Санкт-Петербургский политехнический университет Петра Великого

на правах рукописи

Лазовская Татьяна Валерьевна

Методы анализа сложных систем по разнородной информации на основе нейросетевых и нейроморфных моделей

Специальность 2.3.1 —
Системный анализ, управление и обработка информации, статистика

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель:
д. т. н. Тархов Д. А.

Санкт-Петербург — 2025

Оглавление

Введение

Актуальность темы исследования. Современные технологии сбора и обработки данных из окружающей среды и наблюдаемых объектов играют ключевую роль в развитии киберфизических систем. Цифровые двойники, синхронизирующиеся с моделируемыми объектами, позволяют проводить цифровые эксперименты и прогнозировать поведение объектов без вмешательства в реальный прототип. Цифровой двойник характеризуется двумя триадами, показанными на Рисунке 1.

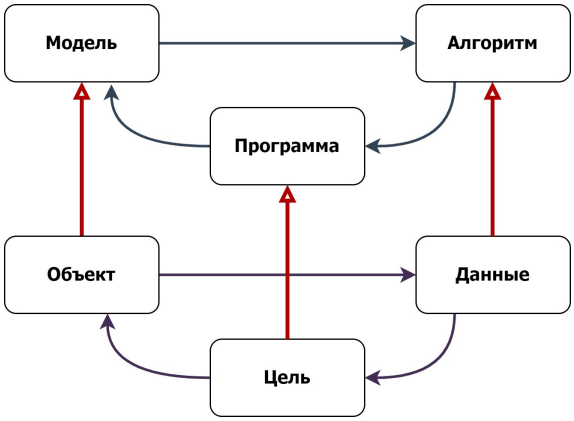

Рисунок 1 – Общая схема устройства цифрового двойника

Верхняя триада относится к традиционному подходу, когда предполагается переход от моделируемого объекта к модели в виде дифференциального уравнения (системы уравнений) с дополнительными условиями (начальными, граничными и т.д.). Далее эта модель рассматривается как сам объект и на ее основе строится приближенное решение. Цифровой двойник строится на основе математической модели, но он также неотделим от алгоритма, реализующего эту модель в цифровой среде в виде программы. В современной парадигме моделирования необходимо предусмотреть возможность адаптации математической модели, которая ранее соответствовала изолированной системе, по результатам функционирования программы, что приводит к соответствующей адаптации алгоритма и программы. Необходимость адаптации наглядно иллюстрируется

нижней триадой. В неё входит объект, в связке с которым функционирует цифровой двойник. Объект и условия его функционирования могут меняться, что требует создания возможности для адаптации всех элементов первой триады по известным данным об объекте. Использование данных позволяет уточнять модель объекта в процессе функционирования двойника, даже если объект и условия функционирования не претерпевают существенных изменений. Кроме того, необходимо постоянно учитывать цель создания цифрового двойника, которая в большинстве случаев предполагает оптимизацию управления им. В зависимости от цели управления выбираются характерные особенности объекта, которые включаются в модель. Цель управления также диктует точность и подробность рассматриваемой модели. Наиболее перспективной математической моделью для применения в киберфизических системах является адаптивная модель типа «серый ящик», когда информации о функционировании объекта не хватает для достаточно точного прогноза его функционирования, но при этом за поведением объекта ведётся наблюдение, что позволяет уточнить его модель и прогноз функционирования, оценить внутреннее состояние изучаемой системы.

Степень разработанности темы исследования.

Фундамент для исследований и разработок в области применения нейронных сетей для решения дифференциальных уравнений и моделирования физических процессов был заложен в работах Дж. Лагариса (J. Lagaris) и Е. Канзы (E. Kansa) еще в 90-е годы. Методы и алгоритмы применения нейронных сетей к решению краевых задач развивались В.И. Горбаченко. Д.А. Тархов и А.Н. Васильев предложили унифицированный нейросетевой подход, который позволяет не только решать задачи математической физики, но и строить модели сложных систем на основе физики объекта и данных измерений. Дж. Эм Карниадакис (G. Em Karniadakis) и М. Раисси (M. Raissi) стали использовать алгоритмы автоматического дифференцирования в ходе обучения нейронных сетей и ввели понятие физически-информированных нейронных сетей (Physics-Informed Neural Networks). В подавляющем числе работ по данной тематике информация об объекте формализуется в виде единой функции потерь. Как правило, взгляд на задачу как многокритериальную недостаточно распространен. Чаще всего в качестве нейросетевой модели рассматриваются многослойные сети прямого распространения, вопрос выбора архитектуры и структуры

которых не разработан и остается на усмотрение исследователя. Таким образом, представляют интерес методы обучения нейронных сетей, позволяющие настраивать не только веса, но и архитектуру сети. Кроме того, недостаточно исследован вопрос адаптации нейросетевых моделей к новым данным. Используемые большинством исследователей нейросетевые модели являются громоздкими, требующими больших вычислительных затрат, что затрудняет их встраивание в сторонние процессы или повторное использование в близких задачах.

Цели и задачи диссертационного исследования. Основной целью исследования является решение проблем, возникающих при реализации унифицированного процесса построения нейросетевых моделей сложных систем.

В диссертации были поставлены и решены следующие задачи:

1. Построение нейросетевой модели по статической выборке плохо определённых данных в случае отсутствия явной статистической зависимости.

2. Разработка нового типа нейроморфных моделей с архитектурой, обусловленной физикой.

3. Сравнительный анализ конечноэлементных, нейросетевых, гибридных моделей и моделей на основе аналитической модификации численных методов на примере задачи с пограничным слоем.

4. Разработка нового типа многослойных нейронных сетей с архитектурой, обусловленной физикой, и методов их построения

5. Разработка и тестирование эволюционных алгоритмов обучения и построения физически информированных нейросетевых моделей на основе приближения к фронту Парето

6. Сравнительный анализ адаптивных свойств непараметрических и параметрических физически-информированных нейросетевых моделей.

7. Разработка методов построения нейроморфных моделей с архитектурой, обусловленной физикой, на основе дифференциальных уравнений в частных производных

Объект исследования. Объектом исследования являются сложные системы и процессы, требующие адаптивного моделирования.

Предмет исследования. Предметом исследования являются методы и алгоритмы для создания физически информированных нейронных сетей и нейроморфных конструкций по гетерогенной информации.

Методология и методы исследования. Методология исследования основана на интеграции классических методов моделирования, основанных на физических законах, с нейросетевыми подходами. Используются эволюционные алгоритмы для структурной и параметрической адаптации нейросетей.

Соответствие содержания диссертации заявленной специальности. Работа выполнена в соответствии с направлениями исследований паспорта ВАК 2.3.1. Системный анализ, управление и обработка информации, статистика (пукнты 1, 2, 4, 5, 7, 11, 12).

Научная новизна:

1. Разработана новая система принятия решений о целесообразности эндоваскулярных вмешательств на основе построенной и протестированной нейросетевой модели прогноза осложнений от параметров крови системы гемостаза.

2. Предложен новый тип нейроморфных моделей, которые в комбинации с традиционными нейронными сетями прямого распространения образуют новый тип глубоких нейронных сетей с архитектурой, основанной на физике, и разработан метод их построения на основе трансформирования обыкновенных дифференциальных уравнений, для решения задач системного анализа.

3. Разработаны новые алгоритмы управления структурой нейронной сети в процессе её обучения на основе предложенных новых эволюционных алгоритмов, использующих управляемую мутацию и отбор на основе приближения к фронту Парето.

4. Впервые обоснованы критерии выбора обычной или параметрической нейронной сети при моделировании и прогнозе динамики объектов, изменяющейся по неизвестному закону, на основе сравнительного анализа адаптивных свойств предложенных типов нейронных сетей.

5. Впервые разработан и протестирован метод построения приближённых нейроморфных решений дифференциальных уравнений в частных произ-

водных с архитектурой, обусловленной физикой, на примере уравнения теплопроводности и волнового уравнения.

Положения, выносимые на защиту:

1. Разработанная и протестированная система принятия решений на основе нейросетевой модели прогноза осложнений после эндоваскулярных вмешательств позволяет выявлять пациентов с высоким риском.

2. Разработанный новый тип нейронных сетей отличается компактностью и гарантированной точностью, что делает его удобным для применения в задачах принятия решений и управления сложными системами.

3. Новые методы анализа систем, описываемых в том числе дифференциальными уравнениями, основанные на физически информированных нейронных сетях, имеют такие преимущества как гибкость и естественная адаптация к новой информации, по сравнению с методами, использующие классический метод конечных элементов.

4. Предложенные алгоритмы управления структурой нейронной сети, проверенные на двух некорректных задачах, могут быть использованы в системах прогноза и управления сложными объектами при наличии противоречивой и неполной информации о них.

5. Разработаны рекомендации для использования непараметрических и параметрических физически-информированных нейросетевых моделей сложных систем в зависимости от характера доступной о данных системах информации.

6. Предложенный новый тип нейронных сетей может быть использован при анализе сложных систем, в информацию о которых входят дифференциальные уравнения в частных производных.

Теоретическая и практическая значимость работы. Теоретическая значимость работы заключается в развитии и углублении методов интеграции данных измерений с классическими методами моделирования сложных систем, основанными на физических законах. В частности, разработанные нейросетевые модели и алгоритмы, включая эволюционные алгоритмы и методы аналитической модификации численных методов, вносят значительный вклад в теорию

моделирования и анализа сложных систем. Эти методы позволяют создавать адаптивные математические модели требуемой точности, которые могут быть использованы для решения широкого спектра задач в различных областях науки и техники. Практическая значимость работы заключается в возможности применения разработанных методов и моделей для решения конкретных индустриальных задач. В частности, разработанная нейросетевая модель прогноза осложнений после эндоваскулярных вмешательств используется в медицинской практике для улучшения качества лечения и снижения риска осложнений. Гибридные модели, сочетающие численные методы и нейросетевые подходы, могут быть применены для создания цифровых двойников и управления киберфизическими системами, что открывает новые возможности для проведения цифровых экспериментов и прогнозирования поведения объектов в различных сценариях их функционирования.

Степень достоверности и апробация результатов. Достоверность полученных автором результатов обеспечивается точностью математических формулировок и выкладок, проверкой вычислительных алгоритмов, а также соответствием результатов вычислительных экспериментов данным реальных и искусственных измерений и результатам, полученным с использованием альтернативных методов, на репрезентативном наборе тестовых и прикладных задач.

Результаты диссертационной работы освещались на международных и всероссийских научных конференциях и симпозиумах:

XI Международная научная конференция, посвященная памяти В.А. Романькова (Омск, 2024); XXVI Международная научно-практическая конференция «Системный анализ в проектировании и управлении» (Санкт-Петербург, 2022); XIX, XXII, XXIII Международная конференция по вычислительной механике и современным прикладным программным системам, ВМСППС'2023(Москва, 2015 2022-2023); V, VI, VII Международная научно-практическая конференция «Информационные системы и технологии в моделировании и управлении» (Симферополь, 2022-2023); XIV, XV, XIX, XX, XXI Всероссийская научная конференция «Нейрокомпьютеры и их применение» (Москва, 2016-2017, 2021-2023); Всероссийская научная конференция «Неделя науки ФизМех» (С.-Петербург, 2023); XIV Международная конференция по прикладной математике и механике в аэрокосмической отрасли (AMMAI'2022) Москва, 2022); 4th

All-Russian Scientific and Practical Conference with International Participation «Distance Learning Technologies»;

International Scientific Conference on Innovations in Digital Economy, SPBPU IDE 2020 (Санкт-Петербург, 2020); International Multi-Conference on Industrial Engineering and Modern Technologies, FarEastCon (Владивосток, 2020) Первая и Вторая всероссийская научно-практическая конференция «Искусственный интеллект в решении актуальных социальных и экономических проблем XXI века» (Пермь, 2016-2017); 13th, 14th International Symposium, ISNN 2016, 2017 (St. Petersburg, Russia, 2016; Sapporo, Hakodate, and Muroran, Hokkaido, Japan, 2017); XI Международная конференция по неравновесным процессам в соплах и струях (NPNJ' Алушта, 2016); XIV, XV, XVI Международная научно-техническая конференция «Проблемы информатики в образовании, управлении, экономике и технике» (Пенза, 2015-2016); 11th International Conference on «Mesh methods for boundary-value problems and applications» (Kazan, 2016); the 1st International Scientific Conference Convergent Cognitive Information Technologies, Convergent 2016 (Москва, 2016); XVIII, XIX, XXIV, XXV международная научно-техническая конференция «Нейроинформатика-20**», (Москва, 2016, 2017, 2022, 2023); 2016 Joint IMEKO TC1-TC7-TC13 Symposium «Metrology Across the Sciences: Wishful Thinking?» (Berkeley, USA, 2016).

Способ прогнозирования возобновления клиники ишемической болезни сердца с помощью нейронных сетей у пациентов после эндоваскулярного вмешательства запатентован.

Результаты исследования также были опубликованы в рецензируемых научных журналах, что подтверждает их значимость и признание в научном сообществе.

Публикации. Основные результаты диссертационной работы опубликованы в 16 статьях в научных журналах из перечня ВАК, в 14 статьях в зарубежных научных изданиях с индексацией в Scopus и Web Of Science, в 48 сборниках всероссийских и в сборниках международных конференций (итого 78 публикаций).

Личный вклад соискателя. Личный вклад автора заключается в разработке эффективных подходов и алгоритмов для построения нейросетевых моделей сложных систем, включая разработку нового типа нейроморфных моделей с архитектурой, обусловленной физикой, и методов их построения. Автор

провел сравнительный анализ конечноэлементных, нейросетевых, гибридных моделей и моделей на основе аналитической модификации численных методов, разработал и протестировал эволюционные алгоритмы обучения и построения физически информированных нейросетевых моделей на основе приближения к фронту Парето. Также автор разработал методы построения нейроморфных моделей на основе дифференциальных уравнений в частных производных и провел вычислительные эксперименты с последующим анализом полученных результатов. Все положения диссертации, выносимые на защиту, получены соискателем самостоятельно и подтверждены публикациями и докладами на конференциях.

Структура и объем работы. Диссертационная работа состоит из введения, семи глав, заключения и списка литературы, занимающих в общей сложности 189 страниц. Список литературы диссертации содержит 171 наименование. Охвачены работы как зарубежных, так и российских авторов.

Глава 1.

Задачи моделирования и связанные с ними нейросетевые конструкции

1.1 Введение

Развитие технологий, позволяющих получать данные из окружающей среды, наблюдаемых объектов и процессов и обрабатывать их с достаточной скоростью и эффективностью, приводит к широкому внедрению киберфизических систем [91]. В рамках функционирования этих систем полученная информация (данные) синхронизируется с соответствующим физическим или информационным объектом. Результатом такой синхронизации является цифровой двойник (ЦД) [126] объекта, т.е. модель сложной системы. ЦД продолжает синхронизацию с моделируемым объектом и отражает его динамические характеристики, изменяющиеся с течением времени. Такие модели можно использовать для проведения цифровых экспериментов, реализации альтернативных внешних сценариев и прогнозирования поведения объекта в будущем, не затрагивая и не изменяя реальный прототип.

Задача построения ЦД и одновременное развитие технологий искусственного интеллекта привели к появлению моделей, управляемых данными, “черных ящиков”, в которых модель создается на основе глубокого обучения с использованием больших данных. Хотя в последнее время возможности получения таких данных растут, стоимость хранения и обработки данных увеличивается вместе с увеличением объемов данных. Кроме того, для некоторых сложных промышленных и технологических процессов получение большого объема данных само по себе может сопровождаться высокими расходами, а иногда и отсутствием возможности получения таких измерений.

В этих случаях имеющиеся небольшие объемы данных могут быть ис-

пользованы для верификации модели, построенной классическими методами моделирования, основанными на физике объекта (например, дифференциальных уравнений), включая численные методы решения соответствующих задач математической физики.

Классический подход к моделированию реальных объектов состоит из двух этапов. На первом этапе на основе анализа известных фактов о физических (и других) процессах, происходящих в объекте, строится модель в виде дифференциального уравнения (системы уравнений) и дополнительных условий (начальных, граничных и т.д.). На втором этапе полученная модель исследуется с использованием численных методов. Конечная модель в виде таблицы чисел позволяет делать необходимые выводы о поведении объекта, строить графики, прогнозировать их динамику и т.д.

Если дифференциальные уравнения и граничные условия достаточно точно отражают поведение моделируемого объекта, для их решения и получения модели объекта разработаны и всесторонне апробированы различные классические методы. В их числе ряд классических методов Рунге-Кутты для обыкновенных дифференциальных уравнений в сочетании с такими методами, как метод стрельбы, для решения краевых задач [25]. Аналогичным образом, для решения уравнений в частных производных были разработаны сеточные методы и методы конечных элементов [112, 78].

В случае, когда операционные данные существенно отличаются от построенной модели, возникает необходимость вернуться к первому этапу и построить более точную дифференциальную модель, затем перестроить таблицу численных решений. При этом, решатели дифференциальных уравнений могут быть дорогостоящими с точки зрения вычислений по сравнению со случаем, когда возможно использование общей параметрической аналитической модели. Более того, если такой подход не приводит к успеху, необходимо строить модель объекта непосредственно по данным измерений, что не является сильной стороной классических численных методов. Та же проблема возникает в инженерном деле, когда требуется тщательный анализ производительности модели при различных параметрах. Лежащие в основе уравнения должны быть решены для большого объема входных данных, отражающих конкретную реализацию интересующего пространства параметров. Таким образом, для моделирования сложных реальных объектов требуются иные методы кроме численных и ис-

пользующих «большие данные».

Задача построения моделей с использованием как дифференциальных уравнений, так и данных измерений, является важной с точки зрения практического применения. Обсуждение определенных аспектов таких задач было начато, например, в работах [14, 15]. Нейросетевой подход к решению подобных задач был впервые предложен в статье [145] российских ученых Д.А. Тархова и А.Н. Васильева, опубликованной в 2005 году. Почти в то же время В.И. Горбаченко развивает нейросетевое направление решения обратных задач [8, 9]. В последующих работах [2, 141, 21, 146] также получила развитие новая парадигма моделирования реальных объктов – построение индивидуальной адаптируемой модели на основе математических моделей, полученных в результате изучения физических и иных процессов, и данных наблюдений над объектом. К этой парадигме относится и данная диссертация. В последние годы к новой парадигме моделирования начинают обращаться в своих работах ученые всего мира [121, 100, 166, 137]. Появляются работы, посвященные классификациям различных подходов внутри и вне новой парадигмы [124, 120, 52, 82, 81]. Среди моделей, сочетающих физику и моделирование на основе данных, выделяют четыре категории [124, 82], образованные взаимной интеграцией следующих подходов к моделированию: моделирование, основанное на физике; моделирование, управляемое данными, в котором отдельно – методы, использующие «большие данные».

В некоторых источниках новая парадигма моделирования получает название гибридного моделирования [120, 81], обеспечивающего с помощью физических законов интерпретируемость и экстраполяционные свойства моделям, управляемым данными, которые, в свою очередь, связывают теоретические модели с реальными объектами. В свою очередь, возникает классификация подходов к характеру взаимодействия физики и моделирования на основе данных. Во-первых, выделяют методы, которые используют физическую теорию для предварительной обработки данных, которые затем поступают на вход нейронной сети для обучения. Второй тип моделей включает те, в которых предварительные знания о физике объекта отражены в архитектуре сети, и развит мало [87]. Наконец, один из наиболее популярных методов гибридного моделирования заключается в использовании физических уравнений в качестве дополнительного члена регуляризации в функции потерь обучаемой нейросетевой

модели. В качестве отдельного класса, кроме того, выделяют модели, в которых физически интерпретируемы функции активации , отдельно указываются подходы [52, 168], в которых модели, управляемые данными, используются в качестве компонентов классических моделей, основанных на физике, для ускорения, упрощения или замены численного решения [10].

Рассмотрим далее типичные задачи построения моделей, решение которых приводит к использованию нейронных сетей, задачи, требующие решения в рамках новой парадигмы моделирования, а также сопутствующие актуальные вопросы.

1.2 Построение нейросетевой модели по выборке данных

Простейшим примером задачи построения модели по имеющейся информации [2] является задача нахождения зависимости между двумя переменными, представленными как матрица исходных данных (входов) и вектор результатов (выходов).

В качестве модели рассматривается функция независимых переменных и параметров

$$y = f(\mathbf{x}, \mathbf{w}), \tag{1}$$

соответствующая экспериментальным данным $\{\mathbf{x}_i, y_i\}_{i=1..N}$, $\mathbf{x}_i \in \mathbb{R}^m$, $y_i \in \mathbb{R}$.

Параметры модели подбираются таким образом, чтобы модель наилучшим образом приближала данные, то есть после выбора функции $f(\cdot)$ ищется вектор $\mathbf{w}$ с помощью минимизации некоторой ошибки аппроксимации:

$$E(\mathbf{w}) = \sum_{n=1}^{N} \delta_n e_n(\mathbf{w}) \to \min. \tag{2}$$

В качестве ошибки аппроксимации могут быть использованы различные функционалы ошибок (функции потерь) $e_n(\mathbf{w})$ [2]. Наиболее часто используются квадратичная или среднеквадратичная ошибка и абсолютная ошибка, вычисляемая как сумма модулей ошибок или их среднее значение, коэффициенты детерминации и корреляции. Для получения более равномерного приближения зависимости входящих и исходящих данных может быть использован так же и максимум модуля ошибки.

Множители δ_n позволяют учитывать влияние конкретных наблюдений на результат, например, в случае наличия данных различной точности или важности. С другой стороны, они могут выступать в качестве регулятора, уравновешивая вклад каждой ошибки.

Рассмотрим типичные виды функции $f(\cdot)$, используемые для решения задачи в виде модели 44.

1.2.1 Регрессии

1.2.1.1 Линейная регрегрессия

Одной из самых простых функций, используемых в качетстве $f(\cdot)$, является линейная. Соответствующая модель называется линейной регрессией и имеет вид

$$y = \sum_{i=1}^{m} w_i x_i + w_0. \tag{3}$$

Для получения весов $\mathbf{w}$ чаще всего пользуются методом наименьших квадратов (минимизация соответствующей ошибки), где минимум и оптимальные параметры линейной регрессии легко получаются после приравнивания к нулю производной. В случае использования других видов ошибок применяются различные численные методы.

1.2.1.2 Логистическая регрессия и задача бинарной классификации

Если в качестве выходных данных выступает принадлежность к тому или иному классу, то есть решается задача классификации, вместо линейной регрессии с ее потенциально неограниченными значениями на выходе используют логистическую регрессию вида

$$y = \sigma(\mathbf{x}, \mathbf{w}) = \frac{1}{1 + \exp(\sum_{i=1}^{m} w_i x_i + w_0)}. \tag{4}$$

В этом случае значения на выходе попадают в промежуток $[0; 1]$ и трактуются как вероятность пренадлежности к выбранному классу. Для получения весов $\mathbf{w}$ чаще всего минимизируют функцию ошибок с логарифмом правдоподобия:

$$L(\mathbf{w}) = -\frac{1}{N}\sum_{n=1}^{N}\big(y_i \log(\sigma(\mathbf{x},\mathbf{w}))_+(1-y_i)\log(1-\sigma(\mathbf{x},\mathbf{w}))\big). \qquad (5)$$

1.2.2 Нейросетевые модели

На сегодняшний день для решения задачи построения модели по статической базе данных обычно применяется два вида нейронных сетей – сети радиальных базисных функций (РБФ-сети) и многослойный персептрон [63]. Оба типа этих сетей относятся к искусственным нейронным сетям прямого распространения, в которых информация передается только в одном направлении без обратной связи. Общая схема таких сетей представлена на Рисунке 2.

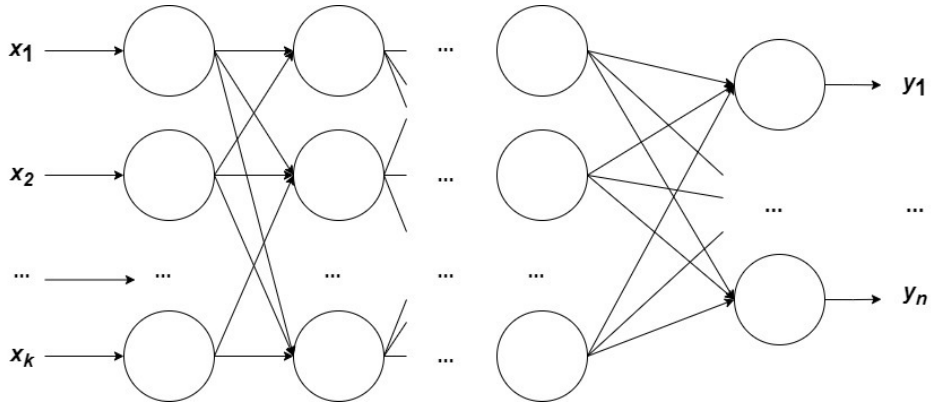

Рисунок 2 – Общая схема искусственной нейронной сети прямого распространения

Опишем ее структуру. Пусть входными данными будет m-мерный вектор. Линейные комбинации входных координат вида $(\mathbf{w},\mathbf{x})$ подаются на вход первого слоя нейронов. Коэффициенты этих комбинаций называются весами первого слоя. Каждый нейрон действует как одномерная нелинейная функция, называемая функцией активации. Линейные комбинации выходных сигналов нейронов передаются на следующий уровень, и линейные комбинации выходных сигналов нейронов последнего уровня формируют выходные данные сети. Дополнительный нейрон, выходной сигнал которого всегда равен 1, часто добавляется ко всем или некоторым слоям. Его значение такое же, как добавление смещения к линейной регрессии, т.е., по сути, вычитание постоянного значения, что значительно улучшает подбор параметров модели, являющейся выходом нейронной сети, в ходе оптимизации (2), то есть обучения нейронной сети [2]. Кроме того, часто добавляется входной сигнал, значение которого всегда равно 1.

1.2.2.1 Многослойный персептрон

Этот тип нейронной сети является наиболее широко используемым и исследованным. Базисная функция в персептроне – это функция, которая по форме приближается к $\mathrm{sign}(x)$. Ее смысл состоит в том, чтобы разделить пространство на два полупространства гиперплоскостями $(\mathbf{w}, \mathbf{x}) = 0$ и $(\mathbf{w}, \mathbf{x}) + w_0 = 0$. В одном из полупространств выходной сигнал нейрона равен $+1$; в другом он равен -1. Если мы возьмем не только один нейрон, но и слой нейронов, то все пространство будет разделено гиперплоскостями на подмножества, каждое из которых соответствует своему набору выходов нейронов этого слоя. Таким образом, выходные данные сети получаются в виде кусочно-постоянной функции. Так же вместо функции $\mathrm{sign}(x)$ используются гладкие функции, что позволяет вычислять производные и применять градиентные методы для обучения сети. В этом случае желательно, чтобы вычисление производных функции активации требовало минимума дополнительных операций. Для таких функций сеть выдает не кусочно-постоянную, а сглаженную зависимость.

Наиболее популярными функциями активации для персептрона являются:

- $\varphi(x) = \frac{x}{1+|x|}$, рекомендованная в работах А.Н. Горбаня [6] и его учеников;
- сигмоида $\varphi(x) = \mathrm{th}(x)$;
- асиметричная сигмоида $\varphi(x) = \frac{\exp x}{1+\exp x}$;
- $\varphi(x) = \arctan(x)$ и другие.

Опишем функционирование многослойного персептрона более формально. Обозначим вектор входных данных l-го слоя $\mathbf{y}_{l-1}$ и вектор выходных данных $\mathbf{x}_l$. Тогда сеть описывается рекуррентными соотношениями $\mathbf{y}_l = \mathbf{W}_l\mathbf{x}_l$, $\mathbf{x}_l = \varphi_l(\mathbf{y}_{l-1})$. Здесь $\mathbf{x}_0 = \mathbf{x}$ – это вход нейронной сети, а $\mathbf{y}_L = \mathbf{f}(\mathbf{x})$ – выход нейронной сети. Соответственно, $\mathbf{W}_l$ – это матрица весов l-го слоя, а φ_l – функция активации, которая действует покоординатно.

Применение градиентных методов обучения сети, т.е. минимизация ошибки (2), требует вычисления производных выходных данных сети относительно весов, которые легко получить, используя формулу для дифференцирования сложной функции.

Для определения желаемых производных вычисления выполняются по формуле

$$\frac{\partial \mathbf{f}}{\partial w_{ij}^{(l)}} = \mathbf{W}_L \mathbf{Z}_L \mathbf{W}_{L-1} \mathbf{Z}_{L-1} \cdots \mathbf{W}_{l+1} \frac{\partial \mathbf{W}_l}{\partial w_{ij}^{(l)}} \mathbf{x}. \tag{6}$$

Матрицы $\mathbf{Z}_L$ вычисляются путем подсчета выходов нейронной сети. В соответствии с (6) вычисления начинаются с последнего уровня и переходят к первому. В литературе по нейронным сетям вычисление производных по этому алгоритму часто сочетается с вычислениями методом градиентного спуска,в результате чего получается метод обратного распространения.

Этот алгоритм обобщен в виде автоматического дифференцирования и широко используется при обучении нейронных сетей, в том числе в [121].

1.2.2.2 РБФ-сети

РБФ-сети состоят из трех слоев, входного, скрытого слоя с радиальной базисной функцией (функции, значения которых монотонны относительно удаления от некоторой центральной точки [113]) и выходного слоя. Выход сети можно представить в виде

$$y = \sum_{k=1}^{m} w_k \varphi(\alpha_k ||\mathbf{x} - \mathbf{c}_k||). \tag{7}$$

Таким образом, в ходе обучения РБФ-сети (7) могут подбираться следующие параметры: $\mathbf{c}_k$ – центры, в окрестности которых происходит локальная аппроксимация; дисперсии α_k, характеризующие разброс данных относительно центров и веса w_k, определяющие вклад каждого центра в результат. Метрика $||\cdot||$ используется для определения близости объектов к центрам.

Примерами радиально-базисных функций являются:

- Гауссова функция (Гауссиан) $f(x) = \exp\big(-\frac{(x-c)^2}{\sigma^2}\big)$;
- Мультиквадратичная функция $f(x) = \sqrt{x^2 + \sigma^2}$;
- Обратная мультиквадратичная функция $f(x) = (x^2 + \sigma^2)^{-\alpha}$, $\alpha > 0$;

Теорема об универсальной аппроксимации для РБФ сетей была опубликована в 1993 году [115].

К обучению сети (7) и подбору ее параметров применяются различные подходы. Во-первых, центры и дисперсии фиксируются заранее по некоторому принципу или случайным образом. В этом случае получается линейная регрессия относительно весов w_k. Кроме того, в ходе многомерной оптимизации функционала типа (2) могут настраиваться все параметры сети. Наконец, может быть использована комбинация данных подходов [144, 145].

Модели, основанные на данных, полностью зависят от данных и не используют другие данные о моделируем объекте. С недавних пор существует консенсус в отношении того, что для моделирования объектов реального мира с его неопределенностью требуется использовать машинное обучение (искусственный интеллект) [118, 120, 124, 52, 82, 81, 21, 146, 127].

1.3 Построение нейросетевой модели по данным и дифференциальным уравнениям

Решение дифференциальных уравнений с помощью нейронных сетей берет свое начало в конце 20-го века [44, 84]. Статья Lagaris исследует применение нейронных сетей для решения обыкновенных и частных дифференциальных уравнений. В этом исследовании решение дифференциального уравнения записывается как сумма двух частей: первая удовлетворяет краевым (или начальным) условиям и не содержит настраиваемых параметров, в то время как вторая часть конструируется таким образом, чтобы не влиять на краевые условия и включает в себя нейронную сеть прямого распространения с настраиваемыми параметрами (весами). Таким образом, построение обеспечивает удовлетворение краевых условий, а сеть обучается удовлетворять дифференциальное уравнение. Этот подход применим как к одномерным обыкновенным дифференциальным уравнениям (ОДУ), так и к системам связанных ОДУ, а также к уравнениям в частных производных (ДУЧП). Общий нейросетевой подход к решению различных задач математической физики был предложен Д.А. Тарховым и А.Н. Васильевым в 2005 году и изложен в серии статей [144, 145], а затем в монографиях [16, 2]. Обучение нейронных сетей для решения задач ДУЧП было рассмотрено и в [7]. В работе [145] впервые была обозначена новая парадигма моделирования на основе ОДУ и ДУЧП при известных реальных измерениях, представлено решение задач математической физики в нестандартной поста-

новке, со свободной и подбираемой границей. В качестве решения рассматривалась нейронная сеть с одним скрытым слоем, параметры которой получались в ходе обучения – минимизации соответствующей функции потерь.

Ежегодно публикуется большое количество статей, посвященных решению задач, связанных с дифференциальными уравнениями, с использованием методов машинного обучения [16, 77, 41, 28, 106, 12, 90, 49, 150, 42, 59, 83, 128, 95, 87, 73, 72, 70, 92, 146, 21, 54].

В 2019 году Raissi, Perdikaris и Karniadakis предложили применять в аналогичном общем нейросетевом подходе методы автоматического дифференцирования, что позволило использовать и быстрее обучать так называемые глубокие нейронные сети с большим числом скрытых слоев [121]. Подход был проиллюстрирован на типовых задачах ОДУ и ДУЧП и получил ставшее известным название Физически-информированных нейронных сетей (Physics-Informed Neural Networks, PINNs).

1.3.1 Нейросетевой подход к решению краевых задач для дифференциальных уравнений

Рассмотрим классическую задачу Коши для системы обыкновенных дифференциальных уравнений

$$\begin{cases} \mathbf{u}'(x) = \mathbf{f}(x, \mathbf{u}(x)),\ x \in [a_1, a_2] = D \subset \mathbb{R}; \\ \mathbf{u}(x_0) = \mathbf{u}_0,\ x \in (a_1, a_2). \end{cases} \tag{8}$$

Приближенное решение этой задачи может быть получено минимизацией соответствующей функции потерь, например, $J(\mathbf{y}) = \int_{a_1}^{a_2} |\mathbf{y}'(x) - \mathbf{f}(x, \mathbf{y}(x))|^2 dx + (\mathbf{y}(x_0) = \mathbf{y}_0)^2$ на некотором выбранном множестве функций $\mathbf{y}(x)$.

Рассмотрим более общую постановку для одномерной функции, в которую в некором смысле входит и задача (8), а именно – краевую задачу

$$\begin{cases} \mathscr{D}u = f,\ u = u(\mathbf{x}),\ \mathbf{x} \in \Omega \subset \mathbb{R}^m; \\ \mathscr{B}u|_\Gamma = g; \end{cases} \tag{9}$$

где $u, f : \mathbb{R}^m \to \mathbb{R}$, $g : \mathbb{R} \to \mathbb{R}$ – некоторые допустимые функции; $\mathscr{D}$ – дифференциальный оператор, то есть некоторое алгебраическое выражение, содержащее производные неизвестной функции u; B – некоторый оператор, задающий гра-

ничные условия на границе Γ рассматриваемой в задаче области Ω.

В 1989 году Цыбенко [40] и Hornik [68] независимо доказали универсальную теорему аппроксимации, которая показала, что искусственная нейронная сеть прямой связи с одним скрытым слоем может с любой точностью приближать любую непрерывную функцию нескольких переменных. В общем нейросетевом подходе Д.А. Тархова и А.Н. Васильева [2] для сетей с одним скрытым слоем приближенное решение ищется в форме линейной комбинации базисных функций v_i,

$$u_N(\mathbf{x}) = \sum_{i=1}^{N} c_i v_i(\mathbf{x}). \tag{10}$$

Поиск решения в виде (10) обычно относят к методу Галеркина [11]. Его частные случаи включают различные варианты МКЭ [4] и нейросетевых моделей **[144, 145, 17]**, в которых уравнения, основанные на физике, используются в качестве дополнительного члена для регуляризации функции потерь. При этом эти модели отличаются особенностями выбора базовых функций и параметров c_i **[144, 145]**. Модели с большим числом скрытых слоев могут быть использованы аналогично, особенности их обучения с помощью автоматического дифференцирования [121] были описаны в разделе о построении моделей по статической выборке. Далее рассматриваются варианты различных базисных функций для общего нейросетевого подхода для сетей с одним скрытым слоем.

1.3.1.1 Методы с фиксированными базисными функциями

Во-первых, выделим группу методов, в которых базисные функции $v_i(\mathbf{x})$ фиксированы и настраиваются только параметры c_i. В современных работах такой подход встречается в алгоритмах экстремального машинного обучения (ELM) [72], где для однослойных нейронных сетей прямого действия случайным образом выбираются внутренние веса и смещения базисных функций, а затем аналитически получаются выходные веса c_i. В [70] аналогичные процедуры применяются поэтапно, показывая большую эффективность при использовании кусочно-непрерывных базисных функций. Аналогичный подход характерен для простейшей версии метода конечных элементов (МКЭ).

Существует два общепринятых подхода к нахождению параметров c_i. Первый, наиболее распространенный из них, заключается в том, что параметры c_i находятся из условия ортогональности остатка уравнения (9) ко всем базисным

функциям $v_i(\mathbf{x})$,

$$(\mathscr{D}u_N(\mathbf{x}) - f(\mathbf{x})) \cdot v_i(\mathbf{x}) = 0. \tag{11}$$

То есть решение ищется минимизацией произведения $a(\mathbf{x}) \cdot b(\mathbf{x})$, простейшей версией которого является произведение в интегральной форме $a(\mathbf{x}) \cdot b(\mathbf{x}) = \int_\Omega a(\mathbf{x}) b(\mathbf{x}) d\mathbf{x}$.

Если интеграл не может быть вычислен явно, он может быть вычислен с помощью кубатурных формул или через среднее значение

$$a(\mathbf{x}) \cdot b(\mathbf{x}) \cong \frac{1}{M} \sum_{j=1}^{M} a(\xi_j) b(\xi_j). \tag{12}$$

Тогда c_i могут быть найдены в качестве решения для системы

$$\sum_{i=1}^{N} c_i \mathscr{D} v_i(\mathbf{x}) \cdot v_j(\mathbf{x}) = f(\mathbf{x}) \cdot v_j(\mathbf{x}). \tag{13}$$

Базисные функции обычно удовлетворяют однородному граничному условию $\mathscr{B} v_i|_\Gamma = 0$. Чтобы удовлетворить неоднородному граничному условию, в (10) добавляется дополнительный член $v_0(\mathbf{x})$. Этот член удовлетворяет неоднородному граничному условию $\mathscr{B} v_0|_\Gamma = f$. Сумма принимает вид $u_N(\mathbf{x}) = v_0(\mathbf{x}) + \sum_{i=1}^{N} c_i v_i(\mathbf{x})$.

Соответственно, подстановка $u_N(\mathbf{x})$ в (11) приводит к линейной системе вида

$$\sum_{i=1}^{N} c_i \mathscr{D} v_i(\mathbf{x}) \cdot v_j(\mathbf{x}) = (f(\mathbf{x}) - \mathscr{D} v_0(\mathbf{x})) \cdot v_j(\mathbf{x}),\ j = 1, \ldots, M. \tag{14}$$

Или, для дискретной формы функции потерь,

$$\sum_{i=1}^{N} c_i \sum_{k=1}^{M} \mathscr{D} v_i(\xi_k) \cdot v_j(\xi_k) = \sum_{k=1}^{M} (f(\xi_k) - \mathscr{D} v_0(\xi_k)) \cdot v_j(\xi_k),\ j = 1, \ldots, M. \tag{15}$$

Этот подход называется методом коллокации.

Второй подход заключается в том, что качество решения задачи (9) характеризуется функцией потерь (называемой также функционалом ошибок) J, минимизация которой приводит к необходимым параметрам c_i. Может использоваться как интегральная форма функции потерь, так и дискретная, если су-

ществуют трудности при явном вычислении скалярного произведения.

МКЭ характеризуется использованием базисных функций (элементов) с компактным носителем, локализованном в небольших подобластях Ω. В этом случае носители базисных функций пересекаются и покрывают всю область Ω. Такие базисные функции обычно выбираются таким образом, чтобы их носители пересекались только для небольшого числа функций, тогда матрица системы (13) будет разреженной, что значительно ускоряет процедуру решения. Для кусочно-линейных элементов прямое применение формулы (13) может оказаться невозможным из-за отсутствия производных требуемого порядка. В нескольких случаях эта проблема может быть решена путем применения формулы Стокса или соответствующей версии формулы интегрирования по частям.

1.3.1.2 Общий нейросетевой подход

При решении сложных задач рассмотренный ранее подход с фиксированными базисными функциями не работает должным образом, и эти функции приходится корректировать в процессе решения. На первый взгляд, общий нейросетевой подход [2] аналогичен предыдущим, только разложение (10) задается выражением

$$u_N(\mathbf{x}) = \sum_{i=1}^{N} c_i v(\mathbf{x}, \mathbf{a}_i). \tag{16}$$

При решении задачи оптимизации для некоторой функции потерь корректируются как линейные параметры c_i, так и векторные параметры (веса и смещения) $\mathbf{a}_i$. В качестве базисных функций используются функции, типичные для нейронных сетей. Обычно это радиальные базисные функции (РБФ) или сигмоидные базисные функции, в случае персептрона.

Как уже было упомянуто в разделе о построении нейросетевых моделей по данным без уравнений, основное различие между РБФ-сетями и персептроном заключается в том, что каждый нейрон отвечает за локальную аппроксимацию вблизи некоторой точки, а не за разницу между значениями функции в полупространствах, как в случае с персептроном. Отсюда следует вывод, что персептрон лучше всего использовать для моделирования проблем со скачками, таких как фазовые переходы и ударные волны. РБФ-сети предпочтительны для приложений, где ожидается достаточно плавное решение [16].

Для решения задачи (9) в качестве функции потерь используется [2] функ-

ционал вида

$$J = J_1 + \delta J_2, \tag{17}$$

где

$$J_1 = \int_\Omega (\mathscr{D}u_N(\mathbf{x}) - f(\mathbf{x}))^2 d\mathbf{x}, \tag{18}$$

отражает удовлетворение дифференциальному уравнению задачи, а

$$J_2 = \int_\Gamma (\mathscr{B}u_N(\mathbf{x}) - g(\mathbf{x}))^2 d\mathbf{x}, \tag{19}$$

соответствует граничному условию. Обычно, из-за невозможности вычисления интегралов в явном виде, используется дискретная форма. Таким образом,

$$J_1 = \sum_{j=1}^{M} (\mathscr{D}u_N(\xi_j) - f(\xi_j))^2, \tag{20}$$

и

$$J_2 = \sum_{k=1}^{K} (\mathscr{B}u_N(\xi'_k) - g(\xi'_k))^2. \tag{21}$$

Точки $\{\xi_j\}_{j=1}^M$ и $\{\xi' k\}_{k=1}^K$ могут быть выбраны внутри области Ω (или в более широком наборе $\tilde{\Omega} \supset \Omega$, если необходимо обеспечить достаточную гладкость приближенного решения вплоть до границы) и на границе обычным образом; например, равномерно в случае ограниченной области или в соответствии с нормальным законом вероятности, если она неограничена. Часто более целесообразно использовать набор точек, выбранных в соответствии с законом вероятностного распределения и повторно сгенерированных после определенного количества периодов обучения (шагов оптимизации) с использованием постоянной (или другой) плотности вероятности. Это обеспечивает более стабильное обучение [146].

Кроме того, такой подход позволяет контролировать качество обучения с использованием стандартных статистических процедур. Таким образом, количество точек выборки может быть значительно уменьшено без ущерба для точности [146]. В то же время процесс обучения становится более стабильным и не застревает на локальных минимумах. На самом деле оптимизируется не одна функция, а последовательность функций, и каждая из них является дискретным приближением к интегральной форме функции потерь. Такой подход, кро-

ме того, позволяет избежать так называемого «проклятия размерности» [63, 2]. Для уменьшения затрат на обучение нейронной сети можно воспользоваться частичной перегенерацией точек [12]. В классических PINNs используются фиксированные точки [121], аналогично методу коллокации, хотя последнее время возникает тенденция в разработке методов адаптивного выбора точек, в которых вычисляется оптимизируемая функция потерь, что позволяет существенно улучшить обучение сети [107, 164, 155].

В качестве алгоритмов оптимизации, используемых для обучения PINNs, в основном применяют различные вариации алгоритмов градиентного спуска. Алгоритмы Broyden-Fletcher-Goldfarb-Shanno (BFDS и L-BFGD) и ADAM являются наиболее популярными в последнее время и показывают наилучшие результаты при обучении глубоких сетей [39].

1.3.1.3 Метод конечных элементов

Общий нейросетевой подход объединяет в себе как сети с РБФ, так и многослойный персептрон с одним скрытым слоем. В этом контексте важно отметить, что классический МКЭ можно рассматривать как частный случай метода РБФ-сетей. Чаще всего при использовании МКЭ применяется кусочно-линейная аппроксимация. В этом случае выполняется триангуляция, то есть область делится на треугольники (граница аппроксимируется ломаной линией) и для каждого треугольника строится линейная функция. На границе треугольников функции непрерывно соединяются. Чтобы получить представление решения в виде разложения по базисным функциям (10), можно рассмотреть поведение функции вдоль лучей, исходящих из определенного центра c_i. В двумерном случае $v_i = v_i(\rho, \psi)$, где (ρ, ψ) – полярные координаты вектора $\mathbf{x} - \mathbf{c}_i$ и $\mathbf{x}(\rho, \psi)$ – текущая точка. Если положение границы опоры базисной функции относительно ее центра характеризуется некоторой функцией $\rho = a_i(\psi)$, можно использовать $\varphi_i = (1 - \rho/a_i(\psi))_+$, где $w_+ = w(\mathrm{sign}(w) + 1)/2$. В этом случае для многоугольной границы получается кусочно-линейная функция, для которой $a_i(\psi)$ вычисляется из полярных уравнений соответствующих прямых линий, т.е. на соответствующих интервалах изменения ψ имеет место $a_i(\psi) = \rho_i / \cos(\psi - \psi_i)$. Если требуется, чтобы базисная функция на границе имела нулевую производную, можно использовать $\varphi_i = \left((1 - \rho/a_i(\psi))_+\right)^2$. Для базисной функции вида $\varphi_i = (1 + 2\rho/a_i(\psi))\left((1 - \rho/a_i(\psi))_+\right)^2$, вершина является гладкой. Некоторая

более сложная форма базисной функции позволяет получить гладкую поверхность с полигональным основанием, если того требует рассматриваемая задача. В многомерном случае мы можем использовать $v_i = v_i(\rho, \mathbf{v})$, где $\rho = \mathbf{x} - \mathbf{c}_i$, и $\mathbf{v}$ – вектор на единичной сфере, который может быть параметризованным соответствующими координатами.

1.3.1.4 Нейронные сети экстремального обучения

Extreme Learning Machine (ELM), экстремальное машинное обучение, – это алгоритм обучения искусственных нейронных сетей, предложенный Guang-Bin Huang, Qin-Yu Zhu и Chee-Kheong Siew в 2006 году [72, 70].

ELM обычно состоит из одного скрытого слоя и выходного слоя. Простота архитектуры ELM способствует ее эффективности при обучении и внедрении, как и в общем нейросетевом подходе. При этом одной из ключевых особенностей ELM является высокая скорость обучения, так как ELM обучает сеть только один раз, используя случайные скрытые узлы: соединениям между входным слоем и скрытым слоем присваиваются случайные веса. Эти случайные веса определяются только один раз на этапе обучения и не требуют корректировки в процессе обучения.

Было показано, что ELM является универсальным аппроксиматором [60]. При этом очевидны недостатки такого подхода по сравнению с общим нейросетевым подходом: естественная зависимость от инициализации весов и переобучение. В связи с чем, например, в [92] изучается алгоритм обучения ELM с настраиваемой базисной функцией.

1.3.2 Нейросетевой подход к построению моделей на основе дифференциальных уравнений и дополнительных данных

Как уже было отмечено, новая парадигма моделирования реальных объектов рассматривает и математические модели (дифференциальные уравнения и уравнения других типов), и данные измерений как некоторую приближенную информацию разной точности. Такой подход позволяет использовать постановки задач, которые с классической точки зрения считались некорректными [148], и успешно решать их [16, 21, 146, 120, 152, 159, 153]. Развитие сенсорных технологий позволяет получать данные об объекте моделирования в режиме ре-

ального времени и своевременно адаптировать модель к изменениям. В этом контексте на первый план выходят подходы, относящиеся к новой парадигме гибридного моделирования и обеспечивающие построение более надежных и точных моделей и смягчающие некоторые недостатки использования чисто физических моделей (основанных на дифференциальных уравнениях) или моделей, управляемых данными [121, 100, 166, 82, 120, 81, 52, 168, 166, 48].

К подобным задачам можно отнести и задачу Коши в том случае, когда начальное условие имеет место для значения переменной вне области, на которой задано дифференциальное уравнение.

Сформулируем такие задачи в общем виде:

$$\mathscr{D}[u(\mathbf{x})] = 0,\ \mathbf{x} \in \Omega. \tag{22}$$

Рассматриваемая задача может иметь различные условия, включая граничные, в виде

$$\mathscr{B}[u(\mathbf{x})] = b(\mathbf{x}),\ \mathbf{x} \in \bar{\Omega}, \tag{23}$$

начальные условия

$$\mathscr{C}[u(\mathbf{x})] = c(\mathbf{x}),\ \mathbf{x} \in \bar{\Omega} \cup \Omega, \tag{24}$$

данные измерений

$$\mathscr{M}[u(\mathbf{x_i})] = m(\mathbf{x}_i),\ \mathbf{x}_i \in \bar{\Omega}, \tag{25}$$

и другие условия различного типа, включая асимптотические соотношения [12]. Здесь, $\mathscr{D}[\cdot]$, $\mathscr{B}[\cdot]$, $\mathscr{C}[\cdot]$, $\mathscr{M}[\cdot]$ – некоторые дифференциальные операторы , $\mathbf{x}$ – пространственно-временная переменная, Ω – область решения с границей $\bar{\Omega}$, и $b(\mathbf{x}), c(\mathbf{x}), m(\mathbf{x})$ – подходящие функции.

Простейший нейросетевой подход [16, 121] предоставляет решение задачи (22–25) в виде полносвязной нейронной сети с прямой связью $\hat{u}(\mathbf{x}, \mathbf{a})$, и неизвестные параметры $\mathbf{a}$ ищутся путем минимизации среднеквадратичных потерь ошибок по пространству этих параметров,

$$L_D(\hat{u}(\mathbf{x}, \mathbf{a}), \mathbf{X}_{N_D}) = \frac{1}{N_D} \sum_{i=1}^{N_D} (\mathscr{D}[\hat{u}(\mathbf{x}_i, \mathbf{a})])^2; \tag{26}$$

$$L_B(\hat{u}(\mathbf{x},\mathbf{a}),\mathbf{X}_{N_B}) = \frac{1}{N_B}\sum_{j=1}^{N_B}(\mathscr{B}[\hat{u}(\mathbf{x}_j,\mathbf{a})] - b(\mathbf{x}_j))^2; \tag{27}$$

$$L_C(\hat{u}(\mathbf{x},\mathbf{a}),\mathbf{X}_{N_C}) = \frac{1}{N_C}\sum_{k=1}^{N_C}(\mathscr{C}[\hat{u}(\mathbf{x}_k,\mathbf{a})] - c(\mathbf{x}_k))^2; \tag{28}$$

$$L_M(\hat{u}(\mathbf{x},\mathbf{a}),\mathbf{X}_{N_M}) = \frac{1}{N_M}\sum_{l=1}^{N_M}(\mathscr{M}[\hat{u}(\mathbf{x}_l,\mathbf{a})] - m(\mathbf{x_i}))^2. \tag{29}$$

Таким образом, возникает множество функций потерь (функционалов ошибок). Для обучения PINNs наиболее распространенным методом является минимизация общих потерь, представляющих собой взвешенную сумму функций потерь, с использованием стандартных алгоритмов нелинейной оптимизации. Классический подход предлагает несколько вариантов построения общих функций потерь с использованием метода линеаризации для задач многоцелевой оптимизации. Эти опции включают в себя изменение количества точек сопоставления N_D, N_B, N_C, используемых для вычисления расхождения для уравнений (22)–(24), объемов точек обучения N_M и набора тестовых точек $(M - N_M)$ для данных измерений, и включение,

$$\begin{aligned} &L(\mathbf{X}_{N_D},\mathbf{X}_{N_B},\mathbf{X}_{N_C},\ldots) \\ &= L_D(\mathbf{X}_{N_D}) + \delta_B L_B(\mathbf{X}_{N_B}) + \delta_C L_C(\mathbf{X}_{N_C}) + \delta_M L_M(\mathbf{X}_{N_M}) + \ldots \end{aligned} \tag{30}$$

или исключение [121],

$$L(\mathbf{X}_{N_D},\mathbf{X}_{N_B},\mathbf{X}_{N_C},\ldots) = L_D(\mathbf{X}_{N_D}) + L_B(\mathbf{X}_{N_B}) + L_C(\mathbf{X}_{N_C}) + L_M(\mathbf{X}_{N_M}),$$

штрафных множителей $\delta_B, \delta_C, \ldots$ для членов в общей функции потерь. Выбор подходящих штрафных параметров является важной задачей, остающейся актуальной и на сегодняшней день. В книге [17] Д.А. Тархов обсуждает различные подходы к выбору штрафных множителей, начиная от фиксации в начале обучения до пересчета через некоторое число итераций обучения сети из принципа равномерного вклада слагаемых в функцию потерь. Кроме того, с помощью вариации штрафных множителей предлагается учитывать достоверность измерений. В современных работах используются аналогичные идеи [171], иногда с более точной настройкой штрафных параметров. Так, адаптивные веса пред-

ложены в [26], где параметр штрафа обновляется на каждом или некоторых этапах обучения сети в соответствии с заранее определенной формулой. В [123] оценивается влияние штрафного параметра на скорость обучения, и для получения окончательного решения используется фиксированное значение. Авторы [71] исследуют влияние веса как на функцию потерь, так и на ее отдельные компоненты, сравнивая их значения для результирующей сети. В [152] приближенные решения, полученные для различных значений весового параметра, сравниваются с точным решением.

Задача построения PINN также рассматривается как многокритериальная. В [97] предложен и протестирован на простых задачах с обыкновенными уравнениями и уравнениями в частных производных многоцелевой метод оптимизации для обучения PINN, использующий алгоритм сортировки без доминирования. В [162] предлагается самоадаптивный алгоритм поиска с градиентным спуском, который вручную определяет скорость обучения на разных этапах в соответствии с различными этапами поиска. Эволюционный алгоритм выборки, динамически эволюционирующий там, где точки коллокации совпадают на протяжении итераций обучения, предложен в [43].

Решение задач с несколькими целями и принятие многокритериальных решений встречаются в различных дисциплинах и ведущут к применению альтернативных подходов, выходящих за рамки традиционных методов. В [30] рассматриваются различные стратегии динамического уравновешивания воздействий множества членов и градиентов в функции потерь путем выбора наилучших коэффициентов масштабирования. В [45] для многоцелевой многозадачной оптимизации применяются эволюционные алгоритмы.

1.3.3 Выбор архитектуры нейронной сети

Подавляющее большинство статей, представляющих основанные на физике модели нейронных сетей и решения, используют предварительно выбранную сетевую архитектуру, упоминая, что соответствующие параметры были найдены эмпирически. Эта дилемма также упоминается в классической книге по нейронным сетям Хайкина [63]. Понятие архитектуры в данном случае может включать как типы функций активации, количество слоев и нейронов в них, так и методы инициализации выходных параметров нейронной сети. Но для целенаправленного отбора желательно использовать эволюционные алгоритмы

[16, 132, 99, 165, 147, 131]. Эти методы оптимизации используют набор потенциальных решений, которые со временем эволюционируют с помощью различных операций, включая отбор, мутацию и скрещивание.

Проблема определения оптимальной архитектуры особенно актуальна при рассмотрении решений нейронной сети для задач, связанных с дифференциальными уравнениями [44, 84]. Хотя преобладающая тенденция в большинстве исследований предполагает фиксированную на начальном этапе архитектуру физически-информированных сетей, в последние годы количество статей, посвященных поиску оптимальной архитектуры нейронных сетей, растет, и вопрос разработки алгоритмов выбора архитектуры сети является актуальным [17, 156, 146, 47]. В [29] применяется схема наложения физических ограничений на архитектуру нейронной сети. Они представляют метод, который преобразует различные типы уравнений в линейные ограничения и использует их для построения выходного уровня нейронной сети с использованием положительно определенных функций активации.

В монографии [17] описываются различные методы изменения структуры нейронных сетей. Предложены комбинированные сети, объединяющие многослойные персептроны и РБФ сети, сети с частичной структурой связей, позволяющие включать квадратичные и более сложные связи с нейросетевую модель; рассмотрены возможности подбора структуры сети в ходе обучения, а также принципы работы генетического алгоритма для оптимизации структуры итоговой нейронной сети, начиная с некоторого набора сетей с различными структурами.

Рассматривая проблему формирования популяции, следует отметить, что построение физически-информированных нейронных сетей по своей сути является многокритериальной задачей, учитывая, что краевые задачи включают в себя множество критериев, поскольку они состоят из уравнения и граничных условий, каждое из которых описывается отдельными функциями потерь для оценки точности. Кроме того, при наличии измерений возникает дополнительный член функции потерь. В статье [47] предлагается эволюционный алгоритм для многоцелевого поиска архитектуры, который позволяет аппроксимировать полный спектр архитектур по Парето в соответствии с несколькими целями, включая прогнозируемую производительность и количество параметров. Эволюционные алгоритмы, в силу их популяционной природы, были использова-

ны для генерации множества элементов оптимального набора по Парето за один прогон [38]. Некоторые практические приложения к инженерным задачам представлены в [132]. Эти методы также хорошо подходят для решения задач, связанных с нерегулярными фронтами Парето [94]. Было замечено, что приближение к фронту Парето [93, 104] дает набор решений, которые можно рассматривать как совокупность для построения эволюционных алгоритмов.

Одним из недостатков эволюционных алгоритмов является их значительная ресурсоемкость. Следовательно, ключевой задачей является повышение их эффективности. Распознавание характеристик рассматриваемой проблемы является важнейшим ресурсом для повышения эффективности таких алгоритмов. Кроме того, важную роль играет осознанный подход к информации о популяции, который включает в себя набор моделей, участвующих в эволюции.

