переходной пьезопроводности. Рассмотрим это уравнение в простейшей одномерной формулировке с точечным источником для случая одной скважины:

$$\frac{1}{\chi}\frac{\partial p}{\partial t} - \frac{\partial^2 p}{\partial x^2} = \mathrm{q}\delta(\mathrm{x}_0, t_0) \tag{4.8}$$

где p - распределение давления; χ - коэффициент пьезопроводности, δ - дельта-функция Дирака, q - расход жидкости; x_0, t_0 - координата и время начала воздействия источника на пласт.

Чтобы решить проблему дискретности обобщенной функции Дирака, мы меняем формулировку задачи без потери физичности, а также учитываем начальные и граничные условия:

$$\frac{1}{\chi}\frac{\partial p}{\partial t} - \frac{\partial^2 p}{\partial x^2} = 0 \tag{4.9}$$

$$p = p_0, \;\; t = 0 \tag{4.10}$$

$$\tilde{q}|_{x=x_w} = \frac{q}{2\pi x} = -\frac{kh}{\mu B}\frac{\partial p}{\partial x}|_{x=x_w}, \qquad t > 0 \tag{4.11}$$

$$p|_{x=x_e} = p_e, \qquad t > 0 \tag{4.12}$$

Для упрощения и обобщения системы уравнений (4.9) - (4.12) введем в рассмотрение величину снижения давления и безразмерные величины:

$$p_D = \frac{p(x,t) - p_0}{p_{ref} - p_0}, \qquad x_D = \frac{x}{x_e}, \qquad x_{eD} = 1, \qquad t_D = \frac{\chi t}{x_e^2}, \qquad C = \frac{khx_e}{\mu B(p_{ref} - p_0)}$$

$$\frac{\partial p_D}{\partial t_D} - \frac{\partial^2 p_D}{\partial x_D^2} = 0$$

$$p_D = p_0, \;\; t = 0 \tag{4.13}$$

$$\tilde{q}|_{x=x_w} = -C\frac{\partial p_D}{\partial x_D}|_{x=x_w}, \qquad t > 0$$

$$p|_{x=x_e} = p_e, \qquad t > 0$$

В результате решения системы (4.13) методом конечных разностей моделируется закачка воды в пласт. PINN с 5 скрытыми слоями по 256 нейронов был обучен 50000 итерациями оптимизатора Adam со скоростью обучения 0.001.

