ЧЕБЫШЕВСКИЙ СБОРНИК
Том 25. Выпуск 5.

УДК 517 DOI 10.22405/2226-8383-2024-25-5-126-139

Численный метод решения интегральных уравнений Фредгольма и Вольтерра с помощью искусственных нейросетей[1]

Т. Д. Нгуен, И. З. Ахметов, А. Ф. Галимянов

Нгуен Тиен Дык — аспирант, Казанский (Приволжский) федеральный университет (г. Казань); Колледж промышленных технологий (г. Бакзанг, Вьетнам).
e-mail: ducnt@bcit.edu.vn
Ахметов Ильшат Зуфарович —- аспирант, Казанский (Приволжский) федеральный университет (г. Казань).
e-mail: ilshat.achmetov@gmail.com
Галимянов Анис Фуатович —- кандидат физико-математических наук, Казанский (Приволжский) федеральный университет (г. Казань).
e-mail: anis_59@mail.ru

Аннотация

Многие задачи математики, механики, физики и других инженерных дисциплин приводят к уравнениям, в которых неизвестная функция стоит под знаком интеграла. Интегральные уравнения являются полезными математическими инструментами во многих областях, поэтому они исследуются во многих различных аспектах, таких, как существование решений, аппроксимация решений, расчет поправки или неисправимости, корректировка решений и т. д. Во многих статьях упоминаются так называемые PINN (physics-informed neural networks, что можно перевести как физически обусловленные нейронные сети), которые нашли применение для решения дифференциальных уравнений, как обыкновенных, так и в частных производных, а также систем дифференциальных уравнений. PINN также применяются для решения интегральных уравнений, однако, в публикациях обычно приводятся методы для решения некоторого класса уравнений, к примеру, уравнения Фредгольма 2-го рода либо уравнения Вольтерра 2-го рода. В этой статье будет описан общий метод решения непрерывных интегральных уравнений с помощью нейросетей, который обобщает их как для интегральных уравнений Фредгольма, так и для интегральных уравнений Вольтерра. Суть метода заключается в том, что искомая функция аппроксимируется нейронной сетью, которая по сути является огромной функцией с большим числом настраиваемых параметров, которые выбираются из условия минимальности квадрата невязки, для чего параметры нейронной сети настраиваются с помощью алгоритма оптимизации L-BFGS. Результаты метода ANN сравниваются с точным решением для нескольких типовых интегральных уравнений.

Ключевые слова: численные методы, интегральные уравнения, уравнения Фредгольма и Вольтерры, приближение функций, нейронные сети.

Библиография: 20 названий.

Для цитирования:
Нгуен, Т. Д., Ахметов, И. З., Галимянов, А. Ф. Численный метод решения интегральных уравнений Фредгольма и Вольтерра с помощью искусственных нейросетей // Чебышевский сборник, 2024, т. 25, вып. 5, с. 126–139.

[1]Работа выполнена за счет средств Программы стратегического академического лидерства Казанского (Приволжского) федерального университета («ПРИОРИТЕТ-2030»).

CHEBYSHEVSKII SBORNIK

Vol. 25. No. 5.

UDC 517 DOI 10.22405/2226-8383-2024-25-5-126-139

Numerical method for solving Fredholm and Volterra integral equations using artificial neural networks

T. D. Nguyen, I. Z. Akhmetov, A. F. Galimyanov

Nguyen Tien Duc — postgraduate student, Kazan (Volga) Federal University (Kazan); College of Industrial Techniques (Bac Giang, Vietnam).

e-mail: ducnt@bcit.edu.vn

Akhmetov Ilshat Zufarovich — postgraduate student, Kazan (Volga) Federal University (Kazan).

e-mail: ilshat.achmetov@gmail.com

Galimyanov Anis Fuatovich — candidate of physical and mathematical sciences, Kazan (Volga) Federal University (Kazan).

e-mail: anis_59@mail.ru

Abstract

Many problems in mathematics, mechanics, physics and other engineering disciplines lead to equations in which the unknown function appears under the integral sign. Integral equations are useful mathematical tools in many fields, so they are studied in many different aspects, such as the existence of solutions, approximation of solutions, calculation of correction or incorrigibility, correction of solutions, etc. Many articles mention the so-called PINN (physics-informed neural networks, which can be translated as physically conditioned neural networks), which have found application for solving differential equations, both ordinary and partial derivatives, as well as systems of differential equations. PINNs are also used to solve integral equations, but publications usually provide methods for solving a certain class of equations, for example, the Fredholm equation of the 2nd kind or the Volterra equation of the 2nd kind. This article will describe a general method for solving continuous integral equations using neural networks that generalizes them to both Fredholm and Volterra integral equations. The essence of the method is that the desired function is approximated by a neural network, which is essentially a huge function with a large number of adjustable parameters, which are selected from the condition of minimal squared residual, for which the parameters of the neural network are adjusted using the L-BFGS optimization algorithm. The results of the ANN method are compared with the exact solution for several typical integral equations.

Keywords: Numerical methods, integral equations, Fredholm and Volterra equations, approximation of functions, neural networks.

Bibliography: 20 titles.

For citation:

Nguyen, T. D., Akhmetov, I. Z. Galimyanov, A. F. 2024, "Numerical method for solving Fredholm and Volterra integral equations using artificial neural networks" , *Chebyshevskii sbornik*, vol. 25, no. 5, pp. 126–139.