В работе [87] впервые рассмотрен метод выбора архитектуры модели, основанный на физике. Для решения системы дифференциальных уравнений первого порядка вида

$$\begin{cases} \mathbf{y}'(x) = \mathbf{f}(x, \mathbf{y}(x)), \\ \mathbf{y}(x_0) = \mathbf{y}_0, \end{cases} \quad x \in [x_0, x_0 + a]. \tag{31}$$

используется аналитическая модификация стандартных численных методов [62], которые в общем виде описываются через применение рекурентной формулы

$$\mathbf{y}_{k+1} = \mathbf{y}_k + F(\mathbf{f}, h_k, x_k, \mathbf{y}_k, \mathbf{y}_{k+1}), \tag{32}$$

где оператор F определяет конкретный метод, точки x_k разбивают $[x_0, x_0 + a]$ на интервалы длиной h_k, $k = 1, \ldots, n$, Если формулу (43) применить итеративно n раз к интервалу с переменным правым концом $[x_0, x] \subset [x_0, x_0 + a]$, в результате получится аналитическое выражение для функции $\mathbf{y}(x)$, которое рассматривается как приближенное решение системы (31).

1.4 Построение параметрических моделей динамических объектов и решение обратных задач

Во многих случаях формулировка задачи включает параметры, которые проявляют вариации в определенном диапазоне. Например, такие свойства, как плот-

ность, коэффициенты теплопроводности, эластичность и другие, могут быть точно неизвестны для реальных сред. Кроме того, такие факторы, как размер объекта и температура окружающей среды, также могут проявлять вариабельность. В таких сценариях возникает задача построения параметрических моделей либо для исследования поведения решения в зависимости от конкретного параметра, либо для оценки значения параметра с использованием данных измерений.

Классические методы традиционно предполагают получение численного решения задачи с использованием репрезентативного набора параметров. Универсальный нейросетевой подход [3, 141] позволяет нам искать решение в виде функции нейронной сети, где соответствующие параметры включаются в качестве аргументов. Формализация параметризованных задач заключается в том, что в формулировке задачи (9) появляется некоторый параметр $\mathbf{r}(\cdot)$, который принимает приемлемые значения в соответствующей области $\Gamma \subset \mathbb{R}^d$ с подходящим d:

$$\begin{cases} \mathscr{D}[u](\mathbf{x}, \mathbf{r}(t)) = \quad f(\mathbf{x}, \mathbf{r}(t)), \ \mathbf{x} \in \Omega; \\ \mathscr{B}[u](\mathbf{x}, \mathbf{r}(t)) = \ g(\mathbf{x}, \mathbf{r}(t)), \ \mathbf{x} \in \partial\Omega. \end{cases} \tag{32}$$

Некоторые значения параметра $\mathbf{r}(t)$, где time $t \in [0, T]$, могут изменить природу задачи, например, уменьшить ее размерность или определить тип дифференциального оператора.

Таким образом, возникает два типа задач. В первом требуется найти решение $u(\mathbf{x}, \mathbf{r}(t))$, для которого (32) выполняется для всех значений параметра $\mathbf{r}(t)$ из некоторого множества. Во втором при наличии соответствующей дополнительной информации (данных измерений) требуется восстановить значение параметра $\mathbf{r}(\cdot)$.

Прямое применение МКЭ [85] к решению задачи (32) приводит к тому, что вместо (10) решение ищется в виде разложения

$$u_N(\mathbf{x}, \mathbf{r}) = \sum_{i=1}^{N} c_i(\mathbf{r}) v_i(\mathbf{x}, \mathbf{r}), \tag{33}$$

и система (13) заменяется системой

$$\sum_{i=1}^{N} c_i(\mathbf{r})\mathscr{D}(\mathbf{r})[v_i](\mathbf{x},\mathbf{r}) \cdot v_j(\mathbf{x},\mathbf{r}) = (g(\mathbf{x},\mathbf{r}) - \mathscr{D}(\mathbf{r})[v_0](\mathbf{x},\mathbf{r}) \cdot v_j(\mathbf{x},\mathbf{r}). \qquad (34)$$

В результате получается система линейных уравнений с коэффициентами, зависящими от параметра. Существует ряд методов для решения этой системы. Это, во-первых, аналитический метод, для некоторых задач представляющий собой высокую вычислительную сложность; второй подход использует интервальный анализ, методы которого зачастую подвержены эффекту Мура (неограниченное увеличение оценок). Эти методы работают в разумные сроки только при небольшом количестве элементов. Таким образом, приемлемая точность достигается только для простых задач. В-третьих, если задача решена для достаточно репрезентативного набора параметров, это может позволить оценить решение для всего набора параметров (методы типа Монте-Карло). В то же время вычислительная сложность возрастает во много раз, и возникает вопрос о достаточной репрезентативности выбранного набора параметров.

Построение параметрических решений с помощью универсального подхода является актуальной задачей. Например, в статье [41] рассматриваются гибридные системы, которые представляют собой класс задач, циклически переключающихся между двумя фазами: первая описывается жесткой системой дифференциальных алгебраических уравнений движения (взаимодействие с землей), вторая соответствует фазе полета и задается нежесткая система. Используя в качестве примера конкретную задачу (прыгающие одноногие роботы), авторы [41] сравнивают различные решатели обыкновенных дифференциальных уравнений, в том числе из пакета Matlab. Отмечено, что, хотя разные решатели хорошо работают для разных типов задач, из-за постоянного перезапуска численного решателя точность решения снижается, а стоимость вычислений увеличивается. Эту проблему можно было бы решить с помощью некоторого универсального параметрического подхода. [28] также рассматривает необходимость решения параметризованной задачи, которая возникает при изучении широкого спектра сложных физических явлений, когда при моделировании множества различных возможных реализаций системы большие требования к вычислительным ресурсам приводят к вынужденному использованию аппарата редуцированных параметрических моделей. Авторы подчеркивают важность

таких параметрических моделей в глобальных задачах проектирования, управления, оптимизации и количественной оценки неопределенности. Разработка контроллера, который эффективно работает при всех значениях параметров модели, рассматривается задача высокой важности. В [106] предлагается решение параметризованной дифференциальной задачи с помощью МКЭ для интервального параметра на основе интервальной арифметики. Однако в результате авторы получают только соответствующие интервальные границы решения, но не параметрическое аналическое решение.

Статья [154] посвящена изучению динамической системы для составления прогнозов, особенно когда некоторые переменные остаются ненаблюдаемыми, с использованием физического моделирования и моделирования, основанного на данных. В частности, неопределенность поверхности рассматривается как один из ограничивающих факторов методов контактного моделирования, основанных на физике. Постановка задачи в этих случаях близка к моделированию на основе данных с высокой точностью и приближенными управляющими уравнениями. Одним из решений является оценка текущей ошибки модели и последующее включение ее в уравнение в качестве корректирующего элемента [53]. Другим вариантом, помимо определения уравнений на основе данных, в случае моделирования неизвестной или частично известной динамики, является идентификация динамической системы [**?**].

1.4.1 Общий нейросетевой подход

Нейросетевой подход также может быть применен к задаче с параметрами. В этом случае параметры включены в число входных данных нейронной сети [1, 3]. Модификация нейросетевого подхода к задаче (32) состоит в нахождении приближенного решения в виде выхода искусственной нейронной сети заданной архитектуры

$$u_N(\mathbf{x}, \mathbf{r}) = \sum_{i=1}^{N} c_i v_i(\mathbf{x}, \mathbf{r}, \mathbf{a}_i), \tag{35}$$

где веса $c_i, \mathbf{a}_i$ определяются в процессе пошагового обучения сети в ходе минимизации функции потерь вида $J = J_1 + \delta J_2$, аналогичного представленному в

формулах (18-19). Соответственно, в дискретной форме

$$J_1 = \sum_{j=1}^{M} (\mathscr{D}(\mathbf{r}_j)[u_N](\xi_j, \mathbf{r}_j) - g(\xi_j, \mathbf{r}_j))^2 \tag{36}$$

и

$$J_2 = \sum_{k=1}^{K} (\mathscr{B}(\mathbf{r}'_k)[u_N](\xi'_k, \mathbf{r}'_k) - f(\xi'_k, \mathbf{r}'_k))^2 \tag{37}$$

В ходе обучения нейронной сети перегенерация точек выполняется следующим образом: сначала $\mathbf{r}_j$ и $\mathbf{r}'_k$ повторно генерируются из области изменения параметров, а затем $\xi_j \in \Omega = \Omega(\mathbf{r}_j)$ и $\xi' j \in \Gamma = \Gamma(\mathbf{r}_j)$.

Если доступны дополнительные данные, например, измерения, к функции потерь, как и ранее, добавляется $J_3 = \sum_{l=1}^{L} (u_N(\xi''_l, \mathbf{r}''_l) - u_l)^2$. Здесь u_l – это измерение, соответствующее значению требуемого решения в точке ξ''_l и значению параметра $\mathbf{r}''_l$.

Несмотря на естественную способность нейросетевых моделей адаптироваться к данным, процесс их построения часто отличается высокой вычислительной сложностью.

В реальных приложениях данные высокой точности часто имеются в ограниченном количестве, а дифференциальные уравнения и другие данные имеют более низкую точность. В этом случае говорят о многоуровневом моделировании, использующем данные, имеющие разную точность [100].

Подходы к многоуровневому моделированию являются актуальными областями исследований в области моделирования [37, 98, 71]. Для сопоставления низкоточных физических данных и высокоточных сенсорных данных используется трансферное обучение, которое включает обновление исходной модели путем повторного обучения. Существуют исследования, использующие активное обучение для повышения точности аппроксимации, например, [98] (на основе данных датчиков) и [23] (на основе моделирования методом конечных элементов).

Способность физически-информированных сетей решать задачи с параметрами исследуется во многих исследованиях и остается актуальной проблемой [121, 167, 138, 23, 98, 61, 170, 139]. В частности, построение параметрических моделей является неотъемлемой частью суррогатного моделирования [138, 61, 23, 57]. В [23] подчеркивается важность построения параметрических

решений по сравнению с классическими численными в смысле удобства анализа суррогатной модели в различных условиях и мгновенного ответа при запросе для любого местоположения в интересующем пространстве пространственных параметров. Авторы демонстрируют, насколько эффективно решения, представляющие целые семейства параметрических моделей, строятся с использованием PINNs. Это позволяет решать с высокой точностью и обратную задачу (задачу идентификации параметров).

Необходимо сделать замечание о точности решения подобных задач. В традиционном подходе предполагается переход от моделируемого объекта к модели в виде дифференциального уравнения (системы уравнений) с дополнительными условиями (начальными, граничными и т.д.). Далее эта модель рассматривается как сам объект и на ее основе строится приближенное решение. Соответственно, чем выше точность такого решения, тем более точной считается модель исходного реального объекта. Второй подход, новая парадигма моделирования, в рамках которой написана данная диссертация, заключается в том, что дифференциальная модель, построенная на основе физических принципов, безусловно, является приблизительной. Для реальных сложных технических объектов точность аналогичных моделей часто невысока. Следовательно, решение дифференциальных уравнений с высокой точностью не имеет практического смысла. Более перспективным является построение адаптивной модели, соответствующей дифференциальному уравнению и дополнительным условиям с разумной точностью, и последующая адаптация этой модели к данным наблюдений объекта. Такой подход особенно актуален в ситуации, когда свойства объекта могут изменяться в процессе его эксплуатации.

Глава 2.

Задача построения нейросетевой модели по статической выборке плохо определённых данных в случае отсутствия явной статистической зависимости

2.1 Возможности нейронных сетей для персонализированных подходов для профилактики осложнений после эндоваскулярных вмешательств

Известно, что большинство заболеваний сердечно-сосудистой системы сопровождаются нарушениями в системе гемостаза. Система гемостаза является одной из самых сложных систем. Она имеет иерархическую структуру с множеством компонентов. Анализируются результаты теста генерации тромбина (ТГТ), который позволяет оценить действия всех компонентов системы гемостаза. Проблема осложняется наличием слишком большого количества различных клинических случаев. Обычные статистические методы не дают глобальных оценок. Применяется универсальный нейросетевой подход для построения моделей системы гемостаза, основанный на факторах, которые не имеют статистически значимого различия для различных типов клинических случаев после операции. Инструменты нейронной сети позволяют учитывать нелинейную иерархическую природу рассматриваемой системы и строить индивидуальные модели для каждого клинического случая. В данной главе представлена разработка системы принятия решений на основе нейронной сети для прогнозирования прогрессирования заболевания и осложнений после эндоваскулярных

вмешательств.

В настоящее время искусственные нейронные сети широко используются для обработки медицинских данных и медицинской диагностики [103, 27, 74, 55, 135]. Большие объемы данных позволяют изучить преимущества и недостатки различных алгоритмов обучения и типов архитектуры нейронных сетей [27, 74, 135]. В то же время медицинские данные, полученные из реальных историй, часто представляют собой образцы небольшого размера. Это могут быть редкие показатели, результаты дорогостоящих медицинских анализов. В таком случае мы можем говорить только об изучении потенциала нейронной сети для использования в медицинской диагностике [55].

Требования к медицинским исследованиям с каждым годом становятся все более строгими, но результаты и выводы часто зависят от дизайна исследования и методов статистической обработки, выбранных для анализа полученных данных. Между тем, все патологические состояния в организме человека представляют собой сложные процессы, многоплановая оценка которых требует комплексных (объемлющих) методов исследования и математического анализа; и основной целью их исследования является возможность прогнозирования индивидуальных осложнений.

Задачи медицинской диагностики – это тип проблем, когда все реальные условия не могут быть приняты во внимание. Исследователь может различать только некоторые предполагаемые наборы наиболее важных показателей. Построенный на таких данных алгоритм будет неточным и приблизительным. Более того, правила построения и нахождения ответа не могут быть четко определены. Имеено в таких задачах оправдано использование нейросетевых методов [32, 51, 80, 56, 140, 31].

Настоящее исследование направлено на поиск возможностей прогнозирования индивидуальных осложнений после чрескожного коронарного вмешательства (ЧКВ), которое является наиболее популярным методом лечения ишемической болезни сердца (ИБС). Трудности в исследовании осложнений, развивающиеся после ЧКВ, заключаются в том, что этот процесс включает изменения в стенке сосудов и системе гемостаза. Показатели системы гемостаза можно оценить с помощью ТГТ, выбранного в нашем исследовании для тщательного изучения.

В статье [135] авторы также исследовали применение искусственных ней-

ронных сетей в прогностической оценке послеоперационных осложнений. Но модель, предложенная там, основана на наборе данных со статистически значимой разницей, и объем данных был достаточно большим. В нашей работе мы исследуем возможность искусственной нейронной сети идентифицировать шаблоны и нелинейные связи в случае небольшой выборки данных и унифицированного набора данных в терминах классических методов статистики.

2.2 Методы и материалы

Материалом в исследовании являются образцы данных венозной крови, которые были взяты до ЧКВ у 66 пациентов с ишемической болезнью сердца в возрасте от 53 до 77 лет. Было оценено состояние системы гемостаза с использованием теста генерации тромбина ТГТ в двух параллельных постановках, выполненных в соответствии с методом, предложенным Hemker H. [65] и соавторами в 2003 году. Клинические исходы считались версией симптомов возобновления заболевания коронарной артерии, никаких событий (51 наблюдения); прогрессирование заболевания (8 наблюдений); острый коронарный синдром (ОКС) или инфаркт миокарда (ИМ) (7 наблюдений).

Были рассмотрены количественные характеристики ТГТ, для которых применение классического статистического метода (корреляционный, регрессионный анализ и др.) не показало существенных результатов. С помощью медицинских экспертов были выбраны наиболее важные характеристики. Таким образом, следующие показатели были использованы как факторы.

- ETP – площадь под кривой тромбограммы (образования тромбина), предложенная Хемкером и соавторами для количественного выражения генерации тромбина [66].;
- пик тромбина $PEAK$, который отражает максимальное количество образовавшегося тромбина
- VI_{TM} – изменение скорости генерации тромбина
- ΔLT – процент уменьшения периода начала образования тромбина.

Для анализа показателей теста генерации тромбина и их связи с клиническими вариантами течения заболевания использовался метод нейронной сети

для математического моделирования бинарного классификатора. Построение бинарного классификатора обусловлено относительно небольшой рассматриваемой выборкой, а также общего числа наблюдений осложнений. В качестве классов рассматривались наличие и отсутствие осложнений после операции. Таким образом, задача построения модели классификатора заключалась в поиске аппроксиматора некоторой неизвестной функции, зависящей от входных данных и отражающей в той или иной степени вероятность появления осложнений у объекта после оперативного вмешательства.

В ходе численных экспериментов мы рассматривали нейронные сети с различной архитектурой, номерами параметров и нейронами (узлами). Как правило, результатом моделирования является выход нейронной сети одного скрытого слоя [2] в виде

$$U(\mathbf{a}, \mathbf{c}; \mathbf{t}) = c_0 + \sum_{i=1}^{n} c_i v(\mathbf{a}_i, \mathbf{t}), \tag{38}$$

где n – число нейронов, через скаляры $\mathbf{c} = (c_0, \ldots, c_n)$ и векторы $\mathbf{a} = (\mathbf{a}_0, \ldots, \mathbf{a}_n)$ обозначены веса (параметры) сети ; v – выбранная базисная функция нейронной сети, используемые данные обозначаются через вектор $\mathbf{t}$. В частности, данные j-го наблюдения обозначаются как $\mathbf{t}_j$.

В качестве базисных функций после различных вычислительных экспериментов нами были выбраны сигмоиды (гиперболические тангенсы), поведение которых как нельзя лучше подходит для классификации данных [2, 80], и которые имеют вид

$$v(\mathbf{a}, \mathbf{t}) = \mathrm{th}(a_0 + \sum_{k=1}^{K} a_k t_k), \tag{39}$$

где K – это количество данных для каждого наблюдения. В нашем случае $\mathbf{t} = (ETP,\ PEAK,\ \Delta LT,\ VI_{TM})$.

Чтобы избежать излишней сложности модели, нейронная сеть строилась в несколько этапов, на каждом из которых добавлялся и обучался один дополнительный нейрон. Подразумевалось, что процесс обучения сети идет до удовлетворительного результата.

На первом этапе, значение G_j функционала ошибки (40) было закодировано как -3 для отражения возникновения осложнения после операции и как

1 для отсутствия осложнений.

На последующих этапах мы приближаем с помощью выхода нейронной сети с одним нейроном ошибки G_j предыдущего этапа для каждого наблюдения. В результате выходы сетей суммируются и число нейронов в текущей модели увеличивается на один. Общее число параметров модели равно $1 + 6n$, где n – число нейронов.

На каждом этапе обучения сети нейросетевые веса настраивались в ходе минимизации так называемого функционала ошибки в дискретной форме

$$I(a, c) = \sum_{j=1}^{M} \delta_j \big(U(\mathbf{a}, \mathbf{c}; \mathbf{t}_j) - G_j\big)^2, \tag{40}$$

где M – количество наблюдений, G_j – значения аппроксимируемой функции в точках $\mathbf{t}_j$, δ_j – положительные штрафные коэффициенты.

На всех этапах обучения сети процесс минимизации функционала ошибки проводился с использованием таких алгоритмов оптимизации как RProp и Particle Swarm [2, 125]. Начальные веса выбирались таким образом, чтобы уравновесить вклад каждого из факторов.

В результате решения задачи глобальной нелинейной оптимизации на каждом этапе была получена нейросетевая функциональная модель $U(\mathbf{a}, \mathbf{c}; \mathbf{t}) = U_{a,c}(\mathbf{t})$ с фиксированными параметрами (весами).

Данная функция принимает значения на числовой прямой. Выбирая пороговое значение на основе ROC-анализа, мы получаем классификатор [50].

Количество нейронов модели также является предметом исследования. Необходимо соблюдать баланс в меньшем числе нейронов с одной стороны и большем качестве построенного классификатора с другой.

2.3 Результаты моделирования и классификации

Качество построенных моделей было оценено с помощью ROC-анализа [135, 50]. Сравнивались результаты для моделей с возрастающим количеством нейронов на каждом шаге построения. В Таблице 1 представлены характеристики площади под ROC-кривой (AUC). AUC является агрегированной характеристикой качества классификации, не зависящей от соотношения различных типов ошибок. Идеальная классификация соответствует AUC, равному 1. Эта мера часто

используется для сравнения различных моделей классификации.

Таблица 1 – Результаты ROC-анализа. AUC – площадь под ROC-кривой; n – число нейронов; p – асимптотическая значимость

n	AUC	Станд. ошибка	p	95% дов. интервал
1	0.695	0.074	0.022	0.551 – 0.840
2	0.778	0.060	0.001	0.661 – 0.895
3	0.824	0.064	0.000	0.697 – 0.950
4	0.886	0.051	0.000	0.785 – 0.987

Стандартные ошибки AUC были оценены на основе непараметрического метода. Следует признать, что качество моделей классификации нейронных сетей хорошее даже в случае трех нейронов. Если число нейронов равно четырем, качество классификации очень хорошее [50].

Таблица 2 иллюстрирует качество классификаций, построенных после выбора порогового значения. Пороговые значения для всех наших моделей были равны нулю. Здесь представлены такие качественные характеристики как: Чувствительность – это процент людей с осложнениями после хирургического вмешательства, которые правильно определены; Специфичность – это процент людей, у которых нет осложнений, которые правильно определены (истинный отрицательный показатель). Зачастую для описания качества классификации используются эффективность (среднее значение по специфичности и чувствительности), истинный положительный коэффициент (TPR) – это процент людей, у которых выявлены осложнения, которые правильно определены, и истинные отрицательные (TNR) показатели.

Модель с одним нейроном недостаточно чувствительна. Добавление второго нейрона улучшает распознавание наблюдения с осложнениями, но специфичность падает. Начиная с трехнейронной модели, все показатели улучшаются. Модели с тремя и четырьмя нейронами обладают достаточным уровнем чувствительности и специфичности.

Таблица 2 – Результаты ROC-анализа

Число нейронов	1	2	3	4
Чувствительность, %.	60,0	80,0	80,0	86,7
Специфичность, %.	72,5	62,7	74,5	80,4
Эффективность, %.	66,3	71,4	77,3	83,5
TPR, %.	39,1	38,7	48,0	56,5
TNR, %.	86,0	91,4	92,67	95,3

2.4 Обсуждение результатов

Результаты анализа подтвердили предположение, что количественные показатели ТГТ (ETP, $PEAK$), отражающие интенсивность его формирования, влияют на симптомы возобновления ИБС независимо от версии рецидива. Полученные данные совпадают с выводами других исследователей-исследователей, которые использовали стандартные математические методы, чтобы найти взаимосвязи между показателями ТГТ и смертностью от сердечно-сосудистых причин после неотложного ЧКВ [24].

Ранее высказанная гипотеза о роли протеина С в развитии осложнений после ЧКВ и возможности использования показателей, характеризующих уровень активности этой системы, путем изменения скорости генерации тромбина (VI_{TM}) и процента уменьшения периода инициации образования тромбина (ΔLT) после добавления тромбомодулина в реакционную смесь также подтвердилась. Роль системы протеина С в тромбозе стента после ЧКВ была подтверждена и в другом исследовании [96], где активность этой системы также оценивалась с помощью ТГТ.

Для описанного выше Способа прогнозирования возобновления клиники ишемической болезни сердца с помощью нейронных сетей у пациентов после эндоваскулярного вмешательства получент Патент $RU2675067C1$, дата регистрации 14.12.2018.

Технический результат, достигаемый изобретением, заключается в повышении прогностической достоверности результатов теста генерации тромбина

при прогнозировании возобновления клиники ишемической болезни сердца с помощью нейронных сетей у пациентов после эндоваскулярного вмешательства, в выявлении пациентов с высоким риском возобновления клинических проявлений ишемической болезни сердца после чрескожного коронарного вмешательства, и индивидуальном подходе к ведению пациентов с высоким риском, позволяющим повысить эффективность лечения.

Исследование иллюстрирует способность нейросетевого моделирования выявлять скрытые зависимости и сложную функциональную природу таких процессов, как патологические состояния человека. Были использованы генетические алгоритмы построения и обучения модели нейронной сети. Повышение числа нейронов может обеспечить соответствующее повышение качества классификации нейросетевой модели, но это возможно только при большом количестве данных. При этом особенностью моделей нейронных сетей является способность к дообучению на основе новой информации, которая может включать новые наблюдения и дополнительные факторы, что позволяет говорить о перспективах уточнения и улучшения нейросетевой модели.

Глава 3.

Сравнение и сочетание нейросетевых и классических подходов

3.1 Введение

В этой главе рассматриваются методы моделирования и их модификации в контексте современной классификации парадигм моделирования сложных динамических систем. Представлены принципы построения искусственных нейроморфных моделей на основе физики, которые естественным образом поддаются дальнейшей адаптации и уточнению с использованием данных измерений и других дополнительных данных.

Ставится вопрос моделирования системы, описываемой краевой задачей с неизвестным параметром. Проблемы, связанные с дифференциальными уравнениями с неизвестными параметрами, часто возникают при моделировании и управлении кибер-физическими системами. Эти параметры можно рассматривать как динамические и изменяющиеся со временем по известному или неизвестному закону, а также подлежащие идентификации по данным измерений. При описании сложных систем параметр может характеризовать условия, в которых существует объект, или определенную часть этого объекта. Такую постановку можно считать сложной задачей, некорректно поставленной задачей, к которым также относятся задачи с малым параметром при старшей производной и, соответственно, задачи с пограничным слоем и другими типами.

При решении задачи с неизвестным параметром не ставится задача идентификации этого параметра, а решается более общая задача – построение параметрического решения, универсального для различных значений параметров, относящихся к заданному интервалу, который используется при построении и обучении решения нейронной сети. Результирующая предварительно обученная

модель может быть дополнительно адаптирована путем обучения на данных измерений. На основе этих измерений неизвестные параметры в свою очередь уже могут быть оценены любым способом, или построенная модель скорректирована с помощью дополнительного обучения сети. Подобные задачи имеют большое практическое значение. Так, в работе [77] рассматривается моделирование температуры озера. Авторы [77] используют функцию потерь с регуляризацией, основанной на физике, для предварительного обучения общей рекуррентной модели нейронной сети, прежде чем адаптировать ее посредством данных измерений, соответствующих конкретному озеру.

В данной главе проводится сравнение нескольких подходов к решению подобных задач. Общий нейросетевой подход был применен автором диссертации для решения одной параметризованной дифференциальной задачи, которая при некоторых значениях параметра является жесткой [12]. Кроме того, для улучшения качества приближенного решения использовалась дополнительная гетерогенная информация. Этот же подход был успешно применен для решения жесткой дифференциальной задачи и дифференциально-алгебраической задачи, имеющей различное число решений в зависимости от значения параметра, а также задачи о катализаторе в области более широкой, чем область существования решения дифференциального уравнения [90].

Не смотря на развитие глубоких нейронных сетей, нейронные сети с одним скрытым слоем успешно применялись для получения решений замкнутой формы для класса параметризованных систем начальных значений с различной степенью жесткости в [49]. В статье [150] подчеркивается растущий интерес к разработке численных методов решения аналогичных задач и предлагается метод, основанный на рациональном спектральном сопоставлении. Методы адаптивной сетки рассматриваются в [42, 83, 59, 128].

Преимущество, заключающееся в низких вычислительных затратах, в случае применения уже обученных нейросетевых моделей неоспоримо. Однако при описании систем с неопределенностью структура самой дифференциальной задачи часто может быть неизвестна заранее. Таким образом, возникает необходимость в построении аналитического решения, которое не требует обучения в режиме реального времени и при этом будет отражать рассматриваемую модель для различных значений параметров. В качестве решения данной задачи автором диссертации предлагается подход, связанный с основанными на физи-

ке сетевыми архитектурами, и представленный как расширение многослойных методов, впервые описанных в работе [87].

Методы построения моделей систем, описываемых в форме классической непараметризованной краевой задачи, были представлены в разделе 1.2, в том числе: метод конечных элементов как один из методов моделирования, использующих только физику объекта, и общий нейросетевой подход [1] как гибридный подход в рамках новой парадигмы моделирования.

В данной главе представлено использование общего нейросетевого подхода и методов, основанных на аналитической модификации классических численных методов для построения моделей для постановки задачи с неизвестным параметром. Кроме того рассматривается способ усовершенствования общей нейросетевой модели с помощью использования предварительных знаний о физике объекта в виде набора обучающих данных. При этом, вместо результатов измерений для дополнительного обучения модели нейронной сети используются значения решения МКЭ в соответствующих узлах. Отдельно обсуждаются подходы к решению задачи с фиксированным параметром, облегчающие понимание методов решения параметризованных задач. В конце главы рассматривается параметризованная задача о сингулярных возмущениях и представлены результаты применения ранее описанных методов к построению модели.

3.2 Нейросетевое сглаживание решения МКЭ

В главе 1 в качестве методов моделирования, использующих физику объекта, были рассмотрены метод конечных элементов (МКЭ) и общий нейросетевой подход. Оба метода имеют свои преимущества и недостатки. Для получения модели с достаточной гладкостью требуется использование в МКЭ полиномиальных элементов высоких степеней, что приводит к увеличению стоимости вычислений. С помощью этого метода невозможно получить решение с бесконечной гладкостью, что реализуется в случае нейросетевого моделирования. Кроме того, если количество элементов метода невелико, точность недостаточно высока; если оно велико, система линейных уравнений может потребовать больших вычислительных ресурсов даже при разреженной матрице системы (13). С другой стороны, МКЭ применяется более широко и реализован во многих коммерческих программных продуктах. При этом, МКЭ плохо подходит для решения задач, в которых для построения модели используются дополнительные дан-

ные. Особенно это касается задач, в которых модель должна адаптироваться к вновь полученным данным с учетом изменений в объекте моделирования. Во многом это связано с локальным характером базисных функций, что приводит, в свою очередь, к локальной реструктуризации решения при попытке адаптировать его путем минимизации функции потерь вида (29), соответствующей данным измерений. В то же время, нейронные сети по своей природе способны адаптироваться к новым данным. Об этом свидетельствует широкое использование нейронных сетей для извлечения информации, скрытой в зашумленных данных. Однако прямое применение описанного в первой главе общего нейросетевого подхода часто приводит к вычислительно сложной задаче нелинейной оптимизации. Если настроить только параметры c_i решения нейронной сети вида (16), то получается система линейных уравнений с плотной матрицей, что резко увеличивает сложность ее решения по сравнению с МКЭ. Более эффективный подход заключается в том, чтобы сначала оптимизировать функцию потерь $J = J_1 + \delta J_2$ (17) или соотвествующую сумму (26, 27), а затем обучать нейронную сеть при поступлении новых данных, минимизируя функцию потерь (29). Однако этот метод также очень ресурсоемкий.

В качестве альтернативы предлагается следующий комбинированный метод. Сначала с помощью МКЭ строится решение задачи (9). Далее это решение сглаживается с помощью нейронной сети. Для подхода, рассмотренного выше, все базисные функции, кроме одной, равны нулю в каждой узловой точке, и этот тип модели построить проще всего. Он состоит из оптимизации (29), при которой вместо результатов измерений используются значения решения МКЭ в узлах. Полученное приближенное решение нейронной сети может быть аналогично адаптировано к вновь полученным данным.

В [169] решение МКЭ также используется в качестве обучающих данных в задаче идентификации структурных повреждений. Однако в этой работе значения в узлах не используются напрямую, но функция потерь включает в себя два члена с соответствующими весовыми коэффициентами. Первый отвечает за потерю перекрестной энтропии, а второй соответствует нормализованным вероятностям повреждения.

3.3 Аналитическая модификация классических численных методов как многослойная сеть с архитектурой, обусловленной физикой объекта

Выражения приближенных решений, построенных с использованием общего нейросетевого подхода или МКЭ, довольно громоздки. Если их записать аналитически, они окажутся трудно обозреваемыми. Гораздо более компактные аппроксимации получаются с помощью разложения в приближенный ряд или различных видов асимптотических методов. Однако эти методы позволяют построить приемлемое решение только в достаточно малой окрестности точки, в которой строится разложение, или при достаточно малых значениях параметра, по которому выполняется разложение. В [87] предлагается другой подход, который имеет существенно более широкую область применения. С точки зрения современной классификации новой парадигмы моделирования, можно отнести этот метод к сетевым архитектурам, основанным на физике. Результирующие аналитические модели довольно компактны.

Рассмотрим задачу Коши для системы линейных дифференциальных уравнений n-го порядка

$$\begin{cases} y^{(n)}(x) = f(x, y(x), y'(x), \dots, y^{(n-1)}(x)), \\ y(x_0) = y_0, \\ y'(x_0) = y'_0, \qquad\qquad x \in [x_0, x_0 + a]. \\ \dots \\ y^{(n-1)}(x_0) = y_0^{(n-1)}, \end{cases} \tag{41}$$

С помощью замены можно перейти от системы высокого порядка к многомерной задаче первого порядка вида

$$\begin{cases} \mathbf{y}'(x) = \mathbf{f}(x, \mathbf{y}(x)), \\ \mathbf{y}(x_0) = \mathbf{y}_0, \end{cases} x \in [x_0, x_0 + a]. \tag{42}$$

Для решения системы (42) существует множество численных методов [62]. Значительная часть таких методов состоит в дроблении данного интервала точками x_k на интервалы длины h_k, $k = 1, \dots, n$, и применении некоторой рекур-

рентной формулы

$$\mathbf{y}_{k+1} = \mathbf{y}_k + F(\mathbf{f}, h_k, x_k, \mathbf{y}_k, \mathbf{y}_{k+1}), \tag{43}$$

где оператор F определяет конкретный численный метод. Если в (43) функция F зависит от от $\mathbf{y}_{k+1}$, возникает необходимость в точном или приближенном решении уравнения (43) относительно $\mathbf{y}_{k+1}$.

Применяя формулу (43) итеративно n раз к интервалу с переменным правым концом $[x_0, x] \subset [x_0, x_0 + a]$, мы получаем аналитическую функцию $\mathbf{y}(x)$, которую можно рассматривать в качестве приближенного решения системы (42). Подчеркнем, что зависимость от x имеют и такие параметры, как длина промежутка $h_k = h_k(x)$, промежуточные выражения $\mathbf{y}_k = \mathbf{y}_k(x)$ и решение на последнем слое, которое берётся в качестве приближённого решения задачи (42).

Для построения такого приближенного решения может быть выбран любой из классических численных методов [62], например:

- Неявный метод Эйлера, где

$$F(\mathbf{f}, h_k, x_k, \mathbf{y}_k, \mathbf{y}_{k+1}) = h_k \mathbf{f}(x_{k+1}, \mathbf{y}_{k+1}); \tag{44}$$

- Метод трапеций, где

$$F(\mathbf{f}, h_k, x_k, \mathbf{y}_k, \mathbf{y}_{k+1}) = \frac{1}{2} h_k (\mathbf{f}(x_k, \mathbf{y}_k) + \mathbf{f}(x_{k+1}, \mathbf{y}_{k+1})); \tag{45}$$

- Метод средней точки, где

$$\begin{cases} \mathbf{y}_{k+2} = F(\mathbf{f}, h, x_k, \mathbf{y}_k, \mathbf{y}_{k+1}), \\ F(\mathbf{f}, h, x_{k+1}, \mathbf{y}_k, \mathbf{y}_{k+1}) = \mathbf{y}_k + 2h\mathbf{f}(x_{k+1}, \mathbf{y}_{k+1}). \end{cases} \tag{46}$$

и другие методы подобного типа.

Если функция $\mathbf{f}(x, \mathbf{y})$ является нейросетевой, например, была получена в результате моделирования некоторого сопряженного процесса, в результате применения метода возникнет глубокая нейронная сеть. В этом случае можно рассматривать этот метод как гибридный. Кроме того, правая часть задачи в принципе может быть аппроксимирована нейронной сетью в случае, если при-

менение исходной функции создает вычислительные трудности. Таким образом, результирующая модель может быть рассмотрена как специальная инициализация нейронной сети, параметры которой далее могут быть подобраны дополнительно в ходе обучения на данных измерений.

Рассмотрим далее краевую задачу

$$\begin{cases} \mathbf{y}'(x) = \mathbf{f}(x, \mathbf{y}(x)), \\ \mathbf{u}(x_0) = \mathbf{u}_0,\ \mathbf{v}(x_0 + a) = \mathbf{v}_0, \end{cases} x \in [x_0, x_0 + a]. \tag{47}$$

Здесь, векторы $\mathbf{v}$ и $\mathbf{u}$ составлены из различных координат вектора $\mathbf{y}$; их общая размерность совпадает с размерностью $\mathbf{y}$.

Для решения этой задачи предлагается следующий подход. На интервале выбирается некоторая точка $t \in (x_0, x_0 + a)$ и строится решение следующей системы

$$\begin{cases} \mathbf{y}'(x) = \mathbf{f}(x, \mathbf{y}(x)), \\ \mathbf{y}(t) = \mu, \end{cases} x \in [x_0, x_0 + a]. \tag{48}$$

где параметр μ подбирается, исходя из граничных условий

$$\begin{cases} \mathbf{u}(x_0) = \mathbf{u}_0, \\ \mathbf{v}(x_0 + a) = \mathbf{v}_0. \end{cases} \tag{49}$$

При этом, параметр t остается неопределенным. Его можно выбрать в конце таким образом, чтобы полученное решение удовлетворяло исходной системе с наибольшей точностью или, что более актуально при моделировании реальных объектов, чтобы решение соответствовало наилучшим образом данным измерений.

Адаптация модели к новой информации об объекте моделирования происходит следующим образом. Параметры t и μ, или один из них, подбираются в процессе минимизации функции потерь $\sum_{l=1}^{L} ||\mathbf{y}_n(\xi_l'') - \mathbf{y}_l||^2$, где норма $||*||$ учитывает только те координаты вектора $\mathbf{y}_l$, для которых имеются данные измерений.

3.4 Нейросетевое сглаживание решения МКЭ для задач с параметром

Чтобы объединить преимущества МКЭ и нейросетевого метода для задач с параметром, мы предлагаем следующий гибридный метод. Предположим, что построено МКЭ решение для набора значений параметра

$$u_N(\mathbf{x}, \mathbf{r}_j) = \sum_{i=1}^{N} c_{i,j} v_i(\mathbf{x}, \mathbf{r}_j),\ j = 1, \ldots, M. \tag{50}$$

Далее, строятся нейросетевые аппроксимации для коэффициентов c_i

$$c_i(\mathbf{r}) = \sum_{k=1}^{n} d_{k,i} \varphi_k(\mathbf{r}, \mathbf{a}_{k,i}), \tag{51}$$

где $\varphi_k(\mathbf{r}, \mathbf{a})$ – нейросетевые базисные фукции. В работах [16, 146] рекомендуется использовать радиально-базисные функции. Веса все нейронных сетей подбираются в ходе минимизации соответствующих функций потерь, например, в форме $\sum_{j=1}^{M}(c_{i,j} - \sum_{k=1}^{n} d_{k,i}\varphi_k(\mathbf{r}_j, \mathbf{a}_{k,i}))^2$. Полученное решение

$$u_N(\mathbf{x}, \mathbf{r}) = \sum_{i=1}^{N} \sum_{k=1}^{n} d_{k,i} \varphi_k(\mathbf{r}, \mathbf{a}_{k,i}) v_i(\mathbf{x}, \mathbf{r}) \tag{52}$$

может использоваться не только для интерполяции в диапазоне параметров, для которых использовался МКЭ. Это также применимо для экстраполяции в область, в которой решение МКЭ затруднено из-за плохо обусловленной матрицы. Но для задач экстраполяции обычно РБФ заменяются на базисные функции сигмоидного типа [2].

3.5 Решение конкретной задачи

Все описанные выше методы были применены для решения эталонной параметризованной краевой задачи[111]

$$\begin{cases} u''(x) + a(s)u'(x) + b(s)u(x) = 0; \\ u(0) = 0, u(1) = 1; \end{cases} x \in [0, 1]. \tag{53}$$

с коэффициентами $a(s) = 1/s + s$ и $b(s) = 1/s - s$.

При малых значениях s задача (53) является жесткой и может рассматриваться как параметризованная сингулярная задача о возмущении с пограничным слоем, близким к нулю. Эта задача имеет аналитическое решение

$$u(x) = (1 + s(1 - x))e^{1-x} - (1 + x + s)e^{1-x/s} + O(s^2),$$

позволяющее наглядно сравнивать результаты различных методов.

3.5.1 Общий нейросетевой подход

Сначала для решения задачи (53) был применен общий нейросетевой подход. Задача решается при фиксированных значениях параметра s; решение также имеет вид (16). Веса нейронной сети подбираются в ходе минимизации функции потерь

$$\sum_{i=1}^{m}(u''(x_i) + a(s)u'(x_i) + b(s)u(x_i))^2 + \delta_1 u^2(0) + \delta_2(u(1) - 1)^2. \qquad (54)$$

Тестовые точки $\{x_i\}$ выбираются случайным образом на интервале $[0, 1]$ с перегенерацией каждые 5 шагов процесса оптимизации. RProp используется в качестве алгоритма минимизации [34]. Что касается контроля точности решения, есть два разных варианта. Первый предназначен для установки порогового значения для оптимизированной функции потерь, а второй – для настройки ограничения на количество эпох обучения сети. Во втором случае проводится несколько перезапусков процесса обучения и среди полученных результатов выбирается тот, для которого функция потерь имеет наименьшее значение. Кроме того, может быть учтено отклонение от реальных данных об объекте.

Для решения задачи использовались два типа базисных функций, сигмоиды и гауссианы, и разное количество нейронов скрытого слоя. Для каждого решения нейронной сети представлен график истории потерь при обучении. Одна эпоха соответствует 100 перегенерациям точек обучения. Между каждыми двумя перегенерациями выполняется 5 шагов алгоритма RProp. Количество точек, в которых вычисляется функция потерь, на каждом шаге равно 100. Рисунок 3 представляет решение (53) для фиксированного $s = 0.1$ в случае гауссовой нейронной сети с 5 нейронами в скрытом слое. Соответствующая ошибка показана

на Рисунке 4. Приближенное нейросетевое аналитическое решение имеет вид

$$\begin{aligned}\mathtt{nn}_5(x) = {} & 0.6934e^{-1.0151(-0.9445+x)^2} + 0.5966e^{-30.8231(0.0819+x)^2} + \\ & 0.4256e^{-63.199(0.1164+x)^2} + 2.1744e^{-1.3991(0.1815+x)^2} - 46.4433e^{-8.6259(0.5628+x)^2}.\end{aligned} \tag{55}$$

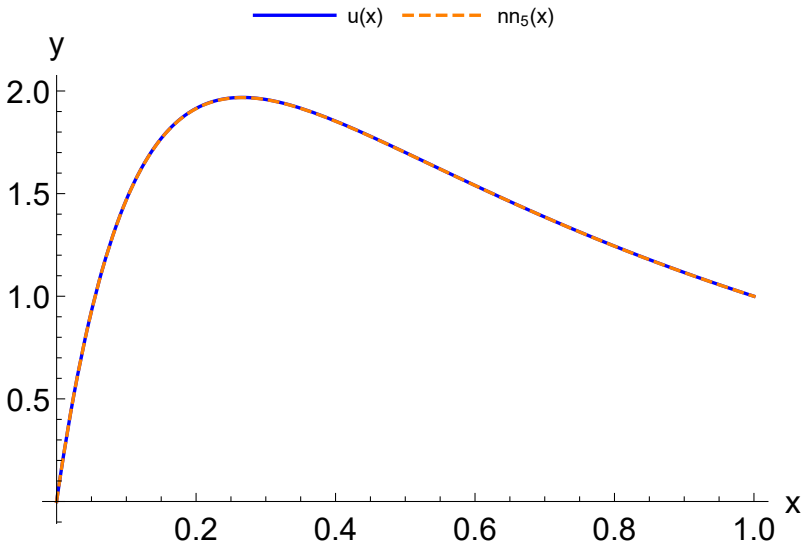

Рисунок 3 – Графики нейросетевых решений $\mathtt{nn}_5(x)$ (16) (число нейронов $n = 5$; Гауссовская базисная функция) для задачи (53) и ее точного решения $u(x)$ (при значении параметра $s = 0.1$), изображенные пунктирной и сплошной линиями, соответственно

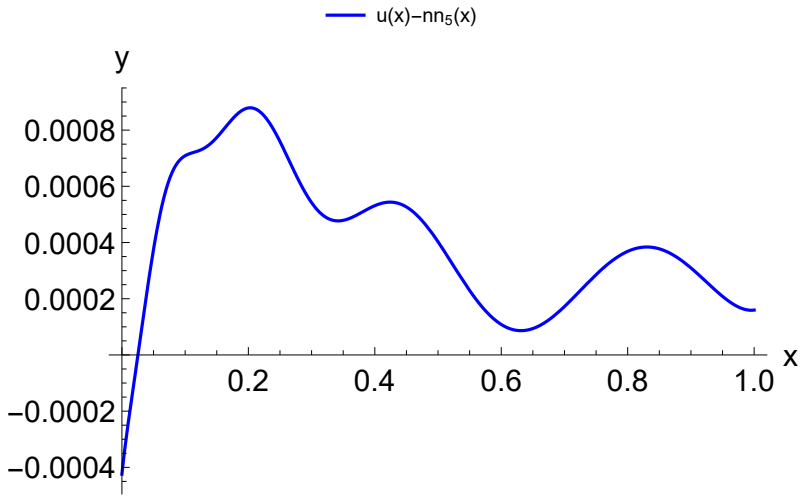

Рисунок 4 – График ошибки нейросетевого решения $\mathtt{nn}_5(x)$ (16) (число нейронов $n = 5$; Гауссовская базисная функция) для задачи (53) (при значении параметра $s = 0.1$)

График динамики функции потерь представлен на Рисунке 5.

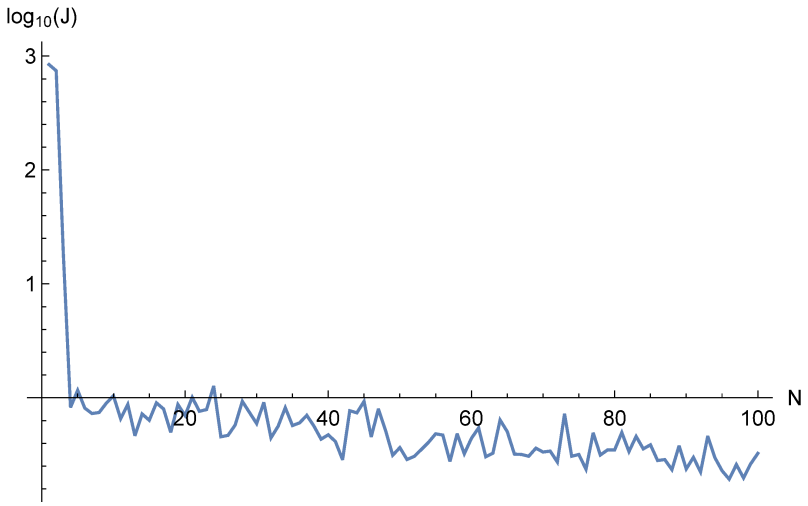

Рисунок 5 – График динамики функции потерь J нейросетевого решения $\mathtt{nn}_5(x)$ (16) для задачи (53),при значении параметра $s = 0.1$, относительно количества N эпох обучения. Число нейронов $n = 5$; Гауссовская базисная функция

Для сигмоидных базисных функций графики решения и ошибок показаны на Рисунках 6 и 7. Аналитическое решение в этом случае выглядит как

$$\begin{aligned}\mathtt{n}_5(x) = {} & 2.9401\mathtt{th}(0.1548(-1291.23 + x)) - 0.2423\mathtt{th}(2.8222(-0.5127 + x)) \\ & - 3.4427\mathtt{th}(0.2677(-0.4512 + x)) + 0.8904\mathtt{th}(6.6248(-0.0469 + x)) \\ & + 3.7422\mathtt{th}(9.4914(0.0895 + x)).\end{aligned} \tag{56}$$

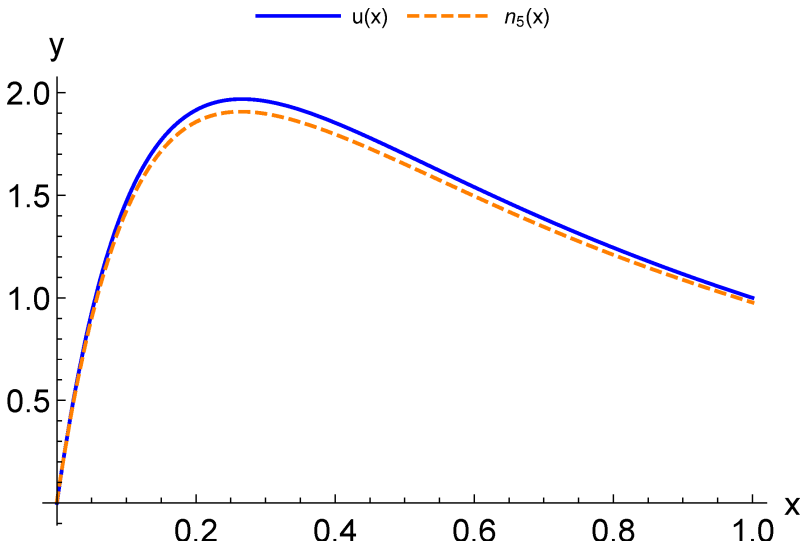

Рисунок 6 – Графики нейросетевого решения $\mathtt{n}_5(x)$ (16) (число нейронов $n = 5$; сигмоидная базисная функция)для задачи (53) и ее точного решения $u(x)$ (при значении параметра $s = 0.1$), изображенные пунктирной и сплошной линиями, соответственно

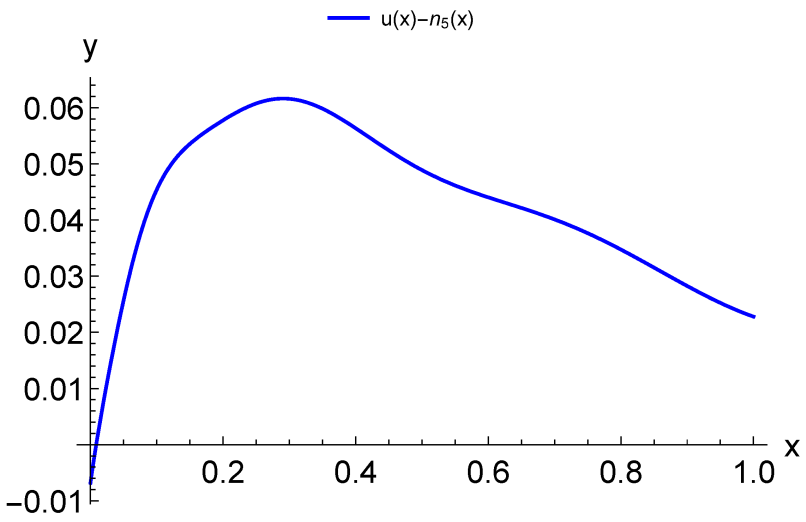

Рисунок 7 – График ошибки нейросетевого решения $\mathtt{n}_5(x)$ (16) (число нейронов $n = 5$; сигмоидная базисная функция) для задачи (53), при значении параметра $s = 0.1$

Рисунок 8 иллюстрирует динамику оптимизации функции потерь для решения (56) в виде десятичного логарифма периода обучения.

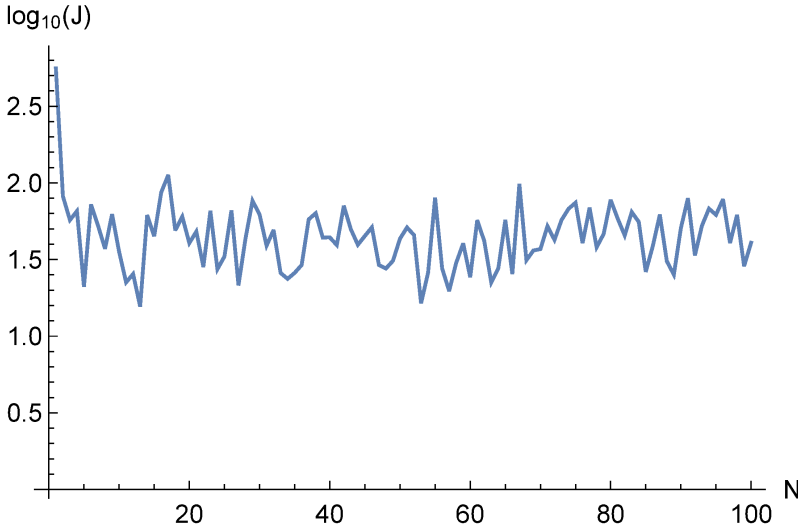

Рисунок 8 – График динамики функции потерь J нейросетевого решения $\mathtt{nn}_5(x)$ (16) для задачи (53), при значении параметра $s = 0.1$, относительно количества N эпох обучения. Число нейронов $n = 5$; сигмоидная базисная функция

Для задачи в более жесткой постановке, например, при $s = 0.01$, без дополнительных усилий можно получить приемлемое решение, только используя сеть с $n = 20$ нейронами скрытого слоя и сигмоидными базисными функциями. Соответствующие графики аппроксимации и ее погрешности показаны на Рисунках 9 и 10.

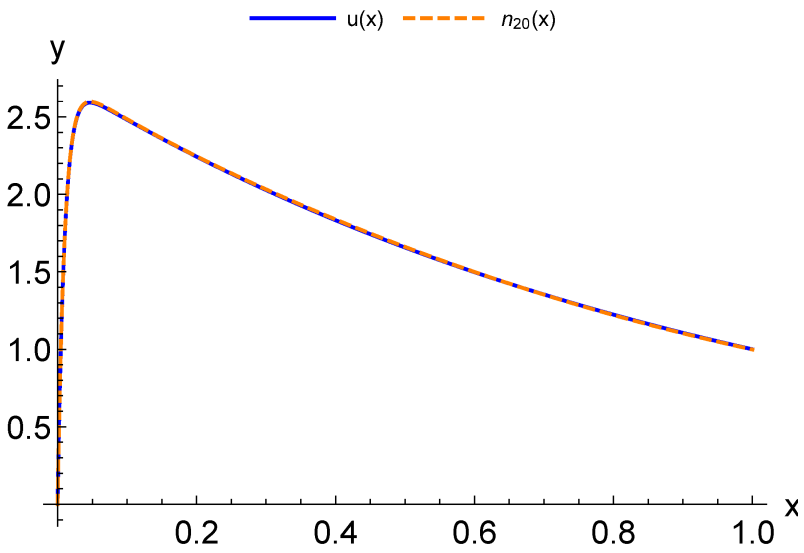

Рисунок 9 – Графики нейросетевого решения $\mathtt{n}_{20}(x)$ (16) (число нейронов $n = 20$; сигмоидная базисная функция) для задачи (53) и ее точного решения $u(x)$ (при значении параметра $s = 0.01$), изображенные пунктирной и сплошной линиями, соответственно

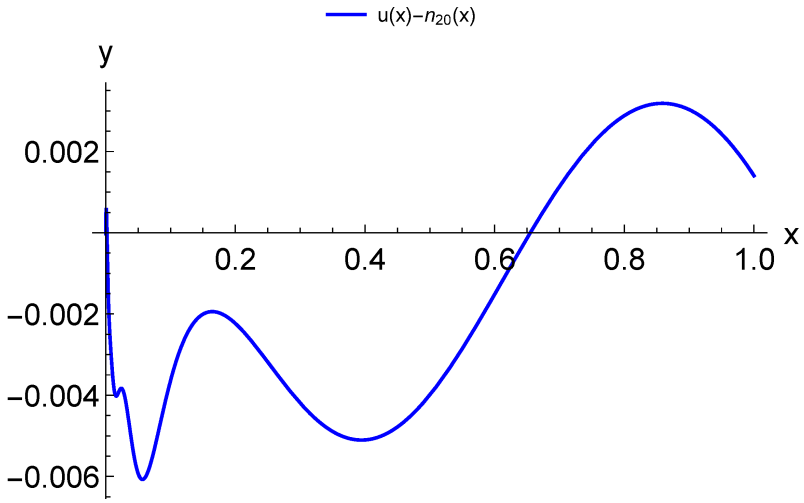

Рисунок 10 – График ошибки нейросетевого решения $\mathtt{n}_{20}(x)$ (16) (число нейронов $n = 20$, сигмоидная базисная функция) для задачи (53), при значении параметра $s = 0.01$

В этом случае нейросетевое решение имеет более громоздкий вид. На Рисунке 11 изображена динамика оптимизации соответствующей функции потерь.

Основываясь на результатах для различных базисных функций в случае постановки задачи с фиксированными параметрами, для задачи с переменным параметром была выбрана сеть-персептрон со 100 нейронами скрытого слоя и сигмоидными функциями активации для построения параметрического нейросетевого решения. Тестовые точки $\{x_i, s_i\}$ выбираются случайным образом вобласти $[0, 1] \times [0.03, 0.1]$. RProp используется в качестве алгоритма мини-

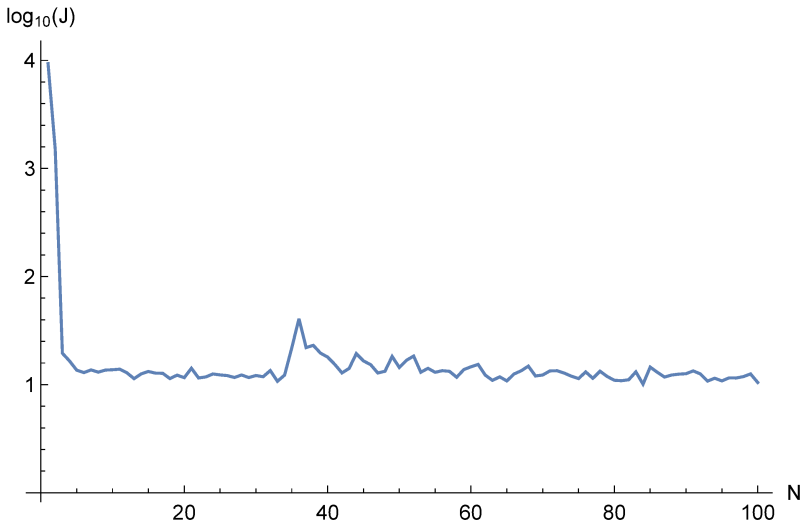

Рисунок 11 – График динамики функции потерь J нейросетевого решения $\mathtt{nn}_{10}(x)$ (16) для задачи (53), при значении параметра $s = 0.01$,относительно количества N эпох обучения. Число нейронов $n = 20$; сигмоидная базисная функция

мизации [34]. Одна эпоха соответствует 10 перегенерациям точек, в которых вычисляется функция потерь. Между каждыми двумя регенерациями выполняется 5 шагов алгоритма RProp. Количество тестовых точек на каждом этапе равно 200.

