УДК 519.633.2

ЧИСЛЕННЫЙ АЛГОРИТМ РЕШЕНИЯ ПОЛНОСТЬЮ НЕЛИНЕЙНЫХ ПАРАБОЛИЧЕСКИХ УРАВНЕНИЙ НА ОСНОВЕ ПРЯМЫХ-ОБРАТНЫХ СТОХАСТИЧЕСКИХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ И НЕЙРОННЫХ СЕТЕЙ

Автономная некоммерческая образовательная организация высшего образования
«Научно-технологический университет «Сириус» (Университет «Сириус»),
федеральная территория «Сириус», г. Сочи, Россия

Статья поступила в редакцию 08.07.2024

В статье рассматривается численный метод решения задачи Коши для полностью нелинейного параболического уравнения. Рассматриваемое уравнение сводится к системе квазилинейных параболических уравнений. Для этой системы построено вероятностное представление решения, основанное на решении системы прямого-обратного стохастических дифференциальных уравнений (ПОСДУ). Решение ПОСДУ сводится к решению оптимизационной задачи, которая численно решается с помощью нейронной сети. Рассмотрен пример применения данного метода для уравнения, описывающего цену оптимального портфеля на рынке Блэка-Шоулса. Численное решение было апробировано на специальных видах функции полезности, для которых существует точное решение.

Ключевые слова: полностью нелинейные параболические уравнения, задача Коши, стохастические дифференциальные уравнения (СДУ), задача оптимизации, глубокое обучение, нейронные сети для прямых-обратных стохастических дифференциальных уравнений.

DOI: 10.37313/1990-5378-2024-26-4-161-169
EDN: FQJDSR

Результаты получены в рамках реализации государственной программы федеральной территории «Сириус» «Научно-технологическое развитие федеральной территории «Сириус».

ВВЕДЕНИЕ

Решение задачи Коши для нелинейных уравнений в частных производных (УЧП) только в редких случаях позволяет получить решение задачи в явном виде. Применение классических разностных методов для решения нелинейных задач, особенно в многомерном случае, создает большие вычислительные трудности. Альтернативным подходом является использование вероятностных представлений решений линейных и нелинейных параболических уравнений. Этот подход позволяет найти решение задачи в одной заданной точке, а не во всей области сразу, что также положительно сказывается на вычислительной трудоемкости, в случаях, когда нам необходимо найти решение только в нескольких заданных точках.

Уточним используемую терминологию. Если коэффициенты зависят от производной (градиента) искомой функции, то такие уравнения называются *квазилинейными*. *Полностью нелинейными* называют уравнения, в которых коэффициенты зависят от второй производной (гессиана) искомой функции.

Один из подходов построения вероятностных представлений решений нелинейных параболических уравнений основан на теории обратных стохастических дифференциальных уравнений (ОСДУ, BSDE). Связь между квазилинейными УЧП и ОСДУ хорошо известна со времен пионерских работ Парду и Пенга [1, 2].

Распространение этой теории на полностью нелинейные параболические уравнения было построено в работе [3] и на системы таких уравнений в [4–6]. В этом случае получается сильно связанная система прямого-обратного СДУ (ПОСДУ, FBSDE). Обычно, чтобы получить замкнутую систему, нам нужно применить теорему Ито о представлении мартингала, но есть и другой способ. Существует ряд работ, посвященных полностью связанным ПОСДУ [7–9]. Как было показано в книге [8], решение ПОСДУ может сводиться к решению оптимизационной задачи, для решения которой оказалось эффективным применять нейронные сети.

В статьях Хорника, Стинчкомба и Вайта [10, 11] впервые показано, что сети прямого распространения с одним или несколькими скрытыми слоями и произвольной ограниченной, но непостоянной функцией активации являются универсальными аппроксиматорами, в том числе в приложениях, требующих одновременной аппроксимации функции и ее производных.

В статьях [12–16] было показано, что можно разработать эффективный алгоритм для получения численного решения задачи Коши для не-

Чубатов Андрей Алексеевич, младший научный сотрудник Научного центра информационных технологий и искусственного интеллекта. E-mail: chaa@inbox.ru

линейных параболических УЧП, объединяющий ПОСДУ подход из [3] и глубокое обучение нейронных сетей [17]. Этот новый подход к решению задачи Коши для нелинейных уравнений представляется очень интересным, поскольку он позволяет численно решать многомерные задачи, позволяя преодолевать так называемое «проклятие размерности» [18, 19].