1. Введение

Теория интегральных уравнений представляет собой широкую область математического анализа и имеет большое теоретическое и практическое значение. Их систематическая теория основана на работах В. Вольтерра (1860-1940), Э. И. Фредгольма (1866-1927), Д. Гильберта (1862-1943) и других математиков. Многие задачи механики, физики и других областей науки приводят к линейным интегральным уравнениям [1, 2], например, проблема Абеля, задача равновесия нагруженной струны, задачи о свободны, вынужденных колебаниях струны. В частности, задача о свободных и вынужденных колебаниях струн привела к задачам Фредгольма первого и второго рода. За долгие годы разработано множество различных алгоритмов для численного решения различных классов интегральных уравнений, таких как метод квадратур, метод Галеркина, метод разложения Тейлора [3, 4], метод гомотопического анализа [5-7], метод полиномов Лежандра [8, 9], метод полиномов Бернштейна [10] и метод разложения Адомиана [11]. Искусственные нейронные сети (ИНС), как известно, очень хорошо могут аппроксимировать неизвестные функции, поскольку сами являются по сути очень сложными функциями. В настоящее время у нейронных сетей много приложений. К примеру, они могут классифицировать тексты, изображения. Как выглядят такие функции, выражаются ли они через элементарные функции неясно, но тем не менее как показывает практика они хорошо справляются с такими задачами. В таком случае возникает вопрос: если ИНС хорошо справляются с аппроксимацией неизвестных функций, быть может, они могут достаточно хорошо аппроксимировать решение интегрального уравнения? В последнее время появилось много исследовательских работ по решению интегральных и дифференциальных уравнений с помощью нейронных сетей [12 - 19]. В англоязычной литературе они часто относятся к так называемым PINN (physical informed neural networks) [20]. Однако, в публикациях приводятся методы для каждого из уравнений Фредгольма и Вольтерра в отдельности. К тому же есть необходимость сравнить точность аппроксимации, выполненной с помощью ANN с классическими методами численного решения интегральных уравнений. В этой статье будет описан общий метод решения непрерывных интегральных уравнений с помощью нейросетей, который обобщает их как для уравнений Фредгольма, так и для уравнений Вольтерра. Так же будет показано, что данный метод применим для решения произвольных нелинейных интегральных уравнений при некоторых условиях, таких как существование и единственность решения, непрерывность ядра и т.д. Для демонстрации эффективности метода проведены численные эксперименты, разработана программа на языке Python на основе фреймворка машинного обучения PyTorch.

2. Постановка задачи

В данной статье рассматривается решение интегральных уравнений Фредгольма 1 и 2 рода, а также уравнения Вольтерры 1 и 2 рода. Как скоро станет ясно, для нейросетевого метода неважно, к какому классу принадлежит интегральное уравнение. Это одно из достоинств данного метода – универсальность. Метод может быть легко обобщен на функции многих переменных. Для начала напомним, как выглядит интегральное уравнение Фредгольма 2-го рода:

$$y(x) = \int_a^b K(x,s)y(s)ds + f(x)$$

где $K(x,s)$ – ядро интегрального уравнения.

Интегральное уравнение Вольтерры 2-го рода:

$$y(x) = \int_a^x K(x,s)y(s)ds + f(x)$$

Обозначим интегральный оператор как $(Iy)(x)$. Для уравнения Фредгольма он будет равен $\int_a^b K(x,s)y(x)ds$, а для уравнения Вольтерры - $\int_a^x K(x,s)y(x)ds$. Оставшуюся часть уравнения обозначим через оператор $F(y,f)(x) = y(x) - f(x)$. Тогда интегральное уравнение можно записать в виде $F(y,f)(x) - (Iy)(x) = 0$. В данной форме с помощью интегрального оператора можно записать как уравнение Фредгольма, так и уравнение Вольтерры, поскольку через (Iy)(x) можно обозначить интегральный оператор для интегральных уравнений обоих типов. Аналогично можно выразить интегральные уравнения первого рода. Мы выбрали такую форму записи, так как нас будет интересовать минимизация квадрата невязки в общем случае.

3. Описание метода

Предлагаемый метод основан на идее, что если норма невязки уравнения на области $C[a,b]$ определения стремится к нулю при замене неизвестной функции $y(x)$ на ее нейросетевую аппроксимацию $NN(x)$, то $NN(x)$ – приближенное решение интегрального уравнения на данной области, то есть если выражение

$$\big|\big|F(y,f)(x) - (Iy)(x))\big|\big|_{(C[a,b])}$$

стремится к нулю, то должно быть, что $NN(x) \sim y(x)$, то есть нейросетевая аппроксимация сходится к истинному решению. Обозначение $NN(x)$ выбрано не случайно, оно означает нейронную сеть, то есть нейронную сеть. Для примера рассмотрим интегральное уравнение Фредгольма 2-го рода:

$$y(x) = \sin(\pi x) + \frac{1}{2}\int_0^1 y(t)dt, x \in [a,b]$$

В таком случае необходимо минимизировать следующее выражение:

$$agrmin_{NN(x)}\big|\big|NN(x) - \sin(\pi x) - \frac{1}{2}\int_0^1 y(t)dt\big|\big|_{C[a,b]}$$

то есть максимальную норму невязки. Обозначим невязку приближенного решения как

$$r\big[\widetilde{u}\big] = F\big(\widetilde{u}, f\big) - \big(I\widetilde{u}\big)$$

Из теории интегральных уравнений известно, что если $\widetilde{u}$ - приближенное решение интегрального уравнения Фредгольма второго рода $u = Ku + f$, уравнение имеет единственное решение u и справедливо

$$\big|\big|u\big|\big|_{C[a,b]} \leq C_1\big|\big|f\big|\big|_{C[a,b]}$$

для некоторого $C_1 > 0$, то справедливо неравенство

$$\big|\big|u - \widetilde{u}\big|\big|_{C[a,b]} \leq C_1\big|\big|r\big[\widetilde{u}\big]\big|\big|_{C[a,b]}$$

Следовательно, об успешной сходимости метода для уравнения Фредгольма можно судить по значению нормы невязки $\big|\big|u - \widetilde{u}\big|\big|_{C[a,b]}$ при выполнении определенных условий.