Рисунок 12 иллюстрирует динамику обучения нейронной сети.

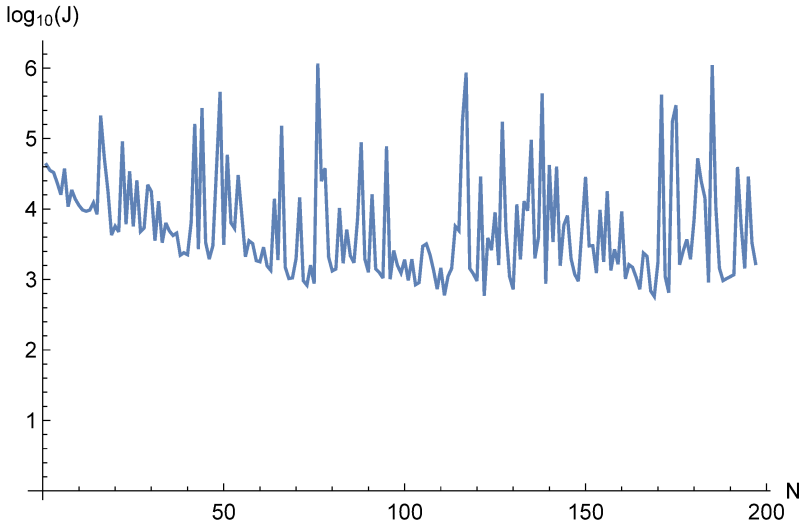

Рисунок 12 – График динамики функции потерь J нейросетевого решения $\mathtt{nn}_{100}(x, s)$ (16) для задачи (53),при значении параметра $s = 0.01$, относительно количества N эпох обучения. Число нейронов $n = 20$; сигмоидная базисная функция

3D-график ошибки аппроксимации представлен на Рисунке 13.

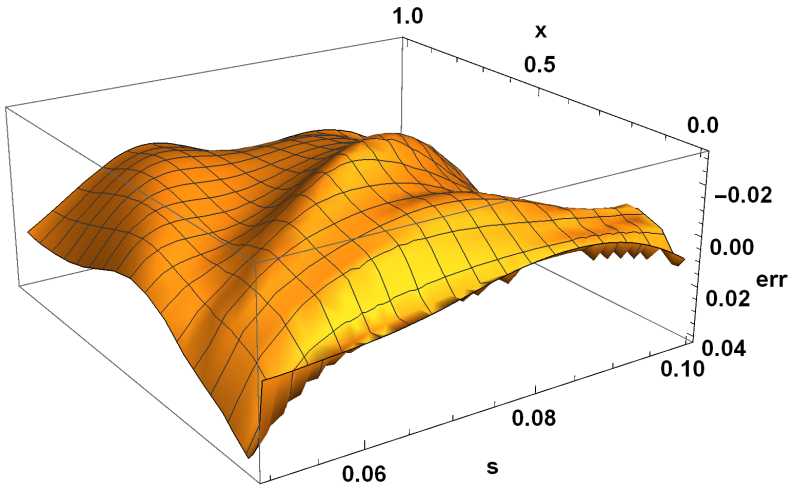

Рисунок 13 3D-график ошибки аппроксимации нейросетевого параметрического решения $\mathtt{nn}_{100}(x, s)$ (16) для задачи (53) в области $x \in [0, 1]$ и $s \in [0.05, 0.1]$. Число нейронов $n = 100$; сигмоидная базисная функция

3.5.2 Общий нейросетевой подход и сглаживание МКЭ

Для решения параметризованной задачи используется нейросетевой подход для сглаживания решений МКЭ, построенных для фиксированных значений параметра. Особенности метода требуют использования единого набора базисных функций МКЭ независимо от значения параметра. Таким образом, не используется МКЭ с адаптивными сетками (адаптивными базисными функциями).

Сначала система (53) решается при помощи МКЭ с различными значениями числа K равноудаленных вершин x_k и различными значениями параметра s. Для простоты вместо $u(x)$ было найдето приближенное решение задачи с однородными граничными условиями, т.е. приближение для $u(x) - x$.

Так как задача является одномерной, приближенное решение системы имеет вид

$$u_N(x) = \mathbf{c}(s) \cdot \mathbf{u}, \tag{57}$$

где вектор $\mathbf{u}$ состоит из K кусочно-линейных функций $u_k(x)$

$$u_k(x) = \begin{cases} \dfrac{x - x_{k-1}}{x_k - x_{k-1}}, & x \in [x_{k-1}, x_k]; \\ \dfrac{x_{k+1} - x}{x_{k+1} - x_k}, & x \in [x_k, x_{k+1}]; \\ 0, & \text{иначе}; \end{cases} \tag{58}$$

Коэффициенты $\mathbf{c}(s) = \{c_k(s)\}$ находятся решением соответствующих систем линейных уравнений.

Таким образом были рассчитаны значения коэффициентов $c_i(s_l)$ для решений МКЭ (57) для значений параметра $s_l = 0.05, \ldots, 0.1$ с шагом 0.01 и числом узлов $K = 10$. В итоге были получены обучающие наборы $\{s_l, c_i(s_l)\}$, которые использовались для аппроксимации нейронной сетью коэффициентов МКЭ в зависимости от параметра s

$$p_i(s) = a_{1i} + a_{2i}\mathtt{th}(a_{3i}(s - a_{4i})). \tag{59}$$

После нейросетевой аппроксимации каждого из коэффициентов было получено общее параметрическое решение

$$\mathtt{znn}(s, x) = \sum_{i=1}^{K-1} p_i(s)\varphi_i(x). \tag{60}$$

Это решение, в свою очередь, можно рассматривать как результат работы нейронной сети с кусочно-заданными функциями активации

$$\varphi_i(x) = \begin{cases} 1 - |K(x - \dfrac{i}{K})|, \ |x - \dfrac{i}{K}| < \dfrac{1}{K}; \\ 0, \ \text{иначе.} \end{cases} \tag{61}$$

Нейронная сеть (60) представляет собой гибридную версию РБФ-сети с предварительно выбранными центрами и шириной и персептрона с одним скрытым слоем.

На Рисунке 14 представлены экстраполяции нейронной сетью решения задачи (53) для параметра s со значением 0.01, который не включен в обучающий набор. Параметрическое решение (60), полученное таким образом, показывает довольно хорошее приближение к решению задачи (53) при малых значениях параметра s. Для сравнения, на рисунке показано решение МКЭ с тем же числом $K = 20$ неадаптивных базисных функций (58), которое отражает сложность задачи (53) при малых значениях параметра. При этом, даже простое сглаживание этого решения МКЭ по вершинам за вычетом x с помощью сети с 3 нейронами скрытого слоя

$$zn\left(x, \mathbf{p}, \mathbf{q}, \mathbf{r}\right) = p_0 + \sum_{i=1}^{3} p_i \mathtt{tanh}\left(q_i\left(x - r_i\right)\right), \tag{62}$$

повышает качество решения (см. Рисунок 14, красные и оранжевые линии).

Параметры $\mathbf{p}$, $\mathbf{q}$, $\mathbf{r}$ выбираются путем оптимизации стандартной функции потерь с использованием метода RProp.

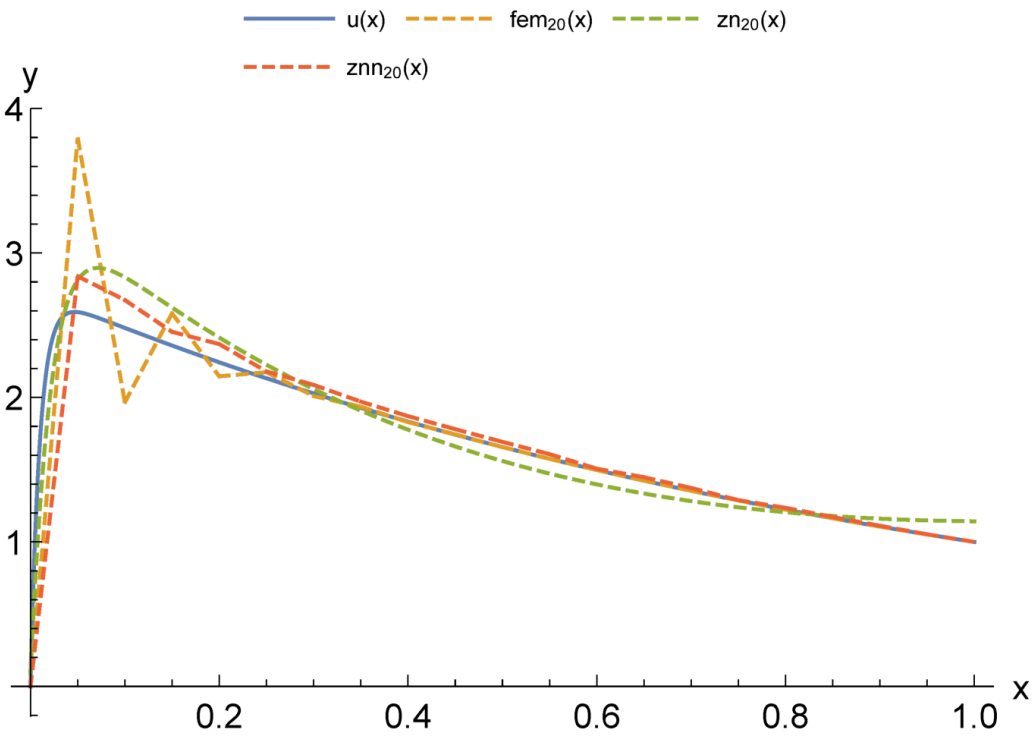

Рисунок 14 – Графики параметрического МКЭ нейросетевого решения $\mathtt{znn}_{20}(x)$ (60), МКЭ решения $\mathtt{fem}_{20}(x)$ и нейросетевого сглаживания МКЭ решения $\mathtt{zn}_{20}(x)$ (62) для параметризованной задачи (53) (значение параметра $s = 0.01$; число элементов МКЭ $K = 20$), и точного решения $u(x)$, представленные пунктирными и сплошной линиями, соответственно

3.5.3 Аналитическая модификация классических численных методов

Чтобы применить идею, описанную в параграфе 3.3, решение системы (53) представляется как линейная комбинация решений $\varphi(x) = (\varphi_1(x), \varphi_2(x))$ and

$\xi(x) = (\xi_1(x), \xi_2(x))$ двух краевых задач, где первая задача имеет вид

$$\begin{cases} y'(x) = z(x); \\ z'(x) = -a(s)z(x) - b(s)y(x),\ x \in [0,1]; \\ y(x_0) = 1; \\ z(x_0) = 0. \end{cases} \tag{63}$$

и вторая

$$\begin{cases} y'(x) = z(x); \\ z'(x) = -a(s)z(x) - b(s)y(x),\ x \in [0,1]; \\ y(x_0) = 0; \\ z(x_0) = 1. \end{cases} \tag{64}$$

Здесь оптимальная точка x_0 зависит от параметра s и может быть выбрана в ходе решения задачи. Далее принимается, что оптимальной точкой считается точка, минимизирующая максимальную погрешность приближенного решения на отрезке $[0,1]$.

Решения задач (63) и (64) могут быть получены с помощью многослойного метода. Тогда решением системы (53) является

$$u(x) = c_1\varphi_1(x) + c_2\xi_1(x). \tag{65}$$

Коэффициенты $c_{1,2}$ определяются граничными условиями системы (53)

$$\begin{cases} c_1\varphi_1(0) + c_2\xi_1(0) = 0; \\ c_1\varphi_1(1) + c_2\xi_1(1) = 1. \end{cases} \tag{66}$$

Для решения задач (63) и (64) были рассмотрены различные типы функций F (см. (43)). В случае использования неявного метода Эйлера (44) в качестве основы для многослойного решения, системы (63) и (64) преобразуются в

$$\begin{cases} y'_k(x) = y_{k-1} + hz_k; \\ z_k = z_{k-1} - h(a(s)z_k + b(s)y_k). \end{cases} \tag{67}$$

Эта система может быть решена относительно y_k и z_k. В качестве начальной точки x_0 были рассмотрены 0 и оптимальная точка, упомянутая выше.

Результирующее многослойное решение задачи (53) для $n = 10$ слоев и оптимальной точки $x_0 = 0.151$ имеет аналитический вид

$$y_n(x, 0.151, 10) = \frac{58.9 + 303x + 850x^2 + 1940x^4 + 1680x^5 + 100x^6 + 391x^7 + 90.6x^8 + 9.44x^9}{10^{-9}x^{-1}(8.59 + 9.9x + x^2)^{10}}.$$

Соответствующие многослойные решения, основанные на неявном методе Эйлера для $x_0 = 0$ и $x_0 = 0.151$ при $s = 0.1$, изображены на Рисунке 15

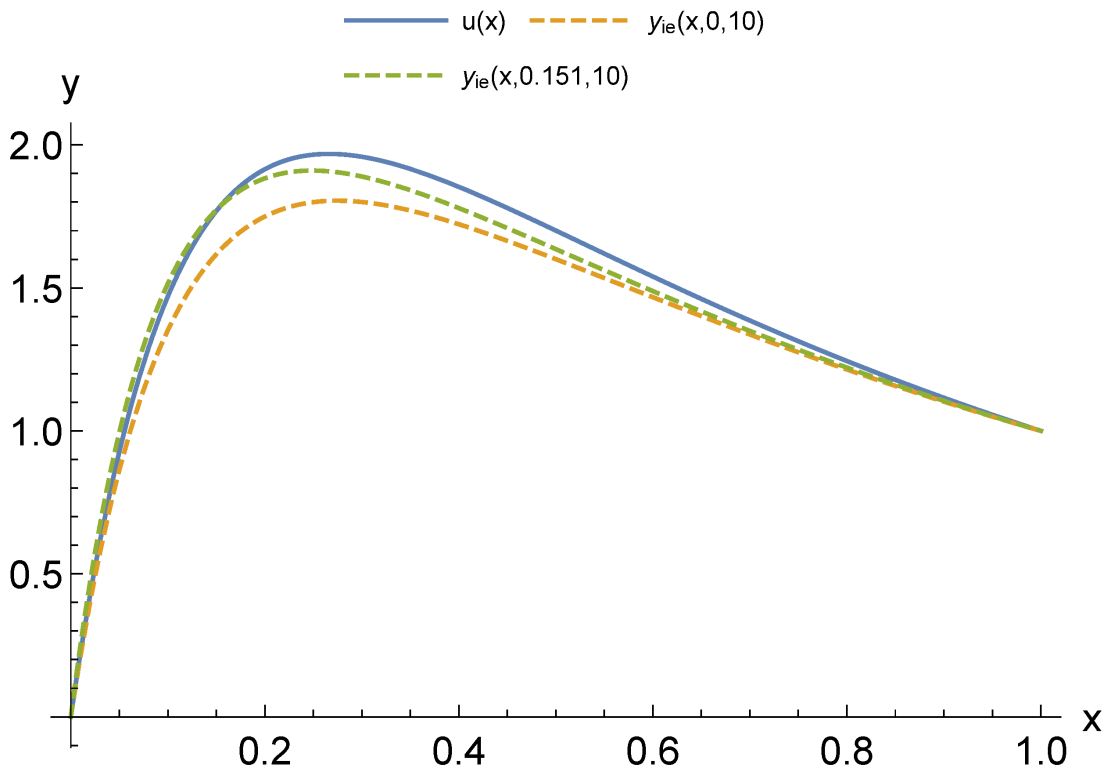

Рисунок 15 – Графики многослойных решений $y_{\mathtt{ie}}(x, 0, 10)$, $y_{\mathtt{ie}}(x, 0.151, 10)$ (65), полученных в результате аналитической модификации неявного метода Эйлера(число слоев $n = 10$) для задачи (53) и ее точное решение $u(x)$ (значение параметра $s = 0.1$), представленные пунктирными и сплошной линиями, соответственно

Если функция F из (43) соответствует методу трапеций, то системы (63) и (64) приводятся к виду

$$\begin{cases} y_k = y_{k-1} + \dfrac{1}{2}h(z_k + z_{k-1}); \\ z_k = z_{k-1} - \dfrac{1}{2}h(a(s)z_k + b(s)y_k + a(s)z_{k-1} + b(s)y_{k-1}). \end{cases} \tag{68}$$

В то же время при значении параметра $s = 0.1$ трехслойная сеть дает решение с погрешностью менее 0.02 даже без выбора оптимальной точки. Этот факт проиллюстрирован на Рисунке 16. На нем также показано, как выбор начальной точки оказывает существенное влияние на результат построения двухуровневой сети.

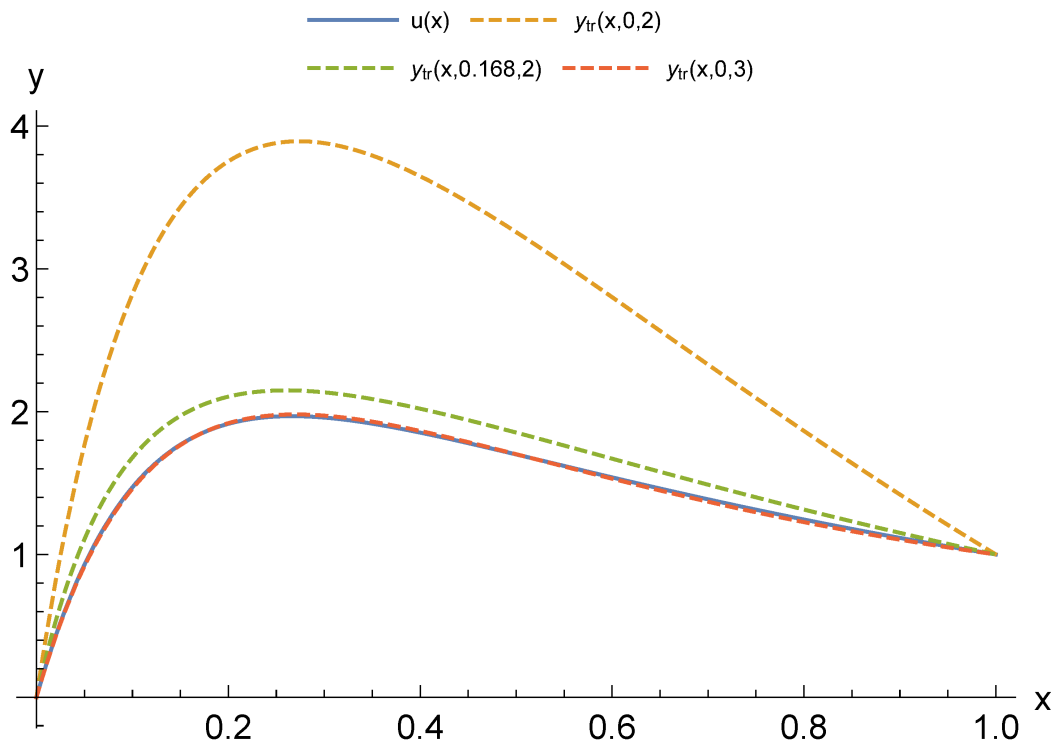

Рисунок 16 – Графики многослойных решений $y_{tr}(x, 0, 2)$, $y_{tr}(x, 0.168, 2)$, $y_{tr}(x, 0, 3)$ (число слоев $n = 2, 2, 3$) для задачи (53)и ее точное решение $u(x)$ (значение параметра $s = 0.1$), представленные пунктирными и сплошной линиями, соответственно

Рассмотрим более жесткую постановку задачи, когда значение параметра s равно 0.001. В этом случае можно получить достаточно точное приближенное решение, используя двухслойную сеть, выбрав оптимальную точку (см. Рисунок 17), а формула (43) приводит к простому виду решения

$$y_{\mathtt{tr}}(x, 0.0046, 2) = \frac{x(-0.0498 + 44.7x - 22.5x^2 + 2.79x^3)}{(0.00238 - 3.99x - x^2)^2}.$$

Увеличение количества слоев сохраняет эффект.

Далее параметризованная задача (53) решается многослойным методом.

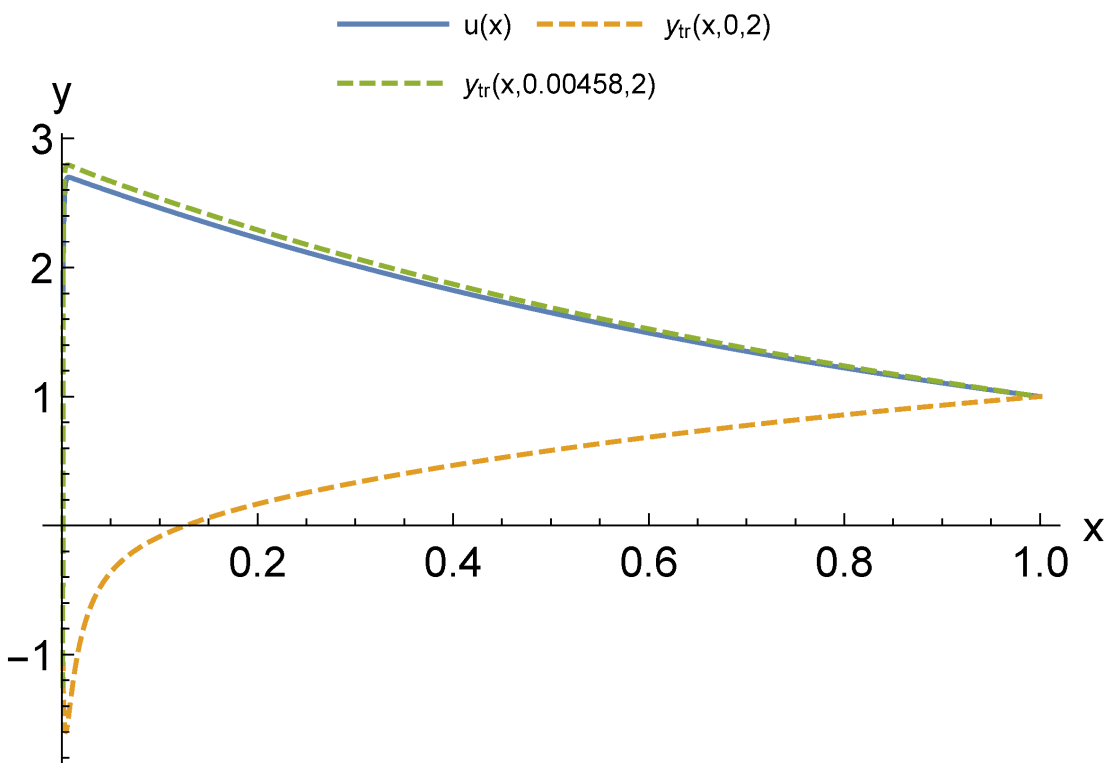

Рисунок 17 – Графики многослойных решений $y_{tr}(x, 0, 2)$, $y_{tr}(x, 0.0048, 2)$ (число слоев $n = 2$) для задачи (53) и ее точное решение $u(x)$ (значение параметра $s = 0.001$), представленные пунктирными и сплошной линиями, соответственно

В этом случае параметр s естественным образом содержится в решении,

$$y(x, s) = c_1(s)\varphi_1(s, x) + c_2(s)\xi_1(s, x). \tag{69}$$

Таким образом, однослойное решение модификации метода трапеции для $x_0 = 1.7$, имеет вид

$$y_1(x, s) = x(s^3 - 0.59s^2 - s - 0.796)^{-1} \cdot \frac{(-1.04 + 0.97s - 1.35s^2 + s^4)(0.06x + s^3 - 0.56s^2x - s - 1.39)}{s^4 + 1.18(1 - x)s^3 - (0.69x + 0.35x^2) + s(1.18x - 0.21) + s^2(0.37x^2 - 0.69x - 1)}.$$

В этом случае точка x_0, как упоминалось ранее, зависит от s, но не должна выходить за пределы рассматриваемого интервала.

Несмотря на то, что задача (53) является сложной для малых s, с помощью аналитической модификации метода трапеции с 8 слоями получается довольно хорошее решение для всех значений s из интервала $[0.001, 0.1]$. Это подтверждается трехмерным графиком ошибки аппроксимации на Рисунке 18.

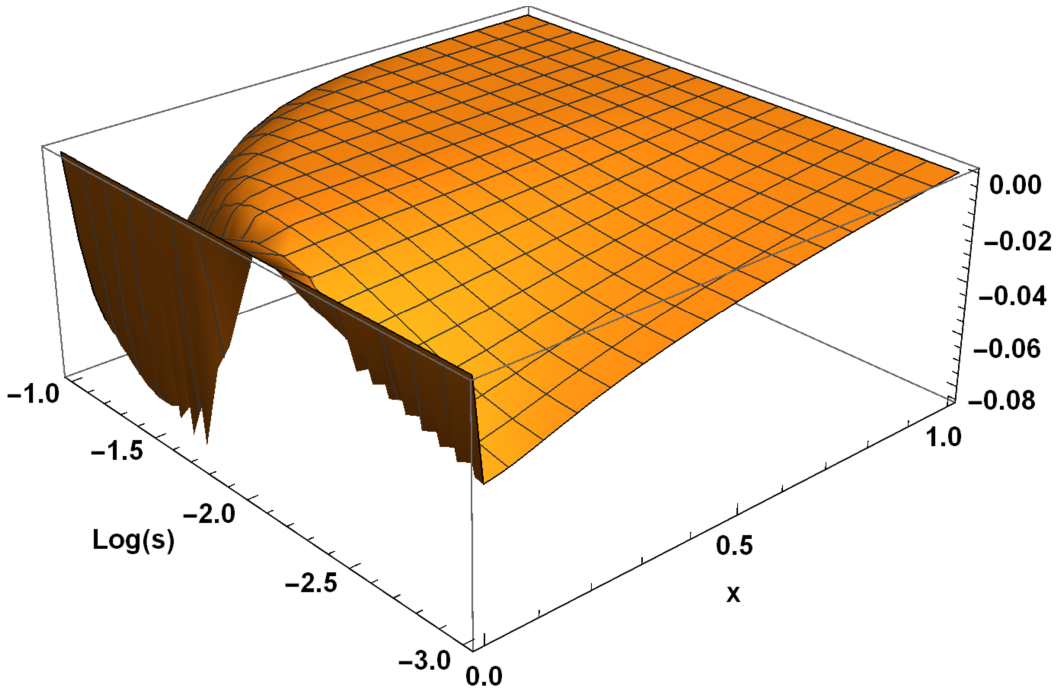

Рисунок 18 – 3D-график ошибки многослойного параметрического решения $y_8(x, s)$ (69) (модификация метода трапеций; число слоев $n = 8$; начальная точка $x_0 = \max(5s, 1)$) для задачи (53), в области $x \in [0, 1]$ и $\log(s)$, $s \in [0.1, 0.001]$

Рисунок 19 иллюстрирует, что выбор необходимого количества слоев позволяет нам получить аппроксимацию с любой точностью.

Приведенные выше результаты применения многослойного метода при решении параметризованной задачи, сингулярно возмущенной малым параметром, позволяют рассматривать этот метод как перспективный при моделировании и управлении сложными системами. Важно отметить, что, если рассматривать задачу построения предварительной модели реального объекта, то достаточно хорошее приближение для малых значений параметра дает решение с одним слоем. Кроме того, известные оценки погрешностей аппроксимации основных рекуррентных соотношений могут быть использованы для определения количества слоев для получения решения с заранее выбранной точностью. Построение многослойного решения занимает мало времени и может быть немедленно использовано для различных значений параметра. В отличие от различных нейросетевых подходов, многослойная модель не требует обучения. В то же время не исключена последующая корректировка параметров такого решения с использованием аппарата нейронных сетей в случае появления дополнительных данных. В данном случае общий нейросетевой подход должен хорошо себя заре-

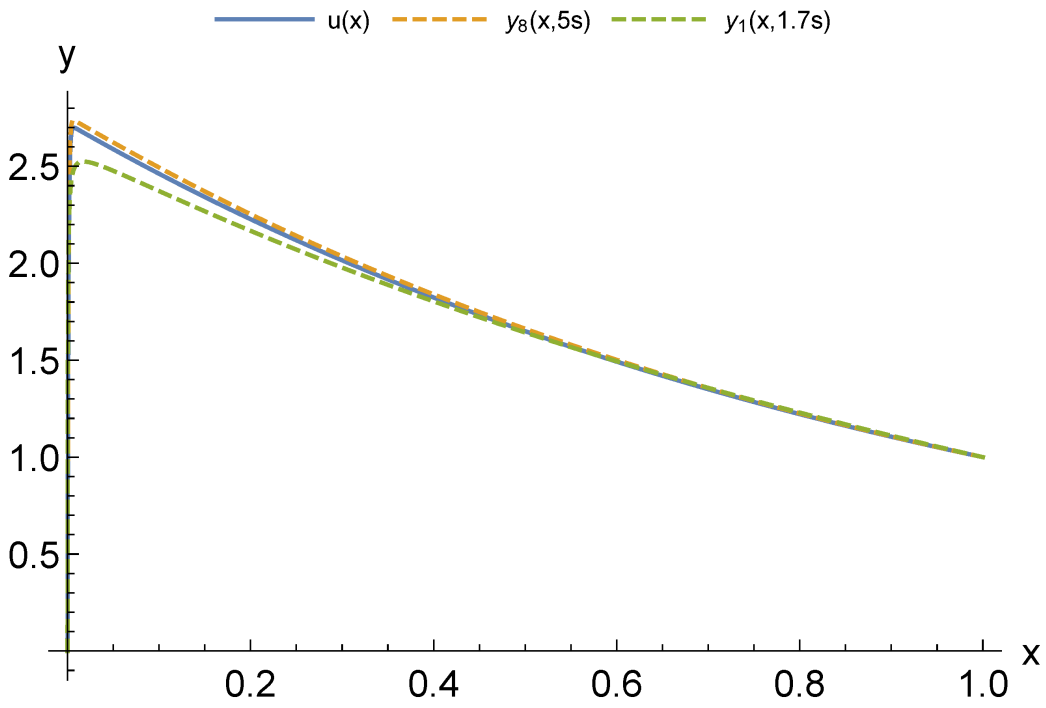

Рисунок 19 – Графики многослойных параметрических решений $y_8(x, 5s)$, $y_1(x, 1.7s)$ с оптимальными начальными точками для задачи (53), и ее точное решение $u(x)$(значение параметра $s = 0.001$), представленные пунктирными и сплошной линиями, соответственно

комендовать, что подтверждается как результатами статьи [12], так и гибридными нейросетевыми моделями, построенными в этой работе с использованием "плохих" результатов классического МКЭ.

В контексте новых вызовов, стоящих перед современной теорией управления и, в частности, перед методологией создания кибер-физических систем, задачи часто некорректны и плохо формализованы и ставятся с использованием различных разнородных данных, параметризованных дифференциальных уравнений неясной структуры или без них. Требования к скорости обработки таких данных и решения динамических уравнений растут. Таким образом, разработка универсальных недорогостоящих методов, представленных в диссертации, является важной задачей.

3.6 Обсуждение результатов

При моделировании кибер-физических систем актуальным вопросом является переход от дифференциальных моделей к функциональным. В то же время важным фактором является возможность адаптации этих моделей в соответ-

ствии с данными наблюдений за объектом. Проведенные вычислительные эксперименты позволяют сделать предварительные выводы о том, какие модели могут быть более подходящими для дальнейшей адаптации.

Основным преимуществом конечно-элементных моделей является то, что они хорошо изучены. Соответственно, существует большое количество программных пакетов и библиотек. В то же время с появлением жесткости в решаемой задаче количество необходимых функций для получения приемлемой точности резко возрастает. Кроме того, возрастают трудности с адаптацией решения, поскольку оно становится неустойчивым к ошибкам в данных. Эта проблема усугубляется локальным характером базисных функций. Нейросетевые модели более способны к адаптации. Однако процесс их построения отличается высокой вычислительной сложностью, которая также значительно возрастает с появлением жесткости. Эта ситуация позволяет перспективно взглянуть на рассмотренные в данной главе диссертации гибридные подходы. Еще больший интерес представляют многослойные модели, которые являются наиболее компактными при заданной точности.

Другой важной проблемой являются вычислительные затраты. Наиболее быстрыми являются многослойные аналитические модификации классических численных методов, поскольку они не требуют обучения. Следующими являются методы с постоянными базисными функциями, но в результате их применения получаются более крупные модели. Также требуется больше времени для их дальнейшей адаптации. Обучение нейронных сетей может быть ускорено, но это актуально для этапа адаптации на основе измерений в реальном времени.

Вопрос о распространении представленных результатов на нелинейные задачи заслуживает отдельного обсуждения. Для этих задач преимущества МКЭ существенно нивелируются, поскольку построение МКЭ-решения больше не сводится к решению линейной системы. В то же время сложность построения нейронной сети и многослойного решения меняется незначительно.

Глава 4.

Методы построения нейронных сетей с архитектурой, основанной на физике с применением к задаче моделирования химического реактора

4.1 Введение

В этой главе диссертации предлагается новый класс нейросетевых моделей с архитектурой, основанной на физике (PBA-PINNs). Главной особенностью моделей такого типа является то, что в отличие от классических физически-информированных нейронных сетей, не только обучение весов, но и архитектура самой сети основаны на физике. Результаты были опубликованы в ... и представлены на конференциях.

В предложенном подходе решается проблема первичной инициализации нейронной сети как с точки зрения архитектуры, так и выбора начальных весов. Напомним, что сетевая архитектура обычно задается вручную [37]. Рассмотренные в этой главе PINN имеют простую архитектуру, что имеет свои преимущества [139, 138].

Построение модели PBA-PINNs происходит на нескольких этапах, позволяя использовать данные о моделируемом объекте различной точности. На первом этапе строится модель PBA на основе аналитической модификации классических численных методов со встроенными модулями нейронной сети. Использование численных методов для улучшения моделей PINN не только в качестве дополнительных данных регуляризации функции потерь набирает популярность [85, 98, 117, 88].

При небольшом числе итераций численного метода предложенная в преды-

дущей главе исходная модель PBA является компактной, но имеет низкую точность. В то же время это позволяет быстро (сравнивая с [23]) обучить сеть на втором этапе, когда потери, соответствующие степени удовлетворения уравнениям физики, минимизируются по всему пространству параметров. На третьем этапе при наличии данных измерений высокой точности модель может быть дополнительно обучена и выходит на уровень наибольшей точности. Использование данных измерений при этом реализует концепцию индустрии 4.0 по активному управлению производством [171].

Эффективность предложенных методов продемонстрирована при решении эталонной задачи, а именно стационарной задачи о тепловом взрыве неизотермического химического реактора в плоскопараллельном случае [67]. Метод Рунге-Кутты и метод стрельбы широко используются при моделировании химических процессов, см. [110, 69, 119], и естественно использовать аналитическую модификацию этих методов для решения задачи. В подобных задачах в качестве измеряемых величин выступают энергия активации, температура и теплопроводность. Измерение теплопроводности в химическом реакторе рассмотрено в работах [116, 58]. В работах [114, 129, 133] исследуются измерения температуры. Параметры, рассматриваемые в этом исследовании, также включают измерения энергии активации химического реактора, которые исследованы в работе [108, 151].

Эта глава структурирована следующим образом. В разделе 4.2 подробно рассматриваются методы, применяемые на каждом этапе предложенного подхода. В разделе 4.3 описывается экспериментальная задача, представлены результаты построения многопараметрических моделей PBA-PINN с высокой точностью и демонстрируется применение высокоточных сетей для решения задачи идентификации параметров. В разделе 4.4 приведены выводы и обсуждение результатов и самого метода.

4.2 Многоуровневое моделирование с использованием нейронной сети и аналитической модификации численных методов

В этом разделе последовательно описываются этапы построения нового класса нейросетевых моделей, основанных на физике, с архитектурой, основанной на

физике, для краевой задачи общего вида. Весь процесс схематично показан на Рисунке 20.

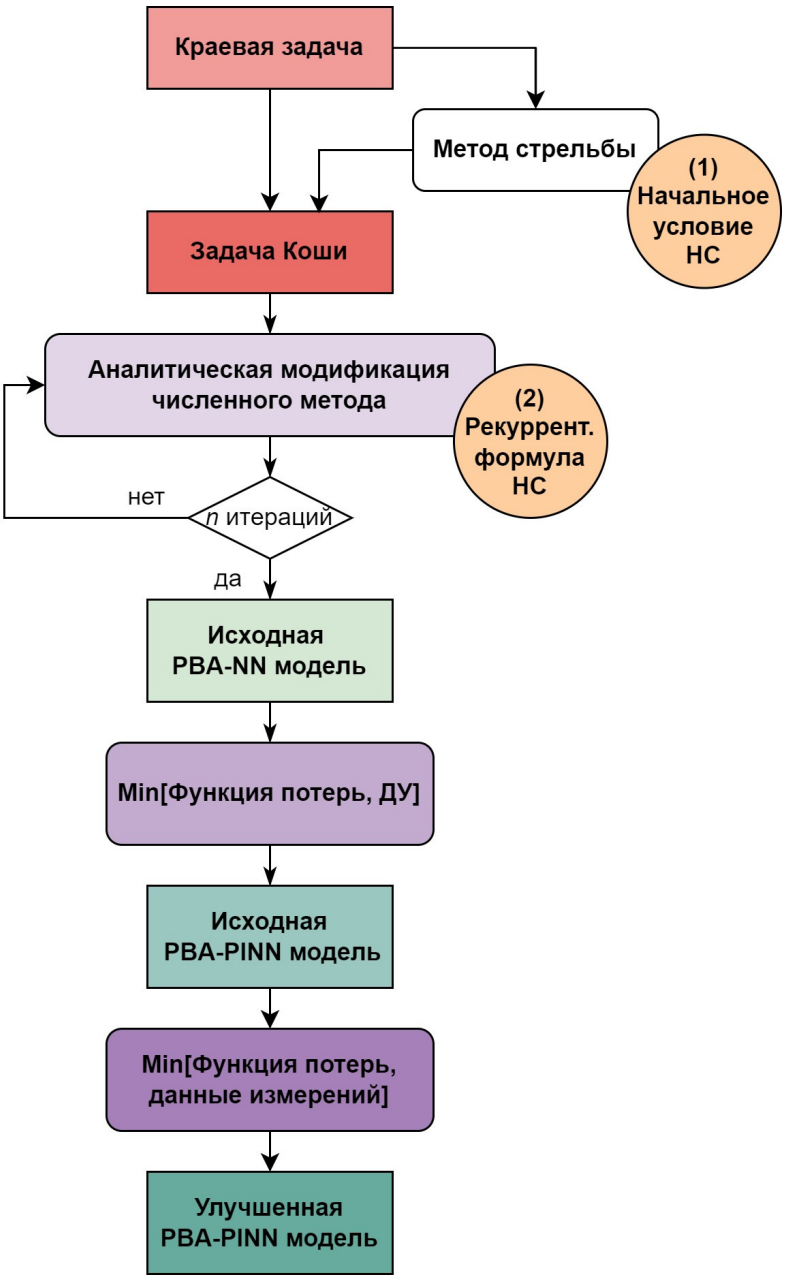

Рисунок 20 – Общая схема построения и обучения многоуровневой модели PBA-PINN

4.2.1 Общая постановка задачи

Рассмотрим краевую задачу в общей форме

$$\mathbf{y}'(x) = \mathbf{f}(x, \mathbf{y}(x)), \tag{70}$$

$x \in [a_1, a_2] = D$, с граничными условиями

$$\mathscr{B}_1[\mathbf{y}](a_1) = \mathbf{b}_1,\ \mathscr{B}_2[\mathbf{y}](a_2) = \mathbf{b}_2; \tag{71}$$

где $\mathbf{f}(x, \mathbf{y}(x))$ – подходящая функция, $\mathscr{B}_1[\cdot], \mathscr{B}_2[\cdot]$ – некоторые операторы, и $\mathbf{y}(x)$ – искомое решение.

4.2.2 Аналитическая модификация метода стрельбы и построение модели с помощью аналитической модификации численных методов

Здесь представлена аналитическая модификация метода стрельбы. Задача (70)-(71) рассматривается как задача Коши с предположением, что

$$\mathbf{y}(x_0) = \mathbf{y}_0, \tag{72}$$

где x_0 – некоторая точка на интервале D. Далее, согласно подходу, описанному в предыдущей главе, аналитическое решение на интервале с переменным правым (левым) концом $x \in D$ строится с помощью известных формул для численного решения задачи Коши для системы обыкновенных дифференциальных уравнений [62].

Классические численные методы состоят в разделении интервала, на котором решается задача, на n частей $x_0 < \ldots < x_n = x_0 + a$. Для нахождения значений приближенного решения в этих точках используется итерационная формула

$$\mathbf{y}_{k+1} = \mathscr{A}[\mathbf{f}, \mathbf{y}_k, \mathbf{y}_{k+1}, h_k, x_k], \tag{73}$$

где $h_k = x_{k+1} - x_k$. Таким образом, $\mathbf{y}_k$ приближает точное значение искомого решения в точке x_k, а $\mathscr{A}[\cdot]$ – это функция, которая определяет конкретный используемый метод. Шаги h_k рассматриваются как функции переменной x, $h_k(x)$. В простейшем случае равномерного разбиения следует, что $h_k = (x - x_0)/n$ и $x_k = x_0 + (x - x_0)k/n$.

Функция $\mathbf{y}_n(x, \mathbf{y}_0)$, построенная на последнем шаге, определяет приближенное аналитическое решение задачи (70) + (72). Это решение включает в себя в качестве векторного параметра начальное значение $\mathbf{y}_0 = \mathbf{y}_0(x)$, которое, подобно классическому методу стрельбы, определяется из граничных условий

$$\mathscr{B}_1[\mathbf{y}_n](a_1, \mathbf{y}_0) = \mathbf{b}_1,\ \mathscr{B}_2[\mathbf{y}_n](a_2, \mathbf{y}_0) = \mathbf{b}_2. \tag{74}$$

Таким образом, получается аналитическая модификация любой классической численной схемы и метода стрельбы. Построенное таким образом решение представляет собой модель с архитектурой, обусловленной физикой, рассмотренную в прошлой главе. Выбор достаточно большого значения n обеспечивает сколь угодно точную аппроксимацию.

4.2.3 Внедрение нейросетевых конструкций в модели

Схема на Рисунке 20 содержит круглые символы рядом с методом стрельбы и аналитической модификацией численных методов, что указывает на возможность внедрения нейронных сетей на этих этапах.

Если формула (73) определяет некоторый явный численный метод, то приближенное решение может быть вычислено в виде явной функции. Если использовать ее напрямую неудобно, например, она имеет громоздкое выражение, можно аппроксимировать эту функцию с помощью нейронной сети. В результате применения итерационной формулы (73), включающей замену на нейросетевую функцию, получается приближенное решение задачи (70)–(71), представляющее собой многослойную нейросетевую модель.

Если функция $\mathscr{A}$ зависит от $\mathbf{y}_{k+1}$, то отношение (73) можно рассматривать как уравнение относительно $\mathbf{y}_{k+1}$. В случае наличия точного решения вместо уравнения (73), получается соотношение вида

$$\mathbf{y}_{k+1} = \mathscr{B}[\mathbf{f}, \mathbf{y}_k, h_k, x_k]. \tag{75}$$

И мы попадаем в ситуацию, аналогичную использованию явного метода.

В случае же, если уравнение (73) не может быть решено точно относительно $\mathbf{y}_{k+1}$, в качестве приближенного решения соответствующего неявного уравнения можно использовать специально обученную заранее однослойную нейронную сеть для получения приближенной формулы вида (75). В результате, как и прежде, строится многослойная (глубокая) нейросетевая модель или набор моделей, с архитектурой, обусловленной физикой задачи, и начальной точностью в соответствии с выбранным базовым численным методом и числом слоев (итераций) его аналитической модификации.

4.2.4 Построение модели PBA-PINN

На данном этапе уточняется исходная нейростевая модель, полученная применением аналитической модификации численных методов,

$$\mathbf{y}_n(x, \mathbf{y}_0, \mathbf{a}), \tag{76}$$

где $\mathbf{a}$ – вектор, включающий все веса нейронной сети, построенной на предыдущем шаге, и далее настраиваемый в ходе минимизации функции потерь, соответствующей дифференциальной задаче, как это обычно бывает в работах, посвященных физически-информированным нейронным сетям или использующих общий нейросетевой подход.

Если формулировка задачи (70)-(71) содержит некоторые параметры $\mathbf{p}$, они автоматически включаются в выражение для модели PBA-PINN $\mathbf{y}_n \to \mathbf{y}_n(x, \mathbf{y}_0, \mathbf{a}, \mathbf{p})$, что приводит к построению параметрической многослойной нейросетевой модели.

Дальнейшее уточнение модели может быть осуществлено путем минимизации функции потерь

$$\begin{aligned}\sum_{j=1}^{m} &\left\|\mathbf{y}_n'(x_j, \mathbf{p}_j, \mathbf{a}) - \mathbf{f}(x_j, \mathbf{y}_n(x_j, \mathbf{p}_j, \mathbf{a}), \mathbf{p}_j)\right\|^2 \\ &+ \lambda \Big(\left\|\mathscr{B}_1[\mathbf{y}_n](a_1(\mathbf{p}_j), \mathbf{y}_0(\mathbf{p}_j), \mathbf{a}) - \mathbf{b}_1(\mathbf{p}_j)\right\|^2 \\ &\qquad + \left\|\mathscr{B}_2[\mathbf{y}_n](a_2(\mathbf{p}_j), \mathbf{y}_0(\mathbf{p}_j), \mathbf{p}_j) - \mathbf{b}_2(\mathbf{p}_j)\right\|^2 \Big). \end{aligned} \tag{76}$$

В этом случае значения параметров берутся из интересующей области, λ – это обычный гиперпараметр, который регулирует вклад каждого члена потерь в значение функции потерь.

Результатом данного этапа является физически-информированная многослойная нейросетевая модель (или набор моделей) с архитектурой, обусловленной физикой объекта.

В процессе обучения нейронной сети зависимость $\mathbf{y}_n(x, \mathbf{y}_0, \mathbf{p})$ от $\mathbf{y}_0$ может быть нарушена. Этого можно избежать, если $\mathbf{y}_0$ в уравнениях (74) заменить на другую нейронную сеть, веса которой определяются в процессе минимизации соответствующей функции ошибки.

Важно отметить, что в соответствии с предложенными методами процесс

обучения физически-информированной нейронной сети начинается не со случайной инициализации веса нейронной сети, а с относительно успешной начальной аппроксимации (точность аппроксимации зависит от точности численного метода, точности решения неявного уравнения при его наличии и количества используемых итераций), что значительно сокращает время обучения нейронной сети, как это продемонстрировано далее на примере параграфа 4.3.

4.2.5 Получение моделей высокой точности на основе данных датчиков

На третьем этапе построенная ранее модель PBA-PINN может быть уточнена еще больше в результате обучения в соответствии с высокоточными данными, поступающими от датчиков. Компактность построенной модели позволяет легко адаптировать ее к реальным измерениям. Веса PBA-PINN переобучаются путем минимизации функции потерь

$$\sum_{j=1}^{M} \left(\mathbf{y}_n(x_j, \mathbf{y}_0, \mathbf{a}, \mathbf{p}_j) - \mathbf{m}_j\right)^2, \tag{77}$$

где $\{\mathbf{m}_j, \mathbf{p}_j\}_{j=1}^{M}$ – данные датчиков в точках x_j.

Получаемая в результате быстрого обучения ввиду своей компактности модель при этом имеет высокую точность соответствия объекту, описываемому уравнениями (70)-(71) и данными измерений.

4.2.6 Идентификация параметров на основе данных

Как отмечалось ранее, построение параметрических решений важно с точки зрения удобства анализа модели в различных условиях и мгновенного ответа на запрос для любого местоположения в интересующем нас пространстве параметров. Таким образом, подобная модель может быть применена для решения обратной задачи (идентификации параметров) на основе нескольких данных датчиков. В этом случае предлагается определить новое текущее значение параметра, используя параметрическую высокоточную многослойную нейросетевую модель PBA-PINN, построенную на предыдущем шаге, и получить прогнозируемое значение параметра путем минимизации функции потерь вида

$$\sum_{j=1}^{L} \left(\mathbf{y}_n(x_j, \mathbf{y}_0, \mathbf{a}, \mathbf{p}) - \mathbf{m}_j\right)^2, \tag{78}$$

где $\{\mathbf{m}_j\}_{j=1}^{L}$ – данные датчиков в точках x_j.

В следующем разделе демонстрируется эффективность предложенных методов при решении контрольной задачи, а именно стационарной задачи о тепловом взрыве неизотермического химического реактора в плоскопараллельном случае.

4.3 Задача моделирования химического реактора

4.3.1 Постановка задачи

Описанные выше методы были применены к стационарной задаче о тепловом взрыве неизотермического химического реактора в плоскопараллельном случае [67] в предположении, что реакция является одностадийной, необратимой, не сопровождается фазовыми переходами, протекает в стационарной среде. В безразмерном виде эта задача может быть записана в виде нелинейного дифференциального уравнения с граничными условиями, которые задаются системой уравнений

$$\begin{cases} \dfrac{d^2\theta}{dx^2} + \delta \exp(\theta) = 0, \\ \dfrac{d\theta}{dx}(0) = 0,\ \theta(1) = 0, \end{cases} \quad x \in [0,1], \delta \in [0.2, 0.8]. \tag{79}$$

У этой задачи есть точное решение, которое может быть получено с помощью стандартного метода понижения порядка [62]. Это позволяет оценить качество решения, построенного с использованием методологии, рассмотренной выше.

Интервал изменения переменной взят из постановки задачи и связан с переходом к безразмерной координате. Пространство параметров выбрано для вычислительных экспериментов исходя из следующих соображений. Для небольшого значения δ можно получить приближенное решение на основе стандартных методов асимптотических разложений. Кроме того, для достижения небольшой относительной погрешности при этих значениях параметра требуется изменение функции потерь. Такие изменения зависят от конкретной задачи и за-

трудняют использование этой проблемы для иллюстрации общей методологии. Точного решения задачи при $\delta > \delta^* \approx 0,878458$ не существует. Когда решение приближается к этой критической границе, оно становится неустойчивым, и задача приобретает жесткие свойства. Соответствующие проблемы в данной главе были оставлены в стороне, чтобы не терять ясности и проиллюстрировать общие методы.

Кроме того, при эксплуатации реального реактора существует тенденция избегать работы вблизи границы стабильности. Малое значение параметра соответствует низкой скорости реакции, что делает работу реактора неэффективной. Поэтому интервал изменения параметра, аналогичный рассматриваемому, представляется наиболее интересным с практической точки зрения.

4.3.2 Преобразование уравнений

Преобразуем задачу (79) к системе

$$\begin{cases} \dfrac{d\theta}{dx} = z, \\ \dfrac{dz}{dx}(0) = -\delta \exp(\theta), \end{cases} \quad x \in [0, 1]. \tag{80}$$

и применим к ней аналитическую модификацию неявного метода Эйлера, тогда

$$\begin{cases} \theta_{k+1} = \theta_k + h_k z_{k+1}, \\ z_{k+1} = z_k - h_k \delta \exp(\theta_{k+1}). \end{cases} \tag{81}$$

Для одной итерации ($k = 1$) имеем

$$\theta_1 = \theta_0 + (x - x_0) z_0 - (x - x_0)^2 \delta \exp(\theta_1). \tag{82}$$

Пусть $x_0 = 0$, тогда $\frac{d\theta}{dx}(0) = 0$ и $\theta(1) = 0$, откуда $z_0 = 0$ и $\theta_0 = \delta$. Таким образом,

$$\theta_1 = \delta - x^2 \delta \exp(\theta_1). \tag{83}$$

Это неявное уравнение далее решается с помощью нейронной сети в соответствии с общей схемой, представленной при описании метода в начале данной главы.

4.3.3 Встраивание нейросетевой конктрукции в решение, полученное аналитической модификацией численных методов

Рассмотрим неявное уравнение

$$y + s\exp(y) = t. \tag{84}$$

Приближенное решение $y(s,t)$ для (84) ищется в виде нейронной сети с одним скрытым слоем и n нейронами в нем

$$\hat{y}(s,t,\{c_i,\mathbf{a}_i\}_{i=1}^n) = c_1 + \sum_{i=2}^{n} c_i v(s,t,\mathbf{a}_i), \tag{85}$$

где

$$v(s,t,\mathbf{a}) = th(a_1 s + a_2)th(a_3 t + a_4) \tag{86}$$

– базисная функция, параметры $\{c_i,\mathbf{a}_i\}_{i=1}^n$ которой подбираются в ходе минимизации квадратичной функции потерь

$$J = \sum_{j=1}^{M}\left(\hat{y}(s_j,t_j,\{c_i,\mathbf{a}_i\}_{i=1}^n) + s_j\exp(\hat{y}(s_j,t_j,\{c_i,\mathbf{a}_i\}_{i=1}^n)) - t_j\right)^2. \tag{87}$$

На протяжении всего решения этой задачи, в процессе обучения тестовые точки перегенерируются каждые 3-5 шагов нелинейной оптимизации соответствующей функции потерь в интересующей области. Эта перегенерация [146] является регуляризацией, направленной на то, чтобы избежать переобучения. В этом случае тестовые точки $\{s_j,t_j\}_{j=1}^M$ выбираются в области $0 < s < t < 1$.

Полученная нейронная сеть используется в качестве приближенного решения уравнения (79), а именно

$$\theta_1(x) = \hat{y}(x^2\delta,\delta,\{c_i,\mathbf{a}_i\}_{i=1}^n). \tag{88}$$

4.3.4 Анализ полученных моделей PBA-NN

Далее рассматриваются и обсуждаются результаты, полученные для различных n от 3 до 20. Таблица 3 показывает, что вместе с увеличением числа нейронов точность неявного решения уравнения (84) увеличивается, как и ожидалось.

Одновременно снижается точность исходного решения задачи. Очевидно, это связано с большой погрешностью метода Эйлера, на основе которого строится решение. Кроме того, сеть с $n = 20$ нейронами не имеет существенных преимуществ в точности соответствующего решения неявного уравнения. Это связано с тем, что обучение всех сетей занимает одинаковое количество эпох (2000), чтобы избежать смещения при сравнении, а для обучения сети с 20 нейронами скрытого слоя требуется более длительное обучение.

Таблица 3 – Результаты решения неявного уравнения (84). Сравнение среднеквадратичной ошибки (MSE) и максимального значения абсолютной ошибки для нейросетевого решения уравнения (84) с базисными функциями (86) и соответствующим решением задачи (79) для различного числа n нейронов скрытого слоя.

n	MSE, (84)	max|Error|, (84)	MSE, (79)	max|Error|, (79)
3	0.0259	0.146	0.0617	0.106
5	0.00354	0.0238	0.0900	0.168
10	0.00189	0.0117	0.0880	0.172
20	0.00223	0.0100	0.0875	0.176

Ошибки нейросетевых решений для задач (84) и (79) в случае $n = 3$ и $n = 10$ представлены в виде графиков на Рисунках 21 и 22. Графики на рисунке 21 демонстрируют, что сеть с $n = 3$ нейронами скрытого слоя дает гораздо более низкое качество приближения к точному решению неявного уравнения, поскольку ошибка значительно отклоняется от 0 в большей части входной области. Сеть с $n = 10$ дает большую точность и имеет большие отклонения от 0 только для малых s и больших t. Обратите внимание, что в этом случае поверхность принятия решений имеет более извилистый характер по сравнению с поверхностью принятия решений сети с $n = 3$ нейронами скрытого слоя.

На Рисунке 22 показано, что максимальная ошибка достигается в левом конце интервала параметров и при среднем значении δ. В то же время, ошибка для сети с $n = 10$ нейронами несколько выше. Это вызвано неточностями, вносимыми формулой (82).