Подход, позволяющий получить на основе ПОСДУ вероятностную интерпретацию решения задачи Коши для полностью нелинейного уравнения, состоит из следующих этапов:

1. приведение задачи Коши для полностью нелинейного УЧП к задаче Коши для системы квазилинейных параболических УЧП;

2. сведение решения задачи Коши для квазилинейной системы УЧП к решению стохастической задачи (ПОСДУ);

3. сведение решения ПОСДУ к решению оптимизационной задачи;

4. решение оптимизационной задачи с помощью нейронной сети.

1. МАТЕМАТИЧЕСКИЙ АППАРАТ И ЧИСЛЕННЫЕ МЕТОДЫ

1.1. Сведение задачи Коши для полностью нелинейного уравнения к задаче Коши для системы квазилинейных УЧП

Рассмотрим задачу Коши для многомерного полностью нелинейного параболического УЧП

$$\frac{\partial u}{\partial \tau} + f\left(x, u, \nabla u, \nabla^2 u\right) = 0, \quad u(T, x) = h(x), \quad (1)$$

где $\tau \in [T_1; \mathrm{T}]$, $x \in R^d$, $u = u(\tau, x) \in R$, $\nabla u \in R^d$ – градиент *u*, $\nabla^2 u \in R^{d \times d}$ – гессиан *u*.

Отметим, что в нелинейных задачах зачастую нет глобальных по времени решений и имеет место явление, называемое «взрыв» («blow up»), при котором решение бесконечно возрастает за конечное время. Поэтому мы рассматриваем локальные по времени решения ($\tau \in [T_1; \mathrm{T}]$). В некоторых случаях удается оценить временные промежутки $[T_1; \mathrm{T}]$, на которых решения ограничены [20].

Для применения подхода, основанного на ПОСДУ, к полностью нелинейному уравнению необходимо свести это уравнение к системе квазилинейных параболических уравнений специального вида – система должна иметь коэффициенты одинаковые для всех уравнений в главной (параболической) части.

Введем обозначение (новую функцию)

$$v = \nabla u \in R^d, \quad (2)$$

и сведем уравнение (1) к системе из двух квазилинейных (скалярного и векторного) УЧП относительно функций *u* и *v*.

Предположение С 1. Будем считать, что

1. функция $u = u(t, x)$ – классическое решение задачи Коши (1);

2. ее градиент $\nabla u(t, x) = v(t, x)$ дважды дифференцируем по *x*.

Учитывая обозначение (2), уравнение (1) можно записать в квазилинейном виде (относительно функций *u* и *v*)

$$\frac{\partial u}{\partial \tau} + f\left(x, u, v, \nabla v\right) = 0, \quad u(T, x) = h(x), \quad (3)$$

где $\nabla v = \nabla^2 u \in R^{d \times d}$.

Для получения квазилинейной параболической системы осталось

1) дописать квазилинейное уравнение для функции *v*, воспользовавшись методом дифференциального продолжения;

2) явно выделить в новом уравнении параболическую часть;

3) записать уравнение (3) с такими же коэффициентами в параболической части.

Дифференцируя уравнение (3) вместе с терминальными условиями по x_k, $k = 1, 2, \ldots, d$, после некоторых преобразований получим систему квазилинейных уравнений для функций v_k

$$\frac{\partial v_k}{\partial \tau} + \frac{\partial f}{\partial x_k} + \frac{\partial f}{\partial u} v_k + \sum_{i=1}^{d} \frac{\partial f}{\partial v_i} \frac{\partial v_k}{\partial x_i} + \sum_{i=1}^{d} \sum_{j=1}^{d} \frac{\partial f}{\partial [\nabla v]_{ij}} \left[\nabla^2 v_k\right]_{ij} = 0,$$

$$v_k(T, x) = \frac{\partial h(x)}{\partial x_k}, \quad k = 1, \ldots, d. \quad (4)$$