Таким образом, неизвестная функция будет аппроксимироваться полносвязной нейронной сетью. В данном исследовании количество скрытых слоев равно 3. На вход нейросетевой функции подается значение $x \in R$, на выходе – ответ $NN(x) \in R$. В данной статье функция активации внутреннего слоя - $Tanh(x)$, количество нейронов на внутреннем слое равно 50. Для оптимизации использован метод L-BFGS. У нейросети имеется большое количество параметров, настраиваемых в ходе обучения. Покажем, как найти их количество на примере

нейросети с двумя скрытыми слоями. Пусть x – значение, поданное на вход. Оно же будет значением функции активации в первом слое.

Пусть $a_j^l(z_j^l)$ - значение функция активации j-го нейрона в l-слое. Значение z_j^l получается, как $z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$ где a_k^{l-1} - значение функции активации k-го нейрона в $(l-1)$ - м слое. Значения w_{jk}^l и b_j^l- параметры весов и смещений на слое l. Они настраиваются в ходе обучения нейронной сети. Если количество нейронов и слоев велико, то, как можно догадаться, таких параметров будет очень много. Для примера рассмотрим нейросеть с двумя скрытыми слоями, в каждом из которых 50 нейронов, с одним входом и одним выходом размерности 1. Количество связей $w_j k^1$ между входным и первым слоем будет 50 и еще 50 параметров смещения b_j^l. Затем нужно установить связь между каждой парой нейронов первого и второго слоя. Это будет 50×50=2500. Плюс параметры смещения – это еще 50 параметров. На выходном слое будет 50 параметров весов (связей) + 1 параметр смещения. Всего в сумме 2601 параметр. В итоге выходит, что решение ищется в пространстве функций с 2601 переменными. В нашем случае применения нейросети с будет применяться нейросеть с 3 слоями. Ниже на рис 1 приведена структура используемой нейросети.

Входом является $x \in R$, выходом является значение

$$y_N(x, \Omega) = \sum_{j=1}^{m} w_j^N a_j^N(z_j^{N-1})$$

Определим функционал, так называемую функцию потерь, с помощью которой в ходе обучения будут настраиваться параметры нейросети следующим образом. Пусть нужно найти решение на области $x \in [a, b]$. Выберем на этой области некоторое количество точек K, получим массив значений $[x_1, x_2, \dots x_K]$, таких, что $x_1 = a$ и $x_K = b$. В данной статье использовано равномерное разбиение, хотя оно не обязано быть таковым. Тогда функцией потерь будет среднее значение квадрата невязки:

$$Loss(\widetilde{u}) = \frac{\sum_k \left(F(\widetilde{u}, f)(x_j) - (I\widetilde{u})(x_j)\right)^2}{K} \tag{1}$$

Именно это выражение будет минироваться в ходе обучения, а не среднее абсолютного значения невязки либо максимальное значение модуля невязки. Был выбран именно квадрат невязки, а не абсолютное значение, поскольку квадрат аргумента – выпуклая функция, следовательно такая функция более склонна к достижению глобального минимума, чем модуль значения.

Таким образом, фактически мы перешли к задаче минимизации некоторого функционала $Loss(\widetilde{u})$. Заметим, что для данного метода не важен тип интегрального уравнения и его вид, алгоритм применения одинаков как для линейных, так и для нелинейных интегральных уравнений. Однако оценивать эффективность алгоритма, сходимость будем по максимальному абсолютному значению невязки, т.е. с помощью выражения:

$$\left|\left|F(c, f)(x) - (I\widetilde{u})(x)\right|\right|_{C[a,b]} \tag{2}$$

Это обусловлено тем, что в отдельных точках модуль невязки может быть очень большим, в то время как в остальных точках значение модуля невязки может быть близко к нулю, из-за чего среднее значение квадрата невязки так же будет небольшим, что приведет к ошибочному выводу, что алгоритм сошелся к истинному решению.

Однако оптимизировать веса нейронной сети с помощью функционала, где вместо среднего квадрата используется максимальная норма – плохая идея, так как в таком случае на каждой итерации алгоритм будет подгонять параметры только под одну некоторую точку

x_i, на которой норма невязки максимальна, игнорируя ошибки на всех остальных точках. Эксперименты, показывают, что такой функционал для оптимизации крайне неэффективен.

Пусть всего имеется N так называемых эпох. Это последовательные итерации, на каждой эпохе алгоритм пытается минимизировать функцию потерь для всех значений x из области определения X. Можно делать это последовательно для каждой точки x, можно сразу для некоторого подмножества, а можно минимизировать одновременно для всех точек $x \in X$, если все элементы могут поместиться в память. Поскольку в отличии от, например, изображений либо больших текстов несколько десятков чисел легко помещается в память, то нет необходимости вычислять среднее значение квадрата невязки отдельно для каждой точки x, можно минимизировать одновременно по всем точкам x_i, то есть с помощью выражения (1).