Для фиксированных значений параметра δ графики решений типа (88) и

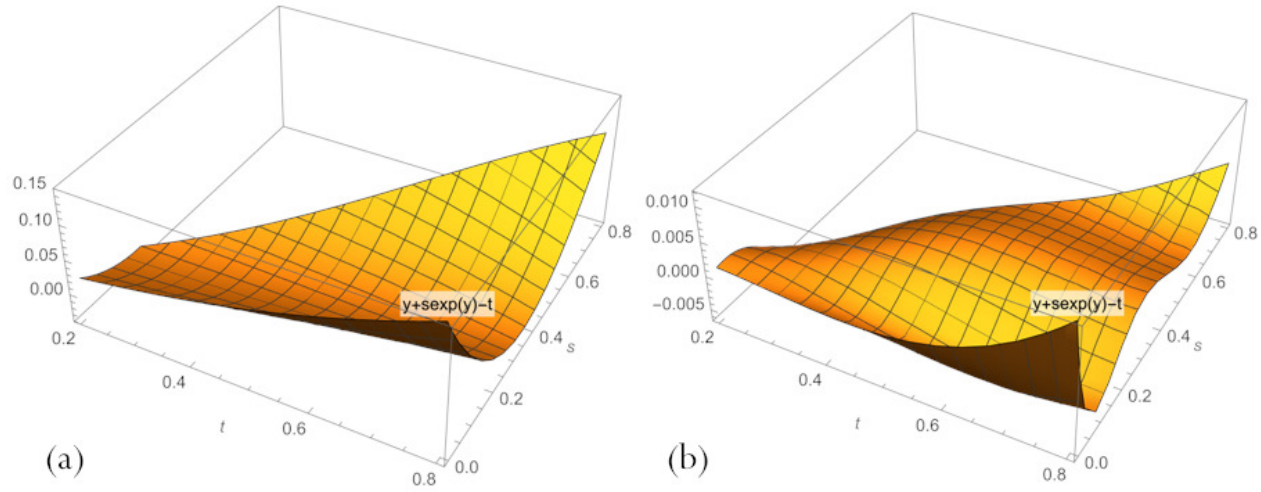

Рисунок 21 – Ошибки решений (85), полученных аналитической модификацией метода Эйлера, уравнения (84), в области $0 < s < t < 1$, для $n = 3$ (**a**) и $n = 10$ (**b**)

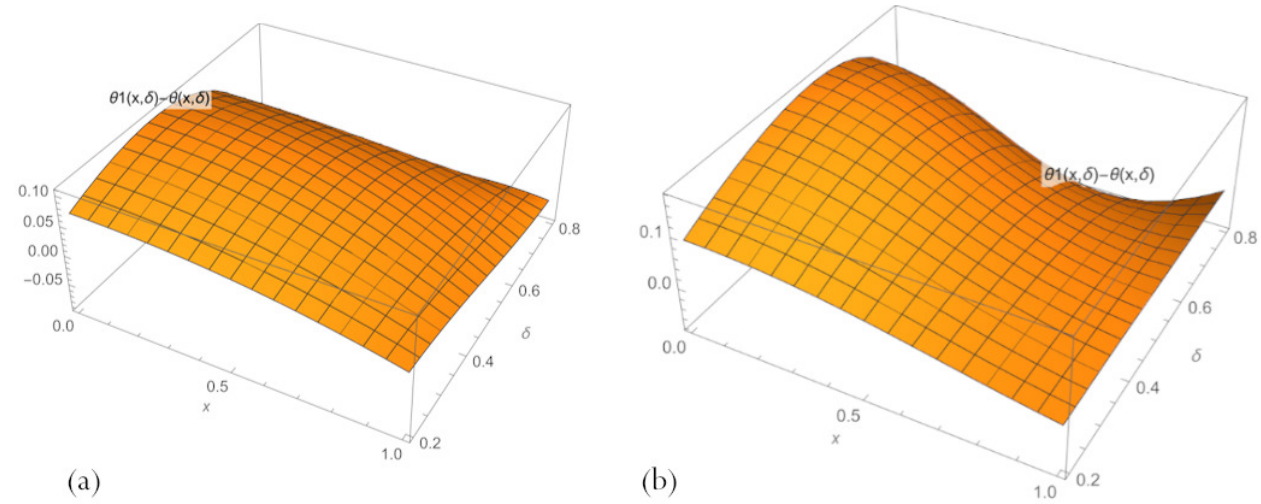

Рисунок 22 – Ошибки решений (88) со встроенной нейронной сетью (85) для задачи (79), в области $0.2 < \delta < 0.8$, для (**a**) $n = 3$ и (**b**) $n = 10$

точного решения задачи (79) изображены на Рисунке 23. Графики, представленные на Рисунке 22, показывают, что, несмотря на значительную ошибку, решения РВА со встроенной нейронной сетью соответствуют общему характеру точного решения, что позволяет получать более точные решения путем дальнейшего обучения.

Сравнение Рисунков 23 и 24, где представлены те же графики для решения РВА с $n = 10$, показывает, что решение (88) с $n = 3$ нейронами скрытого слоя встроенной сети не имеет такого преимущества перед вариантом с $n = 10$, как может показаться из Таблицы 3.

Более того, сравнение точности решений, представленных в этом подразделе, является довольно условным, поскольку эти решения являются лишь качественно верной предварительной аппроксимацией для дальнейшего высокоточного классического обучения PINN для эффективной доработки низкоточных исходных решений PBA со встроенной нейронной сетью.

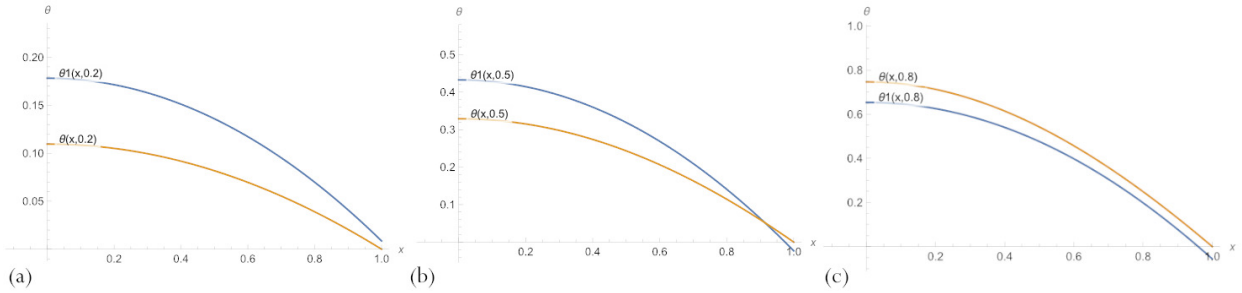

Рисунок 23 – Сравнение PBA решений (88) со встроенной нейронной сетью, $\theta_1(x,\delta)$, $n=3$ нейрона, и точного решения $\theta(x,\delta)$ (79) для значений параметра: (**a**) $\delta=0.2$; (**b**) $\delta=0.5$; (**c**) $\delta=0.8$

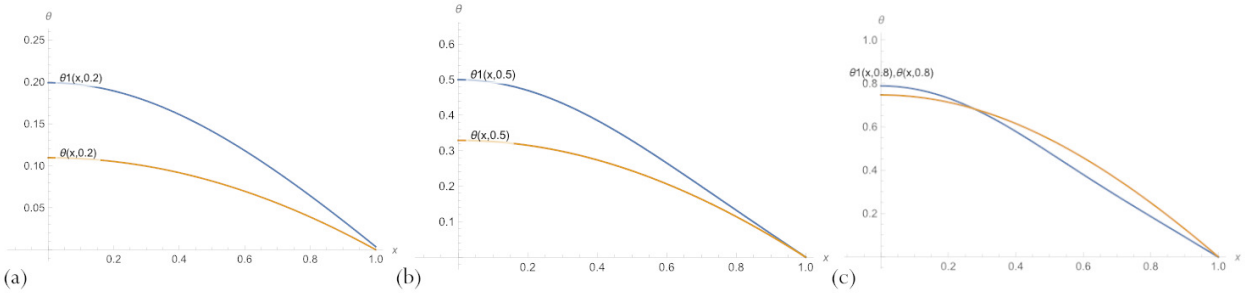

Рисунок 24 – Сравнение решений (88) со встроенной нейронной сетью, $\theta_1(x,\delta)$, $n=10$ нейронов, и точного решения $\theta(x,\delta)$ (79) для значений параметра: (**a**) $\delta=0.2$; (**b**) $\delta=0.5$; (**c**) $\delta=0.8$

4.3.5 Улучшение PBA модели со встроенной нейронной сетью

На данном этапе дорабатываются исходные решения PBA низкой точности со встроенной нейронной сетью, построенные в предыдущем параграфе. Параметры $\{c_i,\mathbf{a}_i\}_{i=1}^n$ параметрического семейства нейронных сетей

$$\theta_1(x)=y(x^2\delta,\delta,\{c_i,\mathbf{a}_i\}_{i=1}^3)$$

подбираются в ходе минимизации функции потерь

$$J_1 = \sum_{j=1}^{m} \left[\left(\theta_1''(x_j) + \delta_j \exp(\theta_1(x_j)) \right)^2 + \lambda \left((\theta_1'(0))^2 + (\theta_1(1))^2 \right) \right]. \qquad (89)$$

Тестовые точки $\{x_j, \delta_j\}$ перегенерируются после 3–5 шагов алгоритма нелинейной оптимизации функционала ошибки (89) внутри области $[0, 1] \times [0.2, 0.8]$.

Результаты вычислительных экспериментов, представленные в Таблице 4, демонстрируют, что нейронные сети, предложенные в этой главе, обладают значительным потенциалом для совершенствования за счет минимизации потерь (89), соответсвующей исходной формулировке задачи (79). Для сравнения представлены результаты классических нейросетевых моделей с функциями активации вида (86).

Таблица 4 – Результаты обучения PBA-PINN. Сравнение среднеквадратичной ошибки (MSE) и максимального значения абсолютной ошибки для PBA-PINN (88) с n нейронами скрытого слоя во встроенной сети и классического PINN с n нейронами скрытого слоя и функцией активации (86) после обучения сетей путем минимизации потерь (89).

n	MSE, (88)	max|Error|, (88)	MSE, PINN	max|Error|, PINN
3	0.0192	0.0947	0.120	0.393
5	0.0118	0.0678	0.0448	0.149
10	0.0113	0.0706	0.0374	0.123
20	–	–	0.0269	0.0792

4.3.6 Дополнительные возможности встраивания нейросетевых конструкций в решения, полученные аналитической модификацией численных методов

В качестве дальнейшего развития принципа встраивания нейросетевых конструкций в полученные на предыдущих этапах решения равенство θ_0 было заменено функцией нейронной сети с одним скрытым слоем

$$\theta_0(\delta, \{\mathbf{b}_i\}_{i=1}^N) = \sum_{i=2}^{N} c_i th(b_i^1 \delta + b_i^2). \tag{90}$$

В результате была получена нейронная сеть с двумя скрытыми слоями, а именно

$$\theta_1(x) = \hat{y}\big(x^2\delta, \theta_0(\delta, \{\mathbf{b}_i\}_{i=1}^N), \{c_i, \mathbf{a}_i\}_{i=1}^n\big). \tag{91}$$

где веса $\{\mathbf{a}_i\}_{i=1}^N$, $\{\mathbf{b}_i\}_{i=1}^N$ подбираются в ходе минимизации функции потерь (89).

Сравнивая значения ошибок в Таблицах 4 и 5, можно сделать вывод, что двухслойная сеть обладает значительным преимуществом.

Таблица 5 – Результаты обучения PBA-PINN. Среднеквадратичная ошибка (MSE) и максимум абсолютной ошибки для решений PBA-PINN (91) с N нейронами первого скрытого слоя и n нейронами второго скрытого слоя после обучения минимизацией функции потерь (89).

Число нейронов	**MSE**	**max|Error|**
$n = 3$, $N = 1$	0.00865	0.0527
$n = 10$, $N = 3$	0.00593	0.0437

Ошибка для различных PINN решений уравнения (79) во всей рассматриваемой области изменения параметра $0.2 < \delta < 0.8$, полученных путем минимизации функции потерь (89), представлены на Рисунке 25. Из этих графиков становится ясно, что ни одно из решений PINN не имеет большого преимущества перед другими. Уточненное решение PBA PINN (88) с $n = 10$ нейронами имеет несколько меньшую ошибку, чем решение с $n = 3$, но обладает чрезмерными колебаниями в области больших значений параметра δ. PBA PINN (91) с $n = 3$, $N = 1$ имеет наименьшую ошибку и соответствует характеру точного решения. Классическая PINN с $n = 20$ нейронами скрытого слоя и функцией активации (86) имеет наибольшую погрешность при максимальной амплитуде колебаний.

Сравним результаты для фиксированных значений параметра δ, наглядно представив их на графиках. На Рисунке 26 показано, что для малых значений δ решения PBA-PINN (88) с $n = 10$ нейронами и PBA-PINN (91) с $n = 3$, $N = 1$

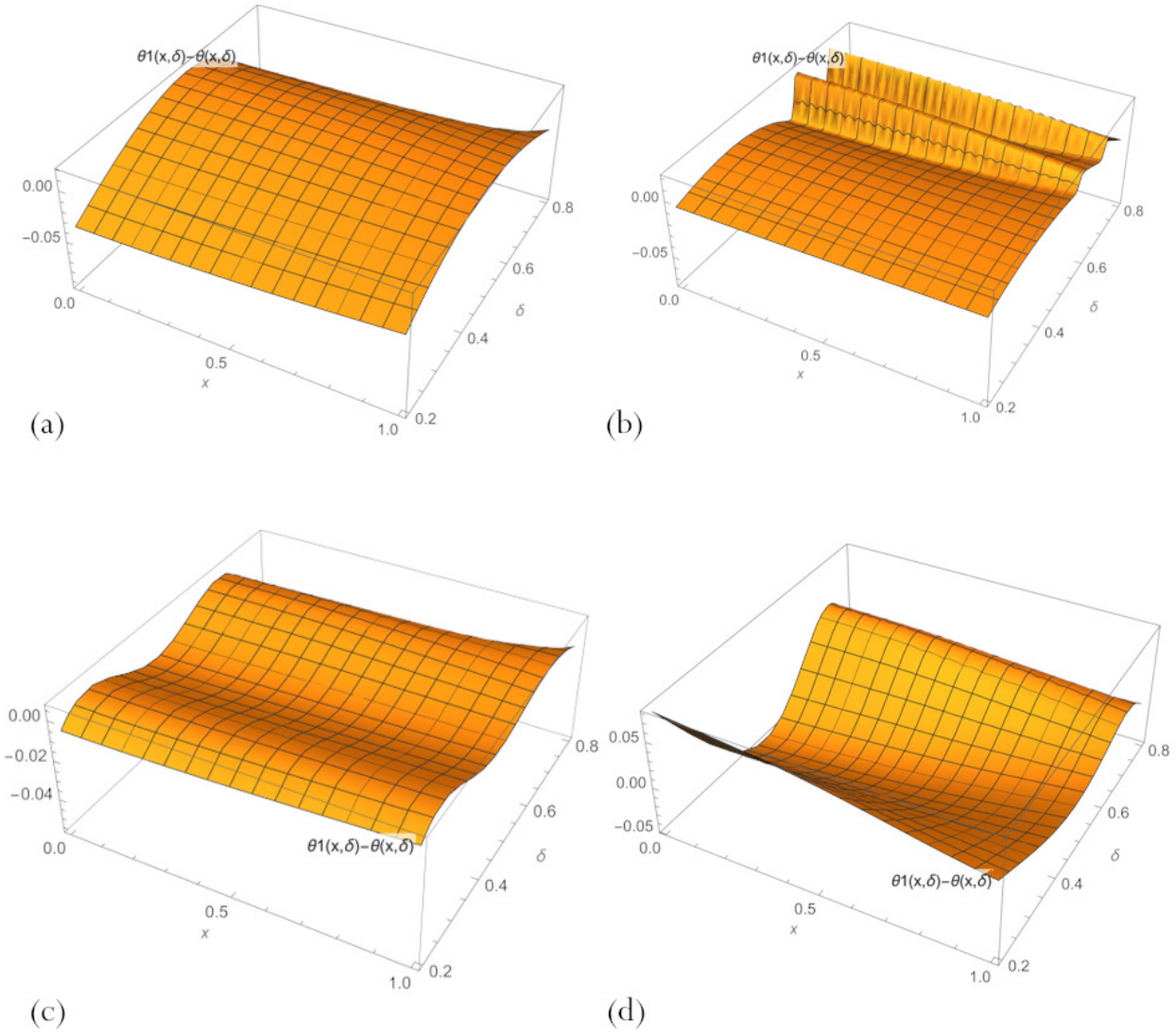

Рисунок 25 – Ошибки решений PBA-PINN (88) при (**a**) $n = 3$ и (**b**) $n = 10$, (**c**) решения PBA-PINN (91) при $n = 3$, $N = 1$, и (**d**) классического решения PINN с $n = 20$ нейронами скрытого слоя и функцией активации (86), в области изменения параметра $0.2 < \delta < 0.8$, после обучения путем минимизации функции потерь (89).

имеют минимальные значения ошибки. При этом классическое решение PINN с $n = 20$ нейронами скрытого слоя имеет самую большую ошибку. Согласно Рисунку 27, в середине промежутка значений параметров решения PBA-PINN (91) с $n = 3$, $N = 1$ имеют минимальную ошибку, классическое решение PINN с $n = 20$ нейронами имеет наибольшую ошибку. В то же время относительные ошибки для всех сетей значительно меньше, чем ошибки для $\delta = 0.2$. На Рисунке 28 показано, что для больших δ классические решения PINN с $n = 20$

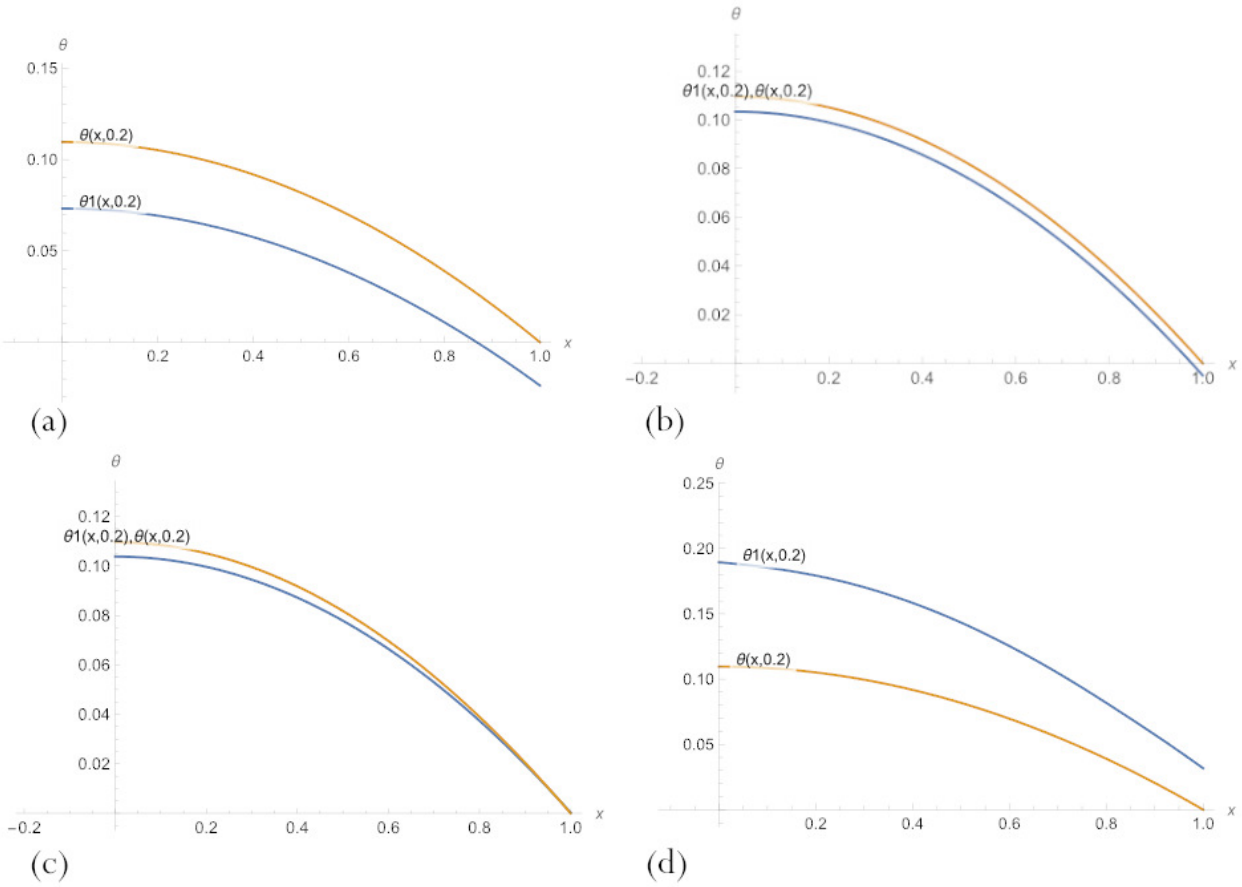

Рисунок 26 – Сравнение решений PBA-PINN (88) при (**a**) $n = 3$ и (**b**) $n = 10$, (**c**) решения PBA-PINN (91) при $n = 3$, $N = 1$, и (**d**) классического решения PINN с $n = 20$ нейронами скрытого слоя и функцией активации (86), после обучения путем минимизации функции потерь (89) и точного решения задачи (79), для значения параметра $\delta = 0.2$

нейронами имеют наименьшую ошибку, а качество других PINN примерно такое же.

Из Рисунков 26–28 видно, что удобнее всего иметь семейство решений PINN и выбирать из них наиболее подходящее в конкретной ситуации.

4.3.7 Уточнение PBA-PINN на основе данных и идентификация параметров

Высокоточные псевдоданные с датчиков используются для эффективного уточнения решений типа PBA-PINN для задачи (79). Компактность PBA-PINN модели (88) с $n = 3$ нейронами скрытого слоя позволяет легко адаптировать его к реальным измерениям. Напомним, что этот веса этой сети уже были настроены

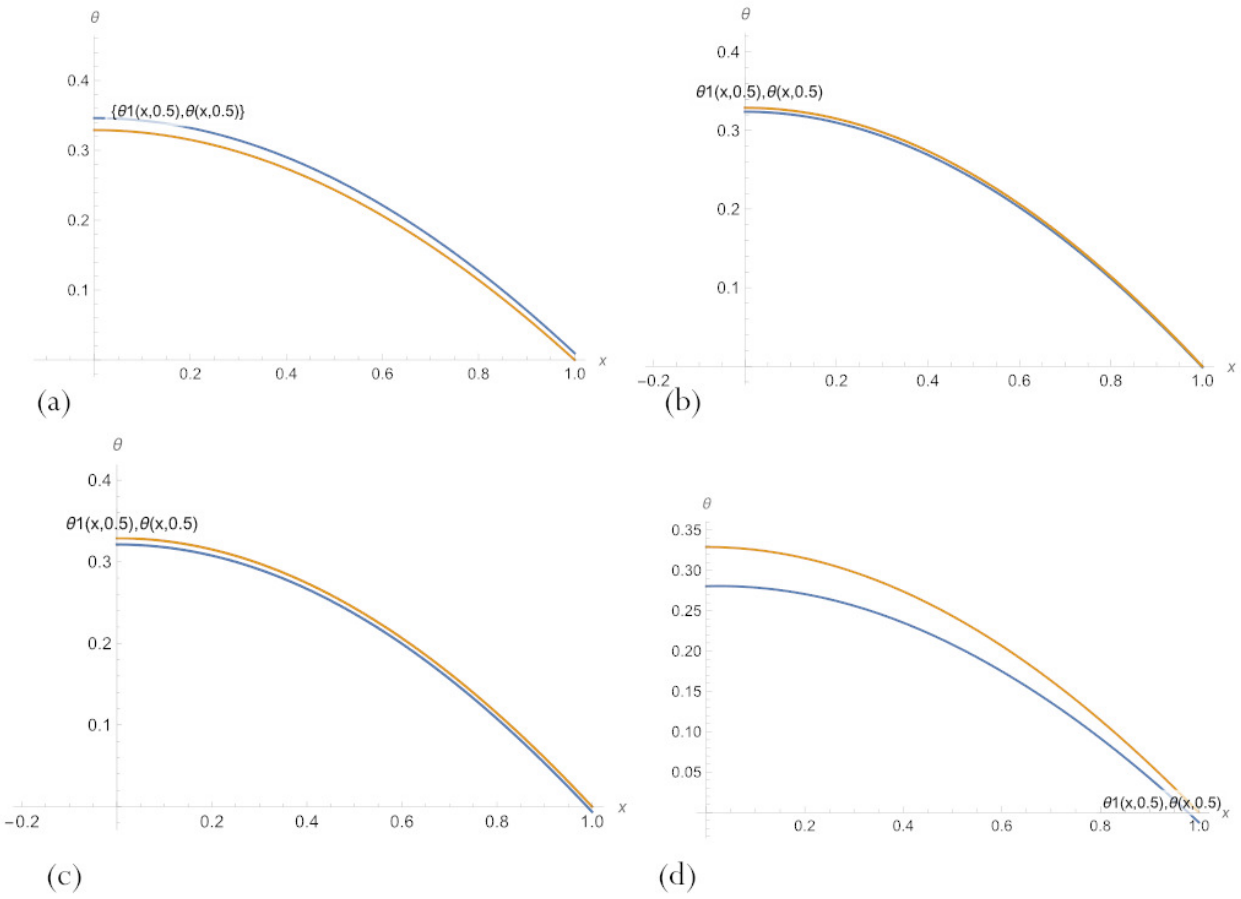

Рисунок 27 – Сравнение решений PBA-PINN (88) при (**a**) $n = 3$ и (**b**) $n = 10$, (**c**) решения PBA-PINN (91) при $n = 3$, $N = 1$, и (**d**) классического решения PINN с $n = 20$ нейронами скрытого слоя и функцией активации (86), после обучения путем минимизации функции потерь (89) и точного решения задачи (79), для значения параметра $\delta = 0.5$

в ходе минимизации функции потерь (89).

4.3.7.1 Параметрические модели PBA-PINN

В первом вычислительном эксперименте в качестве входных данных использовались $M = 50$ случайных (равномерно распределенных) точек $\{x'j, \theta'j\}_{j=1}^{M}$ в *пространственно-параметрической* области $(0, 1) \times (0.2, 0.8)$ и соответствующие искусственные измерения $\theta'j$, сгенерированные с использованием точного решения (79) в качестве выходных данных. Обучающая выборка показана на Рисунке 29. Можно видеть, что случайная выборка используется без намерения равномерно охватить всю интересующую область.

Для обучения используется функция потерь, в которой учитываются эти так называемые данные сенсоров $\{x'j, \delta'j, \theta'j\}_{j=1}^{M}$, а именно

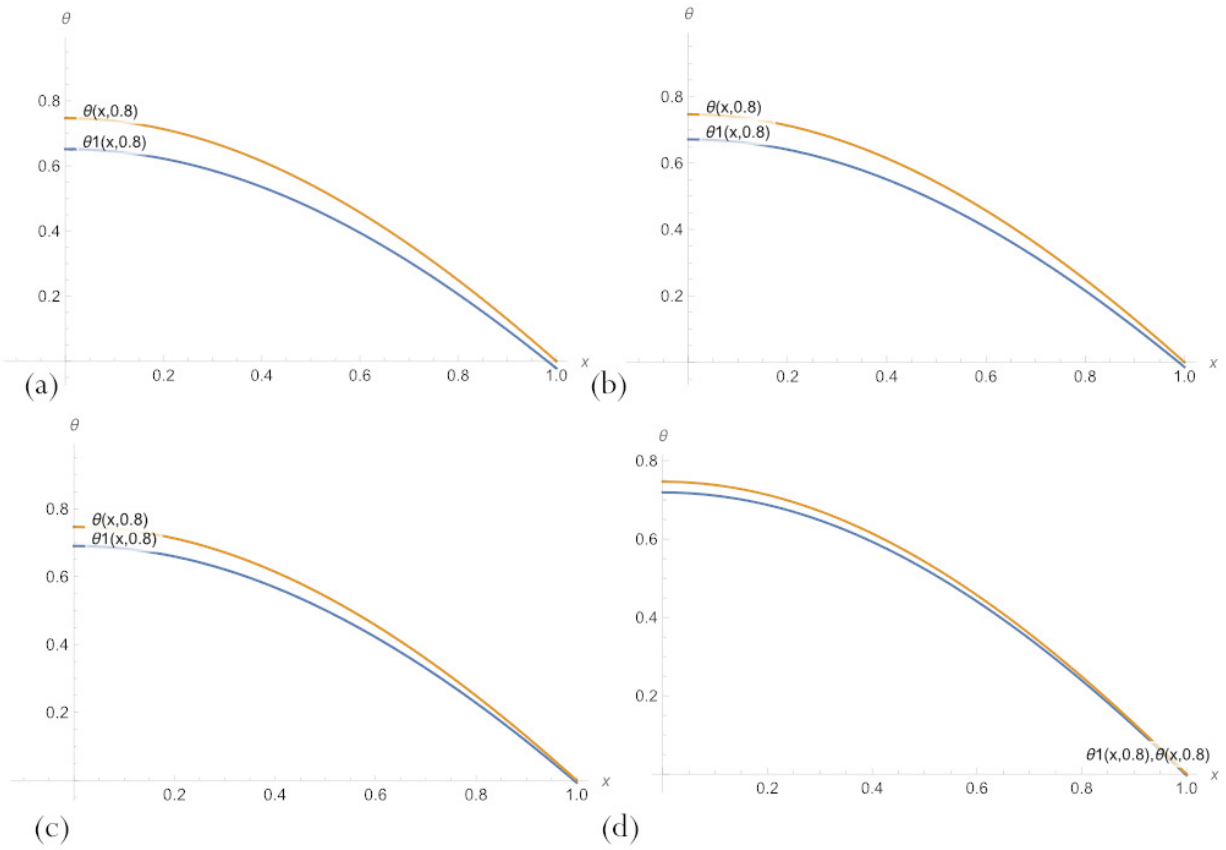

Рисунок 28 – Сравнение решений PBA-PINN (88) при (**a**) $n = 3$ и (**b**) $n = 10$, (**c**) решения PBA-PINN (91) при $n = 3$, $N = 1$, и (**d**) классического решения PINN с $n = 20$ нейронами скрытого слоя и функцией активации (86), после обучения путем минимизации функции потерь (89) и точного решения задачи (79), для значения параметра $\delta = 0.8$

$$J_2' = \sum_{j=1}^{M} \left(\theta_1(x_j', \delta_j') - \theta_j'\right)^2. \tag{92}$$

Рисунок 30 иллюстрирует результаты обучения PBA-PINN (88) минимизацией функции потерь (92). Сравнение их с графиками на Рисунках 26a, 27a и 28a показывает явное улучшение качества решения PBA-PINN, особенно на концах интервала параметров. На Рисунке 31 показаны решения PBA-PINN для фиксированных точек.

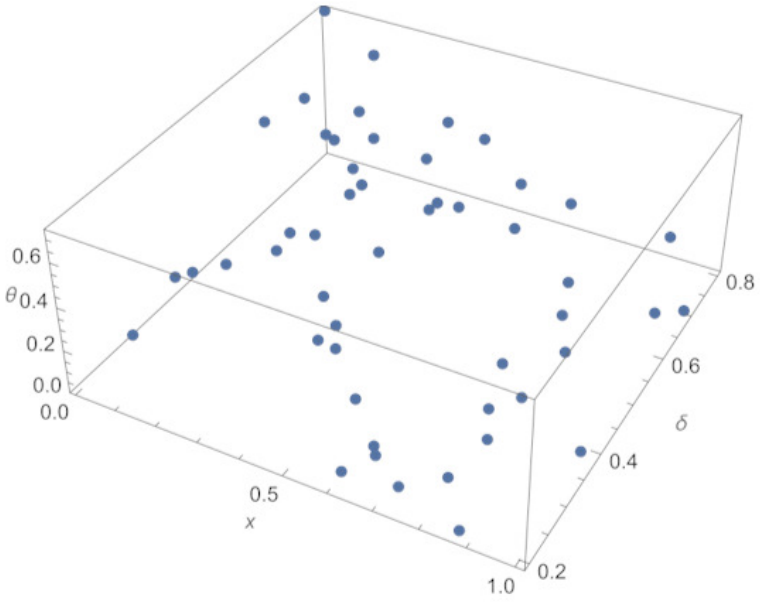

Рисунок 29 – 3D-график обучающей выборки для параметрической улучшенной модели PBA-PINN в интервале изменения параметра $0.2 < \delta < 0.8$

4.3.7.2 Модели PBA-PINN для фиксированного значения параметра

В следующем эксперименте параметр $\delta \in (0.2, 0.8)$ фиксируется и 10 случайных входных данных x_j'' для PBA-PINN (88) берутся на $[0, 1]$. Соответствующие искусственные измерения θ_j'' моделируются, как и ранее.

Для обучения используется функция потерь, учитывающая удовлетворение данным $\{x_j'', \delta, \theta_j''\}_{j=1}^M$, а именно

$$J_2'' \sum_{j=1}^{M} \left(\theta_1(x_j'', \delta) - \theta_j''\right)^2. \tag{93}$$

Результаты вычислительных экспериментов, представленные в Таблице 6, демонстрируют, что ошибки после дополнительного обучения PBA-PINN по данным для всего пространства пространственных параметров в несколько раз меньше ошибок, представленных в Таблице 4.

4.3.7.3 Идентификация параметров

Рассмотрим несколько иную задачу, которая демонстрирует возможности использования параметрической обученной на данных измерений модели PBA-

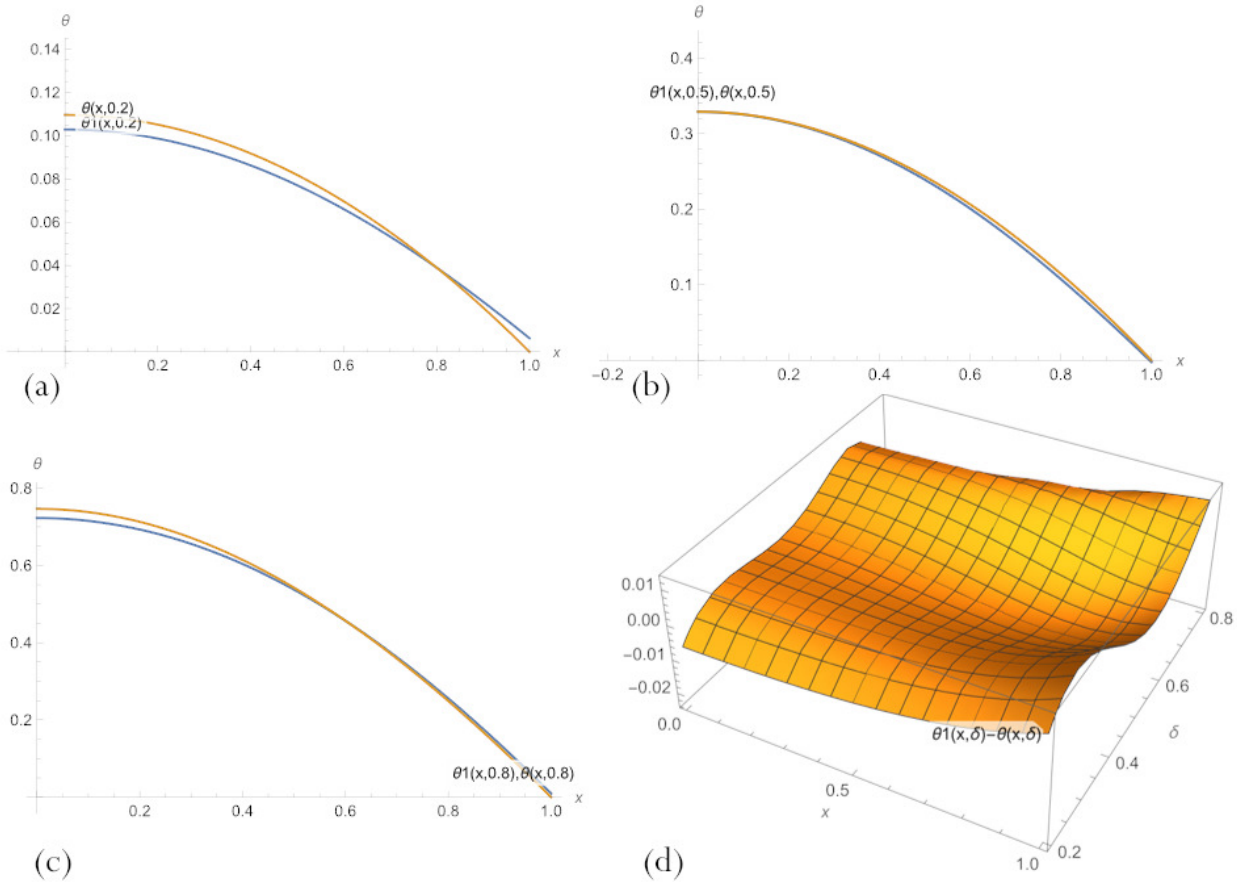

Рисунок 30 – Сравнение параметрического решения PBA-PINN (88), $\theta_1(x, \delta)$, $n = 3$, после двух последовательных обучений путем минимизации функций потерь (89) и (92), соответственно, и точного решения задачи для различных значений параметра: (**a**) $\delta = 0.2$; (**b**) $\delta = 0.5$; (**c**) $\delta = 0.8$; и (**d**) ошибка решения PBA-PINN на все интервале изменения параметра $0.2 < \delta < 0.8$

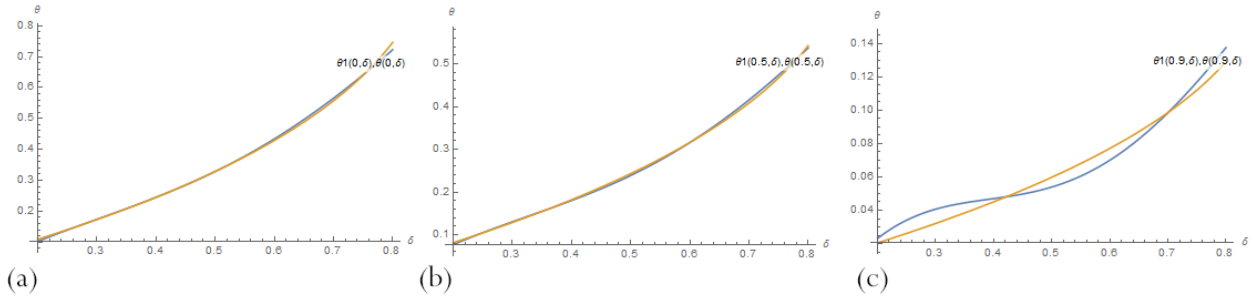

Рисунок 31 – Сравнение параметрического решения PBA-PINN (88), $\theta_1(x, \delta)$, $n = 3$, после двух последовательных обучений путем минимизации функций потерь (89) и (92), соответственно, и точного решения задачи для различных значений переменной x: (**a**) $x = 0$; (**b**) $x = 0.5$; (**c**) $x = 0.9$

Таблица 6 – Среднеквадратичная ошибка (MSE) и максимальное значение абсолютной ошибки для модели PBA-PINN (88) с $n = 3$ нейронами скрытого слоя после двух последовательных обучений путем минимизации функций потерь (89) и (92), соответственно.

$\boldsymbol{\delta}$	**MSE, (79)**	**max|Error|, (79)**
$(0.2, 0.8)$	0.00445	0.0163
0.2	0.0000456	0.000155
0.5	0.000148	0.000272
0.8	0.000798	0.00178

PINN. С этой цели построенная модель применяется для решения обратной задачи, прогнозирования значения параметра δ, которое соответствует определенным данным датчика. Как это было сделано ранее, искусственныее измерения температуры $\{x_j''', \theta_j'''\}_{j=1}^K$ в K случайных точках на интервале $[0, 1]$ моделируются для $\delta = 0.4$ с помощью известного точного решения рассматриваемой задачи. Обучающие точки, использованные в экспериментах, и соответствующие решения PBA-PINN показаны на Рисунках 32a и 33a.

Модель PBA-PINN с $n = 3$, уточненная в результате двух обучений, рассматривается как конечная параметрическая модель $\theta_1(x, \delta)$, удовлетворяющая уравнению (79) в области $(0, 1) \times (0, 2, 0, 8)$. Неизвестный параметр δ подбирается в ходе минимизации функции потерь

$$J_3'' = \sum_{j=1}^{K} \left(\theta_1(x_j''', \delta) - \theta_j'''\right)^2. \tag{94}$$

Значения параметров, предсказанные в результате запроса параметрической модели PBA-PINN, представлены в Таблице 7.

Из Рисунка 33b видно, что использование неточного решения PBA-PINN только с $n = 3$ нейронами и одним измерением с датчиков позволяет прогнозировать достаточно качественное приближенное решение в случае неизвестного параметра δ, что можно сравнить с результатом работы [61], где для решения обратной задачи используется 100 точек измерений.

Таблица 7 – Результаты решения обратной задачи с помощью модели PBA-PINN (88). Оцененное значение параметра δ для различного числа данных измерений и при точном значении $\delta = 0.4$.

Число измерений	**Оценка δ**	**|Error|**
$K = 3$	0.378	0.022
$K = 1$	0.364	0.046

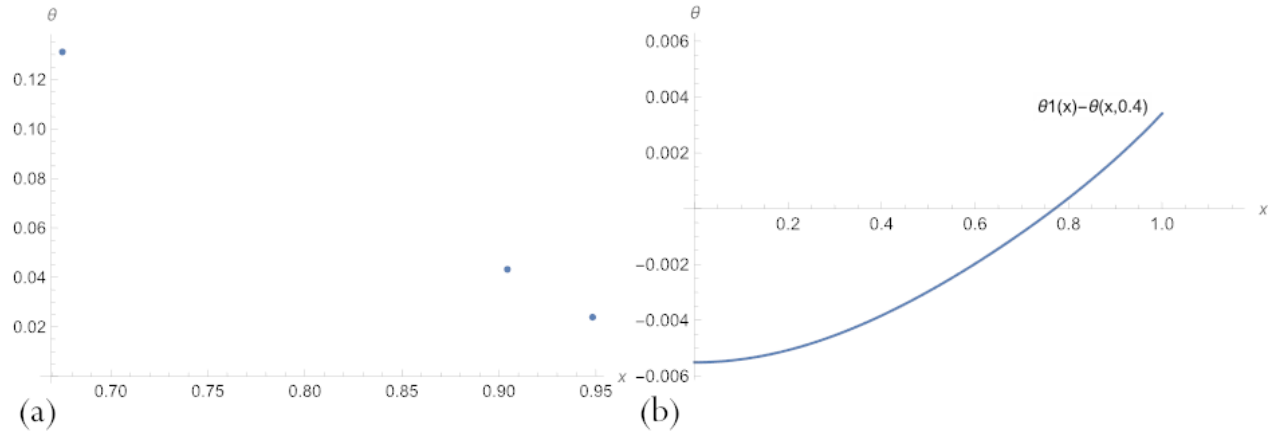

Рисунок 32 – Обратная задача, идентификация параметра. (**a**) Выборка с $K = 3$ искусственными измерениями; (**b**) ошибка соответствующей параметрической модели PBA-PINN (88) для (79) в случае оцененного значения параметра δ

4.4 Обсуждение результатов

В данной главе был предложен новый класс нейронных сетей, основанных на физике, PBA-PINN. Главной особенностью сетей этого типа является то, что не только обучение весам, но и архитектура (структура) самой сети основаны на физике.

Задача обучения нейронной сети путем минимизации функции потерь, кодирующей данные измерений, дифференциальные уравнения и граничные условия, хорошо известна и исследована. Однако вопрос формирования хорошего начального приближения к весам нейронной сети (инициализация), и особенно вопрос выбора архитектуры сети, отвечающей особенностям решаемой задачи, изучен недостаточно. В главе был предложен метод решения этих задач, основанный на использовании дифференциальных уравнений и, соответственно,

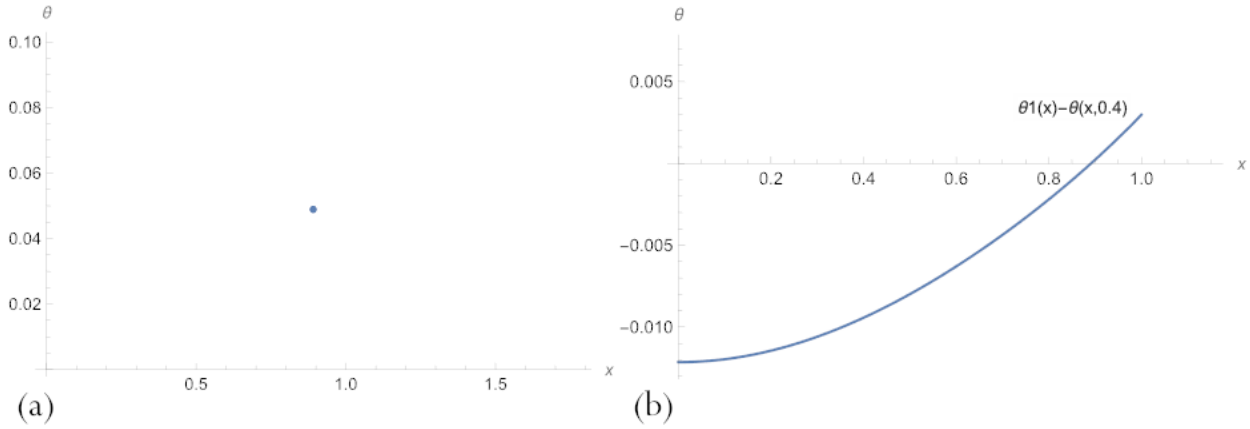

Рисунок 33 – Обратная задача, идентификация параметра. (**a**) Выборка с $K = 1$ искусственными измерениями; (**b**) ошибка соответствующей параметрической модели PBA-PINN (88) для (79) в случае оцененного значения параметра δ

физики процессов, происходящих в моделируемом объекте.

Процесс создания и использования такого рода сетей состоит из следующих трех этапов. На первом этапе, основанном на аналитической модификации классических численных методов, задача построения приближенного нейросетевого решения краевой задачи для дифференциального уравнения сводится к построению приближенного решения с архитектурой, основанной на физике. Чтобы упростить решение уравнения (явного или неявного) на каждой итерации численного метода, это уравнение предлагается решать с помощью нейронной сети. Сетевые веса обучаются обычным способом, основанным на минимизации функции потерь. Важной особенностью сети является то, что параметры задачи могут быть в числе входных данных. При небольшом числе итераций численного метода результирующее решение с архитектурой, основанной на физике, является компактным, но имеет низкую точность. На втором этапе построенная сеть PBA дополнительно обучается решению дифференциального уравнения путем минимизации соответствующей функции потерь. В этом случае сеть обучается не только по набору входных переменных исходной задачи, но и по области изменения параметров. На третьем этапе построенная модель PBA-PINN дополнительно обучается в соответствии с данными, поступающими от датчиков. С помощью полученной высокоточной модели можно решать задачи идентификации параметров, составления уравнений и другие задачи, например, задачу управления.

Эффективность предложенных методов продемонстрирована на примере задачи моделирования процессов в химическом реакторе.

Результаты вычислительных экспериментов показали, что для рассматриваемой задачи предложенный метод позволяет строить очень маленькие нейросетевые модели, основанные на физике, которые отражают моделируемый объект с приемлемой точностью. Недостаточная точность низкоточных моделей, построенных на первом этапе, компенсируется возможностью доработки моделей путем дополнительного высокоточного обучения на втором и третьем этапах. Результаты этого многоточного обучения PBA-PINNs были сопоставлены с результатами общего нейросетевого подхода.

Предложенные PBA-PINN позволили достичь в несколько раз большей точности, чем стандартные полносвязные нейронные сети. Вычислительные эксперименты на основе данных показали, что предложенные параметрические модели с низкой точностью подходят для последующего переобучения и решения задач идентификации параметров на основе данных измерений. В работе [143] та же параметрическая дифференциальная задача была решена путем применения аналитической модификации таких численных методов, как скорректированный метод Эйлера и метод Штермера. Авторы показали, что точность решения повышается с увеличением числа итераций. Отметим, что на основе аналогичной низкоточной модели, с помощью последующего дополнительного обучения, удалось получить параметрическую модель со сравнимой точностью. Более того, при завершении обучения в соответствии с данными для конкретных значений параметров преимущество конечного высокоточного решения было выражено в снижении максимальной погрешности на три порядка. Этот результат сопоставим с решениями, полученными в [105, 163], где были решены аналогичные задачи с фиксированными значениями параметров.

Разработанные PBA-PINN модели целесообразно использовать в задачах суррогатного моделирования и моделирования реальных объектов, когда сложно или нецелесообразно строить достаточно точную физическую модель и, соответственно, математическую модель в виде краевой задачи для дифференциального уравнения (или системы таких уравнений). В этом случае предполагается, что имеются данные датчиков, которые могут повысить точность модели, но которых недостаточно для построения модели без использования дифференциальных уравнений.

Глава 5.

Эволюционные алгоритмы обучения и построения физически-информированных нейросетевых моделей на основе фронта Парето

5.1 Введение

Данная глава посвящена разработке новых эволюционных алгоритмов обучения нейронных сетей, основанных на физике. В связи с тем, что эволюционные алгоритмы обычно ресурсоемки, представляют интерес подходы к их ускорению. Представленные алгоритмы направлены на решение проблем, связанных с некорректно поставленными задачами, путем построения совокупности, близкой к фронту Парето. Алгоритм Парето используется в рамках разработанной методики как механизм, который вводит фундаментальную концепцию в эволюционный алгоритм. Проведено сравнение возможностей алгоритма при использовании трех различных критериев качества решений. Для оценки эффективности алгоритмов были использованы две контрольные задачи в некорректной постановке, с разрывными граничными условиями и при их отсутствии. Кроме того, изучено влияние гиперпараметров на конечные результаты. Были проведены сравнения между предложенными алгоритмами и стандартными алгоритмами общего нейросетевого подхода. Результаты демонстрируют преимущество предложенных алгоритмов в достижении более высокой производительности при решении некорректно поставленных задач. Кроме того, предлагаемые алгоритмы обладают способностью находить решения с требуемой гладкостью. Представленный подход и результаты его применения были опубликованы в журнале Computation (Q2).

Для анализа представленных результатов важно подчеркнуть, что в этой

главе, как и во всей диссертации, рассматриваются методы в рамках новой парадигмы математического моделирования. Классическая парадигма предполагает полное соответствие между дифференциальным уравнением и исходным объектом, и если результаты неудовлетворительны, процесс должен начинаться сначала. В новой парадигме моделирования модель дифференциального уравнения с дополнительными условиями по своей сути рассматривается как приближенная. В результате стремление к очень высокой точности при решении дифференциальной задачи может оказаться бессмысленным. Таким образом, естественно иметь набор нейросетевых моделей, основанных на физике (PINN), для решения дифференциальных задач с различным уровнем точности. При этом важно получить достаточно точное решение, когда дифференциальная задача соответствует объекту, требующему высокой точности. Адаптивные свойства модели PINN, такие как ее способность уточняться на основе дополнительных данных (например, измерений с датчиков, изменений параметров уравнения), являются еще более важными. На данном этапе скромная точность дифференциальной модели не является окончательной оценкой ее качества, поскольку она отражает качественное поведение решения. Дифференциальная задача должна точно отражать качественное поведение решения, а модель должна допускать уточнение на основе дополнительных данных.

5.1.1 Эволюционные алгоритмы на основе фронта Парето: общая идея

Рассмотрим в общем виде задачу, сформулированную в параграфе 1.3.2.

$$\mathscr{D}[u(\mathbf{x})] = 0,\ \mathbf{x} \in \Omega. \tag{95}$$

Помимо основного уравнения задача может включать граничные условия вида

$$\mathscr{B}[u(\mathbf{x})] = b(\mathbf{x}),\ \mathbf{x} \in \bar{\Omega}, \tag{96}$$

и (или) начальные условия

$$\mathscr{C}[u(\mathbf{x})] = c(\mathbf{x}),\ \mathbf{x} \in \bar{\Omega} \cup \Omega, \tag{97}$$

а также данные измерений

$$\mathscr{M}[u(\mathbf{x_i})] = m(\mathbf{x}_i),\ \mathbf{x}_i \in \bar{\Omega}, \tag{98}$$

и другие условия. Здесь, $\mathscr{D}[\cdot]$, $\mathscr{B}[\cdot]$, $\mathscr{C}[\cdot]$, $\mathscr{M}[\cdot]$ – некоторые дифференциальные операторы , $\mathbf{x}$ – пространственно-временная переменная, Ω – область решения с границей $\bar{\Omega}$, и $b(\mathbf{x}), c(\mathbf{x}), m(\mathbf{x})$ – подходящие функции.

С помощью унифицированного нейросетевого подхода все условия кодируются в виде функций потерь с дискретным представлением вида

$$L_D(\hat{u}(\mathbf{x}, \mathbf{a}), \mathbf{X}_{N_D}) = \frac{1}{N_D} \sum_{i=1}^{N_D} (\mathscr{D}[\hat{u}(\mathbf{x}_i, \mathbf{a})])^2; \tag{99}$$

$$L_B(\hat{u}(\mathbf{x}, \mathbf{a}), \mathbf{X}_{N_B}) = \frac{1}{N_B} \sum_{j=1}^{N_B} (\mathscr{B}[\hat{u}(\mathbf{x}_j, \mathbf{a})] - b(\mathbf{x}_j))^2; \tag{100}$$

$$L_C(\hat{u}(\mathbf{x}, \mathbf{a}), \mathbf{X}_{N_C}) = \frac{1}{N_C} \sum_{k=1}^{N_C} (\mathscr{C}[\hat{u}(\mathbf{x}_k, \mathbf{a})] - c(\mathbf{x}_k))^2; \tag{101}$$

$$L_M(\hat{u}(\mathbf{x}, \mathbf{a}), \mathbf{X}_{N_M}) = \frac{1}{N_M} \sum_{l=1}^{N_M} (\mathscr{M}[\hat{u}(\mathbf{x}_l, \mathbf{a})] - m(\mathbf{x_i}))^2. \tag{102}$$

Как уже было отмечено, данная задача является многокритериальной. В данной главе предлагается общая схема эволюционных алгоритмов для обучения физически-информированных нейронных сетей на базисе метода Парето. Предлагаемый подход включает в себя использование схем мутации, скрещивания и отбора особей на основе определенных критериев. В качестве критериев могут использоваться функции потерь (99)-(102), встречающиеся в общем нейросетевом подходе, а также экспертные знания или любая информация, которая становится доступной в режиме реального времени.

Чтобы учесть многоцелевой характер задачи оптимизации, изменяется штрафной множитель в соответствующей функции потерь. Обучая отдельные решения для разных значений штрафного множителя, генерируется аналог фронта Парето, после чего наилучшие варианты выбираются на основе выбранного критерия. Чтобы реализовать принцип выживания наиболее приспособленных особей из популяции, предлагаемый эволюционный алгоритм изби-

рательно выбирает особей из исходной популяции для мутации.

Чтобы проиллюстрировать лежащую в основе предлагаемых эволюционных алгоритмов концепцию, обозначим в качестве $L_1, L_2, \ldots$ различные критерии для принятия решений, такие как функции потерь, логические индикаторы или специально разработанные функции [89].

На Рисунке 34 показана процедура мутации, используемая в предлагаемом семействе эволюционных алгоритмов, называемая мутацией Парето.

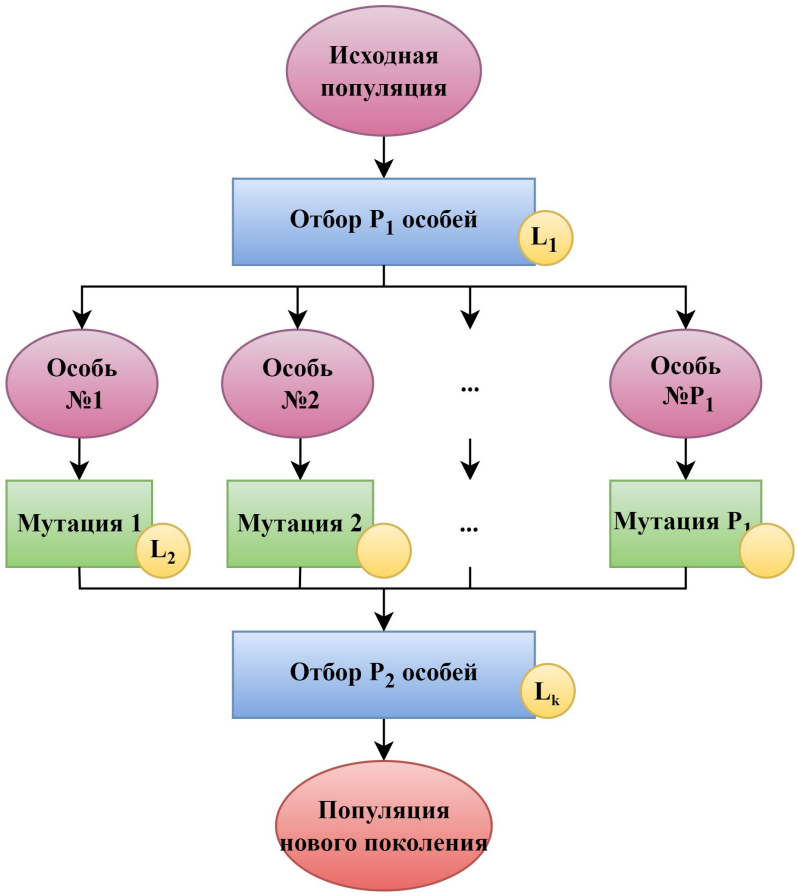

Рисунок 34 – Эволюционный процесс направленной мутации и отбор по Парето. На диаграмме показано построение аналога фронта решений по Парето, а также выбор нового поколения решений в ходе эволюционных процессов. Желтыми кружками обозначены области, где при оценке отдельных лиц учитываются конкретные критерии для решения задачи. Обучение PINN рассматривается как Мутация 1–Мутация P_1

Далее направленная мутация и отбор по Парето включаются в общую структуру эволюционного алгоритма. Стоит отметить, что конкретные критерии используются как на различных этапах общего алгоритма, как показано на Рисунке 35, так и в рамках мутации по Парето. Точки вариативности обозна-

чены желтыми кружками. Пунктирная линия представляет собой возможное расширение эволюционной схемы. Общая структура схемы соответствует стандартному шаблону, используемому в эволюционных алгоритмах. Например, в качестве примера можно привести схему процесса оптимизации генетического алгоритма для искусственных нейронных сетей, представленную в [?].

Рисунок 35 – Эволюционный алгоритм, используемый для генерации совокупности PINN-решений. Желтые кружки обозначают области, в которых при оценке отдельных пользователей учитываются конкретные критерии для искомого решения задачи. «Скрещивание» относится к процессу введения дополнительных нейронов в текущее поколение PINNs

На основе общей схемы для семейства эволюционных алгоритмов, основанную на критериях, описанных в условиях (95)-(98), различные алгоритмы могут быть разработаны с использованием как критериев (99)-(102), так и лю-

бых других. Свойства решений, генерируемых с помощью этих алгоритмов, и влияние гиперпараметров нейронной сети на эти решения являются перспективными областями исследований. В этой главе в качестве применения представленного алгоритма рассматриваются некорректные задачи, которые не могут быть решены с использованием общего нейросетевого подхода.

5.2 Вычислительные эксперименты и результаты

5.2.1 Решение плохо поставленных задач

Для оценки эффективности алгоритмов были использованы две контрольные задачи. Их можно отнести к классу задач построения математических моделей на основе разнородных данных, что является актуальной задачей современной науки [36, 64, 158, 35]. Из-за особых характеристик используемых алгоритмов необходимо, чтобы дифференциальное уравнение и оператор, определяющий граничные условия, были линейными. В данном исследовании в качестве дифференциального уравнения было выбрано классическое уравнение Лапласа, поскольку оно часто используется для тестирования методов, предназначенных для решения краевых задач. Расширение алгоритмов для работы с другими линейными уравнениями, как правило, не создает существенных проблем.