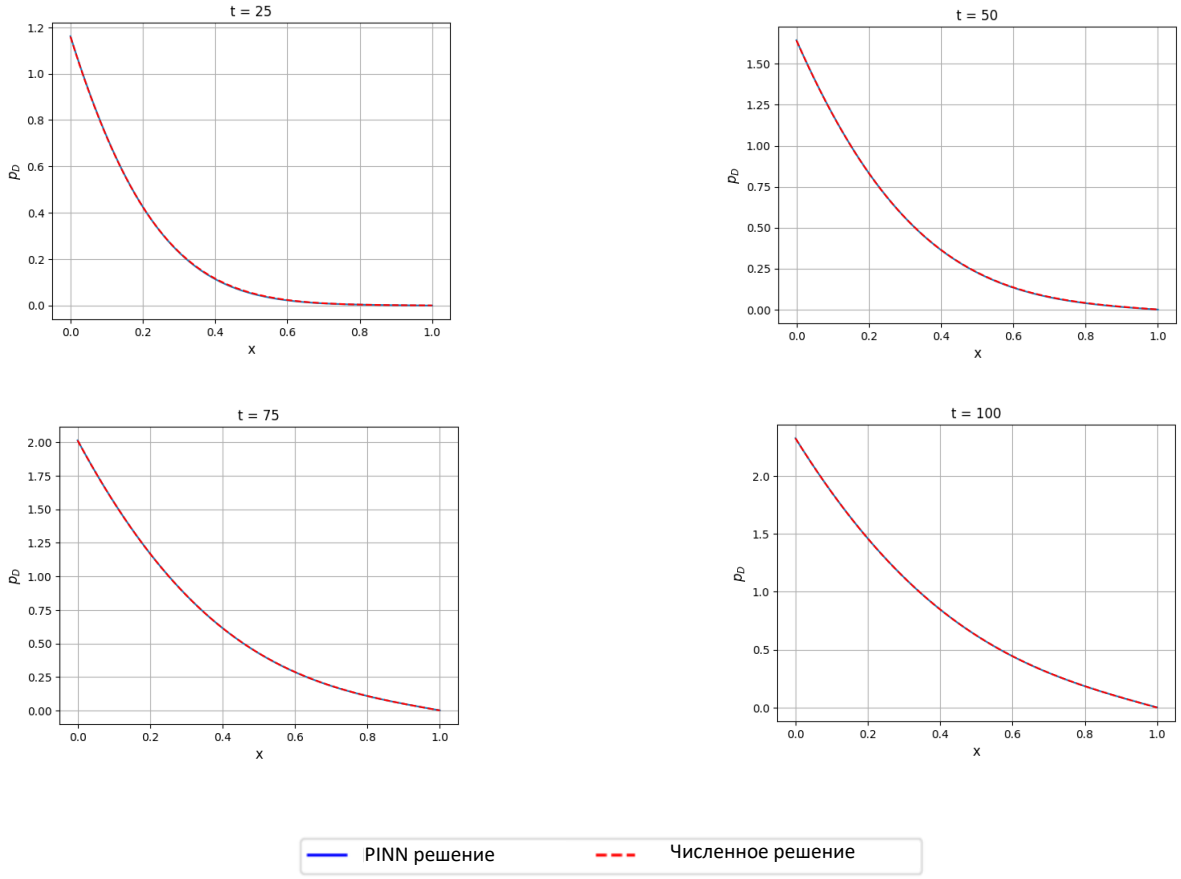


Рис.17. Сравнение работы модели PINN и решения методом конечных разностей в различные моменты времени t.

Средняя квадратическая ошибка (MSE) составляет $6.14 \cdot 10^{-7}$, а относительная процентная ошибка – 0.24%. Результаты, полученные в этой главе, показывают, что PINN способны воспроизводить решение уравнения фильтрации с высокой точностью, что может быть распространено на многоскваженные системы.

4.3 Задача моделирования газонефтяного контакта. PINN

Существуют нефтяные месторождения, где между водоносным горизонтом и газовой шапкой находится тонкий нефтяной слой. Во время сжатия газа добыча уменьшит локальный газонефтяной контакт вблизи скважины. После прорыва газа соотношение газ/нефть (GOR) из скважины может сильно меняться в зависимости от дебита. Возможность предсказать эту зависимость необходима для оптимизации добычи на таких месторождениях.

Математическая модель, получившая название GORM, позволяет предсказать поведение системы газоснабжения и результирующий показатель GOR в зависимости от скорости. Моделируется горизонтальная составляющая потока, перпендикулярная скважине. Водонефтяной контакт (ВНК) рассматривается как непроницаемый - упрощение, которое может быть оправдано до тех пор, пока движение НВК ограничивается локальным скоплением воды вблизи скважины. Уравнение называется уравнением Дюпюи-Форгеймера, граничные и начальные условия характеризуют соотношение газ/нефть:

$$\frac{\partial h}{\partial t} = \frac{k\left(\rho_o - \rho_g\right)g}{\mu\varphi}\frac{\partial}{\partial x}\left(h\frac{\partial h}{\partial x}\right) \tag{4.14}$$

$$h(0,x) = h_0, \;\; t = 0 \tag{4.15}$$

$$\tilde{q}_{oil} = -2uh = \frac{2khg\Delta p}{\mu}\frac{\partial h}{\partial x}|_{x=0}, \qquad t > 0 \tag{4.16}$$

$$\frac{\partial h(t,W)}{\partial x} = 0, \qquad t > 0 \tag{4.17}$$

где $\tilde{q}_{oil}$ - дебит нефти на единицу скважины, φ - коэффициент эффективной пористости, h_0 - начальная высота столба нефти, W - ширина зоны дренирования скважины.