Таким образом, среднее значение квадрата невязки вычисляется сразу на всей области определения, а не на некотором ее подмножестве. Здесь K – количество точек для обучения. На каждой итерации вычисляется градиент данной функции потерь по параметрам нейросети, то есть весам w^l_{jk} и смещениям b^l_j, и тем самым изменяются параметры нейросети в сторону уменьшения функции потерь $Loss(x)$:

$$w_{new} = w_{old} - lr * H(Loss(\widetilde{u}))_w$$

где lr (learning rate) - строго положительная константа - скорость обучения, от которой зависит скорость обновление параметров нейросети и сходимость метода, $H(Loss(\widetilde{u}))_w$ – аппроксимация Гессиана функции потерь по параметрам нейросети с помощью алгоритма L-BFGS.

В отличии от традиционных численных методов решения интегральных уравнений, таких как метод квадратур, метод простой итерации, метод Галеркина и т.д, нейросетевой метод является скорее эвристическим, сильно зависящим от выбранных гиперпараметров обучения, а также метода оптимизации. Гиперпараметры в оптимизации нейронных сетей – параметры, выбираемые до начала обучения и не изменяющиеся на протяжении всей процедуры оптимизации. Для оптимизации параметров нейросетевой модели был выбран метод L-BFGS. Так же важную роль играет архитектура нейросети, выбранная для аппроксимации искомой функции. В данной работе она очень проста. В качестве функций активации, как было отмечено раннее выбран Tanh. Строго не рекомендуется использовать в качестве функции активации такие функции, как ReLu, т.к они не имеют непрерывных вторых производных и являются по сути псевдолинейными, так как $ReLU(X) = [x\ if\ x > 0\ else\ 0]$.

Так же стоит отметить, что в отличии от сеточных методов добавление новых узлов интегрирования обходится значительно дороже, так как для каждого узла необходимо итеративно выполнять операции forwardpropagation и backpropagation для обновления параметров нейросети. Поэтому здесь следует рассмотреть возможность применения квадратур Гаусса вместо, например, метода средних прямоугольников. В данной работе в различных экспериментах для аппроксимации интегрального оператора в точке были использованы как квадратуры Гаусса, так и метод средних прямоугольников.

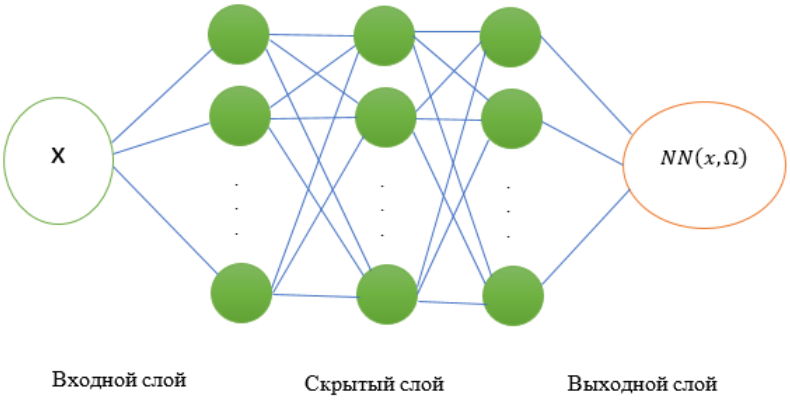

Рис. 1: Архитектура нейронной сети.

4. Численные эксперименты

Для проверки эффективности метода была написана программа для решения интегральных с помощью нейросетей. Для разработки программы использован фреймворк для глубокого обучения PyTorch и проведен ряд экспериментов. Далее будут показаны результаты некоторых из них. Узлы интерполяции были выбраны наиболее примитивно, они представляют собой равномерную сетку. Для оптимизации использован L-BFGS. Приближенное значение интегралов считается с помощью метода средних прямоугольников с выбранным количеством узлов. Узлы получаются путем генерации равномерной сетки. Через $\left|\left|R\right|\right|_{C[a,b]}$ будем обозначать максимальное абсолютное значение невязки на области определения искомой функции, т.е выражение (2). Погрешность будет округляться до третьего знака после запятой. Так же приведем максимальную абсолютную разность между точным решением и аппроксимацией, обозначив ее как $\left|\left|y(x) - NN(x)\right|\right|_{C[a,b]}$.

Пример 1. Рассмотрим интегральное уравнение Фредгольма 2-го рода.

$$y(x) = \sin(\pi x) + \frac{1}{2}\int_0^1 y(t)dt,\ x \in [0,1]$$

Аналитическое решение: $y(x) = \sin(\pi x) + \frac{2}{\pi}$

Мы обучаем сеть на 20 точках в области $[0;1]$ с одним, двумя и тремя скрытыми слоями соответственно. Количество нейронов в слое 10, эпох $= 20$, $lr = 0.225$. В таблице 1 показано сравнение решения методом ANN и аналитического решения при разных количествах скрытых слоев. График абсолютной ошибки между решением ANN и аналитическим решением показан на рис.2. На рис.3 показан график убывания нормы невязки в зависимости от эпохи минимизации. Для аппроксимации интеграла в уравнении были использованы квадратуры Гаусса с 6 узлами. Норма невязки для трехслойной сети: $\left|\left|r[\widetilde{u}]\right|\right|_{C[0,1]} \sim 0.0001$. График ниже может ввести в заблуждение, что с ростом количества скрытых слоев нейросеть все точнее аппроксимирует решение уравнения. Но это далеко не всегда так, причем начиная с некоторого числа скрытых слоев точность уже убывает при прочих равных параметрах. Например, если сделать число слоев равным четырем, то $\left|\left|r[\widetilde{u}]\right|\right|_{C[0,1]} \sim 0.0005$, а при пяти в данном случае уже имеем $\left|\left|r[\widetilde{u}]\right|\right|_{C[0,1]} \sim 0.001$.