При моделировании реальных объектов часто возникают трудности, связанные со сложной природой моделируемого объекта. Одной из таких проблем является возможность получения противоречивых математических моделей при описании различных аспектов объекта. Чтобы проиллюстрировать это, мы приводим базовый пример, в котором решение в углах квадрата должно быть равным как 0, так и 1 одновременно. Однако любое непрерывное решение не может удовлетворять этому условию, что приводит к значительным ошибкам в окрестности этих точек. Тем не менее, важно убедиться, что ошибки в остальной части области невелики и что построенное решение согласуется с решением, полученным с использованием принципиально иного подхода (в данном случае классического метода Фурье). Важно подчеркнуть, что угловые особенности для начально-краевых задач недавно стали предметом исследовательского интереса [149, 102]. Другой проблемой при моделировании реальных объектов является возможность создания неполных математических моделей. В таких случаях дополнительная информация может быть получена путем наблюдений

за объектом. В качестве задачи для применения разработанных эволюционных алгоритмов рассматривается второй пример, который подчеркивает эту проблему. В частности, задача состоит в решении уравнения без достаточных данных о граничных условиях, что делает невозможным получение правильного решения стандартными методами. Вместо этого доступны данные «измерений», что не делает задачу корректной. Вычислительный эксперимент продемонстрировал, что предложенный подход успешно решает эту задачу. Оба примера включают решение уравнения Лапласа в замкнутой области. Эта задача широко используется в качестве эталона для оценки производительности многоцелевых эволюционных алгоритмов [22].

5.2.2 Формулировка модельных задач

Рассмотрим специальные модельные задачи в некорректной постановке, а именно, пусть интересующий объект описывается уравнением Лапласа

$$\Delta u(\mathbf{x}) = 0,\ \mathbf{x} \in [0,1] \times [0,1] \tag{103}$$

с разрывными граничными условия Дирихле

$$u(x,0) = u(0,y) = 0,\ u(x,1) = u(1,y) = 1,\ x,y \in (0,1); \tag{104}$$

и вторая формулировка – то же самое уравнение Лапласа без граничных или начальных условий, но имеются данные измерений об объекте вида

$$u(\mathbf{x}_i) = z_i,\ \mathbf{x}_i \in [0,1] \times [0,1],\ i \in 1,\ldots,M; \tag{105}$$

которые в данном случае будут сгенерированы искусственным образом.

Рассматриваемые задачи уже решались другими методами [17, 146, 89], что позволяет провести сравнительный анализ результатов после применения предложенного в данной главе эволюционного алгоритма.

При решении задач (103)+(104) и (103)+(105) с использованием общего нейросетевого подхода [16] оптимально выбрать нейронную сеть прямого распространения с одним скрытым слоем и радиально-базисными функциями активации. Это связано с замкнутой областью, в которой вычисляется оператор Лапласа, значения за пределами которой знать не требуется, в то время как многослойный персептрон подразумевает необходимость определения значений

за пределами этих границ. Радиальные базисные функции хорошо подходят для локальной аппроксимации в небольшой окрестности каждой точки, что делает их идеальным выбором для этих задач.

Следует отметить, что во всех экспериментах на границах используются фиксированные точки коллокации, а в пределах рассматриваемой области используются случайно распределенные точки с перегенерацией после фиксированного количества эпох обучения нейронной сети.

5.2.3 Описание алгоритма

Недостатком классического решения задачи (103)+(104) с использованием метода Фурье является эффект Гиббса на границе. Наибольшее расхождение между решением Фурье и условием задачи возникает при вычислении среднеквадратичной погрешности производной функции – погрешности равенства нулю производной в пробных точках на верхней границе квадрата. При вычислении по 1000 случайно выбранным точкам его приблизительная величина равна 40. Таким образом, естественно условие постоянства функции и равенство производной функции нулю на границе рассматривать как дополнительные критерии оценки решения.

Таким образом, применение описанного алгоритма приводит к построению наиболее гладких решений на границе области.

В этом подразделе описываются конкретные параметры алгоритмов и гиперпараметры нейронных сетей, используемые в вычислительных экспериментах. Используется схема для всего семейства алгоритмов, представленная на Рисунке 35, а также схема для мутации по Парето, изображенная на рисунке 34.

1. Критерий останова:

   Эволюционный процесс завершается после достижения заданного числа нейронов, N, в модели PINN. Различные значения N рассматриваются для исследования влияния общего числа нейронов на полученные решения и их исходы.

2. Начальная популяция моделей

   В качестве исходной популяции рассматривается популяция $n_1 = 100$ нейронов с РБФ-функцией активации вида

$$\hat{u}(\mathbf{x}, \mathbf{a}) = a_1 + a_2 \exp\big(- a_3((x - a_4)^2 + (y - a_5)^2)\big), \qquad (106)$$

которые случайным образом инициализируется. Здесь $\mathbf{x} = (x, y)$. Впоследствии все особи вводятся во входные данные процесса мутации по Парето. Значение $n_1 = 100$ довольно мало, но обеспечивает стабильное значение глобального штрафного множителя (107) для мутации по Парето.

3. Мутация Парето:

   Для отбора особей из входящей популяции используется следующая процедура. Во время начальной мутации дискретные квадратичные ошибки, соответствующие уравнению Лапласа $L_D(\hat{u}(\mathbf{x}, \mathbf{a}), \mathbf{X}_{N_D})$ и граничным условиям $L_B(\hat{u}(\mathbf{x}, \mathbf{a}), \mathbf{X}_{N_B})$ вычисляются для каждой инициализированной сети. Затем эти ошибки суммируются для получения «глобального» штрафного множителя

$$\delta = \frac{L_D(\hat{u}(\mathbf{x}, \mathbf{a}), \mathbf{X}_{N_D})}{L_B(\hat{u}(\mathbf{x}, \mathbf{a}), \mathbf{X}_{N_B})\sqrt{P_1}}, \qquad (107)$$

   для линеаризованной функции потерь $L_D + \delta L_B$, где P_1 – количество различных мутаций (см. рисунок 34). Затем для каждой последующей мутации i, $i = 1, \ldots, P_1$, выбирается особь, имеющая наименьшие потери $L_D + i\delta L_B$.

   В процессе мутации текущие сети обучаются независимо друг от друга путем минимизации соответствующих функций потерь с использованием алгоритма Rprop [34] и при перегенерации точек для уравнения Лапласа каждые 5 эпох обучения. Общее количество эпох обучения K_1 может быть выбрано отдельно. Для последней мутации используется общее количество эпох обучения K_2. Значения, рассмотренные в экспериментах, представлены в таблице 8.

   Что касается второй процедуры отбора, описанной в мутации Парето, то все мутировавшие сети сохраняются во всех прогонах ($P_2 = P_1$), за исключением последнего.

   Как упоминалось ранее, точки коллокации используются на границе. Пер-

воначально они делятся на два набора: первый набор используется для обучения сети, в то время как второй набор используется в процессе отбора. За исключением первой мутации, все остальные мутации используют первый набор точек для обучения PINN.

4. Скрещивание:

   В экспериментах используется алгоритм со скрещиванием в рамках текущей популяции. Оптимальный набор Парето для набора решений PINN строится на основе критерия, отражающего квадратичную ошибку удовлетворения уравнения Лапласа $L_D(\hat{u}(\mathbf{x}, \mathbf{a}), \mathbf{X}_{N_D})$ и граничных условий $L_B(\hat{u}(\mathbf{x}, \mathbf{a}), \mathbf{X}_{N_B})$ в дискретном наборе точек. Далее нейроны выбираются из индивидов, расположенных на крайнем левом и крайнем правом концах текущего Парето-оптимального набора, и добавляются один за другим ко всем остальным сетям, умноженным на оптимальный параметр в смысле наименьших квадратов. Из полученных новых индивидов выбирается тот, у которого ошибка минимальна, для каждого значения штрафного множителя $i\delta_{new},\ i = 1, \ldots, P_1$. Схема процедуры скрещивания представлена на Рисунке 36.

5. Примечание о наборе штрафных множителей:

   В предыдущем абзаце мы представили два разных набора штрафных множителей. Общая схема алгоритма допускает новые выборки таких множителей в рамках каждой мутации и скрещивания. В этом исследовании рассматриваются два варианта: в первом случае набор штрафных множителей фиксируется во время инициализации первой совокупности сетей ($\delta_{new} = \delta$), а во втором параметр δ обновляется в процессе скрещивания. В последнем случае новый параметр вычисляется следующим образом:

$$\delta_{new} = \sqrt{\frac{j_1 j_{P_1}}{P_1}}\delta, \tag{108}$$

   где j_1 и j_{P_1} это число индивидов из крайнего левого и крайнего правого концов текущего Парето-оптимального набора.

В Таблице 8 приведены значения параметров описанного алгоритма в разных вариантах. P_1, P_2 – это параметры , которые определяют количество особей

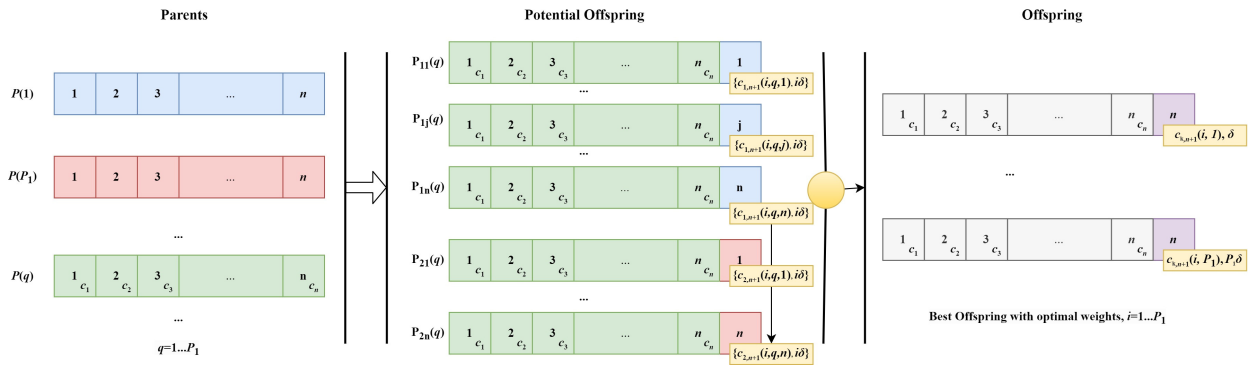

Рисунок 36 – Схема процедуры скрещивания. $P(1)$ и $P(P_1)$ – крайний левый и крайний правый индивидуум текущего Парето-оптимального набора с n нейронами; $P(q)$, $q = 1, \ldots, P_1$, – текущая популяция PINNs с внешним множителем (весом) c_j при каждом нейроне скрытого слоя. $c_{k,n+1}(i, q, j)$ – оптимальные внешние веса нового нейрона в смысле минимального значения ошибки $L_D(\mathbf{X}_{N_D}) + i\delta L_B(\mathbf{X}_{N_B})$. Желтый круг обозначает включение конкретных критериев для задачи в оценку отдельных индивидуумов

Таблица 8 – Конфигурации параметров алгоритма, используемые для серии экспериментов, при которых решается задача (103)+(104).

P_1	P_2	N	K_1	K_2
9	9	10, 20, 30	10, 50, 100	400, 800

в текущей популяции , подвергающихся независимым мутациям; N – это заранее определенное количество нейронов в решении PINN, используемое в качестве условия завершения; K_1 – это общее количество периодов обучения во время мутации Парето, за исключением последнего цикла, а K_2 – общее количество периодов обучения во время последней мутации Парето. Значения параметров были выбраны таким образом, чтобы обеспечить баланс между стабильностью и продолжительностью вычислительных экспериментов. Например, использование $P_1 = 9$ индивидуумов в текущей популяции, подвергшихся независимым мутациям по Парето, дало тот же результат, что и использование $P_1 = 20$, при значительном сокращении времени выполнения. Дальнейшее сокращение потоков привело к ухудшению результатов.

5.2.4 Результаты: Задача (103)+(104)

Метод рестартов был применен к трем вариантам алгоритма, описанным на рисунке 35 и ранее в этой главе, с параметрами, представленными в таблице 8, проведена серия экспериментов. Были использованы различные варианты Алгоритма 1. Алгоритм 1_1 включает обновление набора штрафных коэффициентов, в то время как Алгоритм 1_3 сохраняет значения, полученные на этапе инициализации исходной совокупности. Один из вариантов Алгоритма 1 , Алгоритм 1_2 включает в себя эволюционный механизм обновления весов модели PINN. Во время Парето-мутации некоторые веса заменяются ближайшими элементами текущего парето-оптимального набора.

Algorithm 1 Эволюционный алгоритм обучения PINN на основе приближения к фронту Парето

```
t ← 0
InitPopulation [P(t)]                          ▷ Инициализация популяции P(t)
EvalPopulation1 [P(t)]                         ▷ Оценка популяции P(t) по Критерию 1
Select1 [P(t), P_1]                            ▷ Отбор лучших P_1 индивидов из P(t)
Mutate [P_1(t)]                                ▷ Мутация каждого индивида из P(t)
while Критерий останова do
    EvalPareto [P(t)]          ▷ Оценка популяции P(t) и построение фронта Парето
    (ParentL, ParentR) ← SelectPareto [P(t)]
    for each p(t) in P(t) do
        Crossover (ParentL, p(t))
        Crossover (ParentR, p(t))
    end for
    P(t + 1) ←Select1 [Новое поколение, P_1]
    t ← t + 1
end while
```

Решения, минимизирующие выражения

$$L(\hat{u}(\mathbf{x},\mathbf{a}),\mathbf{X}_{N_D},\mathbf{X}_{N_B}) = \frac{L_D(\hat{u}(\mathbf{x},\mathbf{a}),\mathbf{X}_{N_D})}{\max_{\mathbf{a}} L_D(\hat{u}(\mathbf{x},\mathbf{a}),\mathbf{X}_{N_D})} + \frac{L_B(\hat{u}(\mathbf{x},\mathbf{a}),\mathbf{X}_{N_B})}{\max_{\mathbf{a}} L_B(\hat{u}(\mathbf{x},\mathbf{a}),\mathbf{X}_{N_B})}, \quad (109)$$

были выбраны из окончательного набора оптимальных по Парето решений PINN для оценки результатов. Эффективность последнего критерия была продемонстрирована в предыдущих исследованиях аналогичных задач. Кроме того, были введены верхние границы для характеристик (99) и (100). Для анализа результатов в основном использовались непараметрические тесты Крускала-Уоллиса и Манна-Уитни. Если не указано иное, значимые различия были опре-

делены на уровне 0.05.

Было исследовано влияние количества нейронов N в конечном решении на качество решений, полученных с использованием всех типов алгоритмов. Рисунок 37 показывает, что решения с $N = 20$ и 30 приближаются к фронту Парето. Кроме того, для Алгоритма 1_3 существует значительная разница в характеристиках (26) и (27) на уровне значимости 0.05.

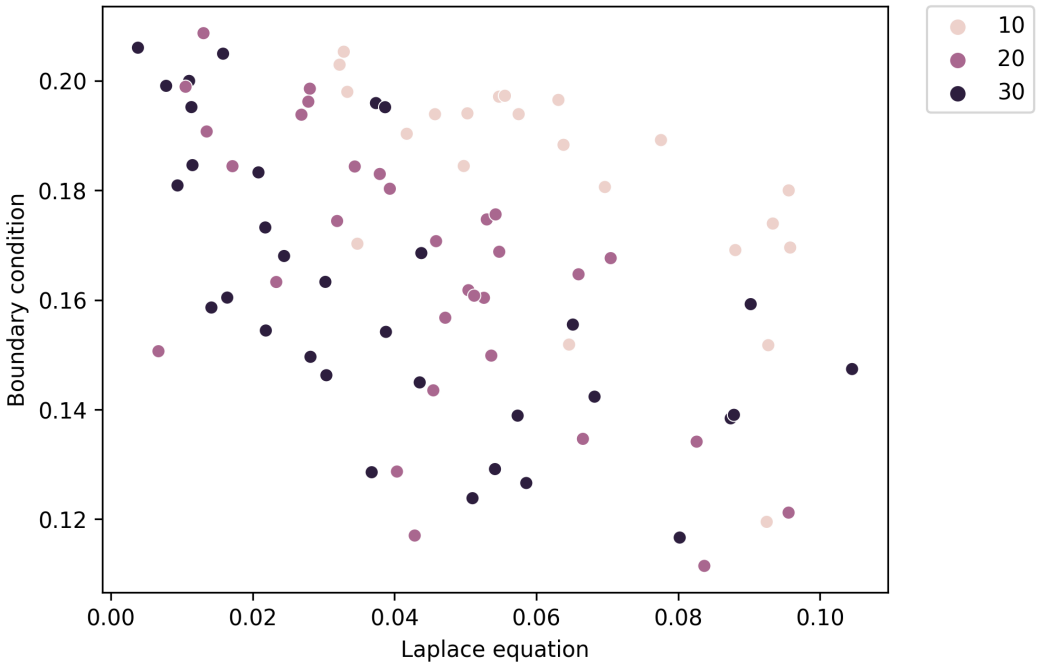

Рисунок 37 – Диаграмма рассеивания показывает все полученные и выбранные эволюционные решения PINN для двухцелевой задачи оптимизации (103)+(104), в зависимости от заранее определенного количества нейронов N в конечной сети

При сравнении результатов работы самих алгоритмов на Рисунке 38 показано, что при сохранении разнообразного диапазона решений в пределах полученного набора особенно наблюдается в случае Алгоритмов 1_1 и 1_2 существенная разница в значениях характеристик (99) и (100) на уровне значимости 0.05. Следует отметить, что при рассмотрении всех полученных решений Алгоритмы 1_1 и 1_2, как правило, повышают важность соответствия критерию удовлетворения уравнению Лапласа.

Экспериментальные результаты показывают, что Парето-оптимальные решения соответствуют наименьшим расхождениям с аналитическим решением, полученным методом Фурье, что подтверждает согласованность оцениваемых методов. Существует статистически значимая разница между результатами Алгоритма 1_3 и Алгоритмами 1_1 и 1_2, что видно на Рисунке 39. Алгоритм

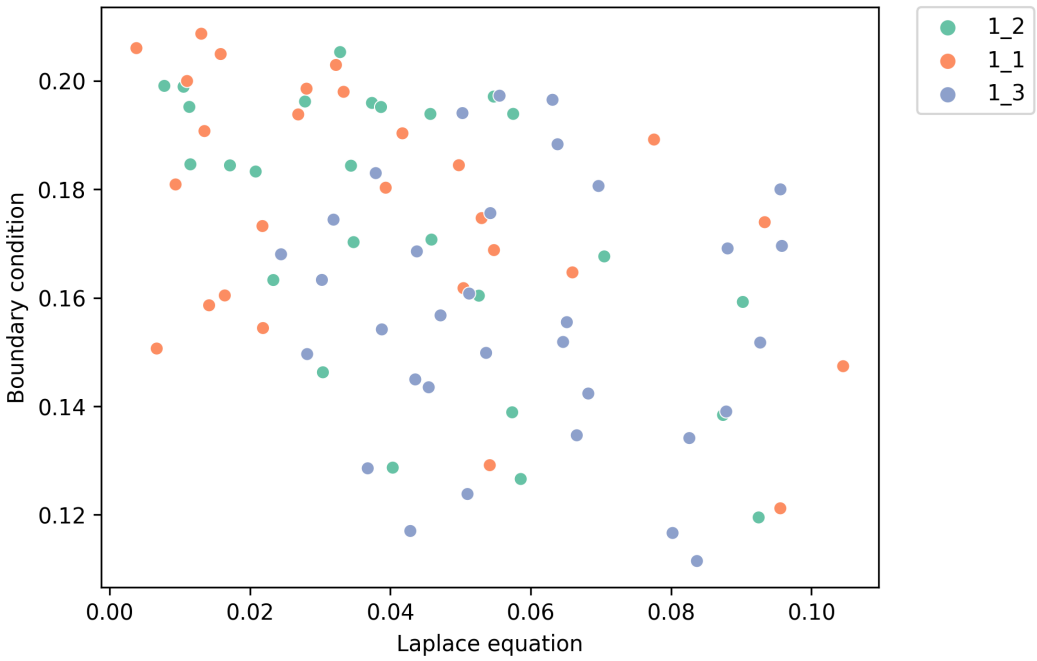

Рисунок 38 – Диаграмма рассеивания показывает все полученные и выбранные эволюционные решения PINN для двухцелевой задачи оптимизации (103)+(104), в зависимости от используемых модификаций алгоритма (1_1, 1_2 и 1_3)

1_3 – единственный, для которого на соответствие аналитическому решению влияет количество нейронов в результирующей сети. В частности, модели с 10 нейронами дают значительно худшие результаты по сравнению с моделями с 20 и 30 нейронами.

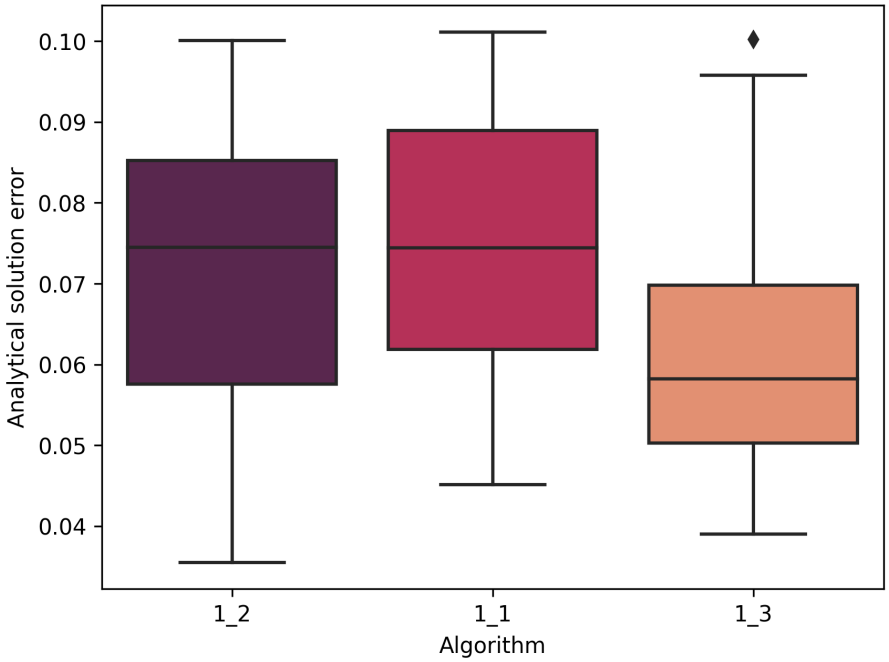

Рисунок 39 – Ящики с усами среднеквадратичной ошибки, удовлетворяющей эволюционным решениям PINN для задачи (103)+(104) авной полученному с использованием метода Фурье решению для различных модификаций алгоритма (1_1, 1_2 and 1_3)

Что касается влияния гиперпараметров, таких как количество эпох обучения PINN на разных этапах алгоритма, то такого влияния на соответствие аналитическому решению не наблюдалось.

Как упоминалось ранее, в дополнение к основным критериям выбора наилучшего решения также может быть принята во внимание погрешность выполнения условия равентсва нулю производной на границе.

При анализе общей диаграммы рассеяния, изображенной на Рисунке 40, можно заметить, что наиболее гладкие на верхней границе решения не относятся к Парето-оптимальному набору. Кроме того, максимальное значение этого критерия равно 2, что значительно ниже значения 40, полученного для аналитического решения с использованием метода Фурье. Кроме того, Алгоритм 1_3 дает значительно худшие результаты по сравнению с Алгоритмами 1_1 и 1_2. Эта тенденция становится более выраженной по мере увеличения числа нейронов. Не наблюдается какого-либо влияния на приверженность аналитическому решению при изменении количества периодов обучения PINN.

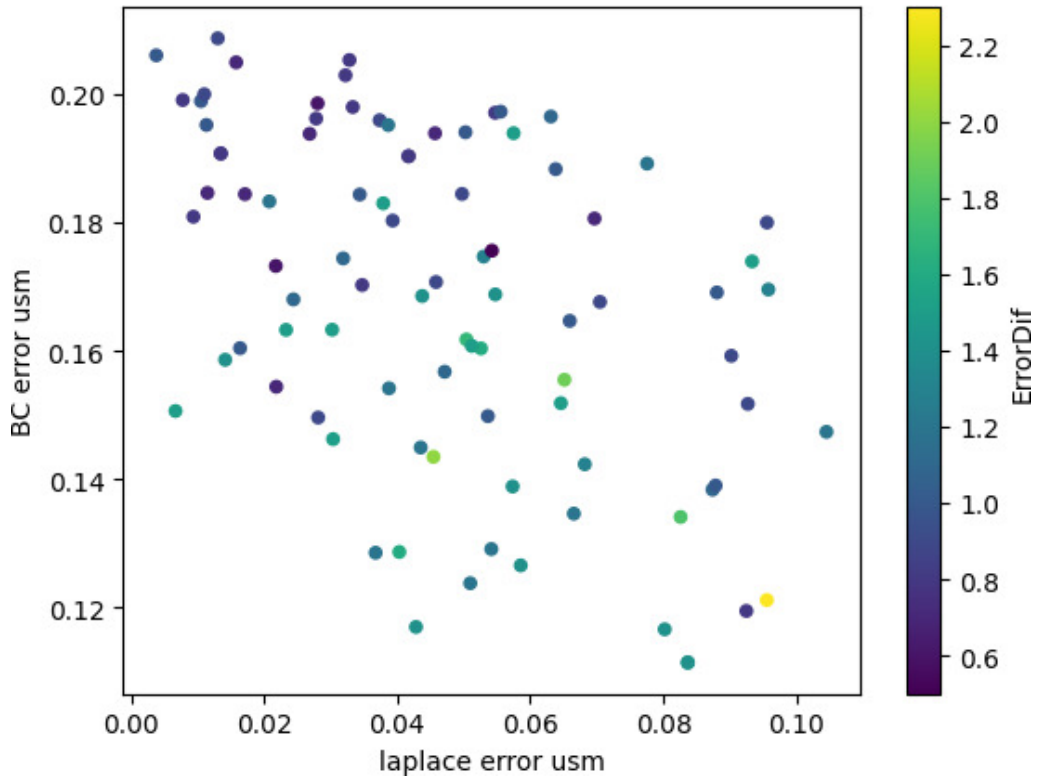

Рисунок 40 – Диаграмма рассеивания показывает все полученные и выбранные эволюционные решения PINN для двухцелевой задачи оптимизации (103)+(104), в зависимости от среднеквадратичной ошибки ErrorDif удовлетворения условию равенства нулю производной на верхней границе области

5.2.5 Результаты: Задача (103)+(105)

В этом параграфе решается задача без граничных условий, в которой данные измерений используются в качестве дополнительного критерия к уравнению Лапласа. Вычислительные эксперименты проводились для параметров алгоритма, представленных в Таблице 9. Алгоритм 1_1 используется для решения этой задачи из-за его меньшей подверженности переобучению в точках коллокации.

Точные данные измерений искусственно моделируются путем аналитического решения уравнения Лапласа с использованием метода Фурье. Чтобы имитировать измерение $u_\varepsilon(x_i, y_i)$ с ошибкой, равномерно распределенная случайная величина t на интервале $[-1, 1]$ добавляется к аналитическому решению $u(x, y)$ по формуле

$$u_\varepsilon(x_i, y_i) = u(x_i, y_i) + t\varepsilon,\ i = 1, \ldots, N_M. \tag{110}$$

Таблица 9 – Конфигурации параметров алгоритма 1_1, используемые для серии экспериментов, при которых решается задача (103)+(105). ε обозначает максимальную случайную погрешность измерения

P_1	P_2	N	K_1, K_2	N_M	ε
9	9	10, 20, 30	30	16, 64	0.01, 0.1

Чтобы предотвратить переобучение, точки измерений распределены по псевдоравномерной сетке внутри квадрата со случайным сдвигом относительно исходной равномерной сетки.

Половина точек измерения выделяется для обучения, в то время как оставшаяся половина используется для отбора лучших индивидов в процессе скрещивания и поиска оптимального решения в заключительном цикле алгоритма.

Было выполнено пять запусков вычислительного эксперимента для каждой конфигурации параметров алгоритма и набора параметров данных измерений. Оптимальный набор Парето из всех решений состоит из нейронных сетей, которые прекратили обучение при $N = 30$ нейронах.

Рисунок 41 иллюстрирует, как алгоритм придает большее значение удовлетворению требуемому решению уравнения Лапласа при меньшем количестве измерений.

В целом, существенных различий при различных уровнях шума обнаружено не было, что указывает на устойчивость алгоритма к ошибкам в измерениях. Это показано на Рисунке 42. Кроме того, большая ошибка в данных никак не влияет на погрешность в получении точного решения.

Для сравнения с общим нейросетевым подходом была обучена сеть из 30 нейронов линеаризованной функции потерь с фиксированным штрафным множителем, рассчитанным во время инициализации. Однако ни одно из полученных решений не попало в целевой диапазон из-за переориентации на одно из условий, на которое повлияли начальные случайные значения весов сети.

5.3 Обсуждение результатов

В данной главе было рассмотрено использование эволюционного алгоритма для построения нейросетевых моделей, основанных на физике, для реальных объ-

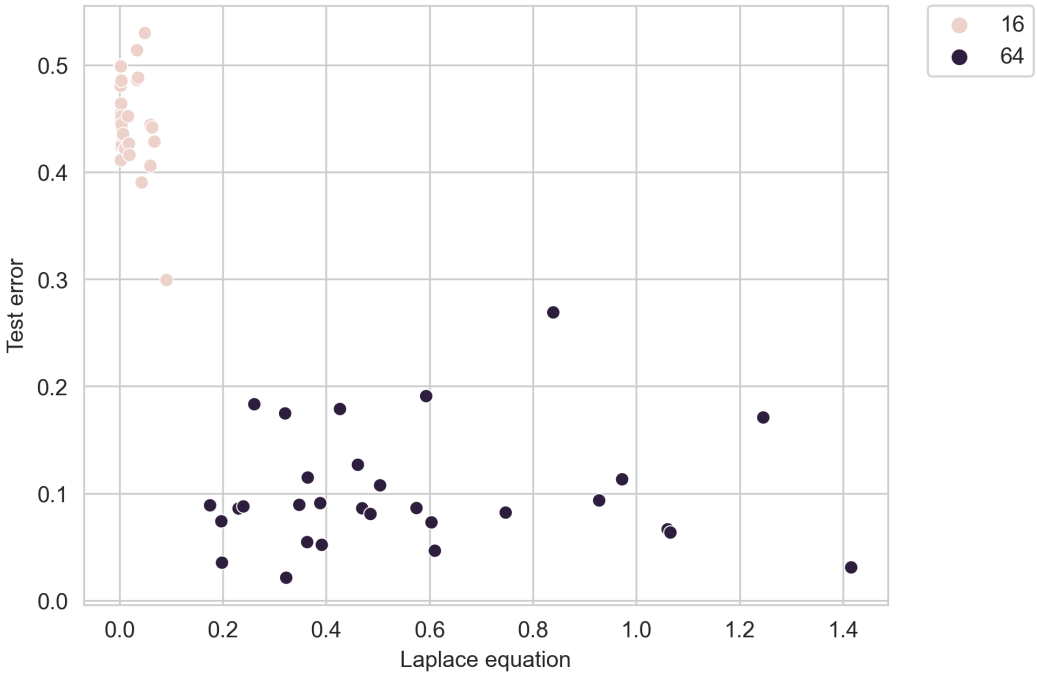

Рисунок 41 – Диаграмма рассеивания всех полученных эволюционных решений PINN для двух-критериальной задачи (103)+(105), в зависимости от общего числа измерений

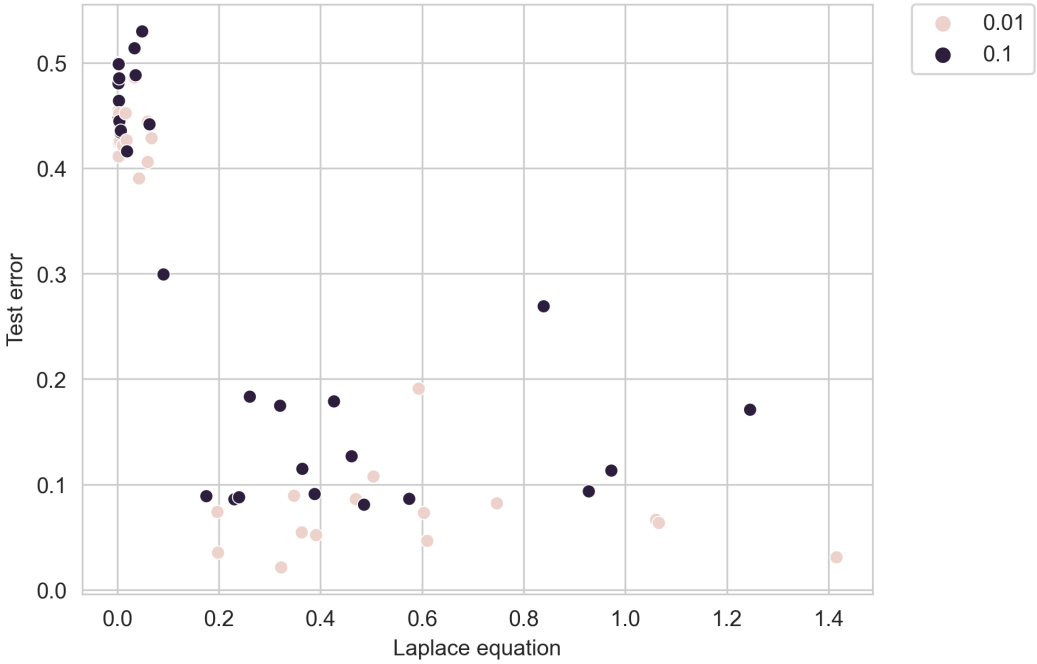

Рисунок 42 – Диаграмма рассеивания всех полученных эволюционных решений PINN для двух-критериальной задачи (103)+(105), в зависимости от максимальной случайной ошибки в данных измерений

ектов, что создает некоторые трудности при их описании и задачи в некорректной постановке. В первой задаче исследуется противоречивая математическая модель, содержащая дифференциальные уравнения и граничные условия. Во

второй задаче полная математическая модель объекта отсутствует (краевые условия), что подразумевается компенсировать с помощью данных наблюдений. Для решения обеих задач были применены различные модификации предложенной впервые общей схемы эволюционного обучения нейронных сетей на основе идеи фронта Парето .

Исследование показало, что предложенный алгоритм может включать различные критерии как на этапах обучения, так и на этапах отбора модельных популяций. Увеличение общего числа нейронов в модели повышает качество решений, в то время как параллельное обучение отдельных нейронов позволяет уделять особое внимание конкретным критериям или получать решения с желаемой гладкостью. Все модели из набора оптимумов по Парето демонстрируют наилучшее соответствие с аналитическим решением, полученным методом Фурье, и также наблюдается устойчивость решений к возрастающим погрешностям измерений.

Будущие исследования могут помочь изучить свойства этого семейства эволюционных алгоритмов для решения других типов некорректных задач.

Глава 6.

Адаптация нейросетевых моделей

6.1 Введение

Как уже было отмечено ранее, в определенных ситуациях, когда знание дифференциального уравнения и граничных условий является неточным или неполным, а результаты измерений доступны, нейронные сети оказываются более эффективными для построения моделей, чем классические методы. Другой класс задач, в которых рекомендуется использовать нейронные сети, – это когда параметры уравнения претерпевают неизвестные изменения и в динамических процессах доступны только результаты измерений. Этими изменениями можно считать плавные увеличения или уменьшения, а также резкие скачки значений параметров, соответствующих каким-либо физическим свойствам моделируемого объекта и его функционированию или условиям эксплуатации. Это могут быть задачи, в которых параметры характеризуют форму области или граничные условия для дифференциальных уравнений [23]. В то же время нерешенным остается вопрос об адаптивных свойствах моделей, полученных с помощью общего нейросетевого подхода, для задач, когда решение качественно меняет свою природу.

В данной главе рассматриваются возможности адаптации приближенных нейросетевых решений краевых задач для дифференциальных уравнений к изменениям в данных о моделируемом физическом объекте. Рассматриваются два типа моделей: PINN и параметрические PINN (PPINN). PINNs создаются с фиксированными значениями параметров с использованием общего нейросетевого подхода [144, 145], а PPINN включают параметры в качестве входных переменных [3, 141, 85, 21, 90]. Включение системных параметров в качестве входных данных в модель позволяет проводить моделирование в реальном времени

для целого ряда задач и сокращает время вычислений [20]. Более того, решение параметризованных систем может быть использовано в широком спектре управления сложными системами [160]. Оба типа моделей тестируются на трех задачах. Первая задача включает моделирование изгиба консольного стержня при различных нагрузках. Вторая задача представляет собой нестационарную задачу о тепловом взрыве в плоскопараллельном случае. Исходная модель строится на основе обыкновенного дифференциального уравнения, в то время как объект моделирования удовлетворяет дифференциальному уравнению в частных производных. Третья задача заключается в решении дифференциального уравнения в частных производных смешанного типа, зависящего от времени. Во всех случаях исходные модели адаптируются к соответствующим псевдоизмерениям, сгенерированным на основе изменяющихся уравнений. Для каждой задачи проводится серия экспериментов с различными функциями параметра, отражающего характер изменений в объекте. В этом исследовании впервые был проведен сравнительный анализ качества моделей PINN и PPINN и их устойчивости к изменениям данных.

6.2 Общая постановка задач и описание типов моделей

Рассмотрим краевую задачу в общем виде, как для обыкновенных дифференциальных уравнений, так и для уравнений в частных производных

$$\begin{cases} \mathscr{D}[u](\mathbf{x}) = \quad f(\mathbf{x}),\ \mathbf{x} \in \Omega; \\ \mathscr{B}[u](\mathbf{x}) = \ \ g(\mathbf{x}),\ \mathbf{x} \in \partial\Omega, \end{cases} \tag{110}$$

где $\Omega \subset \mathbb{R}^m$; $u, f : \mathbb{R}^m \to \mathbb{R}$, $g : \mathbb{R} \to \mathbb{R}$; $\mathscr{D}[\cdot]$ – некоторый дифференциальный оператор; и $\mathscr{B}[\cdot]$ – допустимый оператор, устанавливающий граничные условия.

Общий нейросетевой подход к решению этой задачи заключается в подборе параметров (весов) нейронной сети в процессе минимизации функции потерь, соответствующей рассматриваемой задаче.

Подразумевая изменения в объекте, описываемом с помощью системы (110), в формулировку задачи вводится некоторый параметр δ, который принимает приемлемые значения в соответствующей области $\Gamma \subset \mathbb{R}^d$ и с соответствующими d:

$$\begin{cases} \mathscr{D}[u](\mathbf{x}, \delta(t)) = \quad f(\mathbf{x}, \delta(t)), \ \mathbf{x} \in \Omega; \\ \mathscr{B}[u](\mathbf{x}, \delta(t)) = \ g(\mathbf{x}, \delta(t)), \ \mathbf{x} \in \partial\Omega. \end{cases} \tag{110}$$

Здесь $u, f : \mathbb{R}^{m+1} \to \mathbb{R}$, $g : \mathbb{R}^2 \to \mathbb{R}$. Некоторые значения параметра $\delta(t)$, где время $t \in [0, T]$, могут изменять природу задачи, например, уменьшить ее размерность или тип дифференциального оператора.

В данной главе рассматриваются задачи следующего типа. Изначально объект описывается с помощью системы (110) при фиксированном значении параметра δ. Затем начинают происходить изменения, которые в модельных задачах выражаются в виде изменения параметра в системе (110). Между тем, информация об этих изменениях поступает к нам только в виде измерений в реальном времени. Возникает вопрос, насколько хорошо нейросетевая модель может адаптироваться к таким изменениям. Чтобы ответить на этот вопрос, рассматриваются два типа нейросетевых моделей PINN (111) и PPINN (113), различия в которых проиллюстрированы на Рисунке 43.

В первом случае исходная модель PINN строится для системы (110) и представляет собой выход нейронной сети с одним скрытым слоем вида

$$u_{\mathbf{c},\mathbf{a}}(\mathbf{x}) = c_0 + \sum_{i=1}^{n} c_i \varphi(\mathbf{x}, \mathbf{a}_i). \tag{111}$$

где веса сети (параметры $\mathbf{c} = (c_0, c_1, \ldots, c_n) \in \mathbb{R}^{n+1}$, $\mathbf{a}_i \in \mathbb{R}^q$; q зависят от выбранной базисной функции $\varphi : \mathbb{R}^m \times \mathbb{R}^q \to \mathbb{R}$) и подбираются в ходе процесса минимизации функции потерь вида

$$J(\mathbf{c}, \mathbf{a}) = \sum_{j=1}^{M_D} \left(D[u_{\mathbf{c},\mathbf{a}}](\mathbf{x}_j) - f(\mathbf{x}_j) \right)^2 + A \sum_{j=1}^{M_B} \left(B[u_{\mathbf{c},\mathbf{a}}](\mathbf{z}_j) - g(\mathbf{z}_j) \right)^2 \tag{112}$$

для известного фиксированного начального значения параметра δ, где $\{\mathbf{x}_j\}_{j=1}^{M_D}$ – это обучающая выборка, сгенерированная случайным образом с использованием равномерного распределения внутри области Ω с условием последующих перегенераций в соответствии с выбранным алгоритмом обучения нейронной сети. Обучающая выборка для граничных условий $\{\mathbf{z}_j\}_{j=1}^{M_B}$ представляет собой некоторую сетку на $\partial\Omega$. A – положительный штрафной параметр.

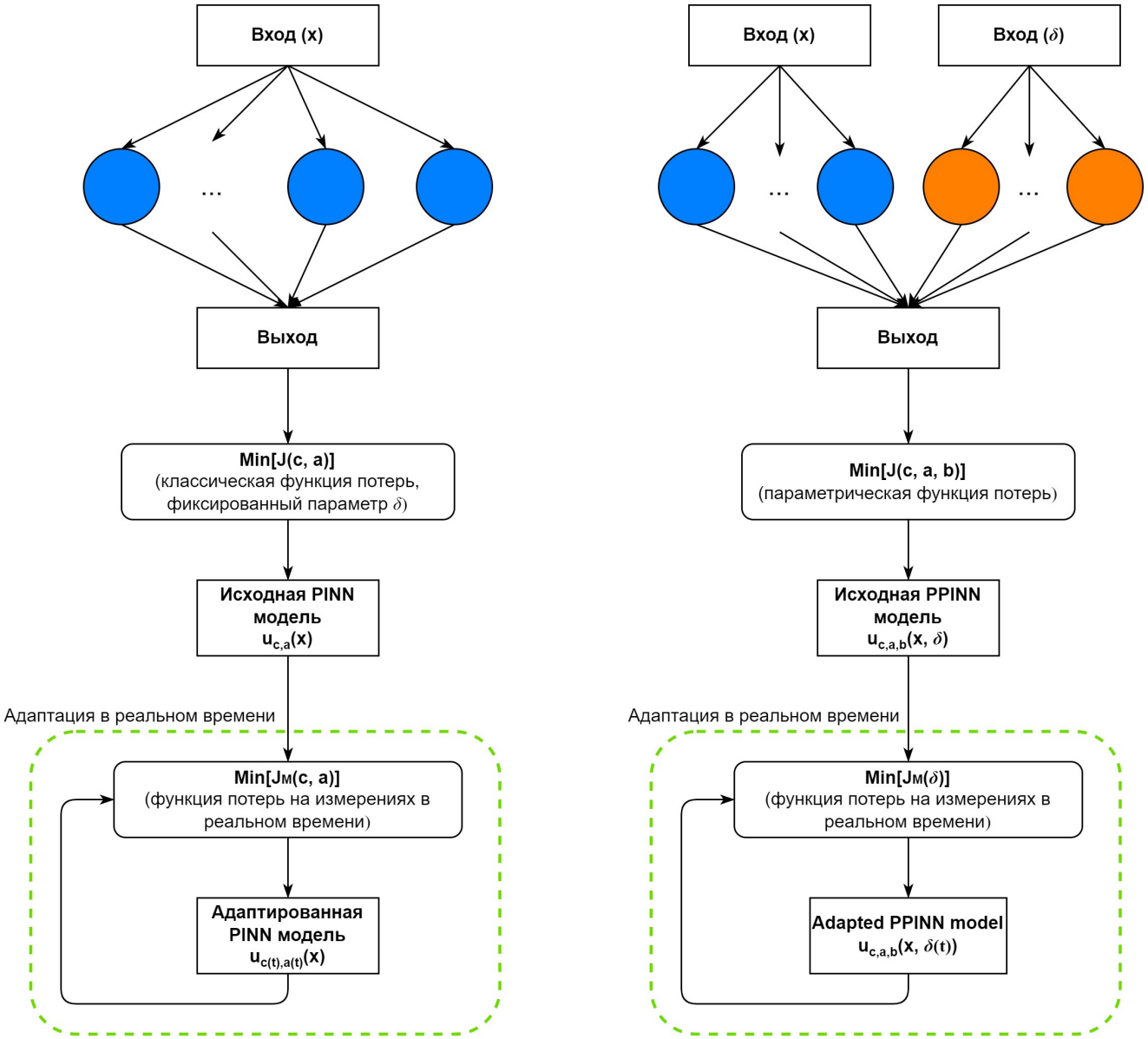

Рисунок 43 – Общая схема построения нейросетевых моделей PINN (слева) и PPINN (справа) на основе решения дифференциальных уравнений и их последующей адаптации по данным измерений

Исходная модель PPINN имеет вид

$$u_{\mathbf{c},\mathbf{a},\mathbf{b}}(\mathbf{x},\delta) = c_0 + \sum_{i=1}^{n} c_i \varphi(\mathbf{x}, \mathbf{a}_i, \delta, \mathbf{b}_i). \tag{113}$$

Веса сети (параметры $\mathbf{c} \in R^{n+1}$, $\mathbf{a} \in R^{q_1}$, $\mathbf{b} \in R^{q_2}$; q_1, q_2, зависящие от базисной функции $\varphi : \mathbb{R}^{m+1} \times \mathbb{R}^{q_1} \times \mathbb{R}^{q_2} \to \mathbb{R}$) подбираются путем минимизации функции потерь, соответствующей системе (110). Дополнительный параметр $\mathbf{b}$ вводится для того, чтобы подчеркнуть разницу между базисными функциями для PINN

и PPINN. А именно,

$$J(\mathbf{c},\mathbf{a},\mathbf{b}) = \sum_{j=1}^{M_D} \left(D[u_{\mathbf{c},\mathbf{a},\mathbf{b}}](\mathbf{x}_j,\delta_j) - f(\mathbf{x}_j,\delta_j)\right)^2 + A\sum_{j=1}^{M_B} \left(B[u_{\mathbf{c},\mathbf{a},\mathbf{b}}](\mathbf{z}_j,\delta_j) - g(\mathbf{z}_j,\delta_j)\right)^2, \tag{113}$$

где $\{\mathbf{x}_j,\delta_j\}_{j=1}^{M_D}$ и $\{\mathbf{z}_j,\delta_j\}_{j=1}^{M_B}$ – это обучающая выборка, сгенерированная случайным образом с использованием равномерного распределения внутри области $\Omega\times\Gamma$ и некоторая сетка на $\partial\Omega\times\Gamma$. A – положительный штрафной параметр.

Под адаптацией обеих нейросетевых моделей PINN и PPINN понимается подбор весов $(\mathbf{c},\mathbf{a})$ для решения (111) и параметра δ для решения (113) в ходе минимизации соответствующих функций потерь

$$J_M(\mathbf{c},\mathbf{a}) = \sum_{j=1}^{M} \left(u_{\mathbf{c},\mathbf{a}}(\mathbf{x}_j) - u_j(t_k)\right)^2 \tag{114}$$

и

$$J_M(\delta) = \sum_{j=1}^{M} \left(u_{\mathbf{c},\mathbf{a},\mathbf{b}}(\mathbf{x}_j,\delta) - u_j(t_k)\right)^2, \tag{115}$$

где $\{\mathbf{x}_j,u_j(t_k)\}_{j=1}^{M}$ – данные измерений в моменты времени t_k.

PINN обучается с фиксированным значением начального параметра. PPINN включает этот параметр в качестве дополнительной входной переменной и обучается с использованием определенного набора изменений начальных параметров. Временное окно, в котором выбираются измерения, сдвигается в процессе прихода новых измерений. Впоследствии параметр продолжает изменяться, но эти изменения не используются для обучения сетей. Вместо этого используются данные измерений, отражающие косвенно изменение этого параметра. PINNs дополнительно обучаются для оптимизации своей работы путем минимизации заданной целевой функции. В отличие от этого, PPINN фокусируется на идентификации параметров, что, следовательно, также влияет на выходную модель.

6.3 Модельные задачи, их решение и анализ типов моделей

Контрольные задачи, которые выбраны для сравнения описанных в предыдущем параграфе нейросетевых моделей, позволяют изучить различные аспекты изменений в объекте моделирования. Псевдоизмерения, используемые для адаптации построенных моделей, генерируются на основе известных аналитических решений. Актуальными являются постановки задачи, когда объект описывается с помощью уравнений, которые не совсем точны. Таким образом, во второй контрольной задаче исходная модель строится на основе обыкновенного дифференциального уравнения, в то время как на самом деле объект моделирования удовлетворяет уравнению в частных производных, согласно которому генерируются соответствующие измерения.

6.3.1 Моделирование изгиба консольного стержня под нагрузкой

Данная задача связана с изучением прогиба консольного стержня под действием нагрузки. Для выбора приближенной дифференциальной модели, было сделано несколько предположений о свойствах стержня, который рассматривался как бесконечно тонкая, однородная, линейно упругая конструкция, которая изначально была прямой.

Таким образом, было выбрано уравнение, описывающее большой статический прогиб такого стержня под действием распределенных и сосредоточенных сил, проецируемых на касательную кривой прогиба, то есть дифференциальное уравнение:

$$\frac{d^2\theta}{dz^2} = a(\delta + z)\cos\theta, \tag{116}$$

где $a = mg/D$, $\delta = F/mg$; D – постоянная жесткость при изгибе; θ обозначает угол наклона касательной к кривой, описывающей стержень; z представляет естественную координату изогнутой оси стержня, измеренную от места крепления; m обозначает вес стержня; а F представляет силу, действующую на незакрепленный конец стержня (см. Рисунок 44).

Таким образом, граничные условия имеют вид

$$\frac{d\theta}{dz}\Big|_{z=0} = 0,\ \theta\Big|_{z=1} = 0. \tag{117}$$

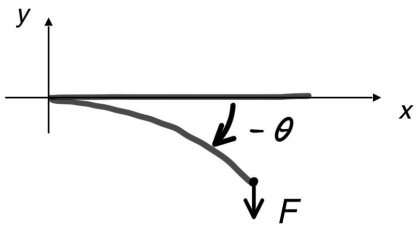

Рисунок 44 – Схема изгиба консольного стержня под нагрузкой

Зависимость между углом и координатами точек на стержне описывается с помощью равенств

$$\frac{dx}{ds} = \cos\theta, \ \frac{dy}{ds} = \sin\theta. \tag{118}$$

Для генерации псевдоизмерений используются различные формы зависимостей параметров от времени, представленные различными функциями $\delta(t)$. Каждая функция $\delta(t)$ обладает уникальными характеристиками, которые показаны на Рисунке 45 и описаны ниже.

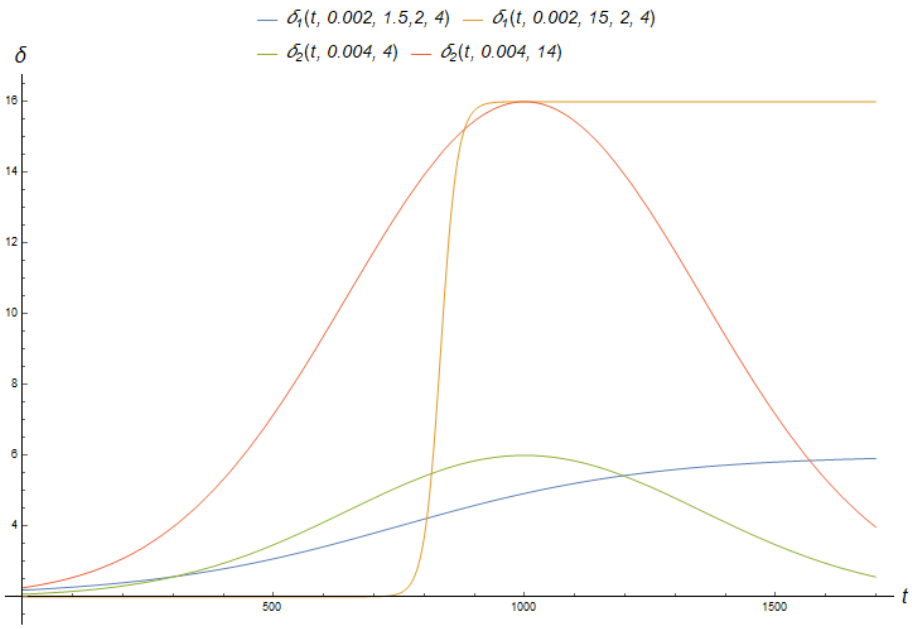

Рисунок 45 – Различные формы зависимостей параметров от времени (119) и (120) от времени

1. Первый тип изменения временной зависимости параметра соответствуют переходному процессу, характеризующемуся максимальной скоростью в течение определенного интервала времени, который в конечном итоге достигает состояния насыщения,

$$\delta_1(t, \alpha, \beta, \gamma, \varepsilon) = \varepsilon + \gamma \tanh(\alpha t - \beta)); \tag{119}$$

2. Вторая временная зависимость параметра имитирует кратковременное отклонение от его стационарного значения,

$$\delta_2(t, \alpha, \gamma) = 2 + \gamma \exp(-(\alpha t - 2)^2). \tag{120}$$

6.3.2 Нестационарная задача о тепловом взрыве в плоскопараллельном случае

В качестве второй задачи была рассмотрена нестационарная задача о тепловом взрыве в плоскопараллельном случае [67] в предположении, что реакция имеет одну стадию и является необратимой; она не сопровождается фазовыми переходами и протекает в стационарной среде.

Такие процессы довольно точно описываются уравнением в частных производных [19, 130]:

$$\begin{cases} \dfrac{\partial^2 u}{\partial x^2} + \delta(t)\exp(u) = & \dfrac{\partial u}{\partial t}, \\ \dfrac{\partial u}{\partial x}(0,t) = & 0, \\ u(1,t) = & 0, \\ u(x,0) = & u_0(x), \end{cases} \tag{120}$$

где $u_0(x)$ – аналитическое решение для системы (120) в случае $\delta = \delta(0)$.

Предполагается, что нестационарный характер рассматриваемого объекта неизвестен заранее и для построения начальной модели используется независящая от времени краевая задача для обыкновенного дифференциального уравнения второго порядка [67]:

$$\begin{cases} \dfrac{d^2 u}{dx^2} + \delta\exp(u) = & 0, \\ \dfrac{du}{dx}(0) = & 0, \\ u(1) = & 0. \end{cases} \tag{120}$$

Эта задача интересна тем, что мы знаем точное решение, область существования решения ($\Omega = [0, 1]$) и значения параметра δ, для которых решение задачи не существует ($\delta > \delta^* \approx 0.878458$). Здесь интервал изменения перемен-

ной определяется из постановки задачи и связан с переходом к безразмерной координате. В случае малых значений параметра довольно удобно получить решение с помощью асимптотических методов. Кроме того, для достижения низкой относительной погрешности для значений этих параметров требуется модификация функции потерь. Очевидно, что режим работы реального реактора не должен приближаться к пределам стабильности. Низкая скорость реакции, соответствующая малым значениям параметра δ, также не соответствует реальным режимам работы из-за низкого КПД. Поэтому для обучения параметрической модели мы предполагаем, что $\delta \in [0.2; 0.85]$ и исследуем возможность адаптации моделей PINN при изменении параметра δ.

Для генерации псевдоизмерительных данных используются различные варианты зависимости параметра от времени. Каждая функция $\delta(t)$ имеет свои особенности, которые показаны на Рисунке 46.

1. Первый вариант временной зависимости параметра является частным случаем (119) и соответствует переходному процессу, который в начальный момент имеет максимальную скорость и становится насыщенным:

$$\delta_1(t, \alpha, 0, 0.6, 0.2) = 0.2 + 0.6\mathrm{th}(\alpha t); \tag{121}$$

2. Второе изменение функции имитирует кратковременное отклонение параметра от стационарного значения:

$$\delta_3(t, \alpha) = 0.2 + 0.67\mathrm{sech}^2(\alpha t - 3); \tag{122}$$

3. Третья зависимость также является частным случаем (119) и отличается от первых двух временным сдвигом в значительном изменении параметра:

$$\delta_1(t, \alpha, 2, 0.3, 0.5) = 0.5 + 0.3\mathrm{th}(\alpha t - 2); \tag{123}$$

4. Последняя зависимость представляет собой скачок в начальный момент времени в область, близкую к границе устойчивости решения. Здесь важно подчеркнуть, что значение параметра в этом случае выходит за пределы интервала, на котором были обучены параметрические модели PINN.