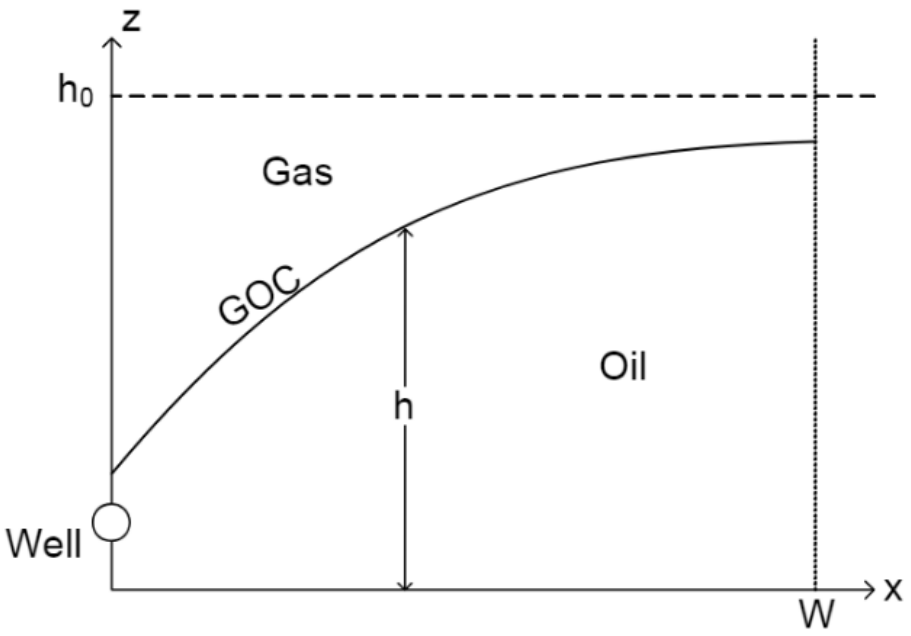


Рис.18. Газонефтяной контакт в разрезе

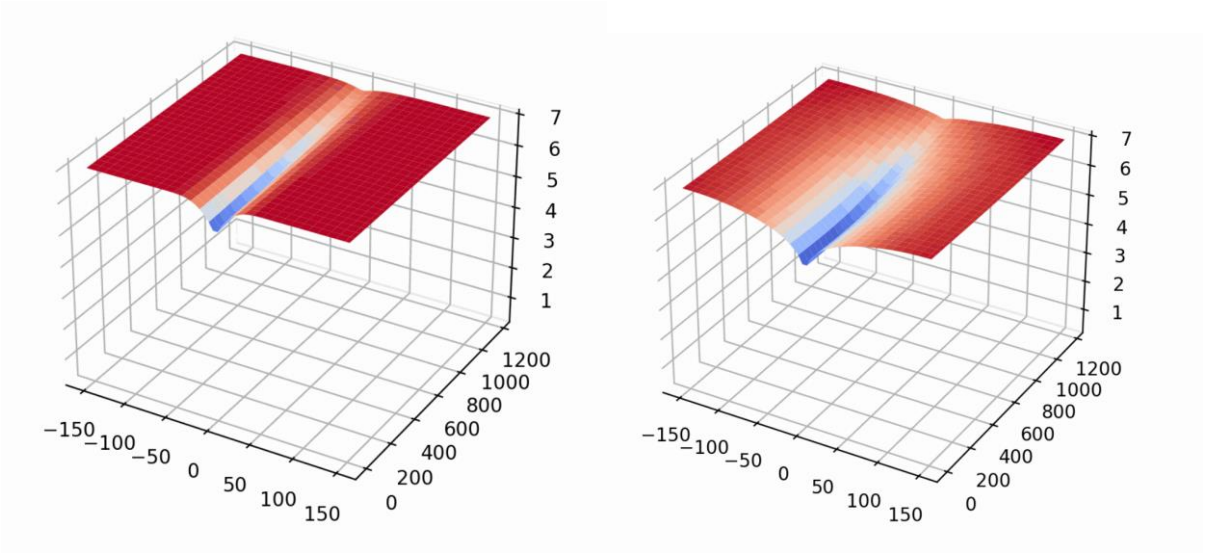


Рис.19. Решение PINN в разные моменты времени

Для этого случая относительная ошибка решения PINN и численного решения составляет менее 1%.

Таким образом, проанализировав применимость PINN для решения различных задач, описываемых гиперболическими и параболическими дифференциальными уравнениями. Точность согласования гибридных моделей зависит в первую очередь

от настройки моделей и может достигать долей процента, что говорит о высокой степени применимости подхода.

Ключевыми проблемами подхода остается проблема работы с произвольными начальными условиями, не участвовавшими в обучении, без переобучения модели.