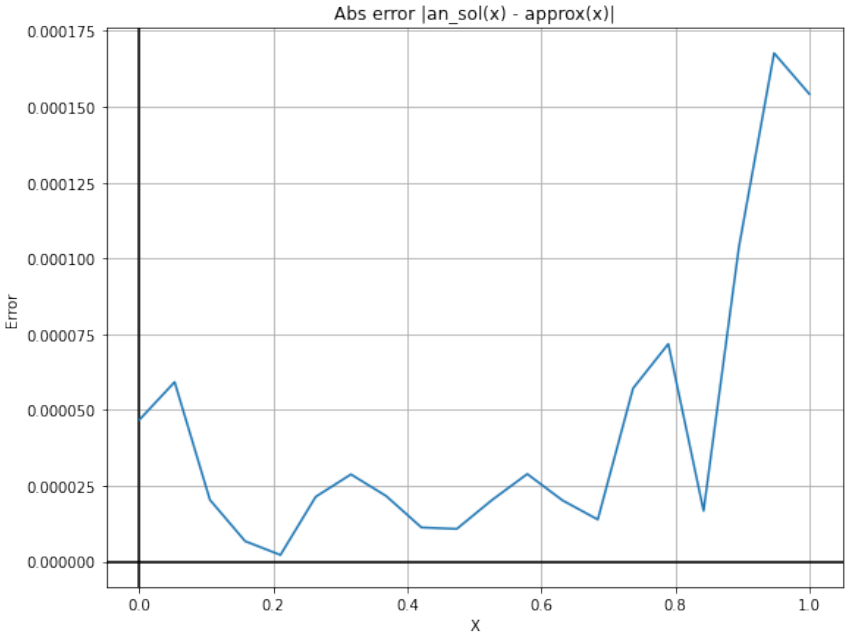

Рис. 2: Абсолютная погрешность аппроксимации (3 скрытый слой, 10 нейронов).

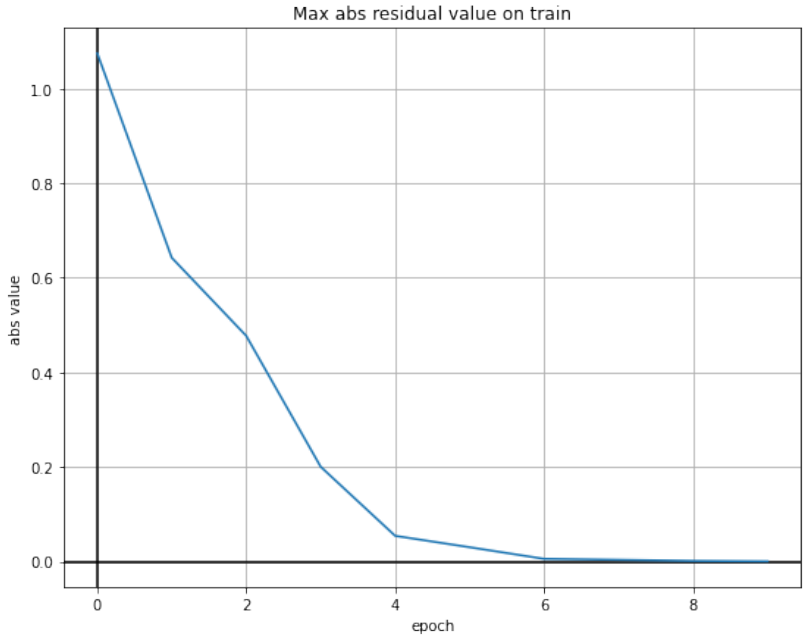

Рис. 3: График убывания значения $\left|\left|r[\widetilde{u}]\right|\right|_{C[0,1]}$ в зависимости от эпохи.

Таблица 1: Сравнение результатов ANN и аналитического решения с 1 скрытым слоем, 2 скрытыми слоями и 3 скрытыми слоями.

X	Analytical	ANN1	Error1	ANN2	Error2	ANN3	Error3
0	0.636620	0.639208	2.59E-03	0.636351	2.69E-04	0.636573	4.67E-05
0.052632	0.801214	0.800647	5.67E-04	0.801588	3.73E-04	0.801274	5.92E-05
0.105263	0.961319	0.959307	2.01E-03	0.961694	3.75E-04	0.961340	2.03E-05
0.157895	1.112567	1.110564	2.00E-03	1.112676	1.09E-04	1.112561	6.68E-06
0.210526	1.250833	1.249789	1.04E-03	1.250660	1.73E-04	1.250835	2.15E-06
0.263158	1.372344	1.372611	2.67E-04	1.372025	3.18E-04	1.372365	2.13E-05
0.315789	1.473786	1.475168	1.38E-03	1.473517	2.69E-04	1.473815	2.87E-05
0.368421	1.552393	1.554311	1.92E-03	1.552342	5.08E-05	1.552415	2.16E-05
0.421053	1.606020	1.607737	1.72E-03	1.606253	2.33E-04	1.606031	1.12E-05
0.473684	1.633204	1.634064	8.59E-04	1.633653	4.48E-04	1.633215	1.07E-05
0.526316	1.633204	1.632827	3.77E-04	1.633682	4.78E-04	1.633225	2.03E-05
0.578947	1.606020	1.604424	1.60E-03	1.606305	2.85E-04	1.606049	2.88E-05
0.631579	1.552393	1.550009	2.38E-03	1.552326	6.70E-05	1.552413	2.01E-05
0.684211	1.473786	1.471365	2.42E-03	1.473370	4.17E-04	1.473772	1.38E-05
0.736842	1.372344	1.370755	1.59E-03	1.371777	5.67E-04	1.372287	5.71E-05
0.789474	1.250833	1.250778	5.48E-05	1.250462	3.71E-04	1.250761	7.18E-05
0.842105	1.112567	1.114231	1.66E-03	1.112725	1.58E-04	1.112550	1.67E-05
0.894737	0.961319	0.963988	2.67E-03	0.962066	7.47E-04	0.961423	1.04E-04
0.947368	0.801214	0.802893	1.68E-03	0.802000	7.86E-04	0.801382	1.68E-04
1	0.636620	0.633683	2.94E-03	0.635902	7.18E-04	0.636466	1.54E-04