Приведем квазилинейную систему (4) для векторной функции *v* к форме с выделенной явно параболической частью

$$\frac{\partial v_k}{\partial \tau} + \frac{1}{2} Tr\left[B \nabla^2 v_k\right] + \psi_k\left(x, u, v, \nabla v\right) = 0,$$

$$v_k(T, x) = \frac{\partial h(x)}{\partial x_k}, \quad k = 1, \ldots, d, \quad (5)$$

где $B = B\left(x, u, v, \nabla v\right)$,

$$Tr\left[B \nabla^2 v_k\right] = 2 \sum_{i=1}^{d} \sum_{j=1}^{d} \frac{\partial f}{\partial [\nabla v]_{ij}} \left[\nabla^2 v_k\right]_{ij} \in R, \quad (6)$$

$$\psi_k = \frac{\partial f}{\partial x_k} + \frac{\partial f}{\partial u} v_k + \sum_{i=1}^{d} \frac{\partial f}{\partial v_i} \frac{\partial v_k}{\partial x_i}. \quad (7)$$

Для использования вероятностного подхода необходимо, чтобы *B* допускала разложение

$$B = AA^{+}, \quad A = A\left(x, u, v, \nabla v\right). \quad (8)$$

Замечание 1. Отметим, что для того, чтобы матрица *B* допускала разложение (8), достаточно чтобы она была симметричной и положительно определенной, т. к. разложение Холецкого всегда существует и единственно для любой симметричной положительно определённой матрицы. Если матрица *B* не допускает разложение (8), то невозможно применить вероятностную интерпретацию решения УЧП, основанную на ПОСДУ.

Из соотношения (6) получим, что

$$B_{ji} = 2\frac{\partial f}{\partial[\nabla v]_{ij}}. \quad (9)$$

Т. к. $\nabla v = \nabla^2 u$, то $[\nabla v]_{ij} = [\nabla v]_{ji}$. Отсюда следует симметричность матрицы *B*.

Замечание 2. Осталось для конкретной матрицы *B* проверить положительную определенность. Для этого можно воспользоваться, например, критерием Сильвестра.

Далее матрица *A* находится с помощью разложения Холецкого в аналитическом либо численном виде. При этом *A* – нижняя треугольная матрица со строго положительными элементами на диагонали.

Представим теперь уравнение (3) в виде с параболической частью

$$\frac{\partial u}{\partial \tau} + \frac{1}{2}Tr\left[B\nabla^2 u\right] + \varphi(x,u,v,\nabla v) = 0,$$

$$u(T,x) = h(x), \quad (10)$$

где $\varphi(x,u,v,\nabla v) = f(x,u,v,\nabla v) - \frac{1}{2}Tr[B\nabla v]$.

Уравнения (10) и (5) образуют замкнутую систему квазилинейных уравнений.

Предположение С 2. Будем считать, что

1) функции $f(x,u,v,\nabla v)$ и $h(x)$ C^1 –гладкие и ограниченные по все аргументам;

2) $\frac{\partial f}{\partial[\nabla v]} > 0$ (матрица B положительно определена).

Предположения С 1 и С 2 обеспечивают возможность сведения полностью нелинейного уравнения (1) к квазилинейной параболической системе (5), (10).

1.2. Сведение решения квазилинейной системы УЧП к решению стохастической задачи

Пусть существует разложение (8) для матрицы B и дана задача Коши для системы квазилинейных УЧП с общей параболической частью

$$\frac{\partial u}{\partial \tau} + \frac{1}{2}Tr\left[AA^+\nabla^2 u\right] + \varphi(x,u,v,\nabla v) = 0,$$

$$u(T,x) = h(x), \quad (11)$$

$$\frac{\partial v_k}{\partial \tau} + \frac{1}{2}Tr\left[AA^+\nabla^2 v_k\right] + \psi_k(x,u,v,\nabla v) = 0,$$

$$v_k(T,x) = g_k(x), \quad k = 1,\ldots,d. \quad (12)$$

Вероятностный подход, основанный на ПОСДУ, позволяет получить решение исходной задачи $u(\tau,x)$. Для этого нужно решить систему из прямого и обратных стохастических дифференциальных уравнений (ПОСДУ), построенных для случайного процесса $X(t)$, начинающегося в точке $X(\tau) = x$, и случайных процессов, определяемых функциями *u* и *v*.