4.4 Задача моделирования вытеснения. PINO

Использование PINN подхода с нейронными сетями, хоть и дает высокую точность на рассмотренных примерах, но достаточно ограничено в возможностях из за обучения только лишь одному варианту параметров.

В следствии чего в данной главе используется физически – информированный нейронный оператор (PINO), который за счет обучения дифференциального оператора, способен без дополнительного переобучения давать высокую точность на всем пространстве параметров.

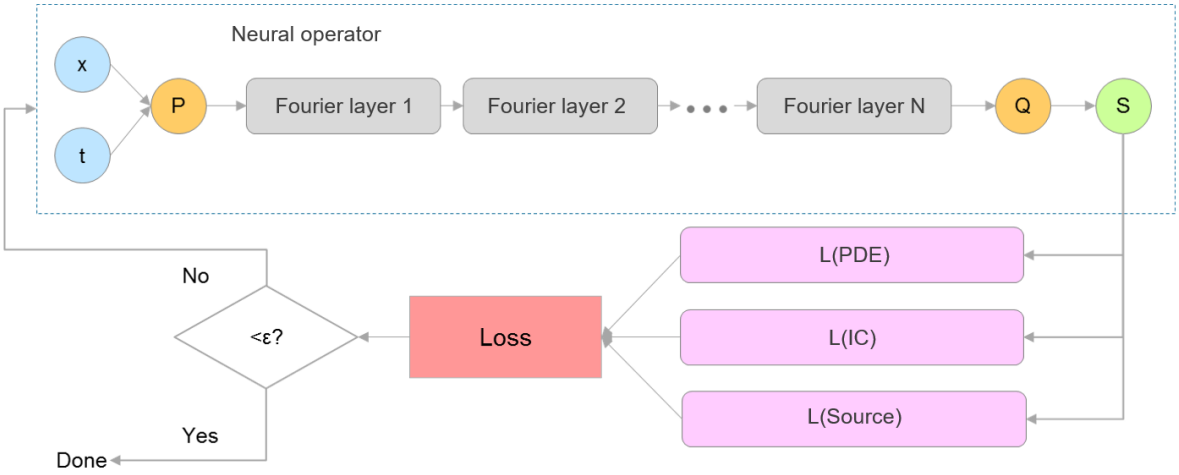


Рис.20. Архитектура модели PINO.

Для обучения модели PINO необходимо сгенерировать входные данные. В качестве примера зафиксируем граничное условие - насыщенность на нагнетательной скважине, и будем варьировать начальные условия. Начальные условия получаются путем построения карты начальной водонасыщенности в пласте M и извлечения значений насыщенности вдоль произвольного направления (рис. 21).