Пример 2. Мы рассмотрим уравнение Вольтерры

$$y(x) = \sin(x) + \int_0^x \sin(x-t)y(t)dt,\ x \in [0,1]$$

Аналитическое решение:

$$y(x) = x.$$

Мы обучаем сеть на 20 точках. Сеть имеет три скрытых слоя. Количество нейронов в слое = 10, эпох = 10, lr = 0.5. В таблице 2 приводится сравнение аппроксимации ANN и аналитического решения. Для аппроксимации интеграла используется формула средних прямоугольников с 50 узлами. На рисунке 4 приведен график абсолютной ошибки.

$$\left|\left|r[\widetilde{u}]\right|\right|_{C[0,1]} \sim 0.0005$$

Таблица 2: Сравнение результатов ANN и аналитического решения.

X	Analytical	ANN	Error
0	0	-0.00018	1.78E-04
0.052632	0.052632	0.052664	3.26E-05
0.105263	0.105263	0.105420	1.57E-04
0.157895	0.157895	0.158097	2.02E-04
0.210526	0.210526	0.210708	1.81E-04
0.263158	0.263158	0.263273	1.15E-04
0.315789	0.315789	0.315813	2.34E-05
0.368421	0.368421	0.368349	7.20E-05
0.421053	0.421053	0.420901	1.52E-04
0.473684	0.473684	0.473486	1.98E-04
0.526316	0.526316	0.526114	2.02E-04
0.578947	0.578947	0.578789	1.58E-04
0.631579	0.631579	0.631509	7.00E-05
0.684211	0.684211	0.684261	5.02E-05
0.736842	0.736842	0.737022	1.80E-04
0.789474	0.789474	0.789763	2.89E-04
0.842105	0.842105	0.842439	3.33E-04
0.894737	0.894737	0.894998	2.61E-04
0.947368	0.947368	0.947379	1.08E-05
1	1	0.999511	4.89E-04

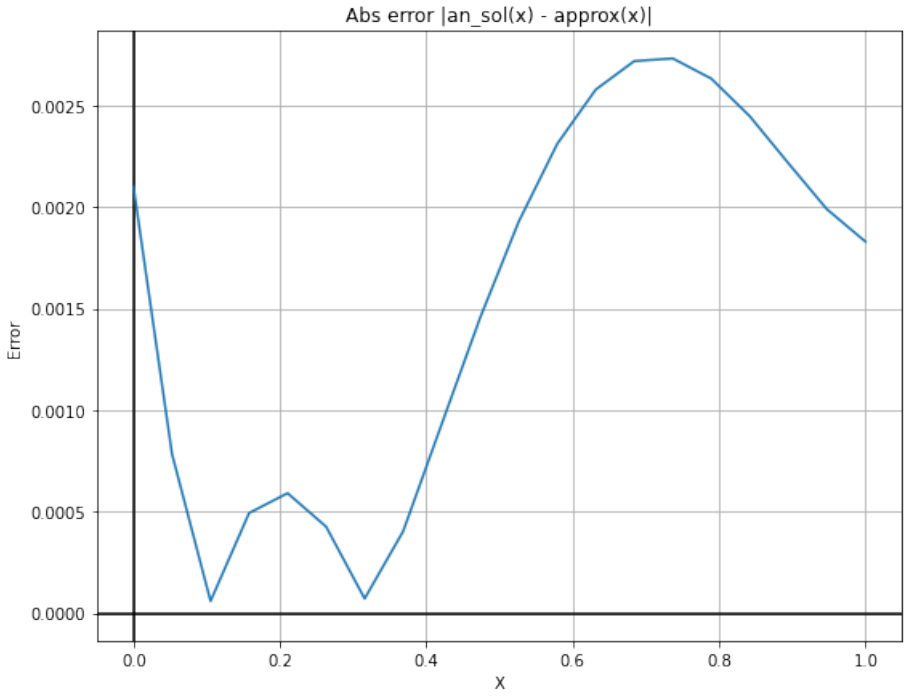

Рис. 4: Абсолютная погрешность аппроксимации (пример 2).

Были проведены и другие эксперименты. Во всех экспериментах нейросеть неплохо справлялась аппроксимацией. Как показывают эксперименты, результаты можно улучшить путем изменения гиперпараметров, например увеличением количества эпох обучения, изменением

количества скрытых слоев, нейронов в слое. Можно так же попробовать изменять learning rate, увеличить количество точек точек для обучения, использовать большее количество узлов для численного вычисления интегралов либо вычислять значение интегралов с помощью методов более высокого порядка точности. Нужно так же отметить, что классические методы решения линейных интегральных уравнений, такие, как например метод квадратур имеют погрешность порядка

$$\left|\left|u-\widetilde{u}\right|\right| \sim o(h^2)$$

где h– размер сетки. Очевидно, что при достаточно плотной сетке они гораздо точнее, чем данный метод. Приведенные в статье результаты являются одними из лучших, найденных авторами статьи для данных примеров. Однако данный метод универсален, применим для разных классов интегральных уравнений и на выходе дает готовую функцию, с помощью которой можно получить значение искомой функции в различных точках области определения, поэтому возможно, что у данного метода есть своя область применения.