Введем в рассмотрение F_t-адаптивный случайный процесс $X(t)$, $t \in [\tau;T]$, $\tau \in [T_1;T]$, заданный (прямым) СДУ

$$dX(t) = A\big(X(t),u(t,X(t)),v(t,X(t)),\nabla v(t,X(t))\big)dW(t),$$

$$X(\tau) = x, \quad (13)$$

где $W(t) \in R^d$ – винеровский случайный процесс, и введем (случайные) процессы

$$U(t) = u(t,X(t)), \quad V(t) = v(t,X(t)),$$

$$Z(t) = \nabla U(t), \quad G(t) = \nabla V(t). \quad (14)$$

Тогда по формуле Ито можно проверить, что $U(t)$ и $V(t)$ удовлетворяют следующим обратным СДУ (ОСДУ)

$$dU(t) = -\varphi\big(X(t),U(t),V(t),G(t)\big)dt +$$

$$+ Z(t)^+ A\big(X(t),U(t),V(t),G(t)\big)dW(t),$$

$$U(T) = h(X(T)), \quad (15)$$

$$dV(t) = -\psi\big(X(t),U(t),V(t),G(t)\big)dt +$$

$$+ G(t)^+ A\big(X(t),U(t),V(t),G(t)\big)dW(t),$$

$$V(T) = g(X(T)). \quad (16)$$

Перепишем уравнение (13) в виде

$$dX(t) = A\big(X(t),U(t),V(t),G(t)\big)dW(t),$$

$$X(\tau) = x. \quad (17)$$

Система уравнений (15)–(17) представляет собой прямое и обратные СДУ (ПОСДУ). Эта система является сильно связанной.

В дальнейшем мы предполагаем, что условия (предположения 1–2) из статьи [9] выполняются. Условно сформулируем их, как условие С 3.

Условие С 3. Будем говорить, что *условие С 3* выполнено если

1) функции $A(x,u,v,\nabla v)$, $\varphi(x,u,v,\nabla v)$, $\psi(x,u,v,\nabla v)$ ограничены и липшицевы по всем аргументам;

2) функции $h(x)$ и $g(x)$ ограничены и липшицевы.

Теорема 1. *Если Условие С 3 выполнено, то существует единственное решение ПОСДУ (15)–(17)* $\big(X(t),(U(t),V(t)),(Z(t),G(t))\big)$, *при этом* $u(\tau,x) = U(\tau)$.

Доказательство этого утверждения можно найти, например, в [5], [7].

1.3. Сведение решения ПОСДУ к решению оптимизационной задачи

Решение ПОСДУ может быть сведено к решению некоторой задачи стохастического оптимального управления [7, 8]. При этом численное решение ПОСДУ может быть сведено к численному решению соответствующей оптимизационной задачи с использованием нейронных сетей [9], [12–16].

В статьях [12, 13] рассматривалась функция потерь, содержащая только слагаемое в момент времени *T* (терминальное слагаемое), которое обеспечивало выполнение только терминального условия. Для оптимизационной задачи, связанной с ПОСДУ (15)–(17) эта функция потерь выглядела бы следующим образом

$$L_{TC} = E\left[\left\|U(T) - h(X(T))\right\|^2 + \left\|V(T) - g(X(T))\right\|^2\right].$$

Такой вид функции потерь использовался в статье [12] для обучения набора из $N-1$ независимой нейронной подсети, каждая из которых имеет ширину S (число нейронов в скрытом слое) равную $d+10$, где N – количество временных интервалов, на которые разбивается промежуток $[\tau, T]$.

В статье [9] в алгоритме 2 и в статье [16] рассматривалось решение квазилинейной задачи и использовалась функция потерь

$$L = L_{TC} + L_{\Sigma}, \qquad (18)$$

содержащая, как терминальное слагаемое L_{TC}, так и интегральную составляющую L_{Σ}, обеспечивающую выполнение определенных соотношений, описываемых ОСДУ, внутри временного промежутка $[\tau, T]$. Особенностью подхода, представленного в статье [16], является то, что интегральная составляющая L_{Σ} функции потерь, описывает разность между решением ОСДУ и контрольными значениями, вычисляемыми по схеме Эйлера-Маруямы. Такой новый вид функции потерь дает возможность использовать для решения задачи всего одну нейронную сеть достаточно большой ширины S, причем S не зависит напрямую от количества интервалов N во временной дискретизации промежутка $[\tau; T]$. Ширина сети S может быть подобрана экспериментально (Раисси использовал значение $S = 256$) и использоваться для решения задач для различных N, меняющихся в достаточно широком диапазоне.