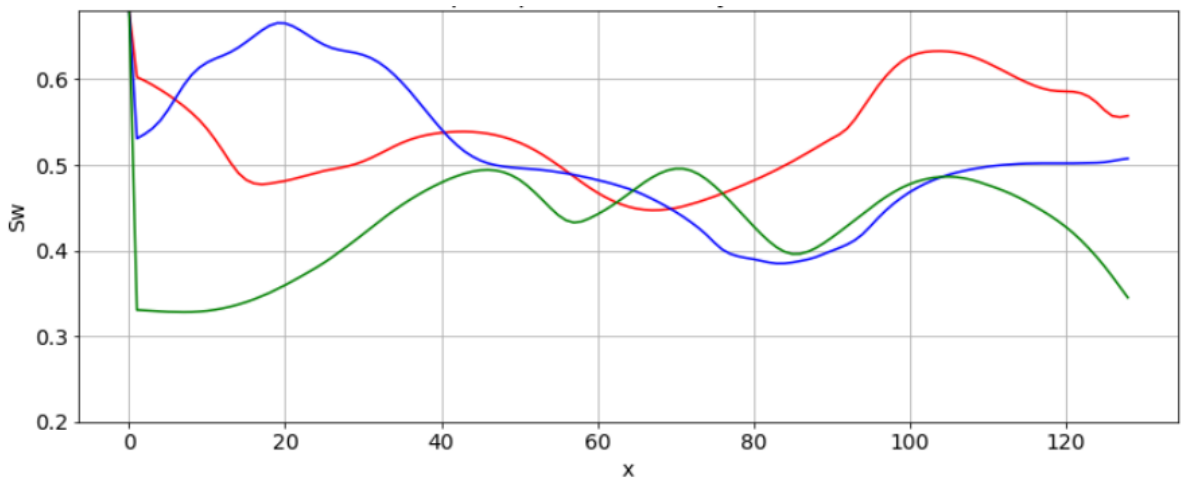


Рис.21. Пример нескольких взятых начальных водонасыщенностей

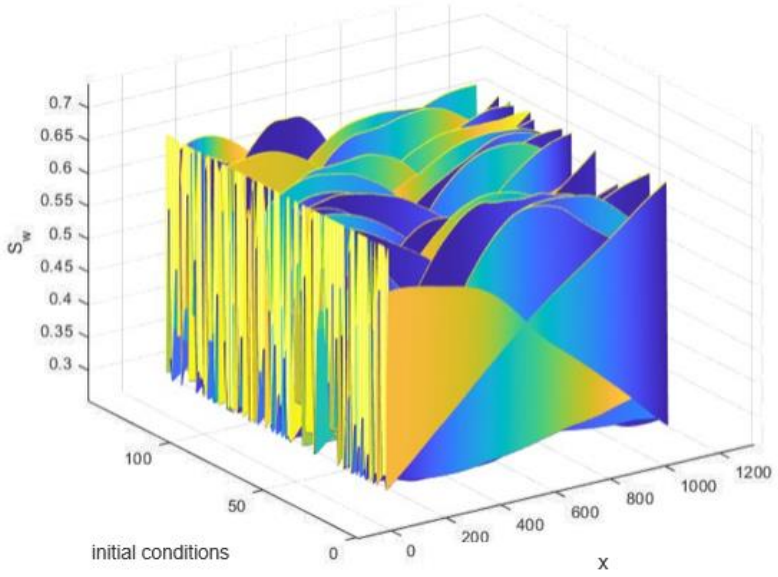


Рис.22. Массив начальных условий

Обученная модель достигла следующих показателей: MSE = 3.9278e-05, MAPE = 1.36%. Примеры предсказаний показаны на рисунке 23:

Аналогично моделям PINN, представим результаты моделирования карты насыщения для модели PINO (рис. 10). В отличие от моделей PINN, модели PINO позволяют строить карты в условиях неоднородной начальной водонасыщенности пласта. Анализируя полученные результаты, можно отметить расхождения в прогнозируемых итогах. Это расхождение можно объяснить появлением длинных

линий трассировки, чья безразмерная координата x превышает максимальное значение x в обучающем наборе данных.