Интересно заметить, что максимальная норма невязки сильно коррелирует с максимальной абсолютной разностью между точным решением и аппроксимацией, что еще раз подтверждает теоретическую оценку

$$\left|\left|u-\widetilde{u}\right|\right|_{C[a,b]} \leq C_1 \left|\left|r[\widetilde{u}]\right|\right|_{C[a,b]}$$

5. Заключение

В статье продемонстрирован численный метод решения интегральных уравнений с помощью нейросетей, проведены численные эксперименты. Как видно из результатов, нейросети вполне неплохо справляются с численным решением интегральных уравнений, причем уравнений различных классов. Несмотря на то, что в данной статье рассматривались лишь уравнения Фредгольма и Вольтерры, в целом этот метод подходит для решения произвольного нелинейного интегрального уравнения при условии, что пределы интегрирования конечны, интегральный оператор непрерывен, а подынтегральная функция ограничена.Алгоритм метода не накладывает ограничений на вид интегрального уравнения. Интуитивно ясно, что чем меньше максимальное значение невязки $\left|\left|r[\widetilde{u}]\right|\right|_{C[a,b]}$, тем ближе аппроксимация к точному решению. Однако строго доказано это лишь для уравнения Фредгольма при некоторых условиях, приведенных раннее, в других случаях это требует доказательства. Но насколько малым должен быть квадрат невязки, чтобы считать аппроксимацию нейросетью достаточно точной? Как лучше выбрать архитектуру нейросети, чтобы искусственная нейронная сеть могла в теории аппроксимировать искомую функцию? Это вопросы, требующие дополнительных исследований.

СПИСОК ЦИТИРОВАННОЙ ЛИТЕРАТУРЫ

1. Hosseini S. M., Shahmorad S. Numerical piecewise approximate solution of Fredholm integro-differential equations by the Tau method //Applied Mathematical Modelling. 2005. Т. 29. №. 11. С. 1005-1021.

2. Maleknejad K., Nouri K., Yousefi M. Discussion on convergence of Legendre polynomial for numerical solution of integral equations //Applied Mathematics and Computation. 2007. Т. 193. №. 2. С. 335-339.

3. Maleknejad K., Aghazadeh N., Rabbani M. Numerical solution of second kind Fredholm integral equations system by using a Taylor-series expansion method //Applied Mathematics and Computation. 2006. Т. 175. №. 2. С. 1229-1234.

4. Gülsu M., Sezer M. The approximate solution of high-order linear difference equations with variable coefficients in terms of Taylor polynomials //Applied mathematics and computation. 2005. Т. 168. №. 1. С. 76-88.

5. Ghasemi M., Kajani M. T., Babolian E. Numerical solutions of the nonlinear Volterra–Fredholm integral equations by using homotopy perturbation method // Applied Mathematics and Computation. 2007. Т. 188. №. 1. С. 446-449.

6. Ghasemi M., Kajani M. T., Babolian E. Numerical solutions of the nonlinear Volterra–Fredholm integral equations by using homotopy perturbation method // Applied Mathematics and Computation. 2007. Т. 188. №. 1. С. 446-449.

7. Abbasbandy S. Numerical solutions of the integral equations: Homotopy perturbation method and Adomian's decomposition method // Applied Mathematics and Computation. 2006. Т. 173. №. 1. С. 493-500.

8. Nasir A. N. K., Tokhi M. O. Novel metaheuristic hybrid spiral-dynamic bacteria-chemotaxis algorithms for global optimisation //Applied Soft Computing. 2015. Т. 27. С. 357-375.

9. Hu Y. C. Flow-based tolerance rough sets for pattern classification //Applied Soft Computing. 2015. Т. 27. С. 322-331.

10. Livi L., Rizzi A., Sadeghian A. Granular modeling and computing approaches for intelligent analysis of non-geometric data //Applied Soft Computing. 2015. Т. 27. С. 567-574.

11. Bildik N., Konuralp A., Yalçınbaş S. Comparison of Legendre polynomial approximation and variational iteration method for the solutions of general linear Fredholm integro-differential equations //Computers and Mathematics with Applications. 2010. Т. 59. №. 6. С. 1909-1917.

12. Saneifard R. Extended artificial neural networks approach for solving two-dimensional fractional-order Volterra-type integro-differential equations //Information Sciences. 2022. Т. 612. С. 887-897.

13. Jadoon I. Integrated meta-heuristics finite difference method for the dynamics of nonlinear unipolar electrohydrodynamic pump flow model //Applied Soft Computing. 2020. Т. 97. С. 106-791.

14. Livi L., Rizzi A., Sadeghian A. Granular modeling and computing approaches for intelligent analysis of non-geometric data //Applied Soft Computing. 2015. Т. 27. С. 567-574.

15. Lagaris I. E., Likas A., Fotiadis D. I. Artificial neural networks for solving ordinary and partial differential equations //IEEE transactions on neural networks. 1998. Т. 9. №. 5. С. 987-1000.