Оно задается с помощью

$$\delta_4(t) = \begin{cases} 0.87, & t > 0; \\ 0.2, & t = 0. \end{cases} \tag{124}$$

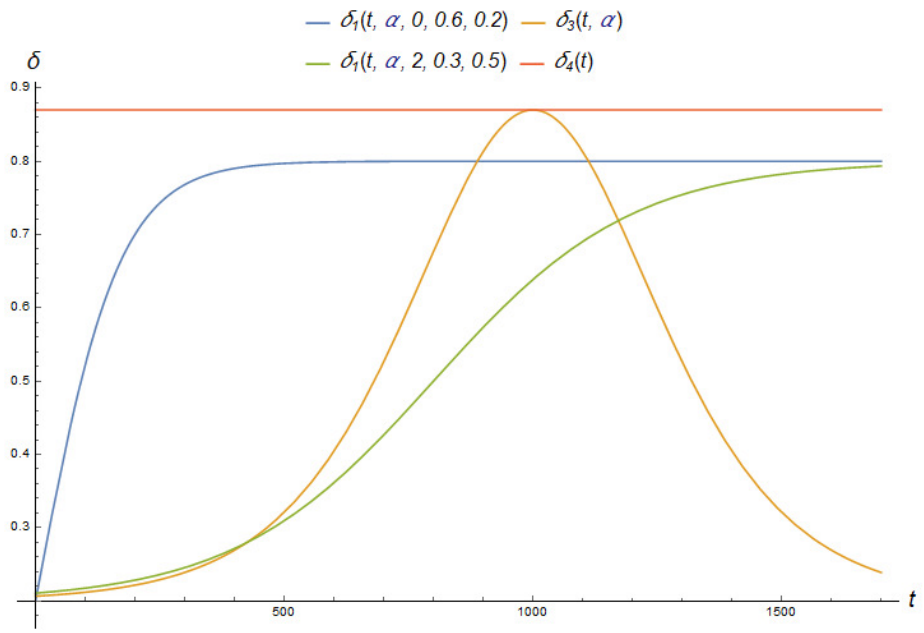

Рисунок 46 – Различные варианты зависимости параметра (121)–(124) от времени

6.3.3 Задача с уравнением смешанного типа

Рассмотрим задачу для дифференциального уравнения в частных производных, в которой изменение параметра приводит к изменению типа уравнения с эллиптического на гиперболическое, то есть краевую задачу на единичном квадрате ($\Omega = [0,1] \times [0,1]$):

$$\begin{cases} \dfrac{\partial^2 u}{\partial x^2} + \delta \dfrac{\partial^2 u}{\partial y^2} = & 0, \\ u(x,0) = & \delta x^2, \\ u(x,1) = & \delta x^2 - 1, \\ u(0,y) = & -y^2, \\ u(1,y) = & \delta - y^2. \end{cases} \tag{124}$$

Точное решение задачи (124) имеет вид

$$w(x, y, \delta) = \delta x^2 - y^2. \tag{125}$$

Предполагается, что параметр может изменяться в пределах интервала $[-1.2, 1.2]$. Снова предлагается рассмотреть различные типы зависимости параметра δ от времени (см. Рисунок 47):

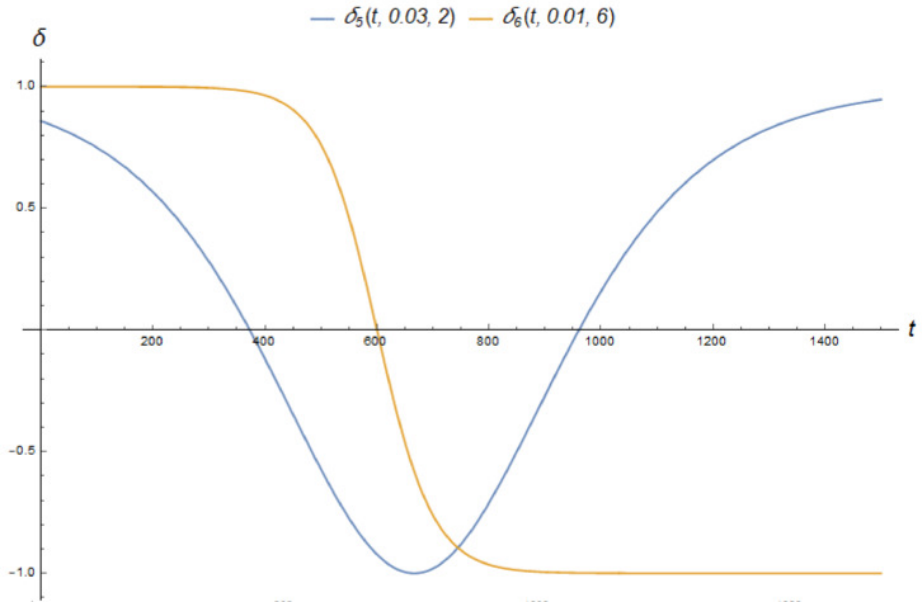

Рисунок 47 – Различные варианты зависимости параметра (126)–(127) от времени

1. Первая зависимость приводит к кратковременному отклонению параметра от стационарного значения:

$$\delta_5(t) = 1 - 2\mathrm{sech}^2(\alpha t - \beta); \tag{126}$$

2. Второй тип зависимости имеет вид сигмоиды со сдвигом по времени (задержкой). Таким образом, рассматривается ситуация, когда со временем происходит переход из одного стационарного состояния в другое. Параметр изменяется в соответствии с

$$\delta_6(t) = \mathrm{th}(\alpha t - \beta); \tag{127}$$

6.3.4 Генерация псевдоизмерений

Как упоминалось выше, псевдоизмерения $u_j(t_k)$ генерируются для всех задач путем вычисления значений решений в моменты времени t_k, $k = 1, \ldots, M$, указывая время в секундах с момента запуска алгоритма адаптации. Для решения первой задачи были использованы искусственные данные измерений $\theta(t_k)$, полученные вычислением значений $\theta(z_j, t_k)$ для решения задачи

$$\frac{d^2\theta}{dz^2} - a(\delta(t) + z)\cos\theta = 0,\ \frac{d\theta}{dz}(0,t) = 0,\ \theta(1,t) = 0. \tag{128}$$

Таким образом, $u_j(t_k)$ для второй задачи – это значения решения $u(t_k, x_j)$ для задачи (120) в случае фиксированного параметра $\delta = \delta(t_k)$.

Для третьей задачи использовалось точное решение $w(x_j, y_j, \delta(t_k))$ (125) для системы (124) и $\delta = \delta(t_k)$.

Были выбраны $M = 100$ для всех задач и $M = 4$ для специального эксперимента с задачей смешанного типа.

6.4 Результаты экспериментов

Для всех задач минимизация функций потерь (112) и (113) выполняется в соответствии с алгоритмом нелинейной оптимизации RProp [125] с перегенерацией [55] тестовых (обучающих) точек каждые пять шагов алгоритма оптимизации. Выбор алгоритма RProp продиктован тем, что он позволяет локализовать адаптацию обновлений веса на основе поведения функции ошибок. В отличие от других адаптивных методов, процесс адаптации с RProp не зависит от непредсказуемого влияния величины производной по каждому конкретному параметру, а зависит исключительно от временной закономерности изменения ее знака. Преимущества RProp включают быструю сходимость и меньшую чувствительность к начальным параметрам.

Метод RProp основан на следующих формулах. Если произведение градиентов на текущем и предыдущем шаге больше нуля, то величина шага увеличивается, обновление веса выполняется по формуле $\Delta w(t) = -\mathtt{sign}\frac{\partial E}{\partial w(t)}\Delta(t)$. Если произведение градиентов на текущем и предыдущем шаге меньше нуля, то величина шага уменьшается, при этом не происходит реального изменения веса, а только корректируется шаг. Если произведение градиентов равно ну-

лю, веса обновляются по предыдущей формуле, но без коррекции шага. Этот алгоритм повторяется до сходимости.

Перегенерация тестовых точек позволяет избежать переобучения нейронной сети. Под перегенерацией подразумевается выбор новых точек обучения путем генерации выборки из равномерного распределения в областях Ω и $\Omega \times \Gamma$ для непараметрической и параметрической нейронных сетей соответственно. Для второго члена функции потерь используются равномерные сетки на $\partial\Omega \times \Gamma$ и $\partial\Omega$.

Адаптация исходных моделей в виде настройки параметров конкретной нейронной сети происходит в ходе минимизации функций потерь (114) и (115) для моделей PINN и PPINN соответственно. Для каждой модели выполняется пять шагов алгоритма минимизации между динамически поступающими данными измерений.

6.4.1 Моделирование изгиба консольного стержня под нагрузкой

6.4.1.1 Модель PINN общего непараметрического нейросетевого подхода

Модель PINN (111) для задачи (116)–(117) будет иметь вид

$$u_{\mathbf{c},\mathbf{a}}(z) = c_0 + \sum_{i=1}^{n} c_i \varphi(z, \mathbf{a}_i), \tag{129}$$

при фиксированном значении параметра δ . В этом случае в качестве базисных функций были выбраны гауссианы

$$\varphi(z, \mathbf{a}_i) = \exp\left(-a_{i1}(z - a_{i2})^2\right), \tag{130}$$

и функции персептронного типа

$$\varphi_p(z, \mathbf{a}_i) = \tanh\left(a_{i1}(z - a_{i2})\right). \tag{131}$$

Параметры $\mathbf{c}$ и $\mathbf{a}$ модели PINN оптимизировались в ходе минимизации

функционала ошибки

$$J(\mathbf{c},\mathbf{a}) = \sum_{j=1}^{M} \Big(\frac{d^2 u_{\mathbf{c},\mathbf{a}}}{dz^2}(z_j) - a(\delta + z_j)\cos u_{\mathbf{c},\mathbf{a}}(z_j)\Big)^2 + A\Big((\frac{du_{\mathbf{c},\mathbf{a}}}{dz}(0))^2 + (u_{\mathbf{c},\mathbf{a}}(1))^2\Big). \quad (131)$$

Здесь, тестовые точки $\{z_j\}_{j=1}^{M}$ случайным образом выбираются равномерно распределенными на интервале $[0,\,1]$. Штрафной множитель A вычисляется с использованием инициализации сети таким образом, что слагаемые функции потерь имеют один порядок.

6.4.1.2 Параметрическая модель PPINN общего нейросетевого подхода

В качестве приближенной модели PPINN (113) строится нейронная сеть вида

$$u_{\mathbf{c},\mathbf{a},\mathbf{b}}(z,\delta) = c_0 + \sum_{i=1}^{n} c_i \varphi(z,\mathbf{a}_i,\delta,\mathbf{b}_i). \quad (132)$$

В данном случае были изучены различные виды базовых функций, и наиболее благоприятные результаты были достигнуты при выборе базовых функций в форме, описываемой следующим уравнением:

$$\varphi(z,\mathbf{a}_i,\delta,\mathbf{b}_i) = \exp\left(-a_{i1}(z-a_{i2})^2\right)\tanh\left(b_{i1}(\delta - b_{i2})\right). \quad (133)$$

Кроме того, для сравнения с гауссовской моделью PINN, были рассмотрены

$$\varphi_p(x,\mathbf{a}_i,\delta,\mathbf{b}_i) = \tanh\left(a_{i1}(x-a_{i2})\right)\tanh\left(b_{i1}(\delta - b_{i2})\right). \quad (134)$$

Веса сети тщательно подбираются таким образом, чтобы свести к минимуму погрешность функции:

$$J(\mathbf{c},\mathbf{a},\mathbf{b}) = \sum_{j=1}^{M} \Big(\frac{d^2\theta}{dz^2}(z_j,\delta_j) - a(\delta_j + z)\cos\theta(z_j,\delta_j)\Big)^2 + A\Big((\frac{d\theta}{dz}(0,\delta_j))^2 + (\theta(1,\delta_j))^2\Big). \quad (134)$$

Здесь $\{z_j, \delta_j\}_{j=1}^{M}$ – это тестовые точки, выбранные случайным образом с использованием двумерного равномерного распределения внутри прямоугольника $[0,1] \times [2,16]$. В вычислительных экспериментах было выбрано значение $M = 100$ для обеих моделей PINN и PINN.

6.4.1.3 Модели PINN и PPINN для стационарного этапа

Изначально нейронные сети PINN и PPINN были обучены без учета динамически изменяющихся данных. Для приближенного нейросетевого решения PINN (111) была успешно достигнута достаточно малая погрешность для сети с двумя нейронами. При выборе определенного значения параметра $\delta = 2$ результирующие сети будут иметь вид

$$u_{\mathbf{c},\mathbf{a}}(z) = 0.5967 - 0.3234e^{-0.6353(z-1.115)^2} - 0.6239e^{-0.3529(z+0.5196)^2} \tag{135}$$

для гауссиан и

$$\begin{aligned} v_{\mathbf{c},\mathbf{a}}(z) = 0.2834 &+ 0.4053\tanh(0.9488(-1.706 + x)) \\ &-0.0474\tanh(1.6289(0.3572 + x)) \end{aligned} \tag{135}$$

для базисных функций типа (131).

Точность полученных решений дополнительно проиллюстрирована на Рисунке 48. Впоследствии эти сети используются в качестве начальных приближений для обработки динамических изменений в данных.

Рисунок 48 наглядно демонстрирует, что PINN-персептрон обеспечивает заметно более высокую точность. Однако последующие вычислительные эксперименты показали, что эта более высокая точность не обязательно приводит к повышению производительности при обработке динамических данных.

При значении параметра $\delta = 16$ нейросетевая аппроксимация с гауссианами имеет вид

$$u_{\mathbf{c},\mathbf{a}}(z) = 3.527 - 3.412e^{-0.4586(z-0.7198)^2} - 2.488e^{-0.6949(z+0.8398)^2}. \tag{136}$$

Абсолютная погрешность этого приближения остается ниже 0,006, демонстрируя потенциальную точность, которая может быть достигнута при увеличении параметра до $\delta = 16$.

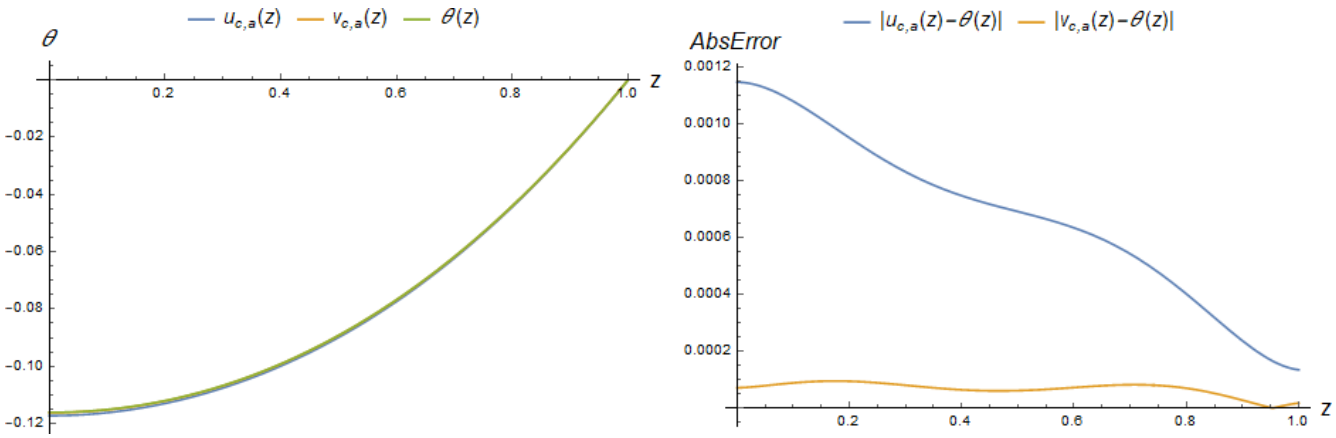

Рисунок 48 – Сравнение точного решения $\theta(z)$ задач (116) и (117) и их приближенных нейросетевых решений $u_{\mathbf{c,a}}(z)$ (135) и $v_{\mathbf{c,a}}(z)$ (135) при фиксированном значении параметра $\delta = 2$ (слева) наряду с абсолютной разницей между точным и приближенным решениями при одном и том же значении параметра (справа)

С другой стороны, PPINN моделям требуется большее количество нейронов скрытого слоя для достижения приемлемой аппроксимации, и это приводит к значительно меньшей точности. Далее приводятся результаты для $n = 10$ и $n = 30$. Точные формулы для этих моделей опущены. При использовании меньшего числа нейронов результаты оказываются явно неудовлетворительными.

На Рисунке 49 показаны аналогичные графики для точек с 30 нейронами, иллюстрирующие значительно улучшенную точность модели. В результате в последующих экспериментах использование сетей с 10 нейронами было прекращено. Для PPINNs точность модели персептрона примерно такая же, как и для радиальных базисных функций.

6.4.1.4 Адаптация моделей PINN и PPINN

Далее были проведены вычислительные эксперименты по адаптации нейросетевых моделей PINN и PPINN, построенных на первом этапе, используя динамически изменяющиеся данные. Для генерации данных использовалась функция (128) с различными параметрами зависимости δ, в зависимости от номера итерации процесса адаптации i.

Время, необходимое для выполнения пяти шагов для минимизации функции потерь (113), обозначалось как $t_{\texttt{adapt}}$. Затем, при каждом номере процесса адаптации $i = t/t_{\texttt{adapt}}$ для каждой зависимости $\delta(t)$, была рассчитана среднеквадратичная ошибка (MSE) для каждой модели $u(z)$ в 100 точках z_k равно-

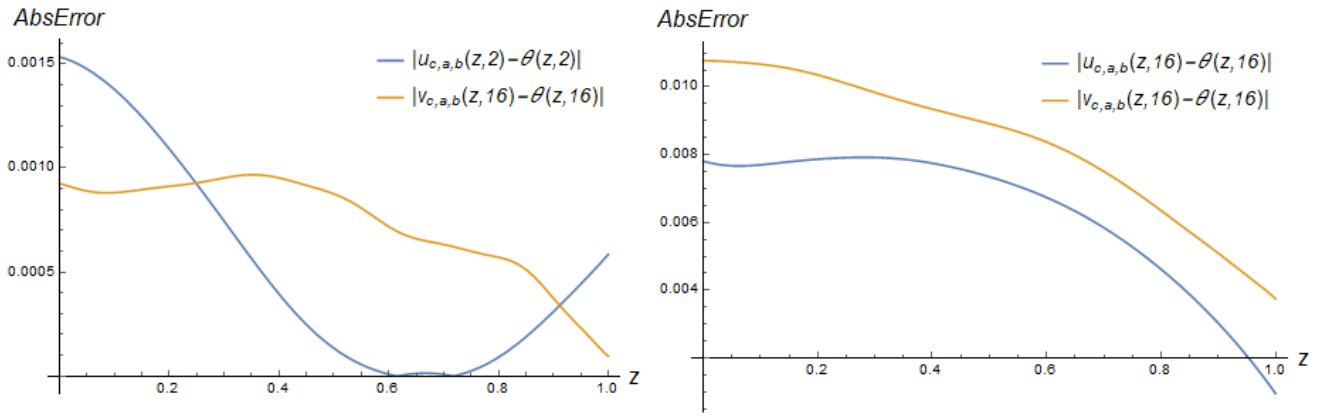

Рисунок 49 – Абсолютные ошибки моделей PPINN (132) $u_{\mathbf{c},\mathbf{a},\mathbf{b}}(z,\delta)$ с базисными функциями (133) и $v_{\mathbf{c},\mathbf{a},\mathbf{b}}(z,\delta)$ с базисными функциями (134) для точного решения $\theta(x,\delta)$ задач (116) и (117) при количестве нейронов $n=30$ и значениях параметра $\delta=2$ (слева) и $\delta=16$ (справа)

мерно распределенных на интервале $[0,1]$

$$\mathrm{MSE}_i(u)=\frac{1}{100}\sum_{k=1}^{100}\left(u(z_k)-\theta(z_k,i)\right)^2, \tag{137}$$

где $\theta(z,i)$ – точное решение системы (116) и (117) при значении параметра $\delta(t)=\delta(it_{\mathtt{adapt}})$ (119) и (120).

В каждой последующей точке i в процессе адаптации к псевдоизмерениям на Рисунке 50 показана взаимосвязь между зависимостью $\delta_1(t,0.1,15,2,4)$ от времени и соответствующей MSE адаптируемых моделей.

На Рисунке 50 также видно, что модель PINN-персептрон демонстрирует существенные преимущества на начальном этапе, но эти преимущества уменьшаются при значительном изменении параметра δ.

На Рисунках 51–53 представлены сопоставимые результаты для зависимостей $\delta_1(t,0.002,15,2,4)$ и $\delta_2(t,0.004,4)$ соответственно, которые подтверждают, что персептроная модель не демонстрирует заметных преимуществ.

Представленные результаты приводят нас к выводу, что ошибка в модели PINN в первую очередь связана с ее адаптивными характеристиками. И наоборот, для моделей PPINN основной источник ошибок может быть связан с неточностями в процессе первоначального обучения. Эти результаты проливают свет на различные источники ошибок в моделях PINN (111) и PPINN (113), подчеркивая важность учета конкретных характеристик и процедур обучения

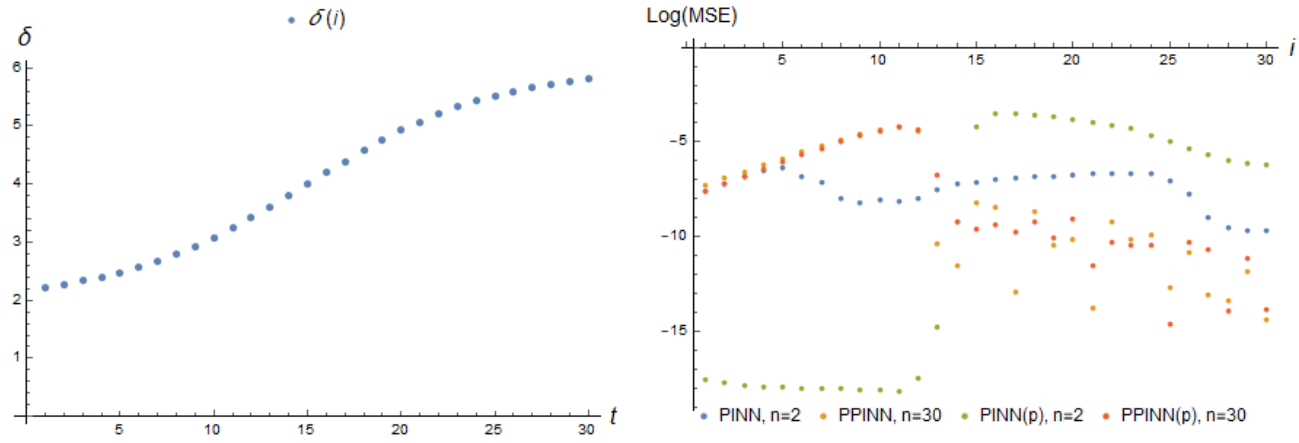

Рисунок 50 – Зависимость значения параметра $\delta(i) = \delta_1(it_{\texttt{adapt}}, 0.1, 1.5, 2, 4)$ (119) и log(MSE) (137) для соответствующей адаптации в реальном времени модели PINN с базисными функциями (130), и модели PINN(p) с базисными функциями (131), для 2 нейронов скрытого слоя и модели PPINN с базисными функциями (133) и PPINN(p) с базисными функциями (134), для 30 нейронов скрытого слоя на каждом итерационном шаге процесса адаптации i для задач (116) и (117)

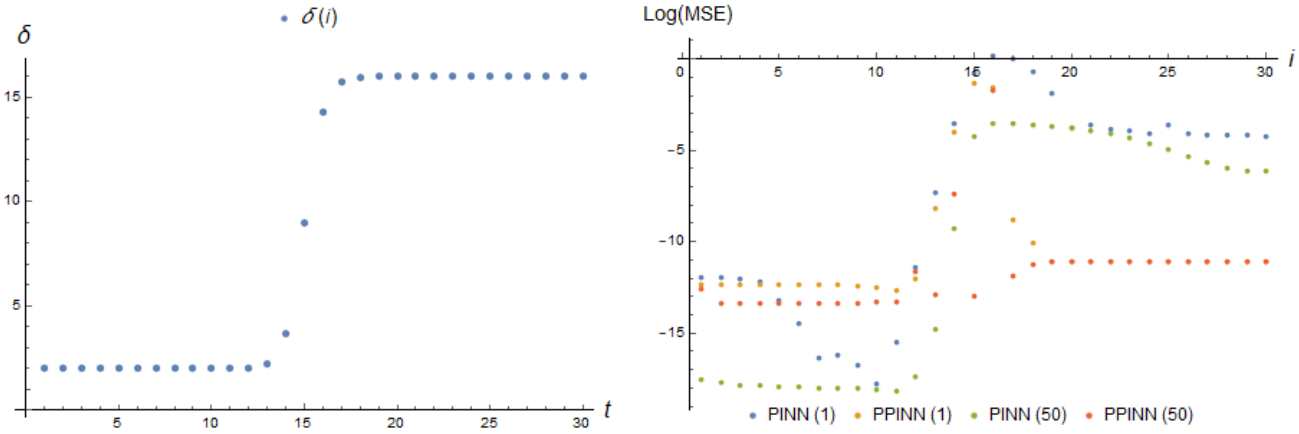

Рисунок 51 – Зависимость значения параметра $\delta(i) = \delta_1(it_{\texttt{adapt}}, 0.1, 15, 2, 4)$ (119) и log(MSE) (137) для соответствующей адаптации в реальном времени модели PINN(1) с базисными функциями (130) для 2 нейронов скрытого слоя; модели PPINN(1) с базисными функциями (133) для 30 нейронов скрытого слоя на каждом итерационном шаге процесса адаптации i; модели PINN(50) с базисными функциями (130) для 2 нейронов скрытого слоя; и модели PPINN(50) с базисными функциями (133) для 30 нейронов скрытого слоя. Данные псевдоизмерений генерируются с помощью функции (128) согласно закону изменения параметра на протяжении 50 итераций обучения между моментами сбора данных для задачи (116) и (117)

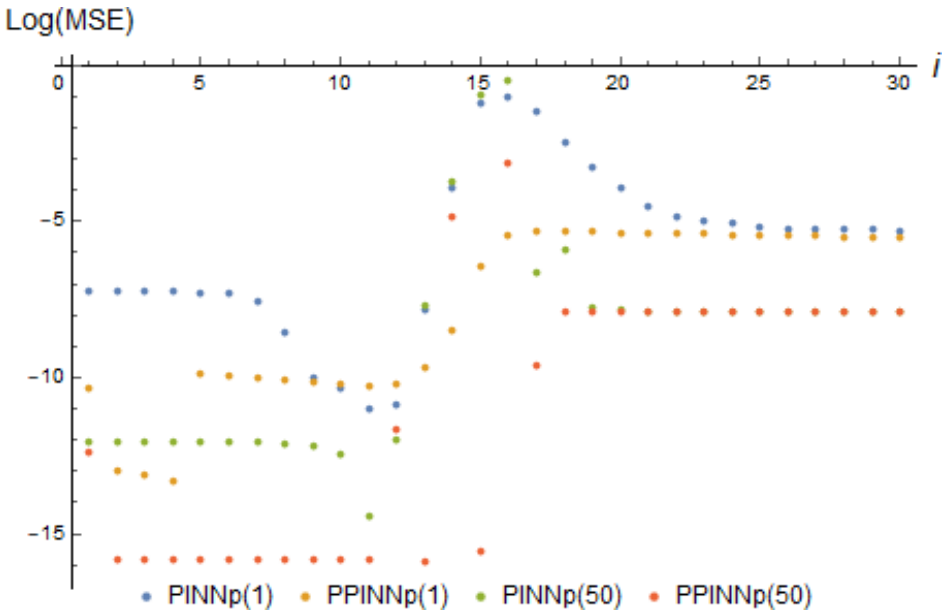

Рисунок 52 – log(MSE) (137) для соответствующей адаптации в реальном времени модели PINNp(1), с базисными функциями (131)для 2 нейронов скрытого слоя; модели PPINNp(1) с базисными функциями (134) для 30 нейронов скрытого слоя на каждом итерационном шаге процесса адаптации i; модели PINNp(50) с базисными функциями (131), для 2 нейронов скрытого слоя ; и модели PPINNp(50) с базисными функциями (134), для 30 нейронов скрытого слоя . Данные псевдоизмерений генерируются с помощью функции (128) согласно закону изменения параметра на протяжении 50 итераций обучения, возникающие между моментами сбора данных для задачи (116) и (117) для зависимости параметра $\delta(i) = \delta_1(it_{\texttt{adapt}}, 0.1, 15, 2, 4)$ (119)

при анализе и улучшении их производительности.

Зависящее от времени поведение $\delta_2(t, 0.004, 4)$ и $\delta_2(t, 0.004, 14)$ имитирует кратковременное отклонение параметра от его стационарного значения. На рисунках 53–55 наблюдается, как зависимости $\delta_2(t, 0.004, 4)$ и $\delta_2(t, 0.004, 14)$ меняются со временем и как это влияет на среднеквадратичную ошибку (MSE) адаптирующихся моделей с различными базовыми функциями.

Рисунок 55 наглядно демонстрирует, что персептрон не обладает никакими преимуществами перед сетью, использующей гауссову базисную функцию. На основании этих результатов модель с персептронными базисными функциями была исключена из последующих экспериментов.

При анализе графиков, представленных на Рисунках 53 и 54, наблюдается резкое увеличение погрешности моделей PINN и PPINN в момент наибольшей

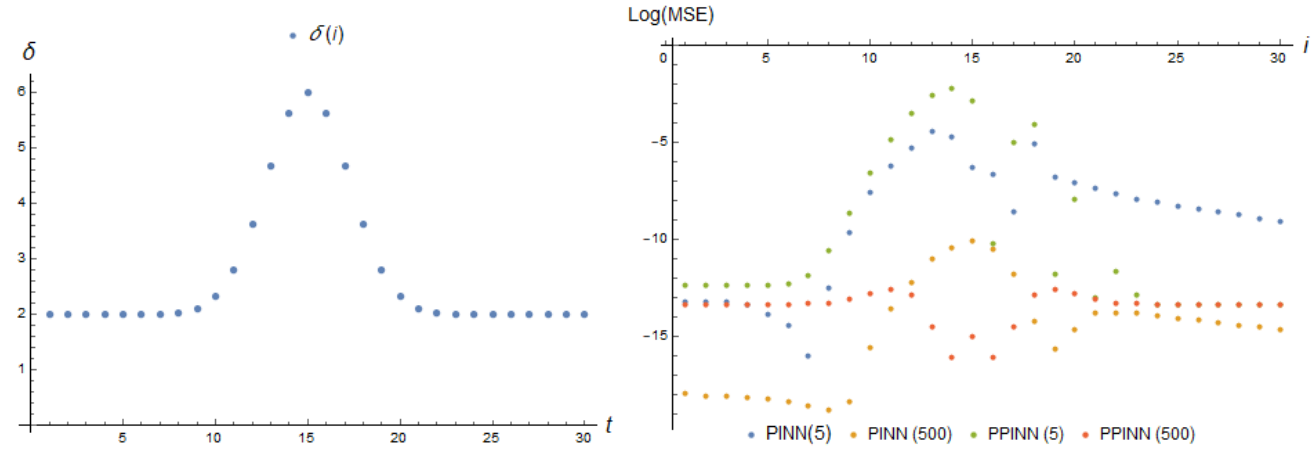

Рисунок 53 – Зависимость параметра $\delta(i) = \delta_2(t, 0.004, 4)$ (120) и log(MSE) (137) для соответствующей адаптации в реальном времени модели PINN с базисными функциями (130) для 2 нейронов скрытого слоя и модели PPINN с базисными функциями (133) для 30 нейронов скрытого слоя на каждом итерационном шаге процесса адаптации i для задач (116) и (117); 5 и 500 итераций процесса обучения между моментами сбора данных

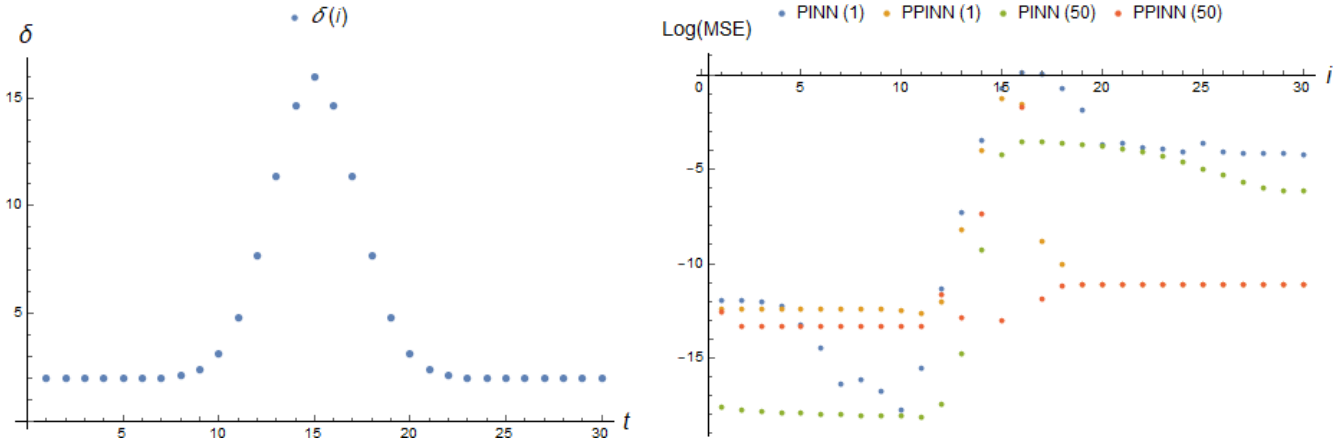

Рисунок 54 – Зависимость параметра $\delta(i) = \delta_2(t, 0.004, 14)$ (120) и log(MSE) (137) для соответствующей адаптации в реальном времени модели PINN с базисными функциями (130) для 2 нейронов скрытого слоя и модели PPINN с базисными функциями (133) для 30 нейронов скрытого слоя на каждом итерационном шаге процесса адаптации i для задач (116) и (117); 5 и 50 итераций процесса обучения между моментами сбора данных

скорости увеличения параметра. Это подтверждает гипотезу, что адаптивные характеристики этих моделей остаются основной причиной их ошибок. Примечательно, что ошибка модели PINN существенно не уменьшается даже при увеличении количества итераций обучения между точками данных, особенно когда параметр существенно увеличивается. Это говорит о том, что основным

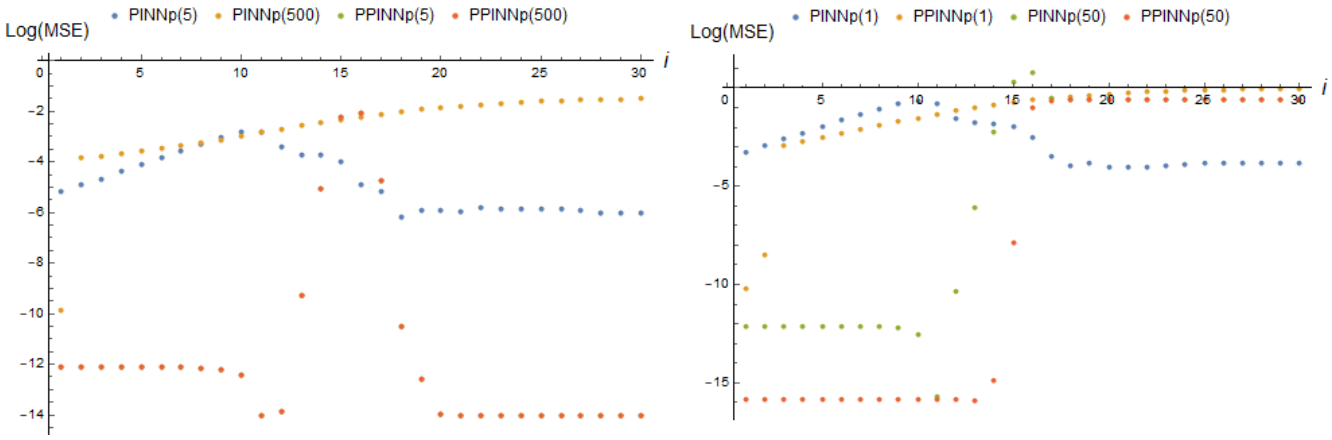

Рисунок 55 – log(MSE) (137) для соответствующей адаптации в реальном времени модели PINNp(1), с базисными функциями (131)для 2 нейронов скрытого слоя и модели PPINNp с базисными функциями (134) для 30 нейронов скрытого слоя на каждом итерационном шаге процесса адаптации i для задач (116) и (117); 5 и 500 итераций процесса обучения между моментами сбора данных, законы зависимости параметров (120) $\delta(i) = \delta_2(t, 0.004, 4)$ (слева), $\delta(i) = \delta_2(t, 0.004, 14)$ (справа)

источником ошибок для моделей PINN (111) является нестабильность процесса обучения при столкновении с резкими изменениями в данных. Стоит отметить, что при значительном увеличении параметра некоторые прогоны модели были неудачными, что указывает на то, что PINN может демонстрировать высокую частоту ошибок, несмотря на длительное обучение. С другой стороны, погрешность модели PPINN существенно уменьшается с увеличением числа итераций обучения между точками данных, особенно в случае значительного увеличения параметра. Следовательно, для моделей PPINN (113) ошибки в основном связаны с неточностями на начальном этапе обучения сети.

6.4.2 Нестационарная задача о тепловом взрыве в плоскопараллельном случае

6.4.2.1 PINN Модель

На первом этапе снова строится решение PINN (111) для задачи (120) при фиксированном значении параметра δ и гауссовскими базисными функциями

$$\varphi(x, \mathbf{a}_i) = \exp\left(-a_{i1}(x - a_{i2})^2\right). \tag{138}$$

Параметры сети $\mathbf{c}$, $\mathbf{a}$ настраиваются в ходе минимизации функции потерь

$$J(\mathbf{c},\mathbf{a}) = \sum_{j=1}^{M_D} \left(\frac{d^2 u_{\mathbf{c},\mathbf{a}}}{dx^2}(x_j) + \delta \exp\left(u_{\mathbf{c},\mathbf{a}}(x_j)\right) \right)^2 + A \left(\left(\frac{du_{\mathbf{c},\mathbf{a}}}{dx}(0) \right)^2 + \left(u_{\mathbf{c},\mathbf{a}}(1)\right)^2 \right). \quad (138)$$

Штрафной параметр A вычисляется во время инициализации сети таким образом, чтобы обеспечить равный вклад каждого из слагаемых функции потерь.

6.4.2.2 PPINN Модель

В качестве приближенной модели PPINN для решения задачи (120) строится нейронная сеть вида (113), которая зависит от параметра δ. В свою очередь, базисная функция имеет вид

$$\varphi(x, \mathbf{a}_i, \delta, \mathbf{b}_i) = \exp\left(-a_{i1}(x - a_{i2})^2\right) \tanh\left(b_{i1}(\delta - b_{i2})\right), \quad (139)$$

а соответствующая функция потерь –

$$J(\mathbf{c},\mathbf{a},\mathbf{b}) = \sum_{j=1}^{M_D} \left[\left(\frac{d^2 u_{\mathbf{c},\mathbf{a},\mathbf{b}}}{dx^2}(x_j,\delta_j) + \delta_j \exp\left(u_{\mathbf{c},\mathbf{a},\mathbf{b}}(x_j,\delta_j)\right) \right)^2 + A \left(\left(\frac{du_{\mathbf{c},\mathbf{a},\mathbf{b}}}{dx}(0,\delta_j) \right)^2 + \left(u_{\mathbf{c},\mathbf{a},\mathbf{b}}(1,\delta_j)\right)^2 \right) \right]. \quad (139)$$

В вычислительных экспериментах для обучения исходных сетей моделей PINN и PPINN было выбрано количество тестовых точек $M = 100$. Начальные инициализации PPINN были обучены с интервалом изменения параметров $[0.2; 0.85]$. Штрафной параметр $A > 0$ снова был вычислен во время инициализации сети для обеспечения равномерного вклада слагаемых в функции потерь.

6.4.2.3 Исходные PINN и PPINN Модели

Непараметрическое PINN решение задачи (120) при фиксированном значении параметра является достаточно точным даже в случае двух нейронов скрытого

слоя. Так, Рисунок 56 иллюстрирует абсолютную погрешность и сравнивает полученное PINN приближение

$$u_{\mathbf{c,a}}(x) = -0.377 + 0.29e^{-0.677(x-0.878)^2} + 0.376e^{-0.603(x+0.538)^2} \tag{140}$$

с точным решением системы (120) при значении параметра $\delta = 0.2$.

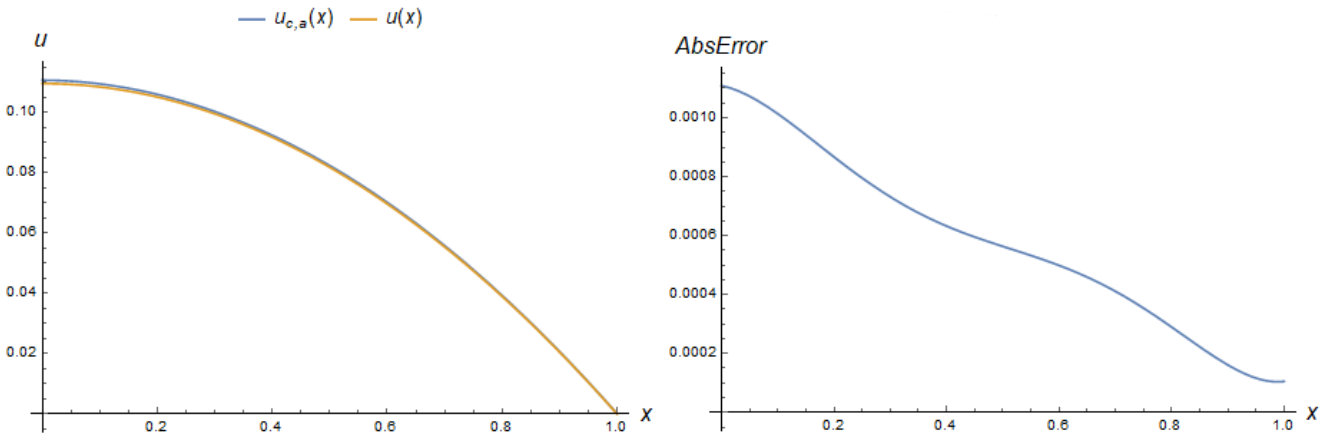

Рисунок 56 – Точное решение $u(x)$ системы (120) при $\delta = 0.2$, его PINN-приближение (111) с двумя нейронами скрытого слоя $u_{\mathbf{c,a}}(x)$ (140), и абсолютная ошибка этого приближения

В случае значения параметра $\delta = 0.8$ было получено решение PINN (111)

$$u_{\mathbf{c,a}}(x) = 1.034 - 2.45e^{-0.483(x-2.34)^2} - 0.496e^{-2.07(x+0.845)^2}, \tag{141}$$

которое имеет абсолютную погрешность, не превышающую 0.0013, что определяет желаемый результат для PINN с двумя нейронами в случае роста параметра δ с 0.2 до 0.8 в процессе адаптации.

Параметрическая модель PPINN (113) требует большего числа нейронов n для приемлемой аппроксимации и дает меньшую точность. Параметр с $n = 10$ нейронами скрытого слоя имеет вид

$$
\begin{aligned}
u_{\mathbf{c,a,b}}(x,\delta) &= -0.416 - 2.48e^{-5.09(x-0.594)^2}\tanh\left(0.047(\delta-1.55)\right) \\
&- 0.354e^{-4.504(x+0.169)^2}\tanh\left(0.771(\delta-0.893)\right) \\
&+ 0.219e^{-7.37(x-0.553)^2}\tanh\left(0.524(\delta-0.689)\right) \\
&+ 0.248e^{-6.55(x+0.133)^2}\tanh\left(0.591(\delta-0.447)\right) \\
&- 2.08e^{-2.43(x-1.42)^2}\tanh\left(0.442(\delta-0.445)\right) \\
&+ 1.76e^{-3.4(x-0.264)^2}\tanh\left(0.21(\delta-0.359)\right) \\
&+ 0.445e^{-4.4(x-1.05)^2}\tanh\left(0.787(\delta-0.045)\right) \\
&+ 3.61e^{-0.848(x+0.106)^2}\tanh\left(0.253(\delta+0.131)\right) \\
&+ 0.096e^{-11.1(x+0.455)^2}\tanh\left(0.983(\delta+0.283)\right). \qquad (141)
\end{aligned}
$$

Формула для полученной модели PPINN с 30 нейронами не приводится из-за ее громоздкости.

На Рисунке 57 представлена абсолютная погрешность и соответсвующее приближенное PPINN решение. Аналогичные результаты были получены для параметрического PINN с 30 нейронами. Значения функции потерь (144) на последнем шаге обучения для этих моделей составили 0.0344 и 0.0280.

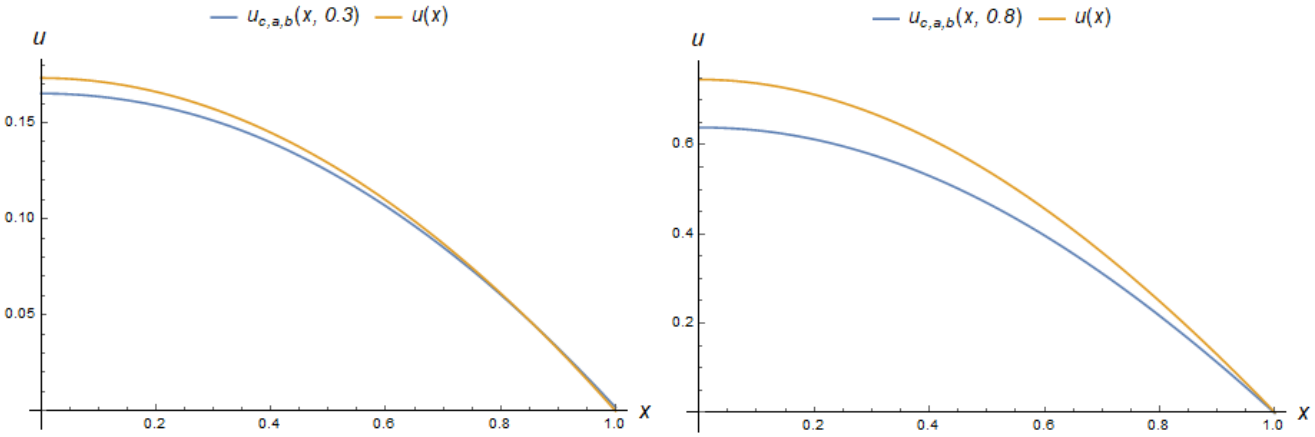

Рисунок 57 – Точное решение $u(x)$ системы (120) и его 10-нейронная PPINN (113) аппроксимация $u_{\mathbf{c,a,b}}(x,\delta)$ (141) при $\delta = 0.3$ (**left**) и $\delta = 0.8$ (**right**)

6.4.2.4 Адаптация построенных PINN и PPINN Моделей

Были проведены численные эксперименты для различных исходных моделей PINN и временных зависимостей параметра $\delta(t)$. В частности, были изучены

начальные модели PINN (140) с 2 нейронами и модели PPINN с 10 (141) и 30 нейронами скрытого слоя, представленные выше.

Аналогично исследованию решений для предыдущих задач, для каждой зависимости $\delta(t)$ и для каждой итерации процесса адаптации $i = t/t_{\texttt{adapt}}$ была вычислена среднеквадратичная ошибка (MSE)

$$\mathrm{MSE}_i(u) = \frac{1}{100}\sum_{k=1}^{100}\left(u(x_k) - v(x_k, i)\right)^2, \tag{142}$$

приближенного решения $u(x)$ в 100 тестовых точках x_k, равномерно распределенных в интервале $[0, 1]$. Здесь, $v(x, i)$ – точное решение задачи (120) при значении параметра $\delta(t) = \delta(it_{\texttt{adapt}})$.

Рисунок 58 иллюстрирует зависимость $\delta_1(t, 0.0045, 0, 0.6, 0.2)$ от времени и соответствующую MSE рассматриваемых адаптированных моделей в последовательных точках итерации i оптимизационного процесса адаптации к псевдоизмерениям. Значение MSE адаптированной модели PINN резко возросло в начальный момент, достигло максимума и быстро снизилось до небольших значений со стабилизацией параметра вблизи нового значения. Погрешность обеих моделей PPINN имела схожее поведение. Значение MSE росло, в то время как $\delta(t)$ стремился к новому стабильному значению. Далее, параметр $\delta(t)$ оставался стабильно выше первоначального значения. Стоит отметить, что несмотря на меньшую разницу в значении функции потерь в исходных моделях, максимальная MSE PPINN с 10 нейронами была примерно в два раза больше, чем у PPINN с 30 нейронами.

Когда скорость изменения параметра $\delta_1(t, \alpha, 0, 0.6, 0.2)$ увеличивалась до $\alpha = 0.1$, процесс адаптации исходной модели PINN – минимизация функции потерь (114) – терял стабильность. Поведение MSE адаптирующегося PPINN аналогично показанному на Рисунке 56 , но с гораздо более ранней стабилизацией ошибки. Ее максимальные значения представлены в Таблице 10.

Следующая зависимость $\delta_3(t, \alpha)$ (122), которая была рассмотрена, имитирует кратковременное отклонение параметра от стационарного значения. На Рисунке 59 показаны результаты для $\delta_3(t, 0.04)$ в зависимости от временного интервала изменения параметра $[0, 150]$. Максимальные значения MSE также представлены в Таблице 10.

Можно видеть, что MSE адаптируемой модели PINN резко возрастает в

Таблица 10 – Максимальные значения ошибки MSE для PINN и PPINN адаптивных решений задачи (124) при различных законах изменения параметра δ

	$\delta_1(t, \alpha, 0, 0.6, 0.2)$		$\delta_3(t, \alpha)$		$\delta_1(t, \alpha, 2, 0.3, 0.5)$		$\delta_4(t)$
Модель, n	$\alpha = 0.0045$	$\alpha = 0.1$	$\alpha = 0.003$	$\alpha = 0.03$	$\alpha = 0.0025$	$\alpha = 0.003$	–
PINN, 2	$14 \cdot 10^{-6}$	–	$19 \cdot 10^{-5}$	$30 \cdot 10^{-5}$	$10 \cdot 10^{-5}$	$33 \cdot 10^{-3}$	$35 \cdot 10^{-5}$
PPINN, 10	$35 \cdot 10^{-6}$	$35 \cdot 10^{-6}$	$33 \cdot 10^{-5}$	$26 \cdot 10^{-5}$	$20 \cdot 10^{-6}$	$23 \cdot 10^{-6}$	$33 \cdot 10^{-5}$
PPINN, 30	$8.8 \cdot 10^{-6}$	$8.8 \cdot 10^{-6}$	$23 \cdot 10^{-6}$	$25 \cdot 10^{-6}$	$5.3 \cdot 10^{-6}$	$5.7 \cdot 10^{-6}$	$23 \cdot 10^{-6}$

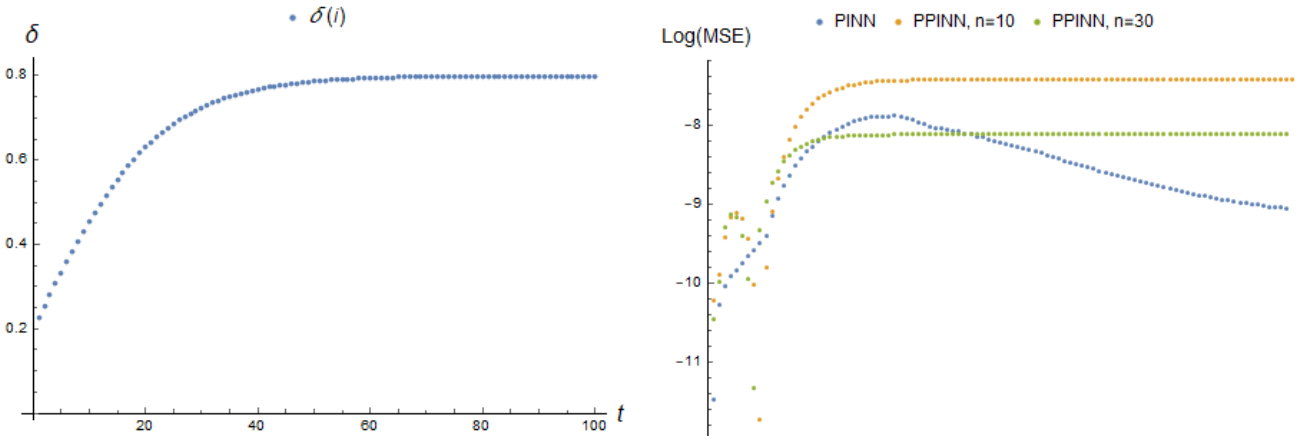

Рисунок 58 – Изменение параметра $\delta(i) = \delta_1(it_{\texttt{adapt}}, 0.0045, 0, 0.6, 0.2)$ и Log[MSE] (142) соответствующих моделей PINN с 2 нейронами и PPINN с 10 и 30 нейронами скрытого слоя, адаптирующихся в реальном времени, на каждом шаге адаптации i

момент наибольшей скорости изменения параметра δ. Еще одна максимальная ошибка достигается, когда значение параметра близко к критическому $\delta^* \approx 0.878458$. MSE адаптирующейся модели PPINN теряет свою стабильность, когда параметр начинает стремиться к δ^*. Максимальная ошибка для PPINN с 10 нейронами скрытого слоя более чем на порядок превышает ошибку для PPINN с 30 нейронами.

Дополнительно были изучены адаптивные свойства моделей для таких закона изменения параметра $\delta_1(t, \alpha, 2, 0.3, 0.5)$ (123). Таким образом, была рассматрена ситуацию, когда момент существенного изменения параметра наступает через некоторое время с момента запуска адаптивного алгоритма (см. Рисунок 46), и проанализированы различные скорости этого изменения. Поведение

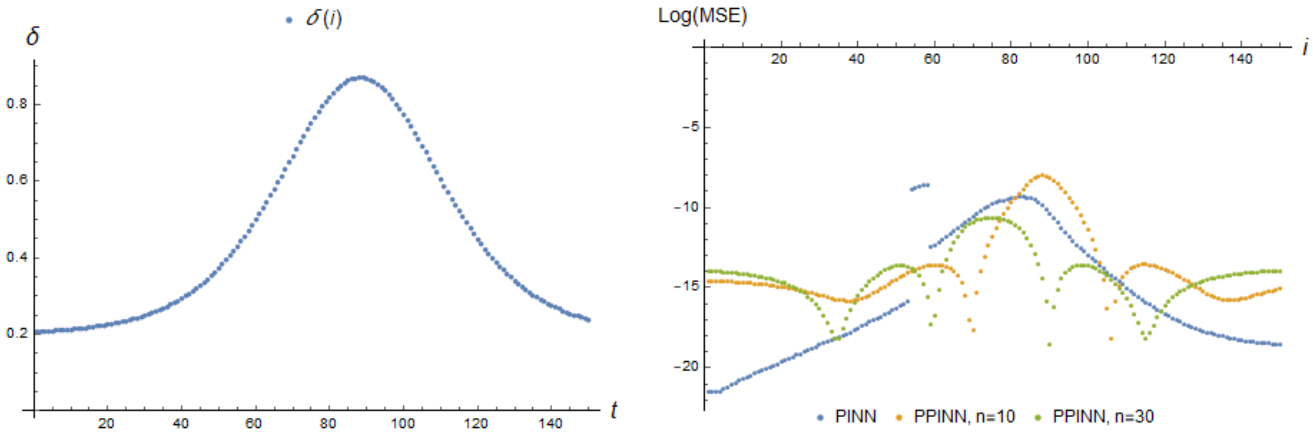

Рисунок 59 – Зависимость изменения параметра $\delta(i) = \delta_3(it_{\texttt{adapt}}, 0.04)$ и Log[MSE] (142) соответствующих моделей PINN с 2 нейронами и PPINN с 10 и 30 нейронами скрытого слоя, адаптирующихся в реальном времени, на каждом шаге адаптации i

MSE адаптирующейся модели PPINN аналогично показанному на Рисунке 57, но со сдвигом максимальной ошибки к правому концу исследуемого временного интервала. Максимальные ошибки также представлены в Таблице 10. Даже небольшое увеличение скорости изменения параметра с 0.025 до 0.03 влияет на точность адаптируемой модели PINN.

Последняя зависимость $\delta_4(t)$ (124) представляет собой скачок в начальный момент времени в окрестности критического значения параметра δ^*. Важно отменить, что значение параметра в этом случае выходит за пределы интервала, для которого были обучены PPINN. На Рисунке 60 представлена MSE адаптируемых моделей PINN и PPINN. Максимальные значения ошибок можно найти в Таблице 10. MSE модели PINN принимает наибольшее значение в начальный момент, когда параметр резко изменяется, и резко уменьшается, когда значение параметра стабильно. Погрешность моделей PPINN изменяется незначительно, что соответствует значениям параметра в окрестности δ^*. MSE для сети с меньшим числом нейронов более чем на порядок превышает ошибку для PPINN с 30 нейронами скрытого слоя.

6.4.3 Задача с уравнением смешанного типа

6.4.3.1 PINN Модель

Для построения PINN решения (111) для задачи (124) при фиксированном δ, в качестве функции активации была использована двумерная гауссовская функ-

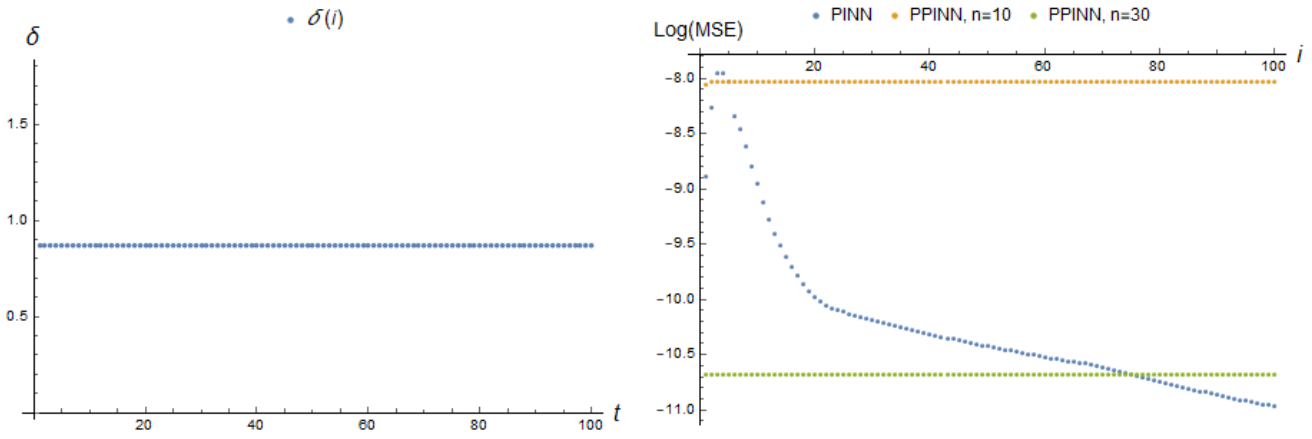

Рисунок 60 – Зависимость изменения параметра $\delta(i) = \delta_4(it_{\texttt{adapt}}) = 0.87$ и Log[MSE] (142) соответствующих моделей PINN с 2 нейронами и PPINN с 10 и 30 нейронами скрытого слоя, адаптирующихся в реальном времени, на каждом шаге адаптации i

ция

$$\varphi(x, y, \mathbf{a}_i) = \exp\left(-a_{i1}\left((x - a_{i2})^2 + (y - a_{i3})^2\right)\right). \tag{143}$$

Параметры сети $\mathbf{c}$ и $\mathbf{a}$ подбирались в ходе минимизации функции потерь

$$J(\mathbf{c}, \mathbf{a}) = \sum_{j=1}^{M_D} \left(\frac{\partial^2 u_{\mathbf{c},\mathbf{a}}}{\partial x^2}(x_j, y_j) + \delta \frac{\partial^2 u_{\mathbf{c},\mathbf{a}}(x_j, y_j)}{\partial y^2} \right)^2$$
$$+A \sum_{j=1}^{M_B} \Bigg(\left(u_{\mathbf{c},\mathbf{a}}(\bar{x}_j, 0) - \delta \bar{x}_j^2\right)^2 + \left(u_{\mathbf{c},\mathbf{a}}(\bar{x}_j, 1) - \delta \bar{x}_j^2 + 1\right)^2$$
$$+ \left(u_{\mathbf{c},\mathbf{a}}(0, \bar{y}_j) + \bar{y}_j^2\right)^2 + \left(u_{\mathbf{c},\mathbf{a}}(1, \bar{y}_j) - \delta + \bar{y}_j^2\right)^2 \Bigg) \tag{143}$$

Штрафной множитель A, как и прежде, вычислялся как уравновешивающий вклад каждого слагаемого функционала ошибки при инициализации сети.