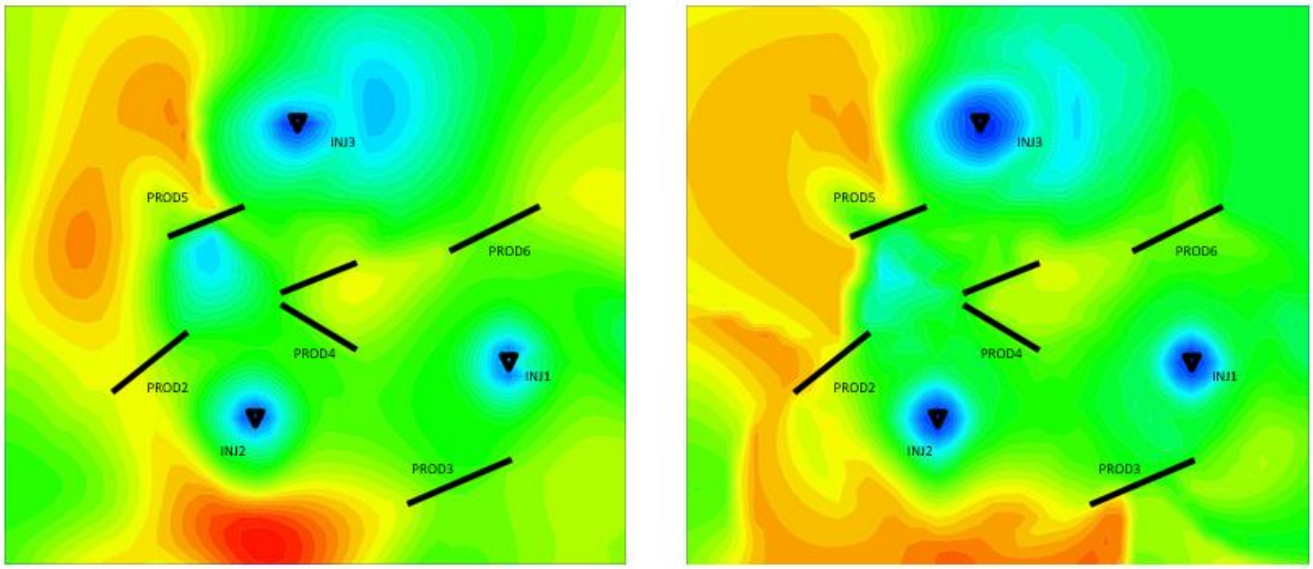


Рис.23. Карта насыщенности (слева - решение по upwind разностной численной схеме, справа - решение по модели PINO).

4.5 Добавление замерных данных в PINN модель

Ключевой характеристикой гибридных моделей является их способность обеспечивать последовательное решение, которое не только удовлетворяет различным физическим моделям, но и улавливает закономерности, выходящие за рамки математических моделей, представленных в реальных данных. Использование возможностей глубокого обучения для объединения физического моделирования и анализа данных позволяет создать единое и согласованное решение, которое сочетает в себе преимущества обоих подходов.

В этом случае общая ошибка оптимизации будет включать дополнительный член для сравнения решения нейронной сети с фактическими данными, что приведет к обновленной формулировке функции потерь в уравнении 4.

$$L_{all} = \alpha L_{ic} + \beta L_{source} + \gamma L_{pde} + \delta L_{data} \quad (4.18)$$

где L_{data} - ошибка совпадения с фактическими данными; δ - весовой коэффициент.

Алгоритм обучения нейронной сети можно схематично представить, как показано на рис. 9. Нейронная сеть принимает на вход пары значений x и t, которые проходят через скрытые слои сети. Прогнозируемое нейронной сетью значение водонасыщенности используется для вычисления двух типов ошибок: ошибка в решении дифференциального уравнения и выполнении начальных и граничных условий, а также ошибка в соответствии с фактическими данными. Суммарная ошибка сравнивается с пороговым значением качества модели, и если она ниже этого значения, обучение прекращается. В противном случае веса нейронной сети корректируются методом обратного распространения, и цикл повторяется.

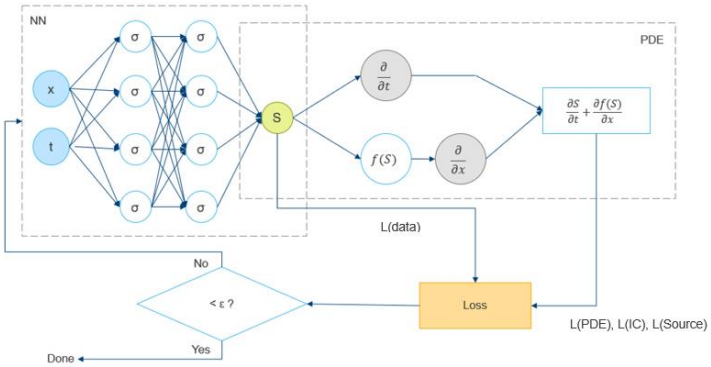


Рис.24. Схема обучения модели.

Рассмотрим задачу с точечным источником из пункта 4, дополнив решение добавлением точек измерения в случае неопределенности, которая не учитывается численной моделью. Более того, эти измерения не могут быть описаны численной моделью из-за ограничений, вызванных исходным управляющим дифференциальным уравнением (рис. 25).

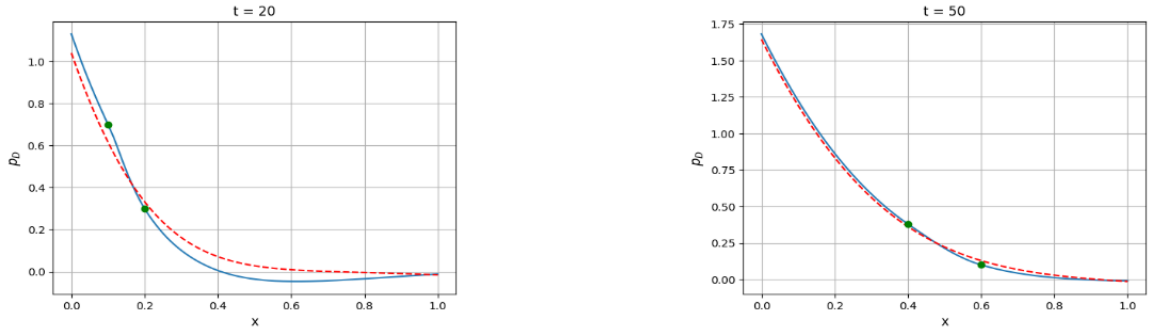


Рис.25. Решение модели PINN с адаптацией к замерным значениям (зеленые точки)