Мы модифицируем функцию потерь (18), предложенную в статье [16], для её применения к решению ПОСДУ (15)–(17), ассоциированному с квазилинейной системой (11), (12).

1.4. Решение оптимизационной задачи с помощью нейронных сетей

Для решения оптимизационной задачи можно использовать специальные подтипы обучения с подкреплением (Reinforcement Learning) – так называемые нейронные сети для ОСДУ: Deep BSDE NNs [12, 13] и Forward-Backward Stochastic Neural Networks (FBSNNs) [16]. В сетях этого типа функция потерь обеспечивает выполнение терминальных условий и разностного соотношения, определяемого ОСДУ.

Отметим, что в подходе Deep BSDE параметр N влияет на размер нейронной сети, аппроксимирующей искомую функцию (суррогатной нейросети). В подходе Раисси (сети типа FBSNNs) предложена архитектура сети, в которой параметр N вынесен за пределы суррогатной нейросети и участвует только в функции потерь. Параметр N управляет точностью, с которой дискретизация по времени аппроксимирует ОСДУ, лежащие в основе функции потерь.

Мы используем подход Раисси для архитектуры нейросети и модифицировали функцию потерь для ее применения к решению систем квазилинейных параболических уравнений.

Для того, чтобы построить численное решение задачи, разобьем промежуток $[\tau; T]$ на N одинаковых частей ($\Delta t = t_{n+1} - t_n = const$): $\tau = t_0 < t_1 < \ldots < t_n < \ldots < t_N = T$.

Запишем соответствующий разностный вид ПОСДУ (15)–(17) с помощью схемы Эйлера-Маруямы

$$X_{n+1} = X_n + A\left(X_n, U_n, V_n, G_n\right) \Delta W_n,$$
$$X_0 = x, \qquad (19)$$

$$U_{n+1} = U_n - \varphi\left(X_n, U_n, V_n, G_n\right) \Delta t +$$
$$+ Z_n^{+} A\left(X_n, U_n, V_n, G_n\right) \Delta W_n, \; U_{\mathrm{N}} = h(X_N), \qquad (20)$$

$$V_{n+1} = V_n - \psi\left(X_n, U_n, V_n, G_n\right) \Delta t +$$
$$+ G_n^{+} A\left(X_n, U_n, V_n, G_n\right) \Delta W_n, \; V_{\mathrm{N}} = g(X_N), \qquad (21)$$

где приняты обозначения $X_{n+1} = X(t_{n+1})$, $\Delta W_n = W_{n+1} - W_n$.

Аппроксимируем функции $\left(u(t,x), v(t,x)\right)$ нейронной сетью

$$\left(u^{\theta}(t,x), v^{\theta}(t,x)\right) = net(t, X(t); \theta),$$

где θ – параметры (весовые коэффициенты) нейронной сети.

Введем обозначения для аппроксимаций случайных процессов

$$\widetilde{U}_n = u^{\theta}(t_n, X_n), \quad \widetilde{V}_n = v^{\theta}(t_n, X_n). \qquad (22)$$

Процедура автоматического дифференцирования [21] позволяет вычислить процессы $\widetilde{Z}_n = Du^{\theta}(t_n, X_n)$ и $\widetilde{G}_n = Dv^{\theta}(t_n, X_n)$, что дает возможность вдвое сократить число неизвестных (искомых процессов). Здесь и далее символ D обозначает операцию автоматического дифференцирования.