6.4.3.2 PPINN Модель

В качестве приближенного решения PPINN мы строим решение PINN, зависящее от параметра δ. В качестве базисной функции была выбрана после некоторых экспериментов функция вида

$$\varphi(x, \mathbf{a}_i, \delta, \mathbf{b}_i) = \exp\left(-a_{i1}(x - a_{i2})^2\right) \tanh\left(b_{i1}(\delta - b_{i2})\right). \tag{144}$$

Для решения данной задачи были рассмотрены различные функции, в том числе произведение гиперболических тангенсов и произведение гауссиан. Было обнаружено, что вариант (144) дает наилучшие результаты. Аналогичные наблюдения были сделаны при решении других задач, где гауссовы функции оказались более подходящими для пространственных переменных (при отсутствии резких изменений решения), в то время как гиперболические тангенсы были более эффективны для параметризации.

Соответствующая функция потерь имеет вид

$$J(\mathbf{c},\mathbf{a},\mathbf{b}) = \sum_{j=1}^{M_D}\left(\frac{\partial^2 u_{\mathbf{c},\mathbf{a},\mathbf{b}}}{\partial x^2}(x_j,y_j,\delta_j) + \delta_j\frac{\partial^2 u_{\mathbf{c},\mathbf{a},\mathbf{b}}(x_j,y_j,\delta_j)}{\partial y^2}\right)^2 + \gamma\sum_{j=1}^{M_B}\left(\left(u_{\mathbf{c},\mathbf{a}}(1,\bar{y}_j,\delta_j) - \delta_j + \bar{y}_j^2\right)^2\right) + A\sum_{j=1}^{M_B}\left(\left(u_{\mathbf{c},\mathbf{a}}(\bar{x}_j,0,\delta_j) - \delta_j\bar{x}_j^2\right)^2 + \left(u_{\mathbf{c},\mathbf{a}}(\bar{x}_j,1,\delta_j) - \delta_j\bar{x}_j^2 + 1\right)^2 + \left(u_{\mathbf{c},\mathbf{a}}(0,\bar{y}_j,\delta_j) + \bar{y}_j^2\right)^2\right) \tag{144}$$

Штрафной множитель A, аналогично предыдущим случаям, определялся путем балансировки вклада от каждого слагаемого функции потерь во время начальной настройки сети.

6.4.3.3 Начальные модели PINN и PPINN

Вычислительный эксперимент для задачи (124), как и для предыдущих сслучаев, был проведен в два этапа. На первом этапе PINN и PPINN были обучены без учета динамически изменяющихся данных для получения исходных моделей. Так, при $\delta = 1$ PINN с четырьмя нейронами скрытого слоя имеет вид

$$\begin{aligned} u_{\mathbf{c},\mathbf{a}}(x,y) &= -39.3 - 1.03e^{-1.92\left((x-0.089)^2+(y-1.71)^2\right)} \\ &+ 0.5e^{0.629\left((x+0.058)^2+(y-0.439)^2\right)} \\ &+ 2.94e^{-0.781\left((x-1.82)^2+(y-0.314)^2\right)} \\ &+ 38.6e^{-0.016\left((x+0.355)^2+(y+0.016)^2\right)} \end{aligned} \tag{144}$$

Для этой моделт имеет место довольно небольшая ошибка по всей области $\Omega = [0,1] \times [0,1]$. Абсолютная разница между точным решением и PINN (144) не превышает 0.02. Рисунок 61 иллюстрирует их поведение на диагонали и границе Ω.

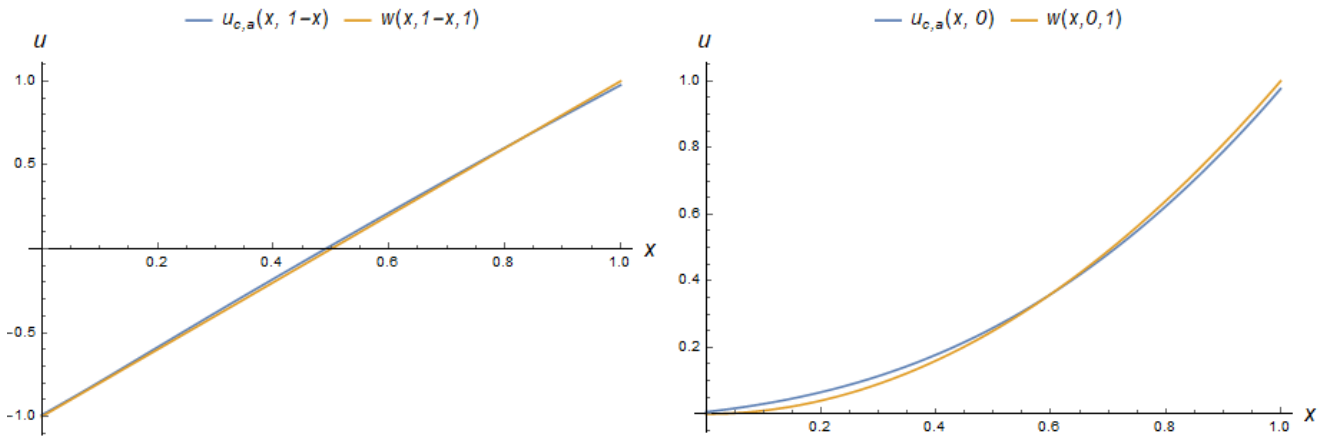

Рисунок 61 – Точное решение $w(x, y, \delta)$ системы (124) и его 4-нейронная аппроксимация $u_{\mathbf{c,a}}(x, y)$ (144) типа PINN (111) при $\delta = 1$ на диагонали $y = 1 - x$ (слева) и границе $y = 0$ (справа) области Ω

Для PPINN (113) требуется гораздо большее количество нейронов. Рассмотрим результаты для модели PPINN с числом нейронов $n = 50$. Сеть была обучена на интервале изменения параметров $[-1.2, 1.2]$. Точная формула для этой модели не приводится из-за ее громоздкости. Абсолютная разница между точным решением и полученной моделью PPINN не превышает 0,02. Среднеквадратичная ошибка для 10 000 случайных точек в области $0 \leq x, y \leq 1$, $-1 \leq \delta \leq 1$ составила 0.0331. На Рисунке 62 сравнивается модель PPINN с точными решениями для различных значений δ.

6.4.3.4 Адаптация PINN и PPINN Моделей

Далее, как и для предыдущей задачи, были исследованы адаптационные свойства начальных моделей PINN и PPINNs, рассмотренных выше, в случае различных временных зависимостей параметра $\delta(t)$. Как и прежде, на каждом этапе процесса адаптации $i = t/t_{adapt}$ для каждой зависимости $\delta(t)$ вычислялись MSE для каждого приближенного решения $u(x)$ в 10000 точках (x_k, y_k)

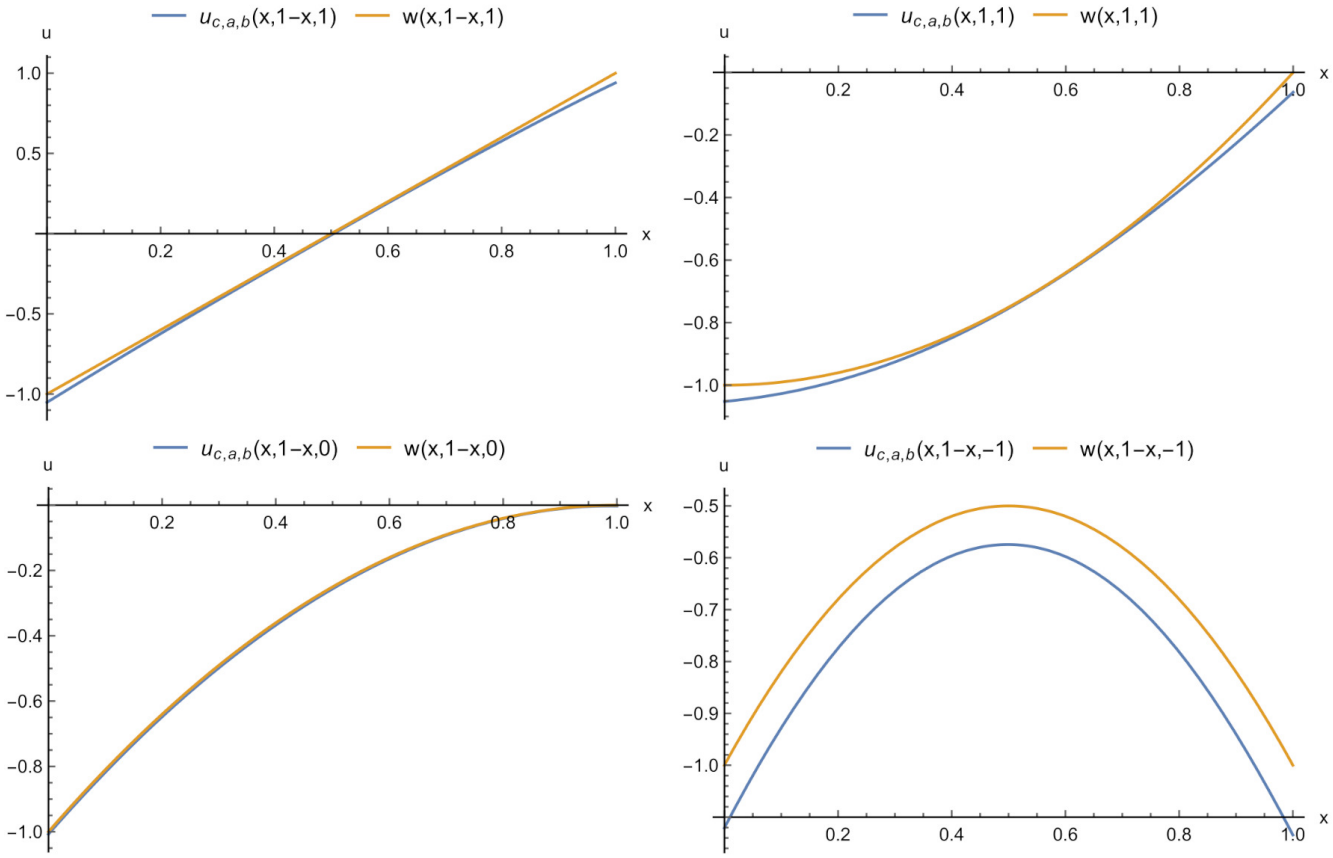

Рисунок 62 – Точное решение $w(x, y, \delta)$ системы (124) и его 50-нейронная аппроксимация $u_{\mathbf{c,a,b}}(x, y, \delta)$ типа (113) при $\delta = 1$ на диагонали $y = 1 - x$ (слева вверху) и границе $y = 1$ (справа вверху) области Ω, на диагонали $y = 1 - x$ области Ω при $\delta = 0$ (слева внизу) и при $\delta = -1$ (справа внизу)

равномерно распределенных в области $[0, 1] \times [0, 1]$

$$\mathrm{MSE}_i(u) = \frac{1}{10,000} \sum_{k=1}^{10,000} \left(u(x_k, y_k) - w(x_k, y_k, \delta(it_{adapt}))\right)^2, \qquad (145)$$

где $w(x, y, \delta)$ – точное решение системы (124).

Первые варианты временной зависимости параметра моделируют кратковременное отклонение параметра от стационарного значения с разной скоростью. Для зависимости $\delta(t) = 1 - 2\mathrm{sech}^2(0.003t + 2)$ результаты представлены на Рисунке 63.

Таблица 11 содержит данные о максимальном значении MSE. Для приближенного решения PPINN оно более чем на порядок превышает погрешность адаптивной модели PINN, причем для отрицательных значений параметра это превышение наблюдается во всех точках.

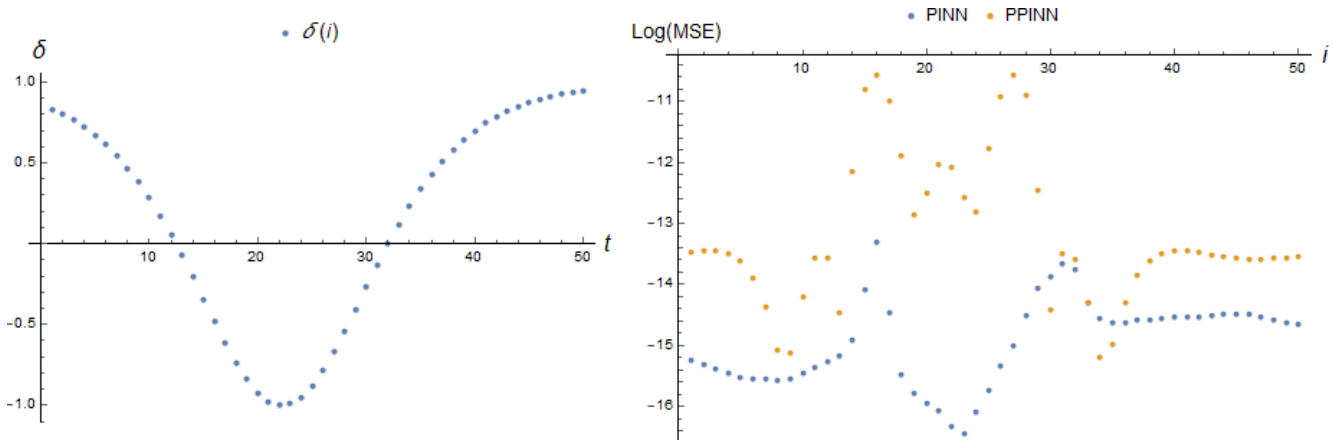

Рисунок 63 – Зависимость изменения параметра $\delta(i) = \delta_5(it_{adapt}, 0.003, 2)$ (126) и Log[MSE] (145)соответствующих моделей PINN с 4 нейронами и PPINN с 50 нейронами скрытого слоя, адаптирующихся в реальном времени, на каждом шаге адаптации i

Таблица 11 – Максимальные значения ошибки MSE для адаптивных решений типа PINN и PPINN для второй задачи и различных зависимостей параметра от времени.

	$\delta_5(t,\alpha,\beta)$		$\delta_6(t,\alpha,\beta)$		$\delta_6(t,\alpha,\beta)$
Model, n	$\alpha = 0.003,$ $\beta = 2$	$\alpha = 0.03,$ $\beta = 20$	$\alpha = 0.01,$ $\beta = 6$	$\alpha = 0.1,$ $\beta = 60$	$\alpha = 0.01,$ $\beta = 2$
PINN, 4	$1.67 \cdot 10^{-6}$	$28.4 \cdot 10^{-6}$	$1.24 \cdot 10^{-6}$	$23.2 \cdot 10^{-6}$	$9.34 \cdot 10^{-4}$
PPINN, 50	$25.8 \cdot 10^{-6}$	$11.2 \cdot 10^{-6}$	$20.2 \cdot 10^{-6}$	$6.21 \cdot 10^{-6}$	$2.48 \cdot 10^{-4}$

Далее были изменены скорость и временная амплитуда отклонения параметра. В случае $\alpha = 0,03$ и $\beta = 20$ для одной и той же зависимости $\delta_5(t,\alpha,\beta)$ наблюдалось, что ошибки обеих адаптивных решений PINN ии PPINN максимальны одновременно с максимальным отклонением параметра от стационарного значения и уменьшаются резко, когда значение параметра стремится к стабильному. Это показано на Рисунке 64. При этом, максимальное значение MSE для PINN более чем в два раза превышает значение ошибки для адаптивного решения PPINN. Конкретные значения приведены в Таблице 11.

Следующие две зависимости имеют сигмоидную форму со сдвигом по времени. Таким образом рассматривается ситуация, когда в процессе работы ал-

горитма происходит переход из одного стационарного состояния в другое. Результаты представлены в таблице 11 и проиллюстрированы на рисунках 65 и 66.

В случае зависимости от параметра $\delta(t) = -\tanh(0.01t - 6)$ (127) значение MSE для PPINN достигает своего максимума одновременно со скоростью изменения параметра. Более того, оно увеличивается в течение всего процесса расчета. Отклонение в MSE для PINN уменьшается в течение всего процесса расчета, за исключением небольшого участка, где им можно пренебречь. MSE для решения PPINN более чем на порядок больше, чем в адаптивной модели PINN.

Более высокая скорость изменения параметров (при сравнении Рисунков 65 и 66) приводит к другим результатам. Для $\delta(t) = -\tanh(0.1t - 60)$ (127) видно, что в момент максимальной скорости изменения параметра MSE для модели PINN имеет один выброс и остается стабильным и небольшим в остальное время. Поведение MSE для PPINN также стабильно почти везде, за исключением небольшого сдвига в момент максимальной скорости изменения параметра. Важно отметить, что максимальная ошибка MSE для PINN примерно в три раза превышает ошибку решения PPINN.

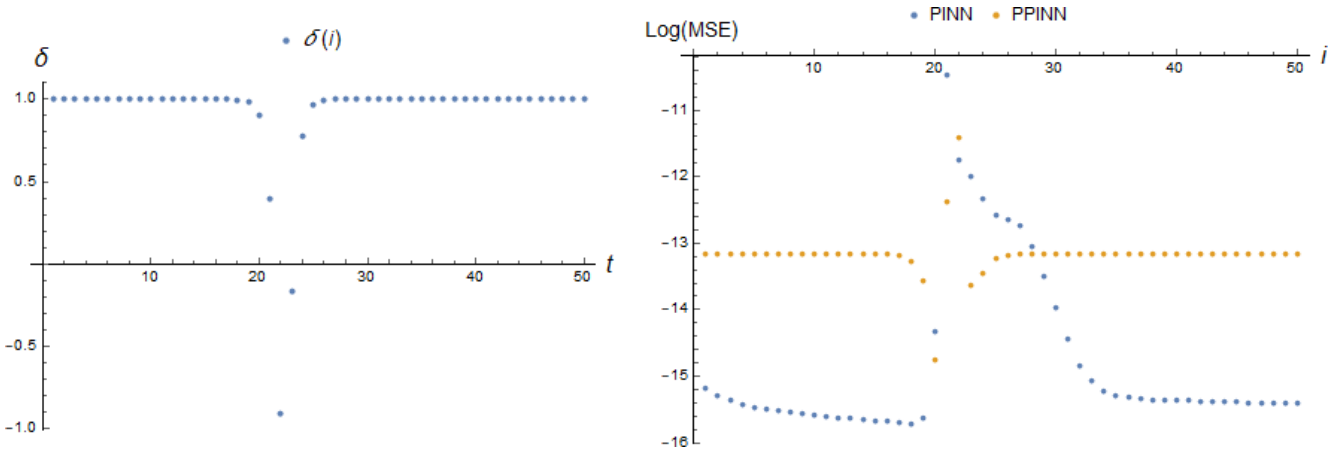

Рисунок 64 – Зависимость изменения параметра $\delta_5(i) = \delta_5(it_{adapt}, 0.03, 20)$ (126) и Log[MSE] (145) соответствующих моделей PINN с 4 нейронами и PPINN с 50 нейронами скрытого слоя, адаптирующихся в реальном времени, на каждом шаге адаптации i

6.4.3.5 Исследование случая с малым числом точек измерений

Очевидно, что большое количество измерений при стабильном поведении параметра позволяет легко минимизировать функцию потерь (139) для адаптивной

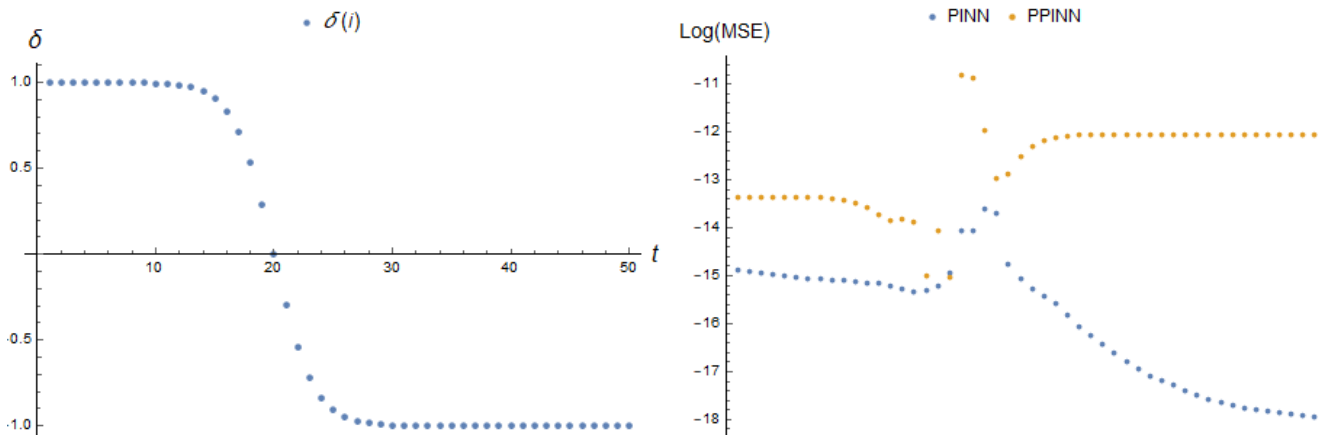

Рисунок 65 – Зависимость изменения параметра $\delta_6(i) = \delta_6(it_{adapt}, 0.01, 6)$ (127) и Log[MSE] (145) соответствующих моделей PINN с 4 нейронами и PPINN с 50 нейронами скрытого слоя, адаптирующихся в реальном времени, на каждом шаге адаптации i

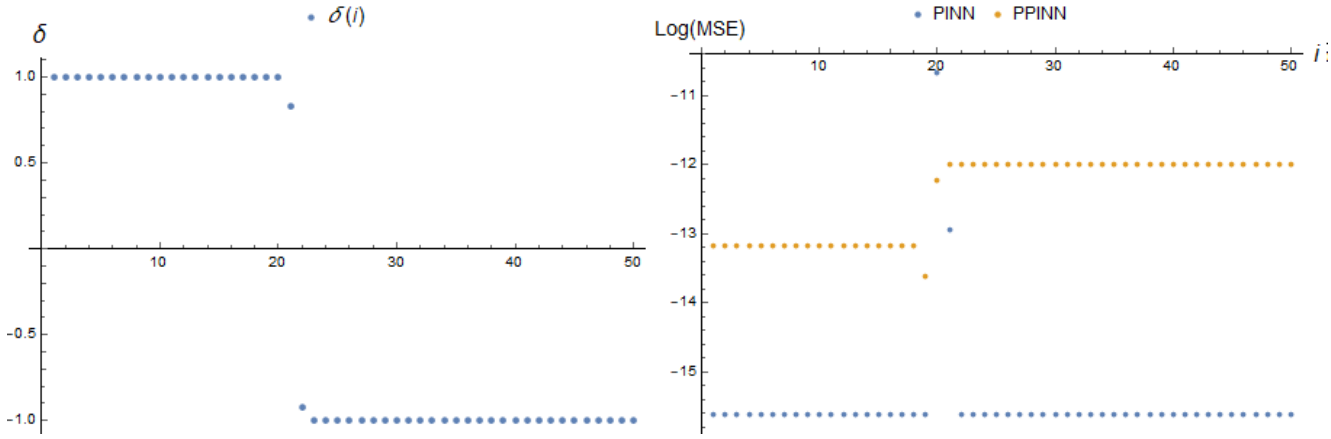

Рисунок 66 – Зависимость изменения параметра $\delta(i) = \delta_6(it_{adapt}, 0.1, 60)$ (127) и Log[MSE] (145) соответствующих моделей PINN с 4 нейронами и PPINN с 50 нейронами скрытого слоя, адаптирующихся в реальном времени, на каждом шаге адаптации i

модели PINN, в то время как PPINN может переобучиться, поскольку в процессе оптимизации на большом количестве данны хиспользуется только один параметр адаптируемой модели. Таким образом, естественно рассмотреть случай, когда в каждый момент времени проводится небольшое количество измерений. Пусть это количество измерений равно $M = 4$, более того, в качестве конкретных точек взяты $\{x_j, y_j\}_{j=1}^4 = \{(1/3, 1/3); (2/3, 1/3); (2/3, 2/3); (1/3, 2/3)\}$.

В качестве функции изменения параметра рассматривается та, где после довольно плавного перехода от одного типа уравнения к другому, $\delta(t)$ некоторое время остается стабильным, $\delta_6(t, 0.01, 2) = -\tanh(0.01t - 2)$. Если сравнить прогнозируемые значения для решения в четырех выбранных точках, полу-

чится, что максимальная абсолютная погрешность решения PINN составляет 1.5×10^{-8} секунд, а для PPINN максимальная погрешность составляет 0,009. Эти результаты могут создать впечатление, что PINN (111) работает намного точнее, чем PPINN (113), хотя это не так. Значения MSE после стабилизации параметра δ для адаптивных моделей PINN и PPINN приведены в последнем столбце таблицы 11, а соответствующие решения показаны на Рисунке 67.

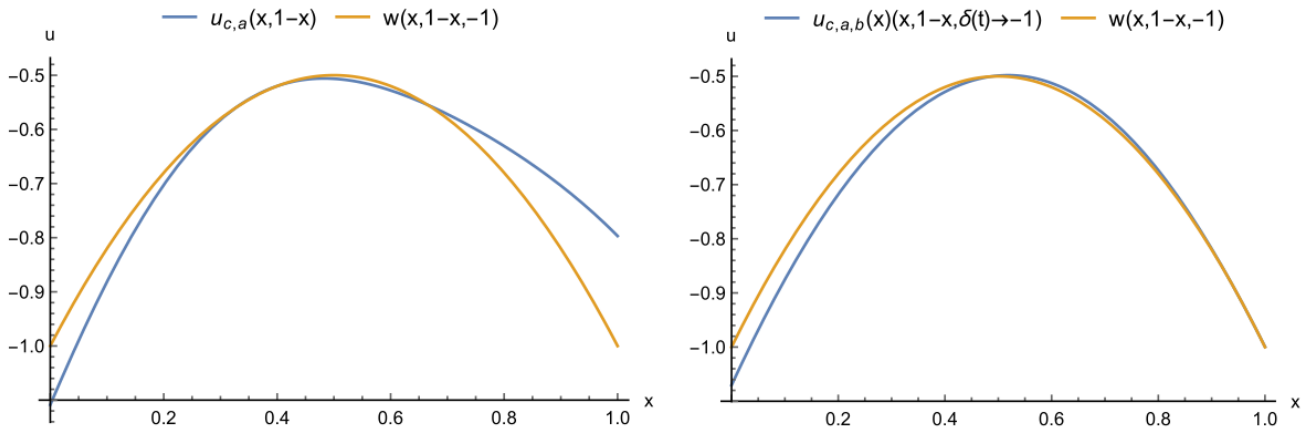

Рисунок 67 – Точное решение $w(x, y, \delta)$ системы (124) и его 4-нейронная аппроксимация $u_{\mathbf{c},\mathbf{a}}(x, y)$ (144) типа PINN (111) (слева) and 50-нейронная аппроксимация $u_{\mathbf{c},\mathbf{a},\mathbf{b}}(x, y, \delta)$ типа PPINN (113) (справа) на диагонали $y = 1 - x$ области Ω, после стабилизации значения параметра $\delta(t) \approx -1$. Адаптация с использованием малого числа измерений $(M = 4)$

Таким образом, ошибка адаптивной модели PINN с небольшим количеством точек, в которых в процессе адаптации берутся значения требуемой функции, плохо характеризуется функцией потерь (114). Для объективной оценки ошибки адаптации необходимо отслеживать значения ошибок в дополнительных тестовых точках. Ошибка адаптивного решения PPINN в том же случае довольно хорошо характеризуется потерей (115).

6.5 Выводы

В данной главе были изучены свойства адаптации решений PINN и PPINN к динамических данным в режиме реального времени. В этом исследовании впервые было проведено сравнение способности параметрической и классической физически-информированных нейросетевых моделей к адаптации. Это сравнение было проведено с использованием примеров задач, в которых исполь-

зовались различные функции зависимости неизвестного динамического параметра. Была проведена серия экспериментов для трех контрольных задач с различными функциями параметра, отражающими характеристику изменений в объекте. Первая задача заключалась в моделировании изгиба консольного стержня при различных нагрузках. Во второй задаче исходная модель была построена на основе обыкновенного дифференциального уравнения, в то время как на самом деле объект моделирования удовлетворяет уравнению в частных производных, согласно которому были сгенерированы соответствующие псевдоизмерения. Третья задача состояла в том, чтобы решить дифференциальное уравнение в частных производных смешанного типа в зависимости от времени.

Важной особенностью исследования является то, что исходные модели были построены с использованием статических дифференциальных математических моделей. Динамические уравнения использовались исключительно для целей генерации данных. Основываясь на результатах вычислительных экспериментов, сделаны следующие выводы. Оба типа моделей, непараметрические и параметрические, обладают свойством адаптивности. Ошибка PINN обусловлена в первую очередь ее адаптивными характеристиками. Наибольшие ошибки связаны с высокой скоростью изменения параметра или резким скачком его значения. Аналогичное поведение MSE для модели PPINN позволяет предположить, что основным источником ошибок является начальное обучение. Больший размер PPINN позволяет лучше отражать тонкие особенности решения задачи на протяжении всего интервала изменения параметров по сравнению с сетью с меньшим количеством нейронов.

В то же время при фиксированном параметре при решении дифференциальной задачи наилучший результат показывает простая PINN и кроме того требует небольшого количества нейронов скрытого слоя. PPINN требует большого количества нейронов, что увеличивает время предварительного обучения. С другой стороны, в процессе адаптации корректируется только один параметр, что требует меньших вычислительных затрат.

Большое количество измерений при стабильном изменении параметра позволяет легко минимизировать функцию потерь в случае адаптивной модели PINN, в то время как модель PPINN в ходе такой адаптации подвержена переобучению, поскольку в процессе оптимизации только один параметр этого решения корректируется с учетом большого количества данных. Если количество

измерений невелико, наилучший результат достигается с помощью адаптивной модели PPINN.

Таким образом, рекомендуется использование адаптивных PINN моделей в ситуации, когда имеется большое количество точек измерения и параметры меняются с низкой скоростью. В то же время точность предварительного обучения сети не имеет большого значения. Предпочтение адаптивным моделям PPINN следует отдавать в тех случаях, когда параметры задачи могут резко изменяться или когда количество точек измерения невелико. Однако требуется тщательный подход к процессу предварительного обучения с использованием сетей с достаточным числом нейронов.

Глава 7.

Применение методов методов построения моделей с архитектурой, основанной на физике для уравнений в частных производных

7.1 Общая постановка задачи и построение моделей

В данной главе рассматривается построение моделей, основанных на аналитической модификации классических аналитических методов и представленных в параграфе 3.3, для объектов, описываемых дифференциальными уравнениями в частных производных.

При решении уравнений в частных производных, проблема громоздкости и длительности процесса обучения приближенных решений, полученных с помощью общего нейросетевой подхода и других методов остается актуальной, как и в случае задач с обыкновенными дифференциальными уравнениями.

Рассмотрим начально-краевую задачу для уравнения в частных производных вида

$$\begin{cases} \dfrac{\partial}{\partial t}\mathbf{y}(x,t) = \mathscr{G}(\mathbf{y}(x,t),x,t), \\ \mathbf{y}(x,0) = \varphi(x), \end{cases} (x,t) \in \mathbb{R}^p \times \mathbb{R}^+; \tag{146}$$

где $\mathscr{G}$ – некоторый линейный оператор, например, дифференциальный. В качестве функции $\varphi(x)$ может рассматриваться как некоторая заданная функция, так и ее нейросетевое приближение, если использование исходной функции напрямую нецелесообразно или ее требуется вычислить дополнительно при решении некоторой связанной задачи, например, путем интерполяции данных измерений.

Аналогично главе 3 приближенное решение для решения задачи (146)

строится для промежутка $[0, t]$ с переменным правым концом. Интервал разбивается точками t_k на интервалы длины h_k, $k = 1, \ldots, n$, затем применяется некоторая рекуррентная формула

$$\mathbf{u}_{k+1}(x) = \mathscr{F}(\mathscr{G}(\mathbf{u}_{k+1}(x), x, t_{k+1}), h_k, t_k, \mathbf{u}_{k+1}(x), \mathbf{u}_k(x), \ldots), \qquad (147)$$

где оператор $\mathscr{F}$ определяет конкретный численный метод. Формула применяется итеративно n раз к интервалу с переменным правым концом $[0, t] \subset [0, +\infty]$, возникает аналитически заданная функция $\mathbf{u}(x, t) = \mathbf{u}_n(x, t)$, которую можно рассматривать в качестве приближенного решения уравнения (146). Важно подчеркнуть, что зависимость от t имеют и такие параметры, как длина промежутка разбиения $h_k = h_k(t)$ и промежуточные выражения $\mathbf{u}_k(x) = \mathbf{u}_k(x, t)$.

Для построения такого приближенного решения может выбрать любой из классических численных методов, таких как метод Рунге-Кутта, Эйлера и другие [62].

В случае неявных методов отдельной задачей является решение уравнения (147) относительно $\mathbf{u}_{k+1}(x)$. После применения подходящего метода решения неявного уравнения возникает рекуррентная явная формула вида $\mathbf{u}_{k+1}(x) = \mathscr{H}(\mathbf{u}_k(x), \ldots, x, t)$, которую можно расмматривать как процесс перехода от одного уровня к другому в некоторой нейросетевой структуре. При этом ее сходство с нейронными сетями становится более заметным, если оператор можно представить в форме интеграла $\mathscr{H}(\mathbf{y}(x), \ldots, x, t) = \int K(\mathbf{y}(x), x, t, v, s) dv ds$. Таким образом, возникает континуальная нейросетевая многослойная модель аналогичная рассмотренной в [5]. Такая нейроподобная структура может быть рассмотрена как расширение многослойной сети прямого распространения или рекуррентной сети Хопфилда. Данная аналогия открывает возможности применения известных методов для доработки решения (оператора $\mathscr{H}$ или ядра K), например, алгоритма обратного распространения ошибки.

Отдельным является случай, когда оператор $\mathscr{G}$ в задаче (146) может быть представлен в виде

$$\mathscr{G}(\mathbf{y}(x, t), x, t) = \mathscr{L}\mathbf{y}(x, t) + \mathbf{g}(x, t),$$

где $\mathscr{L}$ – линейный оператор, например, линейный дифференциальный относительно переменной x. Тогда систему рекуррентных формул (147) можно пере-

писать в виде

$$(\frac{1}{h}\mathscr{I} - \mathscr{L})\mathbf{u}_{k+1}(x) = \frac{1}{h}\mathbf{u}_{k+1}(x) + \mathbf{g}(x, t_{k+1}),\ k = 0, \ldots, n; \tag{148}$$

где $\mathscr{I}$ – тождественный оператор, $\mathbf{u}_0(x, 0) = \varphi(x)$.

Далее, если спектр оператора $\mathscr{L}$ не содержит $1/h$, то существует резольвента $R(\mathscr{L}) = (\mathscr{L} - \frac{1}{h}\mathscr{I})^{-1}$, а приближенные решения вычисляются по формулам

$$\begin{cases} \mathbf{u}_1(x) = -\dfrac{1}{h}R\varphi(x) - R\mathbf{g}(x, t_1), \\ \mathbf{u}_{k+1}(x) = -\dfrac{1}{h}R\mathbf{u}_k(x) - \mathbf{g}(x, t_{k+1}),\ k = 0, \ldots, n-1. \end{cases} \tag{149}$$

Если резольвента записывается в форме интеграла, то результат снова можно рассматривать как континуальную нейросетевую многослойную модель и использовать соответствующие алгоритмы обучения для уточнения решения.

В следующих параграфах представлена аппробация метода для двух задач в частных производных, для которых задача построения резольвенты полностью решена.

7.2 Решение уравнения теплопроводности

7.2.1 Постановка задачи

Рассмотрим краевую задачу для уравнения теплопроводности в безразмерном виде с кусочно-заданным начальным условием

$$\begin{cases} \dfrac{\partial}{\partial t}y(x, t) = \dfrac{\partial^2}{\partial x^2}y(x, t), \\ y(x, 0) = \theta(x), \end{cases} \quad x \in \mathbb{R},\ t \in \mathbb{R}^+; \tag{150}$$

где

$$\theta(x) = \begin{cases} 1,\ x \geq 0; \\ 0,\ x < 0. \end{cases} \tag{151}$$

В данной поставке начальное условие представляет собой одновременно одну и распространенных базисных функций для нейронных сетей, что позволяет рассматривать эту модельную задачу и как частный случай нейросететвого представления для краевого условия.

Задача (150) имеет точное решение $y(x, t) = \frac{1}{2} - \frac{1}{\sqrt{\pi}}\int_0^z \exp(-t^2)dt$, с кото-

рым удобно сравнить и проанализировать результаты работы предложенного метода.

7.2.2 Построение модели с помощью аналитической модификации классических численных методов

В качестве базового метода для построения приближенного решения задачи 150 используется неявный метод Эйлера для интервала $[0, t]$ при $h_k(t) = t/n$, а именно, оператор F имеет вид

$$F(G(u_{k+1}(x), x, t_{k+1}), h_k, t_k, u_k(x), u_{k+1}(x)) = u_k(x) + h(t)u''_{k+1}(x). \tag{152}$$

В результате на каждом шаге k возникает линейное дифференциальное уравнение второго порядка для функции с переменной x

$$u''_{k+1}(x) - \frac{1}{h(t)}u_{k+1}(x) = -\frac{1}{h(t)}u_k(x),\ k \in 1, \ldots, n. \tag{153}$$

Для решения системы 153 используем рекуррентные соотношения при замене $z = x/\sqrt{h}$. Тогда решение системы сводится к решению линейного уравнения второго порядка вида

$$u''(z) - u(z) = A + P_k^1(z)\exp(z) + P_k^2(z)\exp(-z), \tag{154}$$

где $P_k^i = \sum_{j=0}^{k} a_{ij}z^j,\ i = 1, 2.$

Для каждого слагаемого в правой части уравнения (154) частные решения могут быть найдены методом неопределенных коэффициентов, а именно, для $P_k^1\exp(z)$ решение имеет вид $u^1(z) = \exp(z)\sum_{j=0}^{k} b_j^1 z^{j+1}$, где

$$b_k^1 = \frac{a_{1,k}}{2(k+1)};\ b_j^1 = \frac{a_{1,j}}{2(j+1)} - \frac{a_{1,j+1}(j+2)}{2};\ j = 0, \ldots, k-1;$$

и для $P_k^2\exp(-z)$ $u^1(z) = \exp(-z)\sum_{j=0}^{k} b_j^2 z^{j+1}$

$$b_k^2 = -\frac{a_{2,k}}{2(k+1)};\ b_j^2 = -\frac{a_{2,j}}{2(j+1)} + \frac{a_{2,j+1}(j+2)}{2};\ j = 0, \ldots, k-1.$$

Далее с помощью системы (153) для $k + 1 = n$ приближенное решение

будет иметь вид

$$u_n(x,t) = \begin{cases} 1 - P_n^1(x\sqrt{n/t})\exp(-x\sqrt{n/t}),\ x \geq 0; \\ P_n^2(x\sqrt{n/t})\exp(x\sqrt{n/t}),\ x \leq 0 \end{cases} \tag{155}$$

7.2.3 Результаты вычислений

Приближенные решения задачи (150), построенные с помощью аналитической модификации неявного метода Эйлера вида (155) были построены для разного числа итераций n для оценки порядка предложенного метода. Рисунок 68 позволяет оценить амплитуду и характер ошибки аппроксимации точного решения для различных значений n.

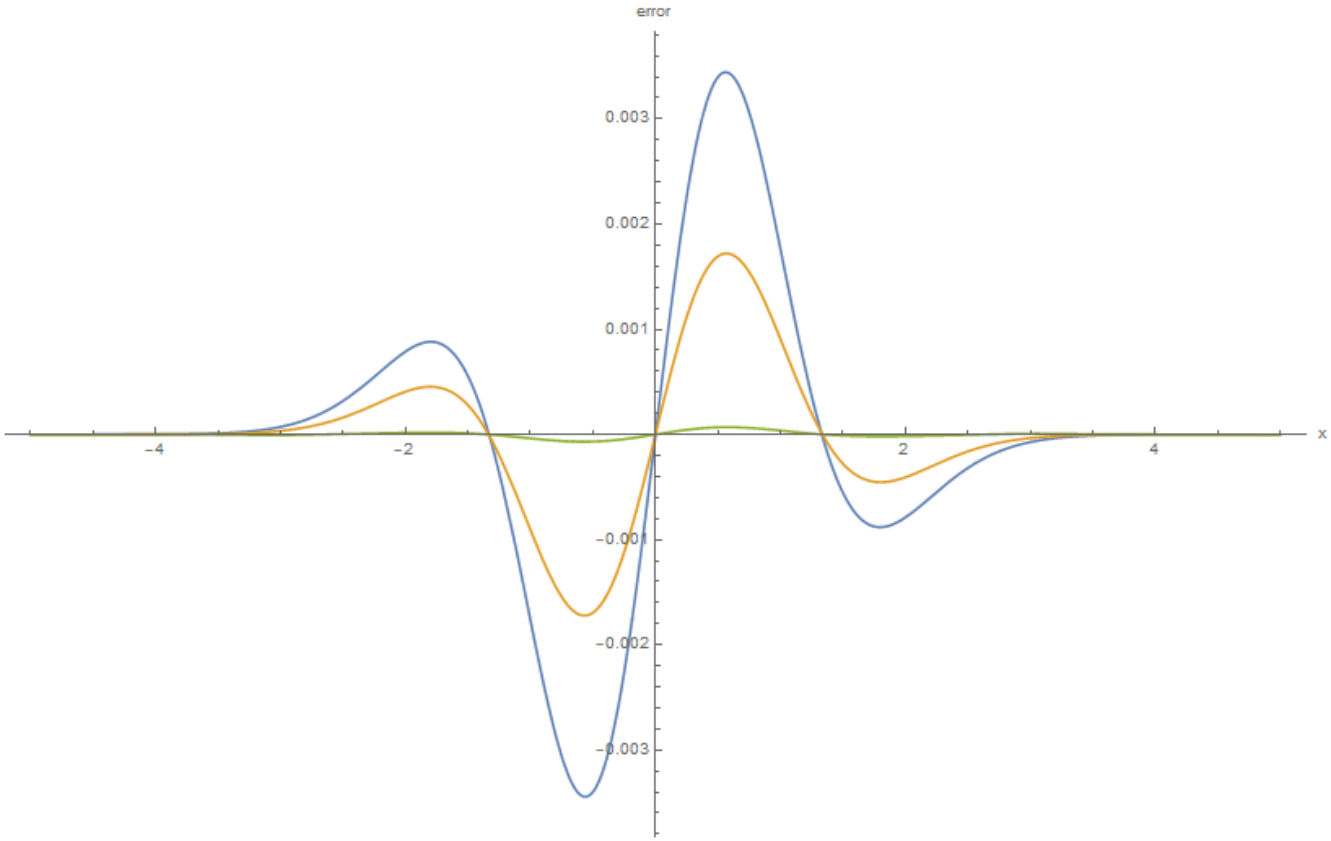

Рисунок 68 – Срез ошибки аппроксимации (155) точного решения задачи (150) при $t = 0.3$, $x \in [-5, 5]$, построенной с помощью аналитической модификации неявного метода Эйлера при значении числа итераций метода $n = 20, 40, 1000$

При этом, порядок значения максимума ошибки для $x \in [-5, 5]$ не зависит от переменной времени t, а только от числа итераций n, требующихся для построения модели. На Рисунке 69 представлена ошибка по всей области

изменения переменных при $n = 1000$ (так как решение симметрично, рассматривается $x \in [0, 5]$)

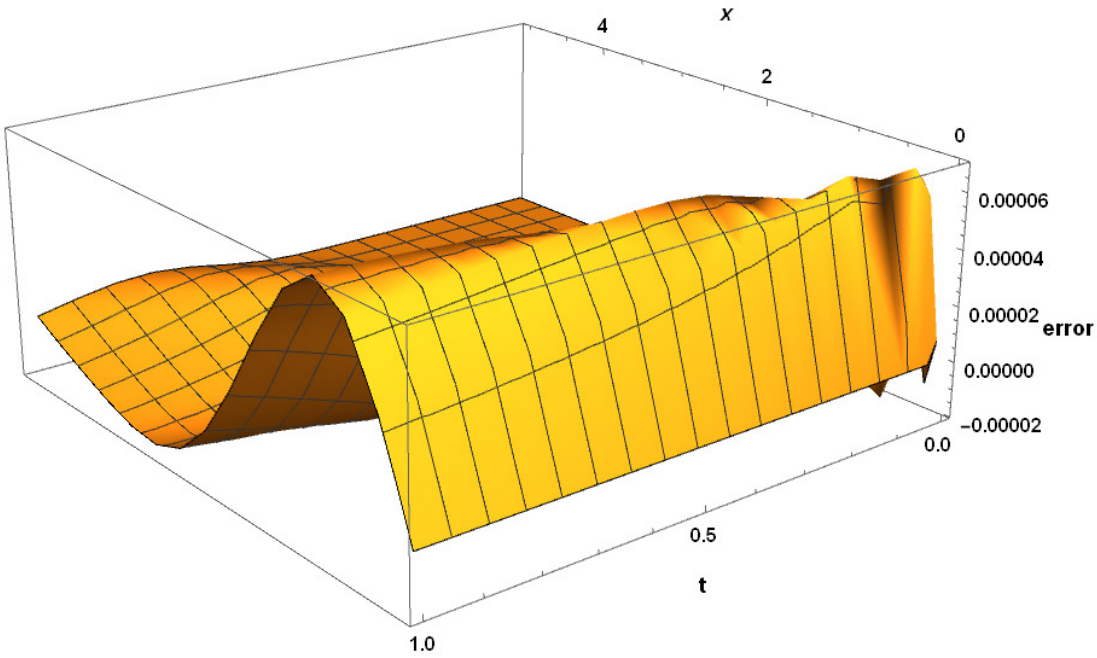

Рисунок 69 – Ошибка аппроксимации (155) точного решения задачи (150) при $t \in [0, 1]$ и $x \in [0, 5]$, построенной с помощью аналитической модификации неявного метода Эйлера при значении числа итераций метода $n = 1000$

7.3 Решение волнового уравнения

7.3.1 Постановка задачи

Рассмотрим линейное уравнение в частных производных гиперболического типа, а именно одномерное волновое уравнение и краевую задачу для него,

$$\begin{cases} \dfrac{\partial^2}{\partial t^2} y(x,t) = \dfrac{\partial^2}{\partial x^2} y(x,t), \\ y(x,0) = \theta(x), \qquad\qquad x \in \mathbb{R},\ t \in \mathbb{R}^+; \\ \dfrac{\partial}{\partial t} y(x,0) = 0; \end{cases} \tag{156}$$

Как и в случае задачи с уравнением теплопроводности рассмотрим в качестве функции $\theta(x)$ функцию, типичную для нейросетевых конструкций, простейшую РБФ-функцию

$$\theta(x) = \exp(-x^2),\ x \in \mathbb{R}. \tag{157}$$

Таким образом, данная постановка является частным случаем представления граничного условия в виде нейросетевой аппроксимации вида

$$\sum_{i=1}^{N} c_i \exp(-(a_i(x - b_i))^2).$$

7.3.2 Построение моделей с помощью аналитической модификации классических численных методов

7.3.2.1 Базовые численные методы первого порядка

Чтобы представить задачу (156) в виде (146), введем векторную функцию $\mathbf{y}(x,t) = (y_1(x,t), y_2(x,t))$, где $y_1(x,t) = y(x,t)$ и $y_2(x,t) = \frac{\partial y}{\partial t}$. Тогда система (156) преобразуется к виду

$$\begin{cases} \dfrac{\partial}{\partial t} y_1(x,t) = y_2(x,t), \\ \dfrac{\partial}{\partial t} y_2(x,t) = \dfrac{\partial^2}{\partial x^2} y_1(x,t), \ x \in \mathbb{R},\ t \in \mathbb{R}^+; \\ y_1(x,0) = \theta(x), \\ y_2(x,0) = 0; \end{cases} \tag{158}$$

При решении данной задачи в качестве базовых численных методов первого порядка

для интервала $[0,t]$ при $h_k(t) = t/n$, рассматриваются несколько вариантов, а именно, для приближенного решения $\mathbf{u}_k(x) = (u_{1k}(x), u_{2k}(x))$, $k = 0, \ldots, n$, оператор F имеет вид

$$F(x,t,\mathbf{u}_k(x)) = \begin{pmatrix} u_{1k}(x) + h(t)u_{2k}(x) \\ u_{2k}(x) + h(t)u'_{1k}(x) \end{pmatrix} \tag{159}$$

для явного метода Эйлера,

$$F(x,t,\mathbf{u}_k(x),\mathbf{u}_{k-1}(x)) = \begin{pmatrix} u_{1,k-1}(x) + 2h(t)u_{2k}(x) \\ u_{2,k-1}(x) + 2h(t)u'_{1k}(x) \end{pmatrix} \tag{160}$$

для улучшенного метода Эйлера и

$$F(x,t,\mathbf{u}_k(x)) = \begin{pmatrix} u_{1k}(x) + h(t)u_{2k}(x) + h^2(t)u'_{1k}(x) \\ u_{2k}(x) + h(t)u'_{1k}(x) + h^2(t)u'_{2k}(x) \end{pmatrix} \tag{161}$$

для Исправленного метода Эйлера.

7.3.2.2 Базовые численные методы второго порядка

Для применения методов, предназначенных для решения дифференциальных уравнений второго порядка, не требуется преобразование исходной формулировки задачи (156). Так, для метода Штермера имеем рекуррентное соотношение для приближенных решений вида

$$F(x,t,u_k(x),u_{k-1}(x)) = 2u_k(x) - u_{k-1}(x) + h^2(t)\frac{\partial^2 u_k(x)}{\partial x^2}. \tag{162}$$

При этом, непосредственное вычисление производной может быть заменено ее разностной аппроксимацией, то есть, вводя дополнительный параметр для шага приближенного вычисления производной d, получим формулу

$$u_{k+1}(x) = 2u_k(x) - u_{k-1}(x) + h^2(t)\frac{u_k(x+d,d) + u_k(x-d,d) - 2u_k(x,d)}{d^2}. \tag{163}$$

В качестве конкретных значений параметра d могут быть рассмотрены, например, $\sqrt{h(t)}$ или $h(t)$.

Для использования метода Штермера необходимо заранее вычислить $u_1(x)$. Для это используем либо условие $u_1(x) = \exp(-x^2)(1+h(t)(2x^2-1))$, либо применим на первом шаге Исправленный метод Эйлера (161).

7.3.3 Результаты вычислений

Приближенные решения задачи $u_n(x,t) = u_n(x)$ (156) были построены с помощью аналитических модификаций (159)-(162) для различного итогового числа итераций n. При различных значениях временной переменной $t \in [0,3]$, были посчитаны максимальные абсолютные ошибки аппроксимации в равноотстоящих точках $x \in [0,15]$,

$$\max E = \max\{|y(x,t) - u_n(x,t)|\big|x = 3i/200,\ i = 1,\ldots,1000\}. \qquad (164)$$

Таблица 12 – Максимум абсолютной ошибки (164) для приближенных решений задачи (156), полученных с помощью аналитических модификаций классических численных методов (159)-(162) в различных точках t при числе итераций n

$\boldsymbol{t = 0.1}$	$\boldsymbol{n = 5}$	$\boldsymbol{n = 10}$	$\boldsymbol{n = 20}$	$\boldsymbol{n = 50}$
Метод Эйлера	0.0001723	0.0000926	0.0000479	0.0000195
Улучшенный метод Эйлера	0.0000129	0.0000131	0.0000033	0.0000005
Исправленный метод Эйлера	0.0000130	0.0000033	0.0000008	0.0000001
Метод Штермера	0.0000131	0.0000033	0.0000008	0.0000001
Метод Штермера (2)	0.0000132	0.0000033	0.0000008	0.0000001
$\boldsymbol{t = 1}$	$n = 5$	$n = 10$	$n = 20$	$n = 50$
Метод Эйлера	0.0969773	0.0419807	0.0195704	0.0075341
Улучшенный метод Эйлера	0.0071519	0.0028264	0.0006968	0.0001111
Исправленный метод Эйлера	0.0053022	0.0013861	0.0003568	0.0000581
Метод Штермера	0.0028264	0.0006968	0.0001737	0.0000278
Метод Штермера (2)	0.0049493	0.0012288	0.0003067	0.0000490
$\boldsymbol{t = 3}$	$n = 5$	$n = 10$	$n = 20$	$n = 50$
Метод Эйлера	1.32383	0.62337	0.195054	0.0480651
Улучшенный метод Эйлера	8.39208	3.11714	0.0131722	0.0019675
Исправленный метод Эйлера	0.875653	0.0660036	0.0115796	0.0018030
Метод Штермера	3.11714	0.0131722	0.0030932	0.0004877
Метод Штермера (2)	0.0333585	0.0067234	0.0016255	0.0002578

Как видно из Таблицы 12 и Рисунка 70, в случае, когда необходимо ап-

проксимировать решение в краткосрочной перспективе ($t = 0.1$), для относительно хорошего результата достаточно применить обычный метод Эйлера, выбрав необходимое количество слоев (итераций).

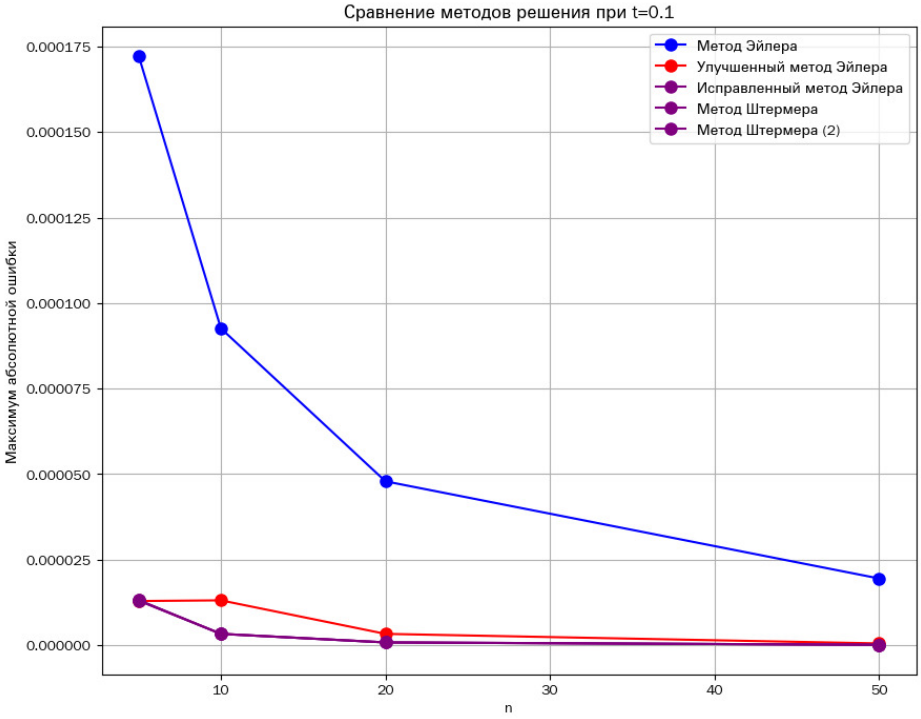

Рисунок 70 – Максимум абсолютной ошибки (164) для приближенных решений задачи (156), полученных с помощью аналитических модификаций классических численных методов (159)-(162) в точке $t = 0.1$ при числе итераций $n = 5, 10, 20, 50$

Уже при $t = 1$ аналитическая модификация метода Эйлера даже при числе итераций $n = 50$ дает большую ошибку, наглядно сравнить остальные базовые методы можно на рис.71. При необходимости более долгосрочных результатов при $n = 10$ и 20 относительно приемлемый результат показывают модели, построенные на базе модификации метода Штермера (Рисунок 72).

Наилучший результат в смысле максимума абсолютной ошибки среди всех рассмотренных моделей для различных моментов времени и числа итераций дает использование в том или ином виде метода Штермера, соответствующего формулам (162) и (163) при $h(t) = t/n$.

Метод Эйлера показывает удовлетворительные результаты только при $t < 1$. Далее поведение приближенного решения значительно отличается от точного решения задачи. На Рисунке 73 показан характер решений в точке $t = 2$ для различного числа итераций аналитической модификации метода.