16. Koryagin A., Khudorozkov R., Tsimfer S. PyDEns: A python framework for solving differential equations with neural networks // ArXiv preprint arXiv: 1909.11544. 2019.

17. Guan Y. Solving Fredholm integral equations using deep learning //International Journal of Applied and Computational Mathematics. 2022. Т. 8. №. 2. С. 87.

18. Effati S., Buzhabadi R. A neural network approach for solving Fredholm integral equations of the second kind //Neural Computing and Applications. 2012. Т. 21. С. 843-852.

19. Jafarian A., Measoomy S., Abbasbandy S. Artificial neural networks based modeling for solving Volterra integral equations system //Applied Soft Computing. 2015. Т. 27. С. 391-398.

20. Zhang J.B., Yu D.M., Pan X.M. Physics-Informed Neural Networks For the Solution of Electromagnetic Scattering by Integral Equations //2022 International Applied Computational Electromagnetics Society Symposium (ACES-China). IEEE, 2022. С. 1-2.

REFERENCES

1. Hosseini, S.M., Shahmorad, S. 2025, "Numerical piecewise approximate solution of Fredholm integro-differential equations by the Tau method", *Applied Mathematical Modelling.*, T. 29, no. 11, pp. 1005–1021.

2. Maleknejad, K., Nouri, K., Yousefi, M. 2007, "Discussion on convergence of Legendre polynomial for numerical solution of integral equations", *Applied Mathematics and Computation.*, T. 193, no. 2. pp. 335–339.

3. Maleknejad, K., Aghazadeh, N., Rabbani, M. 2006, "Numerical solution of second kind Fredholm integral equations system by using a Taylor-series expansion method", *Applied Mathematics and Computation.*, T. 175, no 2, pp. 1229–1234.

4. Gülsu, M., Sezer, M. 2005, "The approximate solution of high-order linear difference equations with variable coefficients in terms of Taylor polynomials", *Applied mathematics and computation.*, T. 168, no. 1, pp. 76–88.

5. Ghasemi, M., Kajani, M.T., Babolian, E. 2007, "Numerical solutions of the nonlinear Volterra–Fredholm integral equations by using homotopy perturbation method", *Applied Mathematics and Computation.*, T. 188, no. 1, pp. 446–449.

6. Ghasemi, M., Kajani, M.T., Babolian, E. 2007, "Numerical solutions of the nonlinear Volterra–Fredholm integral equations by using homotopy perturbation method", *Applied Mathematics and Computation.*, T. 188, no. 1, pp. 446–449.

7. Abbasbandy, S. 2006, "Numerical solutions of the integral equations: Homotopy perturbation method and Adomian's decomposition method", *Applied Mathematics and Computation.*, T. 173, no. 1, pp. 493–500.

8. Nasir, A.N.K., Tokhi, M.O. 2015, "Novel metaheuristic hybrid spiral-dynamic bacteria-chemotaxis algorithms for global optimisation", *Applied Soft Computing.*, T. 27, pp. 357–375.

9. Hu, Y.C. 2015, "Flow-based tolerance rough sets for pattern classification", *Applied Soft Computing.*, T. 27, pp. 322–331.

10. Livi, L., Rizzi, A., Sadeghian, A. 2015, "Granular modeling and computing approaches for intelligent analysis of non-geometric data", *Applied Soft Computing.*, T. 27, pp. 567–574.

11. Bildik, N., Konuralp, A., Yalçınbaş, S. 2010, "Comparison of Legendre polynomial approximation and variational iteration method for the solutions of general linear Fredholm integro-differential equations", *Computers and Mathematics with Applications.*, T. 59, no. 6, pp. 1909–1917.

12. Saneifard, R. 2022, "Extended artificial neural networks approach for solving two-dimensional fractional-order Volterra-type integro-differential equations", *Information Sciences.*, T. 612, pp. 887–897.

13. Jadoon, I. 2020, "Integrated meta-heuristics finite difference method for the dynamics of nonlinear unipolar electrohydrodynamic pump flow model", *Applied Soft Computing.*, T. 97, pp. 106–791.

14. Livi, L., Rizzi, A., Sadeghian, A. 2015, "Granular modeling and computing approaches for intelligent analysis of non-geometric data", *Applied Soft Computing.*, T. 27, pp. 567–574.

15. Lagaris, I. E., Likas, A., Fotiadis, D. I. 1998, "Artificial neural networks for solving ordinary and partial differential equations". *IEEE transactions on neural networks.*, T. 9, no. 5, pp. 987–1000.

16. Koryagin, A., Khudorozkov, R., Tsimfer, S. 2019, "PyDEns: A python framework for solving differential equations with neural networks", *arXiv preprint arXiv:1909.11544.*

17. Guan, Y. 2022, "Solving Fredholm integral equations using deep learning", *International Journal of Applied and Computational Mathematics.*, T. 8, no 2, pp. 87.

18. Effati, S., Buzhabadi, R. 2012, "A neural network approach for solving Fredholm integral equations of the second kind", *Neural Computing and Applications.*, T. 21, pp. 843–852.

19. Jafarian, A., Measoomy, S., Abbasbandy, S. 2015, "Artificial neural networks based modeling for solving Volterra integral equations system", *Applied Soft Computing.*, T. 27, pp. 391–398.

20. Zhang, J. B., Yu, D. M., Pan, X. M. 2022, "Physics-Informed Neural Networks For the Solution of Electromagnetic Scattering by Integral Equations", *2022 International Applied Computational Electromagnetics Society Symposium (ACES-China).*, IEEE, pp. 1–2.

Получено: 17.05.2024
Принято в печать: 26.12.2024