Для нахождения математического ожидания в разностной аппроксимации функции потерь (18) L используем метод Монте-Карло с пакетами-батчами (batch) по *M* реализаций винеровского процесса $W^m(t)$, который сформирует пакет (батч) из *M* траекторий (реализаций) процесса $X^m(t)$, $m = 1, \ldots, \mathrm{M}$, выходящих из одной точки $X^m(t_0) = x$

$$X_{n+1}^m = X_n^m + A\left(X_n^m, U_n^m, V_n^m, G_n^m\right) \Delta W_n^m,$$
$$X_0^m = x. \qquad (23)$$

Тогда оценка $\hat{L}(\theta; N, M)$ для функции потерь (18) примет вид

$$\hat{L}(\theta; N, M) = \frac{1}{M}\left[\sum_{m=1}^{M}\left|\widetilde{U}_N^m - h(X_N^m)\right|^2 + \sum_{m=1}^{M}\left\|\widetilde{V}_N^m - g(X_N^m)\right\|^2 + \right.$$
$$\left. + \Delta t\sum_{n=0}^{N-1}\sum_{m=1}^{M}\left|\widetilde{U}_{n+1}^m - \overline{U}_{n+1}^m\right|^2 + \Delta t\sum_{n=0}^{N-1}\sum_{m=1}^{M}\left\|\widetilde{V}_{n+1}^m - \overline{V}_{n+1}^m\right\|^2\right], \tag{24}$$

где $\widetilde{U}_{n+1}^m = u^\theta(t_{n+1}, X_{n+1}^m)$ и $\widetilde{V}_{n+1}^m = v^\theta(t_{n+1}, X_{n+1}^m)$ – решение обратного СДУ, аппроксимируемое нейросетью по формуле (22); $\overline{U}_{n+1}^m$ и $\overline{V}_{n+1}^m$ – контрольные значения, вычисляемые по схеме Эйлера-Маруямы

$$\overline{U}_{n+1}^m = \widetilde{U}_n^m - \varphi\left(X_n, \widetilde{U}_n^m, \widetilde{V}_n^m, \widetilde{G}_n^m\right)\Delta t +$$
$$+ \widetilde{Z}_n^{m+} A\left(X_n, \widetilde{U}_n^m, \widetilde{V}_n^m, \widetilde{G}_n^m\right)\Delta W_n^m, \tag{25}$$

$$\overline{V}_{n+1}^m = \widetilde{V}_n^m - \psi\left(X_n, \widetilde{U}_n^m, \widetilde{V}_n^m, \widetilde{G}_n^m\right)\Delta t +$$
$$+ \widetilde{G}_n^{m+} A\left(X_n, \widetilde{U}_n^m, \widetilde{V}_n^m, \widetilde{G}_n^m\right)\Delta W_n^m. \tag{26}$$

1.5. Программный комплекс

Для численного решения задачи для полностью нелинейных параболических уравнений был написан программный комплекс (framework) FBSNN_tf1_FnL, позволяющий решать ПОСДУ, связанные с задачей Коши для системы квазилинейных параболических уравнений. Фреймфорк FBSNN_tf1_FnL написан на языке Python с использованием библиотеки TensorFlow и основан на оригинальном фреймворке Раисси FBSNNs [22], который позволял решать ПОСДУ, связанные с квазилинейными параболическими уравнениями. Базовая архитектура программы сохранилась, но был модифицирован (путем наследования) класс FBSNN – создан класс FBSNN_tf1, позволяющий использовать различные функции активации нейронов в нейронной сети и дающий возможность получать на выходе некоторые параметры обученной нейросети (например, история значений функции потерь, полученных входе обучения нейросети). В качестве архитектуры суррогатной нейросети была сохранена оригинальная архитектура из фреймворка [22] – полносвязная нейронная сеть. Далее был создан класс FBSNN_tf1_FnL, предназначенный для решения ПОСДУ (15–17), задающих вероятностное представление решения системы квазилинейных параболических уравнений (11), (12). Также были созданы специальные классы, описывающие конкретный класс задач Коши (например, для уравнения (27)), что позволяет удобно модифицировать и использовать программные коды для решения нескольких подобных задач, отличающихся друг от друга коэффициентами или терминальными условиями. Эти классы могут содержать в себе, в качестве методов, функции, описывающие точные решения соответствующих задач, при условии, что таковые существуют.

Размер полносвязной нейронной сети задается глубиной (H – количество скрытых слоев) и шириной (S – количество нейронов в скрытом слое). Входной слой принимает пару аргументов (t, x) и содержит $1+d$ нейронов. Выходной слой выдает пару функций $(u(t,x), v(t,x))$ и содержит $1+d$ нейронов.

Тогда количество весовых коэффициентов нейросети выражается формулой

$$\dim(\theta) = (2+d)S + (H-1)(S+1)S + (S+1)(1+d).$$

Далее представлен алгоритм для нахождения значения $u(\tau, x)$.