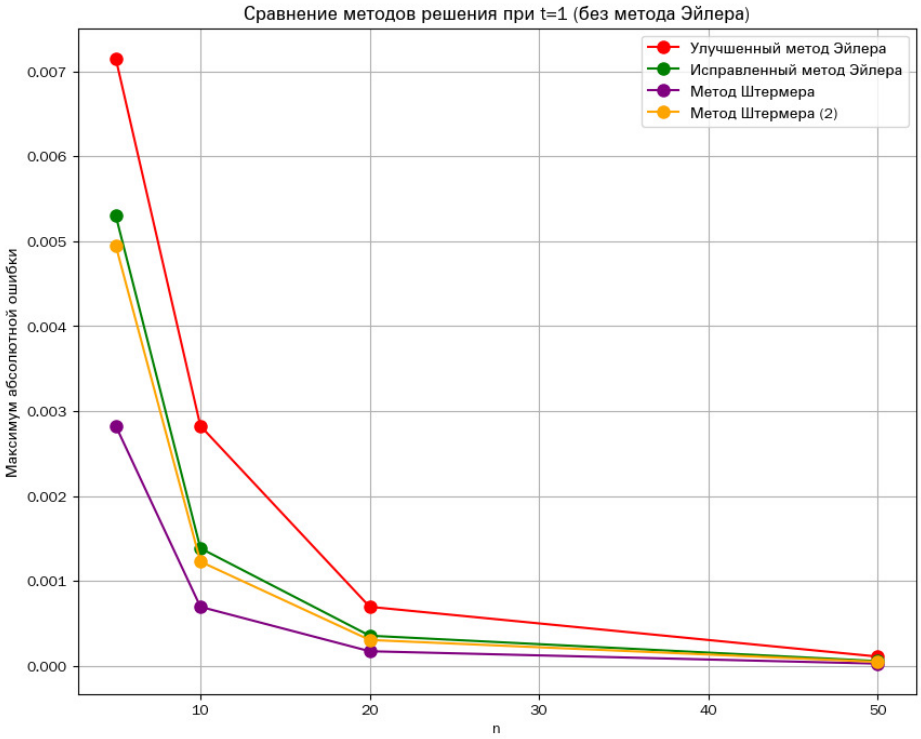

Рисунок 71 – Максимум абсолютной ошибки (164) для приближенных решений задачи (156), полученных с помощью аналитических модификаций классических численных методов (160)-(162) в точке $t = 1$ при числе итераций $n = 5, 10, 20, 50$

Результаты Улучшенного метода Эйлера существенно лучше, хотя рекуррентная формула (160) этого метода является одной из самых простых, поскольку переход осуществляется пошагово.

Исправленный метод Эйлера (161) является наиболее точным из рассмотренных базовых методов для решения уравнений первого порядка. Уже при $n = 5$ решение является довольно близким к точному (Рисунок 75).

Что касается модели, построенной с помощью аналитической модификации метода Штермера, *при аналогичной трудоемкости,* ее результаты даже для $t = 3$, представленные на Рисунке 76, значительно лучше (кроме модели с 5 итерациями), чем у модели базисными вариациями метода Эйлера для $t = 2$, что позволяет рекомендовать использование данного метода в качестве базисного для решения подобных задач с помощью аналитической модификации численных методов.

Для снижения вычислительных затрат в метод Штермера была внесена модификация (163), при этом точность полученных приближенных решений остается на том же уровне. Заметим, что в случае такого вычисления произ-

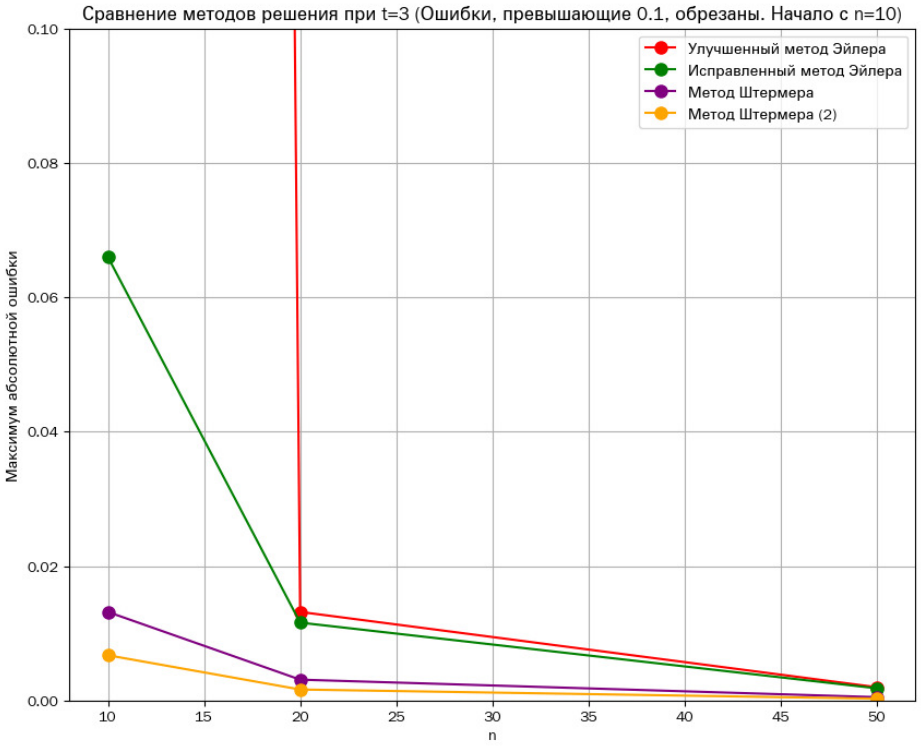

Рисунок 72 – Максимум абсолютной ошибки (164) для приближенных решений задачи (156), полученных с помощью аналитических модификаций классических численных методов (160)-(162) в точке $t = 1$ при числе итераций $n = 10, 20, 50$

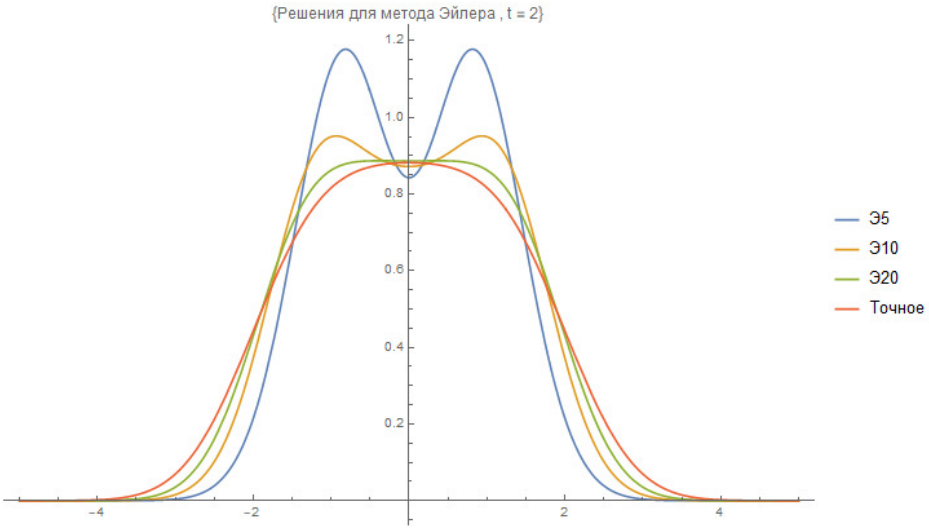

Рисунок 73 – Точное решение задачи (156) и его приближения, полученные с помощью аналитической модификации метода Эйлера (159) в точке $t = 2$ при числе итераций $n = 5, 10, 20$

водной результатом является непрерывный аналог обычной для данной задачи разностной схемы. На Рисунке 77 показано, как ведет себя ошибка метода с ростом числа итераций для $t = 3$.

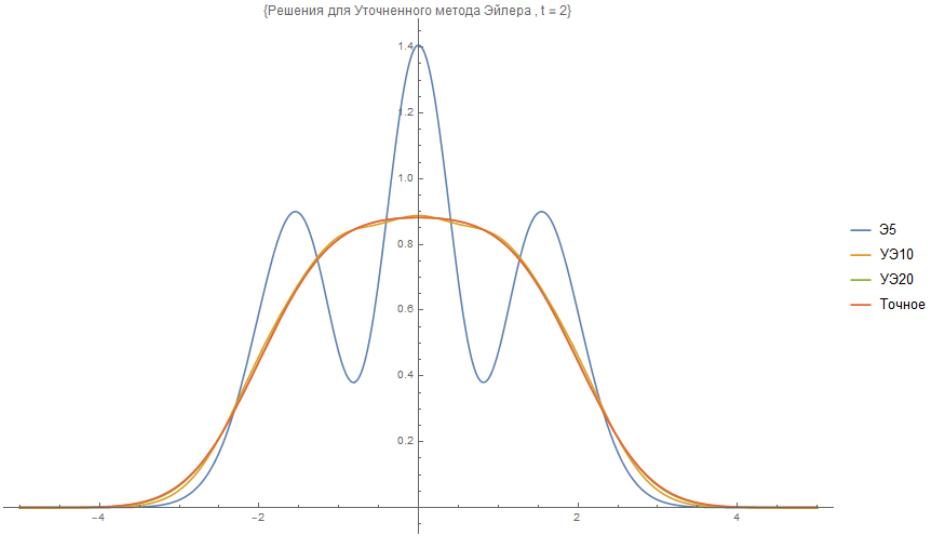

Рисунок 74 – Точное решение задачи (156) и его приближения, полученные с помощью аналитической модификации Улучшенного метода Эйлера (160) в точке $t = 2$ при числе итераций $n = 5, 10, 20$

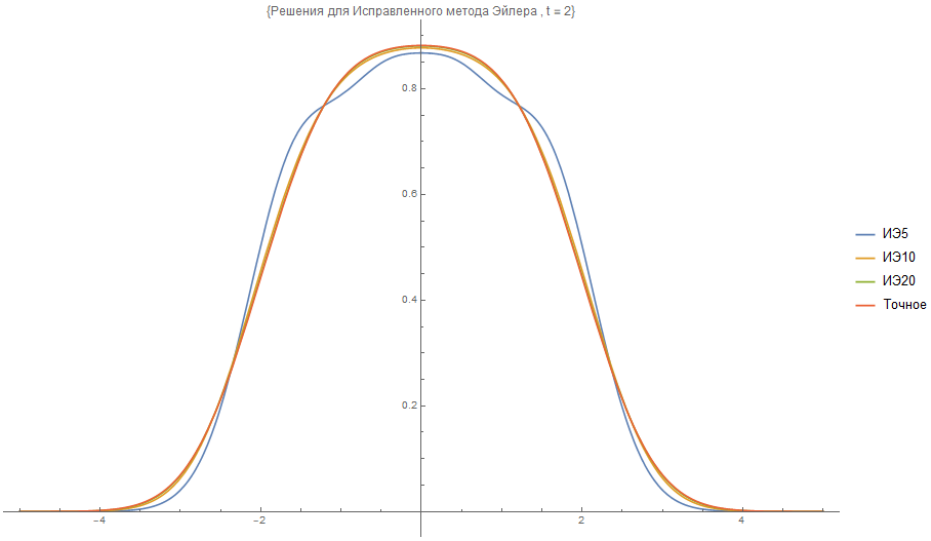

Рисунок 75 – Точное решение задачи (156) и его приближения, полученные с помощью аналитической модификации Исправленного метода Эйлера (161) в точке $t = 2$ при числе итераций $n = 5, 10, 20$

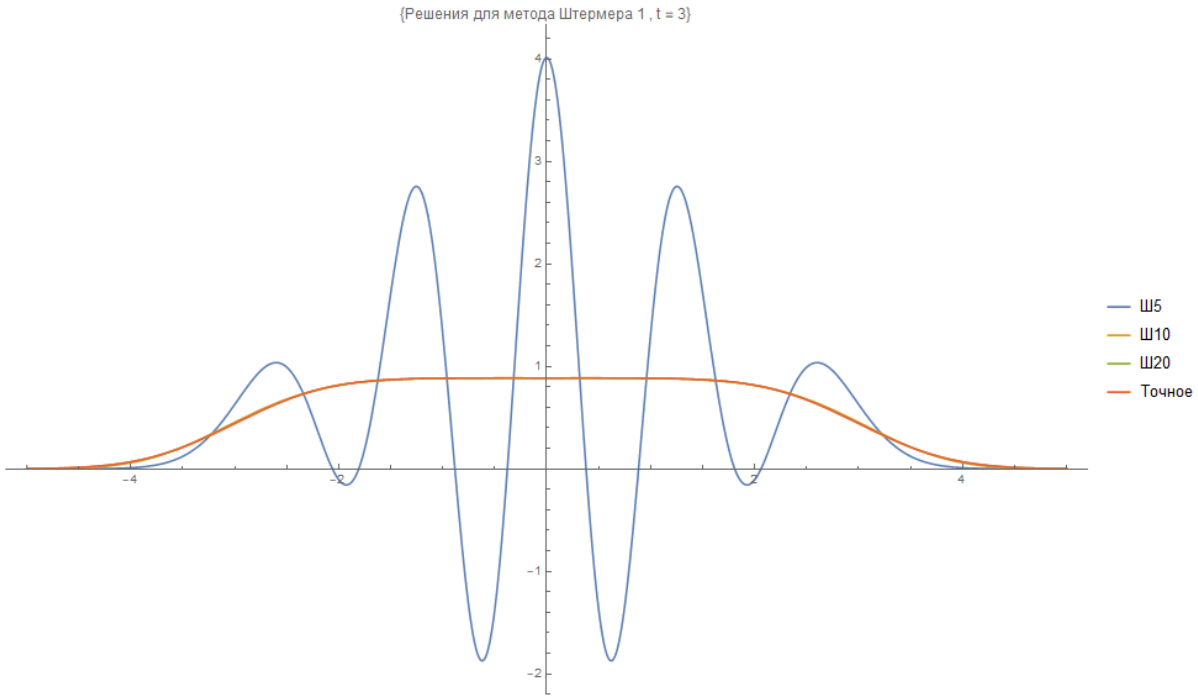

Рисунок 76 – Точное решение задачи (156) и его приближения, полученные с помощью аналитической модификации метода Штермера (162) в точке $t = 3$ при числе итераций $n = 5, 10, 20$

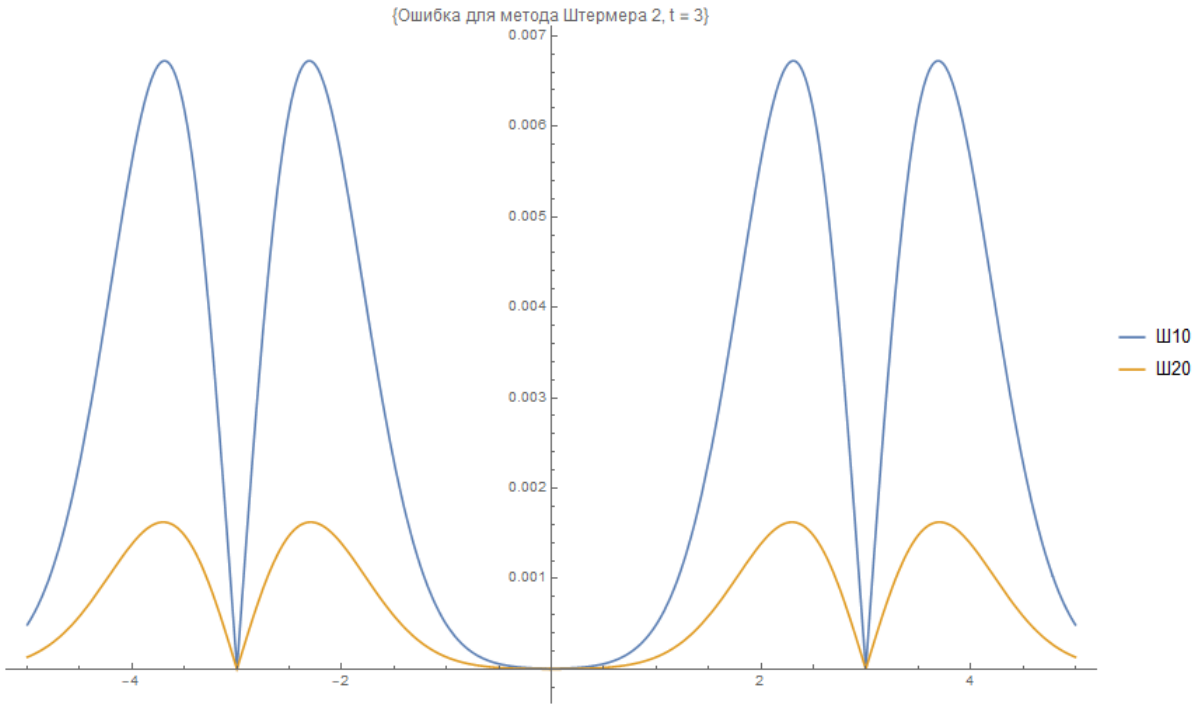

Рисунок 77 – Абсолютная ошибка приближений, полученных с помощью аналитической модификации метода Штермера (163) в точке $t = 3$ при числе итераций $n = 10, 20$

Заключение

Разработана и протестирована нейросетевая модель прогноза осложнений после эндоваскулярных вмешательств. Модель показала высокую точность и адаптивность, что подтверждает ее практическую значимость для медицинской практики.

Разработан новый тип нейроморфных моделей на основе аналитической модификации численных методов с архитектурой, основанной на физике. Эти модели позволяют более точно и эффективно моделировать сложные системы и процессы.

Проведен сравнительный анализ конечноэлементных, нейросетевых, гибридных моделей и моделей на основе аналитической модификации численных методов на примере задачи с пограничным слоем. Анализ показал преимущества и недостатки каждого подхода, что позволяет выбрать наиболее подходящий метод для конкретных задач.

Разработан и протестирован метод построения многослойной сети прямого распространения с архитектурой, основанной на физике, на основе аналитической модификации численных методов. Этот метод обеспечивает высокую точность и устойчивость моделей.

Разработаны и протестированы эволюционные алгоритмы обучения и построения физически информированных нейросетевых моделей на основе приближения к фронту Парето на двух некорректных задачах. Алгоритмы показали высокую эффективность и адаптивность.

Проведен сравнительный анализ адаптивных свойств непараметрических и параметрических нейросетевых моделей. Анализ выявил преимущества и области применения каждого типа моделей.

Разработан и протестирован метод построения приближенных нейроморфных решений с архитектурой, основанной на физике, уравнений в частных производных, на примере уравнения теплопроводности и волнового уравнения. Метод показал высокую точность и эффективность.

Разработанные методы и модели демонстрируют высокую точность и адаптивность, что подтверждает их практическую значимость для решения различных задач в области киберфизических систем и цифровых двойников.

Применение гибридных моделей, сочетающих численные методы и нейросетевые подходы, открывает новые возможности для проведения цифровых экспериментов и прогнозирования поведения объектов.

Эволюционные алгоритмы и методы аналитической модификации численных методов позволяют создавать устойчивые и достаточно точные математические модели, которые могут быть использованы в различных областях науки и техники.

Эти результаты подтверждают эффективность предложенных подходов и их значимость для решения практических задач.

В завершение данной работы автор выражает глубокую благодарность и искреннюю признательность научному руководителю Тархову Д. А. за ценные советы, поддержку, плодотворные обсуждения и научное руководство, оказанные на всех этапах исследования.

Список литературы

1. Васильев, А. Н., Тархов, Д. А. Нейронные сети как новый универсальный подход к численному решению задач математической физики // Нейрокомпьютеры: разработка, применение. — 2004. — № 7–8.

2. Васильев, А. Н., Тархов, Д. А. Нейросетевое моделирование. Принципы. Алгоритмы. Приложения / А. Н. Васильев, Д. А. Тархов. — Санкт-Петербург: Изд-во Политехн. ун-та, 2009. — 526 с.

3. Васильев, А. Н., Тархов, Д. А. Нейросетевое решение задачи о пористом катализаторе // Научно-технические ведомости СПбГПУ. Физико-математические науки. — 2008. — № 6 (67). — С. 110–113.

4. Галлагер, Р. Метод конечных элементов. Основы: Пер. с англ. / Р. Галлагер. — М. : Мир, 1984. — 428 с.

5. Галушкин, А. И. Теория нейронных сетей. Кн. 1. — М. : ИПРЖР, 2000. — 416 с.

6. Горбань, А. Н., Россиев, Д. А. Нейронные сети на персональном компьютере. — Новосибирск : Наука, 1996. — 276 с.

7. Горбаченко, В. И., Артюхина, Е. В. Обучение радиально-базисных нейронных сетей при решении дифференциальных уравнений в частных производных // Нейрокомпьютеры: разработка, применение. — 2007. — № 9. — С. 150–159.

8. Горбаченко, В. И., Москвитин, А. С. Нейросетевой подход к решению коэффициентной обратной задачи математической физики // Нейроинформатика. – 2005. – Т. 7. – С. 66.

9. Горбаченко, В. И., Москвитин, С. А. Решение коэффициентной обратной задачи математической физики с использованием нейронных сетей // Вопросы радиоэлектроники. – 2007. – Т. 4. – № 2. – С. 19–29.

10. Егорчев, М. В., Козлов, Д. С., Тюменцев, Ю. В., Чернышев, А. В. Нейросетевые полуэмпирические модели управляемых динамических систем // Вестник компьютерных и информационных технологий. — 2013. — № 9(111). — С. 3–10.

11. Канторович, Л. В., Крылов, В. И. Приближённые методы высшего анализа / Л. В. Канторович, В. И. Крылов. — 5-е изд. — Л.-М. : Физматгиз, 1962. — 708 с.

12. Лазовская, Т. В., Тархов, Д. А. Новые подходы к построению параметризованного нейросетевого решения жесткого дифференциального уравнения // Научно-технические ведомости СПбГПУ. Физико-математические науки. — 2015. — № 2 (218). — С. 138–146.

13. Лазовская, Т. В., Тархов, Д. А. О модификации классических итерационных методов для получения непрерывных решений на примере задачи теплопроводности // В сб.: Искусственный интеллект в решении актуальных социальных и экономических проблем XXI века: Сборник статей по материалам Второй всероссийской научно-практической конференции, проводимой в рамках Пермского естественнонаучного форума «Математика и глобальные вызовы XXI века». — 2017. — С. 214–219.

14. Льюнг, Л. Идентификация систем: Теория для пользователя / Пер. с англ. А. С. Манделя, А. В. Назина; Под ред. Я. З. Цыпкина. — М. : Наука, 1991. — 431 с.

15. Самарский, А. А., Вабищевич, П. Н. Численные методы решения обратных задач математической физики. — М. : Едиториал УРСС, 2004. — 480 с.

16. Тархов, Д. А. Нейронные сети. Модели и алгоритмы / Д. А. Тархов. — М.: Радиотехника, 2005. — (Научная серия "Нейрокомпьютеры и их применение ред. А. И. Галушкин; кн. 18). — 253 с.

17. Тархов, Д. А. Нейронные сети как средство математического моделирования // Нейрокомпьютеры: разработка, применение. — 2006. — № 2. — С. 47–48.

18. Хайрер, Э., Ваннер, Г. Решение обыкновенных дифференциальных уравнений. Жёсткие и дифференциально-алгебраические задачи. — М. : Мир, 1999. — 685 с.

19. Худяев, С. И. Пороговые явления в нелинейных уравнениях. — М. : Наука, 2003. — 268 с.

20. Amini Niaki, S., Haghighat, E., Campbell, T., Poursarti, A. P., Vaziri, R. Physics-informed neural network for modelling the thermochemical curing process of composite-tool systems during manufacture // Computer Methods in Applied Mechanics and Engineering. — 2021. — Vol. 384. — Article 113959.

21. Antonov, V., Tarkhov, D., Vasilyev, A. Unified approach to constructing the neural network models of real objects. Part 1 // Mathematical Methods in the Applied Sciences. — 2018. — Vol. 41. — P. 9244–9251.

22. Araujo, A., Martins, F., Velez, W., Portela, A. Automatic mesh-free boundary analysis: Multi-objective optimization // Engineering Analysis with Boundary Elements. — 2021. — Vol. 125. — P. 264–279.

23. Arthurs, C. J., King, A. P. Active training of physics-informed neural networks to aggregate and interpolate parametric solutions to the Navier-Stokes equations // Journal of Computational Physics. — 2021. — Vol. 438. — Article 110364.

24. Attanasio, M., Marcucci, R., Gori, A. M., Paniccia, R., Valente, S., Balzi, D., Barchielli, A., Carrabba, N., Valenti, R., Antoniucci, D., Abbate, R., Gensini, G. F. Residual thrombin potential predicts cardiovascular death in acute coronary syndrome patients undergoing percutaneous coronary intervention // Thrombosis Research. — 2016. — Vol. 147. — P. 52–57.

25. Aziz, I., ?arler, B. The numerical solution of second-order boundary-value problems by collocation method with the Haar wavelets // Mathematical and Computer Modelling. — 2010. — Vol. 52. — P. 1577–1590.

26. Basir, S., Inanc, S. Physics and Equality Constrained Artificial Neural Networks: Application to Partial Differential Equations // arXiv preprint, 2021. — arXiv:2109.14860.

27. Beheshti, Z., Beheshti, E. Enhancement of artificial neural network learning using centripetal accelerated particle swarm optimization for medical diseases diagnosis // Soft Computing Methodology and Applications. — 2013. — Vol. 18, No. 11. — P. 2253–2270.

28. Benner, P., Gugercin, S., Willcox, K. A Survey of Projection-Based Model Reduction Methods for Parametric Dynamical Systems // SIAM Review. — 2015. — Vol. 57. — P. 483–581.

29. Beucler, T., Pritchard, M., Rasp, S., Ott, J., Baldi, P., Gentine, P. Enforcing Analytic Constraints in Neural Networks Emulating Physical Systems // Physical Review Letters. — 2021. — Vol. 126. — Article 098302.

30. Bischof, R., Kraus, M. Multi-objective loss balancing for physics-informed deep learning // arXiv preprint, 2021. — arXiv:2110.09813.

31. Blagoveshchenskaya, E. A., Dashkina, A. I., Lazovskaya, T. V., Ryabukhina, V. V., Tarkhov, D. A. Neural network methods for construction of sociodynamic models hierarchy // In: Cheng, L., Liu, Q., Ronzhin, A. (eds.) Proceedings of the 13th International Symposium on Neural Networks (ISNN 2016), St. Petersburg, Russia, July 6–8, 2016. — Cham : Springer, 2016. — Lecture Notes in Computer Science, Vol. 9719. — P. 513–520.

32. Bolgov, I., Kaverzneva, T., Kolesova, S., Lazovskaya, T., Stolyarov, O., Tarkhov, D. Neural network model of rupture conditions for elastic material sample based on measurements at static loading under different strain rates // Journal of Physics: Conference Series. — 2016. — Vol. 772. — Article 012032.

33. Boyarsky, S., Lazovskaya, T., Tarkhov, D. Investigation of the Predictive Capabilities of a Data-Driven Multilayer Model by the Example of the Duffing Oscillator // Proceedings of the 2020 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon 2020), Vladivostok, Russia, 6–9 October 2020.

34. Braun, H., Riedmiller, M. A direct adaptive method for faster backpropagation learning: The Rprop algorithm // Proceedings of the IEEE International Conference on Neural Networks, San Francisco, CA, USA, 28 March – 1 April 1993. — P. 586–591.

35. Calimeri, F., Cauteruccio, F., Cinelli, L., Marzullo, A., Stamile, C., Terracina, G., Durand-Dubief, F., Sappey-Marinier, D. A logic-based framework leveraging neural networks for studying the evolution of neurological disorders // Theory and Practice of Logic Programming. — 2021. — Vol. 21. — P. 80–124.

36. Cauteruccio, F., Stamile, C., Terracina, G., Ursino, G., Sappey-Mariniery, D. An automated string-based approach to White Matter fiber-bundles clustering // Proceedings of the 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 12–17 July 2015.

37. Chakraborty, S. Transfer learning based multi-fidelity physics-informed deep neural network // Journal of Computational Physics. — 2021. — Vol. 426. — Article 109942.

38. Coello, C. A. C., Lamont, G. B., Van Veldhuizen, D. A. Evolutionary Algorithms for Solving Multi-Objective Problems. — 2nd ed. — New York, NY, USA : Springer, 2007.

39. Cuomo, S., Di Cola, V. S., Giampaolo, F., et al. Scientific Machine Learning Through Physics–Informed Neural Networks: Where We Are and What's Next // Journal of Scientific Computing. — 2022. — Vol. 92. — Article 88.

40. Cybenko, G. Approximation by superpositions of a sigmoidal function // Mathematics of Control, Signals and Systems. — 1989. — Vol. 2, No. 4. — P. 303–314.

41. Dallas, S., MacHairas, K., Papadopoulos, E. A Comparison of Ordinary Differential Equation Solvers for Dynamical Systems with Impacts // Journal of Computational and Nonlinear Dynamics. — 2017. — Vol. 12, No. 6.

42. Das, P. Comparison of a priori and a posteriori meshes for singularly perturbed nonlinear parameterized problems // Journal of Computational and Applied Mathematics. — 2015. — Vol. 290. — P. 16–25.

43. Daw, A., Bu, J., Wang, S., Perdikaris, P., Karpatne, A. Rethinking the importance of sampling in physics-informed neural networks // arXiv preprint, 2022. — arXiv:2207.02338.

44. Dissanayake, M. W. M. G., Phan-Thien, N. Neural-network-based approximations for solving partial differential equations // Communications in Numerical Methods in Engineering. — 1994. — Vol. 10. — P. 195–201.

45. Du, K. J., Li, J. Y., Wang, H., Zhang, J. Multi-objective multi-criteria evolutionary algorithm for multi-objective multi-task optimization // Complex & Intelligent Systems. — 2022. — Vol. 9. — P. 1211–1228.

46. Eberhart, R., Kennedy, J. Particle swarm optimization // Proceedings of the ICNN'95 — International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995. — Vol. 4. — P. 1942–1948.

47. Elsken, T., Metzen, J. H., Hutter, F. Efficient Multi-Objective Neural Architecture Search via Lamarckian Evolution // arXiv preprint, 2018. — arXiv:1804.09081.

48. Erge, O., van Oort, E. Combining physics-based and data-driven modeling in well construction: Hybrid fluid dynamics modeling // Journal of Natural Gas Science and Engineering. — 2022. — Vol. 97. — Article 104348.

49. Famelis, I. T., Kaloutsa, V. Parameterized neural network training for the solution of a class of stiff initial value systems // Neural Computing and Applications. — 2020.

50. Fawcett, T. An introduction to ROC analysis // Pattern Recognition Letters. — 2006. — Vol. 27, No. 8. — P. 861–874.

51. Filkin, V., Kaverzneva, T., Lazovskaya, T., Lukinskiy, E., Petrov, A., Stolyarov, O., Tarkhov, D. Neural network modeling of conditions of destruction of wood plank based on measurements // Journal of Physics: Conference Series. — 2016. — Vol. 772. — Article 012041.

52. Frank, M., Drikakis, D., Charissis, V. Machine-Learning Methods for Computational Science and Engineering // Computation. — 2020. — Vol. 8. — Article 8010015.

53. Garg, S., Chakraborty, S., Hazra, B. Physics-integrated hybrid framework for model form error identification in nonlinear dynamical systems // Mechanical Systems and Signal Processing. — 2022. — Vol. 173. — Article 109039.

54. Geneva, N., Zabaras, N. Modeling the dynamics of PDE systems with physics-constrained deep auto-regressive networks // Journal of Computational Physics. — 2020. — Vol. 403. — Article 109056.

55. Gorbachenko, V. I., Kuznetsova, O., Silnov, D. S. Investigation of neural and fuzzy neural networks for diagnosis of endogenous intoxication syndrome in patients with chronic renal failure // International Journal of Applied Engineering Research. — 2016. — Vol. 11, No. 7. — P. 5156–5162.

56. Gorbachenko, V. I., Lazovskaya, T. V., Tarkhov, D. A., Vasilyev, A. N., Zhukov, M. V. Neural network technique in some inverse problems of mathematical physics // In: Cheng, L., Liu, Q., Ronzhin, A. (eds.) Proceedings of the 13th International Symposium on Neural Networks (ISNN 2016), St. Petersburg, Russia, July 6–8, 2016. — Cham : Springer, 2016. — Lecture Notes in Computer Science, Vol. 9719. — P. 310–316.

57. Goswami, S., Yin, M., Yu, Y., Karniadakis, G. E. A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials // Computational Methods in Applied Mechanics and Engineering. — 2022. — Vol. 391. — Article 114587.

58. Gui, N., Jiang, S., Yang, X., Tu, J. A review of recent study on the characteristics and applications of pebble flows in nuclear engineering // Experimental and Computational Multiphase Flow. — 2022. — Vol. 2. — P. 339–349.

59. Guglielmi, N., Hairer, E. Solutions leaving a codimension-2 sliding // Nonlinear Dynamics. — 2017. — Vol. 88. — P. 1427–1439.

60. Guliyev, N. J., Ismailov, V. E. On the approximation by single hidden layer feedforward neural networks with fixed weights // Neural Networks. — 2018. — Vol. 98. — P. 296–304. — ISSN 0893-6080.

61. Haghighat, E., Raissi, M., Moure, A., Gomez, H., Juanes, R. A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics

// Computational Methods in Applied Mechanics and Engineering. — 2021. — Vol. 379. — Article 113741.

62. Hairer, E., Wanner, G. Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems. — Berlin/Heidelberg, Germany : Springer, 1996.

63. Haykin, S. Neural Networks: A Comprehensive Foundation. — Hoboken, NJ, USA : Prentice Hall, 1999. — 842 p.

64. He, Q., Barajas-Solano, D., Tartakovsky, G., Tartakovsky, A. M. Physics-informed neural networks for multiphysics data assimilation with application to subsurface transport // Advances in Water Resources. — 2020. — Vol. 141. — Article 103610.

65. Hemker, H. C., Giesen, P., Al Dieri, R., Regnault, V., de Smedt, E., Wagenvoord, R., Lecompte, T., Beguin, S. Calibrated automated thrombin generation measurement in clotting plasma // Pathophysiology of Haemostasis and Thrombosis. — 2003. — Vol. 33. — P. 4–15.

66. Hemker, H. C., Wielders, S., Kessels, H., Beguin, S. Continuous registration of thrombin generation in plasma, its use for the determination of the thrombin potential // Thrombosis and Haemostasis. — 1993. — Vol. 70. — P. 617–624.

67. Hlavacek, V., Marek, M., Kubicek, M. Modelling of chemical reactors—X. Multiple solutions of enthalpy and mass balances for a catalytic reaction within a porous catalyst particle // Chemical Engineering Science. — 1968. — Vol. 23. — P. 1083–1097.

68. Hornik, K., Tinchcombe, M., White, H. Multilayer Feedforward Networks are Universal Approximators // Neural Networks. — 1989. — Vol. 2. — P. 359–366.

69. Hosseinzadeh, K., Afsharpanah, F., Zamani, S., Gholinia, M., Ganji, D. D. 3D free convective MHD flow of nanofluid over permeable linear stretching sheet with thermal radiation // Case Studies in Thermal Engineering. — 2018. — Vol. 12. — P. 228–236.

70. Huang, G., Chen, L., Siew, C. Universal approximation using incremental constructive feedforward networks with random hidden nodes // IEEE Transactions on Neural Networks. — 2006. — Vol. 17. — P. 879–892.

71. Huang, Y., Hao, W., Lin, G. HomPINNs: Homotopy physics-informed neural networks for learning multiple solutions of nonlinear elliptic differential equations // Computers and Mathematics with Applications. — 2022. — Vol. 121. — P. 62–73.

72. Huang, G., Zhu, Q., Siew, C. Extreme learning machine: Theory and applications // Neurocomputing. — 2006. — Vol. 70. — P. 489–501.

73. Hughes, T. J. R. The Finite Element Method: Linear Static and Dynamic Finite Element Analysis. — Englewood Cliffs, NJ, USA : Prentice-Hall, 1987.

74. Ince, T., Kiranyaz, S., Pulkkinen, J., Gabbouj, M. Evaluation of global and local training techniques over feed-forward neural network architecture spaces for computer-aided medical diagnosis // Expert Systems with Applications. — 2010. — Vol. 37. — P. 8450–8461.

75. Jagtap, A. D., Kawaguchi, K., Karniadakis, G. E. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks // Journal of Computational Physics. — 2020. — Vol. 404. — Article 109136.

76. Jagtap, A. D., Kawaguchi, K., Karniadakis, G. E. Locally adaptive activation functions with slope recovery for deep and physics-informed neural networks // Proceedings of the Royal Society A. — 2020. — Vol. 476. — Article 20200334.

77. Jia, X., Willard, J., Karpatne, A., Read, J., Zwart, J., Steinbach, M., Kumar, V. Physics guided RNNs for modeling dynamical systems: A case study in simulating lake temperature profiles // Proceedings of the SIAM International Conference on Data Mining (SDM19), Calgary, Canada, 2–4 May 2019. — P. 558–566.

78. Johnson, C. Numerical Solution of Partial Differential Equations by the Finite Element Method. — Chelmsford, MA, USA : Courier Corporation, 2012.

79. Kaveh, A., Khavaninzadeh, N. Efficient training of two ANNs using four meta-heuristic algorithms for predicting the FRP strength // Structures. — 2023. — Vol. 52. — P. 256–272.

80. Kaverzneva, T., Lazovskaya, T., Tarkhov, D., Vasilyev, A. Neural network modeling of air pollution in tunnels according to indirect measurements // Journal of Physics: Conference Series. — 2016. — Vol. 772. — Article 012035.

81. Kim, S. W., Kim, I., Lee, J., Lee, S. Knowledge Integration into Deep Learning in Dynamical Systems: An Overview and Taxonomy // Journal of Mechanical Science and Technology. — 2021. — Vol. 35. — P. 1331–1342.

82. Kovalchuk, S. V., Metsker, O. G., Funkner, A. A., Kisliakovskii, I. O., Nikitin, N. O., Kalyuzhnaya, A. V., Vaganov, D. A., Bochenina, K. O. A Conceptual Approach to Complex Model Management with Generalized Modelling Patterns and Evolutionary Identification // Complexity. — 2018. — Article 5870987.

83. Kudu, M. A parameter uniform difference scheme for the parameterized singularly perturbed problem with integral boundary condition // Advances in Differential Equations. — 2018. — Article 170.

84. Lagaris, I. E., Likas, A., Fotiadis, D. I. Artificial neural networks for solving ordinary and partial differential equations // IEEE Transactions on Neural Networks. — 1998. — Vol. 9. — P. 987–1000.

85. Lazovskaya, T., Malykhina, G., Tarkhov, D. Comparing Physics-Based Neural Network Methods for Solving Parameterized Singular Perturbation Problem // Computation. — 2021. — Vol. 9. — Article 97.

86. Lazovskaya, T., Malykhina, G., Tarkhov, D. Construction of an Individual Model of the Deflection of a PVC-Specimen Based on a Differential Equation and Measurement Data // Proceedings of the 2020 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon 2020), Vladivostok, Russia, 6–9 October 2020.

87. Lazovskaya, T., Tarkhov, D. Multilayer neural network models based on grid methods // IOP Conference Series: Materials Science and Engineering. — IOP Publishing, Bristol, UK, 2016.

88. Lazovskaya, T., Tarkhov, D., Chernukha, D., Korchagin, A., Malykhina, G. Analysis of Predictive Capabilities of Adaptive Multilayer Models with Physics-Based Architecture for Duffing Oscillator // Proceedings of the International Conference on Neuroinformatics, 2023. — Vol. 1064. — P. 54.

89. Lazovskaya, T., Tarkhov, D., Chistyakova, M., Sergeeva, A., Shemyakina, T. Evolutionary PINN learning algorithms inspired by approximation to Pareto front for solving ill-posed problems // Computation. — 2023. — Vol. 11, No. 8. — Article 166.

90. Lazovskaya, T. V., Tarkhov, D. A., Vasilyev, A. N. Parametric neural network modeling in engineering // Recent Patents on Engineering. — 2017. — Vol. 11. — P. 10–15.

91. Lee, J., Bagheri, B., Kao, H. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems // Manufacturing Letters. — 2015. — Vol. 3. — P. 18–23.

92. Li, B., Li, Y., Rong, X. The extreme learning machine learning algorithm with tunable activation function // Neural Computing and Applications. — 2013. — Vol. 22. — P. 531–539

93. Lin, X., Zhen, H., Li, Z., Zhang, Q., Kwong, S. T. Pareto Multi-Task Learning // arXiv preprint, 2019. — arXiv:1912.12854.

94. Liu, Q., Jin, Y., Heiderich, M., Rodemann, T., Yu, G. An Adaptive Reference Vector-Guided Evolutionary Algorithm Using Growing Neural Gas for Many-Objective Optimization of Irregular Problems // IEEE Transactions on Cybernetics. — 2022. — Vol. 52, No. 5. — P. 2698–2711.

95. Liu, H., Xing, B., Wang, Zh., Li, L. Legendre Neural Network Method for Several Classes of Singularly Perturbed Differential Equations Based on Mapping and Piecewise Optimization Technology // Neural Processing Letters. — 2020.

96. Loeffen, R., Godschalk, T. C., van Oerle, R., Spronk, H. M., Hackeng, C. M., ten Berg, J. M., ten Cate, H. The hypercoagulable profile of patients with stent thrombosis // Heart. — 2015. — Vol. 101, No. 14. — P. 1126–1132.

97. Lu, B., Moya, C., Lin, G. NSGA-PINN: A Multi-Objective Optimization Method for Physics-Informed Neural Network Training // Algorithms. — 2023. — Vol. 16. — Article 194.

98. Lye, K. O., Mishra, S., Molinaro, R. A multi-level procedure for enhancing accuracy of machine learning algorithms // European Journal of Applied Mathematics. — 2021. — Vol. 32, No. 3. — P. 436–469.

99. Maier, H. R., Razavi, S., Kapelan, Z., Matott, L. S., Kasprzyk, J., Tolson, B. A. Introductory overview: Optimization using evolutionary algorithms and other metaheuristics // Environmental Modelling and Software. — 2019. — Vol. 114. — P. 195–213.

100. Meng, X., Karniadakis, G. E. A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems // Journal of Computational Physics. — 2020. — Vol. 401. — Article 109020.

101. Mirjalili, S., Mirjalili, S. M., Lewis, A. Grey Wolf optimizer // Advances in Engineering Software. — 2014. — Vol. 69. — P. 46–61.

102. Mitchell, S. L., Vynnycky, M. An accuracy-preserving numerical scheme for parabolic partial differential equations subject to discontinuities in boundary conditions // Applied Mathematics and Computation. — 2021. — Vol. 400. — Article 125979.

103. Moein, S. Medical Diagnosis Using Artificial Neural Networks. — Hershey, PA, USA : IGI Global, 2014.

104. Momma, M., Dong, C., Liu, J. A Multi-objective / Multi-task Learning Framework Induced by Pareto Stationarity // Proceedings of the 39th International Conference on Machine Learning (PMLR), Baltimore, MD, USA, 17–23 July 2022. — Vol. 162. — P. 15895–15907.

105. Motsa, S. S., Makinde, O. D., Shateyi, S. On New High Order Quasilinearization Approaches to the Nonlinear Model of Catalytic Reaction in a Flat Particle // Advances in Mathematical Physics. — 2013. — Vol. 2013. — Article 350810.

106. Muhanna, R. L., Mullen, R. L., Zhang, H. Penalty-based solution for the interval finite-element methods // Journal of Engineering Mechanics. — 2005. — Vol. 131. — P. 1102–1111.

107. Nabian, M. A., Gladstone, R. J., Meidani, H. Efficient training of physics-informed neural networks via importance sampling // Computer-Aided Civil and Infrastructure Engineering. — 2021. — Vol. 36, No. 8. — P. 962–977.

108. Najeeb, A., Khan, F. S., Alshomrani, A. S. Binary chemical reaction with activation energy in radiative rotating disk flow of Bingham plastic fluid // Heat Transfer—Asian Research. — 2020. — Vol. 49. — P. 1314–1337.

109. Nasruddin, S., Satrio, P., Mahlia, T. M. I., Giannetti, N., Saito, K. Optimization of HVAC system energy consumption in a building using artificial neural network and multi-objective genetic algorithm // Sustainable Energy Technologies and Assessments. — 2019. — Vol. 35. — P. 48–57.

110. Nayak, M. K., Akbar, N. S., Pandey, V. S., Khan, Z. H., Tripathi, D. 3D free convective MHD flow of nanofluid over permeable linear stretching sheet with thermal radiation // Powder Technology. — 2017. — Vol. 301. — P. 205–215.

111. Nayfeh, A. H. Perturbation Methods. — Weinheim, Germany : Wiley-VCH Verlag GmbH, 1973.

112. Nobile, F., Tempone, R., Webster, C. G. A sparse grid stochastic collocation method for partial differential equations with random input data // SIAM Journal on Numerical Analysis. — 2008. — Vol. 46. — P. 2309–2345.

113. Orr, M. J. L. Introduction to Radial Basis Function Networks. — Edinburgh : Centre for Cognitive Science, University of Edinburgh, 1996. — 67 p.

114. Pantopoulou, S., Ankel, V., Weathered, M. T., Lisowski, D. D., Cilliers, A., Tsoukalas, L. H., Heifetz, A. Monitoring of Temperature Measurements for Different Flow Regimes in Water and Galinstan with Long Short-Term Memory Networks and Transfer Learning of Sensors // Computation. — 2022. — Vol. 10. — Article 108.

115. Park, J., Sandberg, I. W. Approximation and Radial-Basis-Function Networks // Neural Computation. — 1993. — Vol. 5, No. 2. — P. 305–316.

116. Parker, S. S., Newman, S., Fallgren, A. J., White, J. T. Physics-Based Neural Networks Methods for Solving Thermophysical Properties of Mixtures of Thorium and Uranium Nitride // JOM. — 2021. — Vol. 73. — P. 3564–3575.

117. Peng, P., Pan, J., Xu, H., Feng, X. RPINNs: Rectified-physics informed neural networks for solving stationary partial differential equations // Computers & Fluids. — 2022. — Vol. 245. — Article 105583.

118. Platzer, A. The Logical Path to Autonomous Cyber-Physical Systems: (Invited Paper) // Quantitative Evaluation of Systems: 16th International Conference, QEST 2019, Glasgow, UK, September 10–12, 2019, Proceedings. — Berlin, Heidelberg : Springer-Verlag, 2019. — P. 25–33.

119. Puneeth, V., Manjunatha, S., Makinde, O. D., Gireesha, B. J. Bioconvection of a radiating hybrid nanofluid past a thin needle in the presence of heterogeneous-homogeneous chemical reaction // Journal of Heat Transfer. — 2021. — Vol. 143, No. 4. — Article 042502.

120. Rai, R., Sahu, C. K. Driven by Data or Derived through Physics? A Review of Hybrid Physics Guided Machine Learning Techniques with Cyber-Physical System (CPS) Focus // IEEE Access. — 2020. — Vol. 8. — P. 71050–71073.

121. 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. — 2019. — Vol. 378. — P. 686–707.

122. Rajendra, P., Brahmajirao, V. Modeling of dynamical systems through deep learning // Biophysical Reviews. — 2020. — Vol. 12. — P. 1311–1320.

123. Rao, C., Sun, H., Liu, Y. Physics-informed deep learning for incompressible laminar flows // arXiv preprint, 2021. — arXiv:2002.10558.

124. Rasheed, A., San, O., Kvamsdal, T. Digital Twin: Values, Challenges and Enablers From a Modeling Perspective // IEEE Access. — 2020. — Vol. 8. — P. 21980–22012.

125. Riedmiller, M., Braun, H. A direct adaptive method for faster backpropagation learning: The RPROP algorithm // Proceedings of the IEEE International

Conference on Neural Networks, San Francisco, CA, USA, 28 March–1 April 1993. — P. 586–591.

126. San, O., Rasheed, A., Kvamsdal, T. Hybrid analysis and modeling, eclecticism, and multifidelity computing toward digital twin revolution // GAMM Mitteilungen. — 2021. — Vol. 44, No. 2.

127. Sarker, I. H. Machine Learning: Algorithms, Real-World Applications and Research Directions // SN Computer Science. — 2021. — Vol. 2. — Article 160.

128. Shakti, D., Mohapatra, J. Parameter-Uniform Numerical Methods for a Class of Parameterized Singular Perturbation Problems // Numerical Analysis and Applications. — 2019. — Vol. 12. — P. 176–190.

129. Shen, L., Xie, F., Xiao, W., Ji, H., Zhang, B. Thermal Analyses of Reactor under High-Power and High-Frequency Square Wave Voltage Based on Improved Thermal Network Model // Electronics. — 2021. — Vol. 10. — Article 1342.

130. Shemyakina, T. A., Tarkhov, D. A., Vasilyev, A. N. Neural Network Technique for Processes Modeling in Porous Catalyst and Chemical Reactor // Lecture Notes in Computer Science, including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics. — Berlin/Heidelberg, Germany : Springer, 2016.

131. Sirignano, J., Spiliopoulos, K. DGM: A deep learning algorithm for solving partial differential equations // Journal of Computational Physics. — 2018. — Vol. 375. — P. 1339–1364.

132. Slowik, A., Kwasnicka, H. Evolutionary algorithms and their applications to engineering problems // Neural Computing and Applications. — 2020. — Vol. 32. — P. 12363–12379.

133. So, S., Jeong, N., Song, A., Hwang, J., Kim, D., Lee, C. Measurement of Temperature and H2O Concentration in Premixed CH4/Air Flame Using Two Partially Overlapped H2O Absorption Signals in the Near Infrared Region // Applied Sciences. — 2021. — Vol. 11. — Article 3701.

134. Soltoggio, A., Stanley, K. O., Risi, S. Born to learn: the inspiration, progress, and future of evolved plastic artificial neural networks // Neural Networks. — 2018. — Vol. 108. — P. 48–67.

135. Souza, C., Pizzolato, E., Mendes, R., Borghi-Silva, A. Artificial neural networks prognostic evaluation of post-surgery complications in patients underwent to coronary artery bypass graft surgery // Proceedings of the International Conference on Machine Learning and Applications, 2009.

136. Stanley, K., Clune, J., Lehman, J., Miikkulainen, R. Designing neural networks through neuroevolution // Nature Machine Intelligence. — 2019. — Vol. 1. — P. 24–35.

137. Stenkin, D., Gorbachenko, V. Mathematical Modeling on a Physics-Informed Radial Basis Function Network // Mathematics. – 2024. – Vol. 12, No. 2. – P. 241.

138. Sun, L., Gao, H., Pan, S., Wang, J. Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data // Computational Methods in Applied Mechanics and Engineering. — 2020. — Vol. 361. — Article 112732.

139. Tang, S., Feng, X., Wu, W., Xu, H. Physics-informed neural networks combined with polynomial interpolation to solve nonlinear partial differential equations // Computers and Mathematics with Applications. — 2023. — Vol. 132. — P. 48–62.

140. Tarasenko, F. D., Tarkhov, D. A. Basis functions comparative analysis in consecutive data smoothing algorithms // In: Cheng, L., Liu, Q., Ronzhin, A. (eds.) Proceedings of the 13th International Symposium on Neural Networks (ISNN 2016), St. Petersburg, Russia, July 6–8, 2016. — Cham : Springer, 2016. — Lecture Notes in Computer Science, Vol. 9719. — P. 482–489.

141. Tarkhov, D. A., Vasilyev, A. N. Mathematical Models of Complex Systems on the Basis of Artificial Neural Networks // Nonlinear Phenomena in Complex Systems. — 2014. — Vol. 17. — P. 327–335.

142. Tarkhov, D., Lazovskaya, T., Antonov, V. Adapting PINN Models of Physical Entities to Dynamical Data // Computation. — 2023. — Vol. 11. — Article 168.

143. Tarkhov, D. A., Vasilyev, A. N. The Construction of the Approximate Solution of the Chemical Reactor Problem Using the Feedforward Multilayer Neural Network // Proceedings of the International Conference on Neuroinformatics, 2020. — Vol. 856. — P. 41.

144. Tarkhov, D., Vasilyev, A. New neural network technique to the numerical solution of mathematical physics problems. I: Simple problems // Optical Memory and Neural Networks (Information Optics). — 2005. — Vol. 14. — P. 59–72.

145. Tarkhov, D., Vasilyev, A. New neural network technique to the numerical solution of mathematical physics problems. II: Complicated and nonstandard problems // Optical Memory and Neural Networks (Information Optics). — 2005. — Vol. 14. — P. 97–122.

146. Tarkhov, D., Vasilyev, A. N. Semi-Empirical Neural Network Modeling and Digital Twins Development. — Cambridge, MA, USA : Academic Press, 2019.

147. Tian, Y., Zhang, X., Wang, C., Jin, Y. An Evolutionary Algorithm for Large-Scale Sparse Multiobjective Optimization Problems // IEEE Transactions on Evolutionary Computation. — 2020. — Vol. 24. — P. 2380–2393.

148. Tikhonov, A. N., Arsenin, V. Y. Solutions of Ill-Posed Problems. — New York, NY, USA : Winston, 1977.

149. Trogdon, T., Biondini, G. Evolution partial differential equations with discontinuous data // Quarterly of Applied Mathematics. — 2019. — Vol. 77. — P. 689–726.

150. Wang, Y., Chen, S., Wu, X. A rational spectral collocation method for solving a class of parameterized singular perturbation problems // Journal of Computational and Applied Mathematics. — 2010. — Vol. 233. — P. 2652–2660.

151. Wang, K., Deng, P., Liu, R., Ge, C., Wang, H., Chen, P. A Novel Understanding of the Thermal Reaction Behavior and Mechanism of Ni/Al Energetic Structural Materials // Crystals. — 2022. — Vol. 12. — Article 1632.

152. Wang, J., Li, Y., Gao, R. X., Zhang, F. Hybrid physics-based and data-driven models for smart manufacturing: Modelling, simulation, and explainability // Journal of Manufacturing Systems. — 2022. — Vol. 63. — P. 381–391.

153. Wang, H., Li, B., Gong, J., Xuan, F. G. Machine learning-based fatigue life prediction of metal materials: Perspectives of physics-informed and data-driven hybrid methods // Engineering Fracture Mechanics. — 2023. — Vol. 284. — Article 109242.

154. Wang, R., Maddix, D., Faloutsos, C., Wang, Y., Yu, R. Bridging Physics-based and Data-driven modeling for Learning Dynamical Systems // Proceedings of the 3rd Conference on Learning for Dynamics and Control, PMLR, Virtual, 7–8 June 2021. — Vol. 144. — P. 385–398.

155. Wu, C., et al. A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks // Computer Methods in Applied Mechanics and Engineering. — 2023. — Vol. 403. — Article 115671.

156. Vasilyev, A. N., Tarkhov, D. A. Mathematical models of complex systems on the basis of artificial neural networks // Nonlinear Phenomena in Complex Systems. — 2014. — Vol. 17, No. 3. — P. 327–335.

157. Vasilyev, A., Tarkhov, D., Lazovskaya, T. Comparison of Multilayer Methods of Solutions to the Spectral Cauchy Problem for the Wave Equation // In: Sukhomlin, V., Zubareva, E. (eds.) Modern Information Technology and IT Education. Proceedings of the International Conference SITITO 2017. — Cham : Springer, 2021. — Communications in Computer and Information Science, Vol. 1204.

158. Viguerie, A., Lorenzo, G., Auricchio, F., Baroli, D., Hughes, T. J. R., Patton, A., Reali, A., Yankeelov, T. E., Veneziani, A. Simulating the spread of COVID-19 via a spatially-resolved susceptible–exposed–infected–recovered–deceased (SEIRD) model with heterogeneous diffusion // Applied Mathematics Letters. — 2021. — Vol. 111. — Article 106617.

159. Vadyala, S. R., Betgeri, S. N., Matthews, J. C., Matthews, E. A review of physics-machine learning in civil engineering // Results in Engineering. — 2022. — Vol. 13. — Article 100316.

160. Xu, H., Zhang, W., Wang, Y. Explore missing flow dynamics by physics-informed deep learning: The parameterized governing systems // Physics of Fluids. — 2021. — Vol. 33. — Article 095116.

161. Xue, Y., Tong, Y., Neri, F. An ensemble of differential evolution and Adam for training feed-forward neural networks // Information Sciences. — 2022. — Vol. 608. — P. 453–471.

162. Xue, Y., Wang, Y., Liang, J. A self-adaptive gradient descent search algorithm for fully-connected neural networks // Neurocomputing. — 2022. — Vol. 478. — P. 70–80.

163. Yadav, N., Yadav, A., Deep, K. Artificial neural network technique for solution of nonlinear elliptic boundary value problems // Advances in Intelligent Systems and Computing. — 2015. — Vol. 335. — P. 113–121.

164. Yang, Z., Qiu, Z., Fu, D. DMIS: Dynamic mesh-based importance sampling for training physics-informed neural networks // Proceedings of the AAAI Conference on Artificial Intelligence. — 2023. — Vol. 37, No. 4.

165. Yu, X., Gen, M. Introduction to Evolutionary Algorithms. — London, UK : Springer Science & Business Media, 2010.

166. Zhang, D., Del Rio-Chanona, E. A., Petsagkourakis, P., Wagner, J. Hybrid physics-based and data-driven modeling for bioprocess online simulation and optimization // Biotechnology and Bioengineering. — 2019. — Vol. 116. — P. 2919–2930.

167. Zhang, D., Lu, L., Guo, L., Karniadakis, G. E. Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems // Journal of Computational Physics. — 2019. — Vol. 397. — Article 108850.

168. Zhang, Z., Rai, R., Chowdhury, S., Doermann, D. MIDPhyNet: Memorized Infusion of Decomposed Physics in Neural Networks to Model Dynamic Systems // Neurocomputing. — 2021. — Vol. 428. — P. 116–129.

169. Zhang, Z., Sun, C. Structural damage identification via physics-guided machine learning: A methodology integrating pattern recognition with finite element model updating // Structural Health Monitoring. — 2020.

170. Zhao, X., Gong, Z., Zhang, Y., Yao, W., Chen, X. Physics-informed convolutional neural networks for temperature field prediction of heat source layout without labeled data // Engineering Applications of Artificial Intelligence. — 2023. — Vol. 117A. — Article 105516.

171. Zobeiry, N., Humfeld, K. D. A Physics-Informed Machine Learning Approach for Solving Heat Transfer Equation in Advanced Manufacturing and Engineering Applications // Engineering Applications of Artificial Intelligence. — 2021. — Vol. 101. — Article 104232.