ЗАКЛЮЧЕНИЕ

Гибридное моделирование - инновационный метод, соединяющий машинное обучение и математические модели физических процессов. Основой этого подхода является использование как собранных реальных данных, так и знаний о физических свойствах системы при обучении искусственного интеллекта, что позволяет обеспечить более точное и эффективное управление моделируемой системы. В рамках этого подхода были рассмотрены наиболее эффективные на данный момент методы физически – информированного машинного обучения в задачах нефтяной индустрии, а именно моделирования пласта.

Модель PINN показала высокую точность при решении уравнения, моделирующего скважину как точечный источник, а также при решении задачи моделирования соотношения нефти и газа в пласте с газовой шапкой. Для решения задачи прогнозирования насыщенности с использованием модели текущего водовода и для решения задачи одномерного вытеснения с использованием модели Баклея-Леверетта были реализованы модели PINN с функцией решения обратной задачи и модели PINO, способные получать решения при различных начальных и граничных условиях. Модель PINN работает медленнее, чем численные решения, из-за необходимости обучения нейронной сети для каждого случая начальных и граничных условий. Для ускорения решения можно использовать модель PINO, которая обучается один раз и может получать решения для различных начальных и граничных условий. Важным преимуществом физического подхода является то, что нейронная сеть может использовать фактические данные (измерения) для решения обратной задачи, что устраняет необходимость адаптации модели и значительно сокращает общее время решения. В целом адаптируясь к новым данным, PINN продемонстрировала гибкость и способность учитывать значения, выходящие за рамки исходной численной модели. Однако из-за ограничений в случае, например, задачи о вытеснении, интеграция данных измерений в нейронные сети может оказаться невозможной.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Beda L. M., Korolev L. N., Sukkikh N. V., and Frolova T. S. Programs for automatic differentiation for the machine BESM (in Russian). Technical report, Institute for Precise Mechanics and Computation Techniques, Academy of Science, Moscow, USSR, 1959.
2. Boltyanskii V. G., Gamkrelidze R. V., and Pontryagin L. S.. The theory of optimal processes I: The maximum principle. Izvest. Akad. Nauk S.S.S.R. Ser. Mat., 24:3–42, 1960.
3. Bottou L´eon. Online learning and stochastic approximations. On-Line Learning in Neural Networks, 17:9, 1998.
4. Bryson Arthur E. and Ho Yu-Chi. Applied Optimal Control: Optimization, Estimation, and Control. Blaisdell, Waltham, MA, 1969.
5. Bryson E. and Denham W. F. A steepest ascent method for solving optimum programming problems. Journal of Applied Mechanics, 29(2):247, 1962. doi: 10.1115/1.3640537.
6. Chiyu Max Jiang, Esmaeilzadeh Soheil, Azizzadenesheli Kamyar, Karthik Kashinath, Mustafa Mustafa, Tchelepi Hamdi A, Marcus Philip, Anandkumar Anima, et al. Meshfreeflownet: A physics-constrained deep continuous space-time super-resolution framework. arXiv preprint arXiv:2005.01463, 2020.
7. Ciyou Zhu, Byrd Richard H., Lu Peihuang, and Nocedal Jorge. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. ACM Transactions on Mathematical Software (TOMS), 23(4):550–60, 1997. doi: 10.1145/279232.279236.
8. Dennis John E. and Schnabel Robert B. Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, Philadelphia, 1996.
9. Diab W., Kobaisi M. Al. PINNs for the Solution of the Hyperbolic Buckley-Leverett Problem with a Non-convex Flux Function. 2021; doi: https://doi.org/10.48550/arXiv.2112.14826.
10. Greenfeld Daniel, Galun Meirav, Basri Ronen, Yavneh Irad, and Kimmel Ron. Learning to optimize multigrid pde solvers. In International Conference on Machine Learning, pp. 2415–2423. PMLR, 2019.
11. Griewank Andreas and Walther Andrea. Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation. Society for Industrial and Applied Mathematics, Philadelphia, 2008. doi: 10.1137/1.9780898717761.