Алгоритм решения задачи (11), (12) в точке (τ, x)

<u>*Input*</u>: point (τ, x)

Initial parameters: network weights θ^0, learning rate η; max epoch *K*.

Output values: learned neural networks θ^K, loss function $L(\theta^K; N, M)$.

<u>*Output:*</u> $\left(X_n^m, \widetilde{U}_n^m, \widetilde{V}_n^m\right)$, $n = 0, \ldots, N-1$, $m = 1, \ldots, M$

01: Generate Brownian motions ΔW_n^m, $n = 0, \ldots, N-1$, $m = 1, \ldots, M$.
02: *for k = 1 to K do*
03: $L_\Sigma = 0$; $\theta = \theta^{k-1}$
04: $X_0^m = x$, $m = 1, \ldots, M$
05: *for n = 0 to (N–1) do*
06: $\widetilde{U}_n^m = u^\theta(t_n, X_n^m)$; $\widetilde{V}_n^m = v^\theta(t_n, X_n^m)$ из (22), $m = 1, \ldots, M$
07: $\widetilde{Z}_n^m = Du^\theta(t_n, X_n^m)$; $\widetilde{G}_n^m = Dv^\theta(t_n, X_n^m)$, $m = 1, \ldots, M$
08: $\overline{U}_{n+1}^m$ из (25), $\overline{V}_{n+1}^m$ из (26), $m = 1, \ldots, M$
09: X_{n+1}^m из (23), $m = 1, \ldots, M$
10: $\widetilde{U}_{n+1}^m = u^\theta(t_{n+1}, X_{n+1}^m)$; $\widetilde{V}_{n+1}^m = v^\theta(t_{n+1}, X_{n+1}^m)$ из (22), $m = 1, \ldots, M$
11: $L_{n+1} = \sum_{m=1}^{M}\left|\widetilde{U}_{n+1}^m - \overline{U}_{n+1}^m\right|^2 + \sum_{m=1}^{M}\left\|\widetilde{V}_{n+1}^m - \overline{V}_{n+1}^m\right\|^2$
12: $L_\Sigma = L_\Sigma + L_{n+1}$
13: end for n
14: $L_{TC} = \sum_{m=1}^{M}\left|\widetilde{U}_N^m - h(X_N^m)\right|^2 + \sum_{m=1}^{M}\left\|\widetilde{V}_N^m - g(X_N^m)\right\|^2$
15: $L = \frac{1}{M}\left(\Delta t\, L_\Sigma + L_{TC}\right)$
16: $\theta^k = Adam(\theta^{k-1}, \eta, \nabla L)$
17: *end for k*
18: *return* $\left(X_n^m, \widetilde{U}_n^m, \widetilde{V}_n^m\right)$, $n = 0, \ldots, N-1$, $m = 1, \ldots, M$
19: # Remark $u(\tau, x) = U(\tau) = \widetilde{U}_0^m$, $\widetilde{U}_0^m$ is same for any $m = 1, \ldots, M$

2. УРАВНЕНИЕ И ЧИСЛЕННЫЕ РЕЗУЛЬТАТЫ

2.1. Полностью нелинейное уравнение, возникающее на рынке Блэка-Шоулса

Рассмотрим в качестве примера одномерное уравнение ($d = 1, x \in R$), описывающее задачу оптимального управления портфельными инвестициями на рынке Блэка-Шоулса (при безрисковой процентной ставке $r = 0$) [23]

$$\frac{\partial u}{\partial \tau} - \frac{1}{2}\frac{\mu^2}{\sigma^2}\frac{u_x^{\ 2}}{u_{xx}} = 0,\ u(T,x) = h(x), \quad (27)$$

где нижний индекс обозначает переменную, по которой происходит дифференцирование, μ и σ – снос и волатильность в уравнении для логарифма цены базового актива, $h(x)$ – функция полезности.

Задача (27) имеет точное решение для функций $h(x)$ специального вида: логарифмической, экспоненциальной и степенной. Это обстоятельство позволяет осуществить проверку правильности решения, выдаваемого нейронной сетью, на соответствующих модельных примерах.