12. Guo Hongwei, Zhuang Xiaoying, Zhuang Xiaoyu, and Rabczuk Timon. Analysis of three dimensional potential problems in non-homogeneous media with deep learning based collocation method. arXiv preprint arXiv:2010.12060, 2020.
13. Halton John H. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. Numerische Mathematik, 2(1):84–90, 1960.
14. Hammersley JM and Handscomb DC. Monte-Carlo methods, mathuen, 1964.
15. Jagtap Ameya D. and Karniadakis George Em. Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. Communications in Computational Physics, 28(5):2002–2041, 2020.
16. Krishnapriyan Aditi, Gholami Amir, Zhe Shandian, Kirby Robert, and Mahoney Michael W. Characterizing possible failure modes in physics-informed neural networks. Advances in Neural Information Processing Systems, 34:26548–26560, 2021.
17. Li Zongyi, Kovachki Nikola, Azizzadenesheli Kamyar, Liu Burigede, Bhattacharya Kaushik, Stuart Andrew, and Anandkumar Anima. Neural operator: Graph kernel network for partial differential equations. arXiv preprint arXiv:2003.03485, 2020b
18. Linnainmaa Seppo. The representation of the cumulative rounding error of an algorithm as a taylor expansion of the local rounding errors. Master's thesis, University of Helsinki, 1970.
19. Lu Lu, Xuhui Meng, Mao Zhiping, and Karniadakis George Em. DeepXDE: A deep learning library for solving differential equations. SIAM Review, 63(1):208–228, 2021.
20. Meng Xuhui, Li Zhen, Zhang Dongkun, and Karniadakis George Em. PPINN: Parareal physicsinformed neural network for time-dependent pdes. Computer Methods in Applied Mechanics and Engineering, 370:113250, 2020.
21. McKay Michael D, Beckman Richard J, and Conover William J. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics, 42(1):55–61, 2000.
22. Nabian Mohammad Amin, Gladstone Rini Jasmine, and Meidani Hadi. Efficient training of physics-informed neural networks via importance sampling. Computer-Aided Civil and Infrastructure Engineering, 2021.
23. Nolan John F. Analytical differentiation on a digital computer. Master's thesis, Massachusetts Institute of Technology, 1953.
24. Pang Guofei, Lu Lu, and Karniadakis George Em. fPINNs: Fractional physics-informed neural networks. SIAM Journal on Scientific Computing, 41(4), 2019.

25. Raissi, P. Perdikaris, G. Em Karniadakis. Physics informed deep learning (part 2): Data-driven discovery of nonlinear partial differential equations. 2017; doi: https://doi.org/10.48550/arXiv.1711.10566.
26. Mattey Revanth and Ghosh Susanta. A novel sequential method to train physics informed neural networks for allen cahn and cahn hilliard equations. Computer Methods in Applied Mechanics and Engineering, 390:114474, 2022.
27. Rozonoer I. L. Pontryagin's S. maximum principle in the theory of optimum systems— Part II. Automat. i Telemekh., 20:1441–1458, 1959.
28. Shukla Khemraj, Jagtap Ameya D., and Karniadakis George Em. Parallel physics-informed neural networks via domain decomposition. Journal of Computational Physics, 447:110683, 2021.
29. Sobol' Il'ya Meerovich. On the distribution of points in a cube and the approximate evaluation of integrals. Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki, 7(4):784–802, 1967.
30. Speelpenning Bert. Compiling Fast Partial Derivatives of Functions Given by Algorithms. PhD thesis, Department of Computer Science, University of Illinois at UrbanaChampaign, 1980.
31. Stein Michael. Large sample properties of simulations using Latin hypercube sampling. Technometrics, 29(2):143–151, 1987.
32. Sra Suvrit, Nowozin Sebastian, and Stephen J. Wright. Optimization for Machine Learning. MIT Press, 2011.
33. Werbos Paul J. Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. PhD thesis, Harvard University, 1974.
34. Wight Colby L and Zhao Jia. Solving Allen-Cahn and Cahn-Hilliard equations using the adaptive physics informed neural networks. arXiv preprint arXiv:2007.04542, 2020.