Для функции $h(x) = \ln x,\ x > 0$ точное решение задачи (27) имеет явный вид

$$u(\tau, x) = \ln x + \frac{1}{2}\frac{\mu^2}{\sigma^2}(T - \tau). \quad (28)$$

Для функции $h(x) = -e^{-\gamma x},\ \gamma > 0,\ x \in R$ точное решение задачи (27) имеет явный вид

$$u(\tau, x) = -e^{-\gamma x}\exp\left(-\frac{1}{2}\frac{\mu^2}{\sigma^2}(T - \tau)\right). \quad (29)$$

Для функции $h(x) = \frac{1}{\gamma}x^\gamma,\ \gamma \in (0;1),\ x > 0$

точное решение задачи (27) имеет явный вид

$$u(\tau, x) = \frac{1}{\gamma}x^\gamma \exp\left(\frac{1}{2}\frac{\gamma}{1-\gamma}\frac{\mu^2}{\sigma^2}(T - \tau)\right). \quad (30)$$

2.2. Квазилинейная система и ПОСДУ, ассоциированные с УЧП (27)

Применим методы, описанные в разделе 1 (1.1 и 1.2), для уравнения (27).

Задача Коши для параболической квазилинейной системы УЧП, полученной из (27) с помощью дифференциального продолжения, имеет вид

$$\frac{\partial u}{\partial \tau} + \frac{1}{2}A^2(v, v_x)u_{xx} + \varphi(v, v_x) = 0,$$
$$u(T, x) = h(x), \quad (31)$$

$$\frac{\partial v}{\partial \tau} + \frac{1}{2}A^2(v, v_x)v_{xx} + \psi(v, v_x) = 0,$$
$$v(T, x) = g(x), \quad (32)$$

где $0 < A^2(v, v_x) = \frac{\mu^2}{\sigma^2}\left(\frac{v}{v_x}\right)^2 \in R$, $\varphi(v, v_x) = -\frac{\mu^2}{\sigma^2}\frac{v^2}{v_x}$,

$\psi(v, v_x) = -\frac{\mu^2}{\sigma^2}v \in R$, $g(x) = h_x(x) \in R$.

Решению задачи Коши (31), (32) $(u(\tau, x), v(\tau, x))$ соответствует $(U(\tau), V(\tau))$, где $(X(t), (U(t), V(t)), (Z(t), G(t)))$ является решением ПОСДУ

$$dX(t) = A(V(t), G(t))dW(t),\quad X(\tau) = x, \quad (33)$$

$$dU(t) = -\varphi(V(t), G(t))dt +$$
$$+ Z(t)A(V(t), G(t))dW(t),$$
$$U(T) = h(X(T)), \quad (34)$$

$$dV(t) = -\psi(V(t), G(t))dt +$$
$$+ G(t)A(V(t), G(t))dW(t),$$
$$V(T) = g(X(T)). \quad (35)$$

2.3. Численные эксперименты

В качестве алгоритма оптимизации, используемого для обучения нейросети, использовался наиболее распространенный сейчас алгоритм Adam (ADAptive Momentum) [24], который является модификацией стохастического градиентного спуска, использующей метод моментов Нестерова. Для начальной инициализации весов нейронной сети использовалась инициализация Ксавье Глоро [25].

Для оценки качества аппроксимации решения, полученного с помощью нейросети, используем относительные погрешности численного решения $(u(\tau, x), v(\tau, x)) = (\widetilde{U}(\tau), \widetilde{V}(\tau))$ системы (31), (32), вычисляемые по формулам

$$err_{rel}^U(\tau, x) = \frac{\left|\widetilde{U}(\tau) - u_{exact}(\tau, x)\right|}{\left|u_{exact}(\tau, x)\right|}, \quad (36)$$

$$err_{rel}^V(\tau, x) = \frac{\left\|\widetilde{V}(\tau) - v_{exact}(\tau, x)\right\|}{\left\|v_{exact}(\tau, x)\right\|}. \quad (37)$$

Численные эксперименты проводились для различных функций активации (sin, tanh, ReLU) и глубоких нейронных сетей различных размеров, отличающихся глубиной H (количеством скрытых слоев) и шириной S (количеством нейронов в скрытом слое).

На рисунке 1 приведены численное и точное решения для экспоненциального терминального условия (29). На рисунке 1 слева приведены 5 траекторий для процесса $\widetilde{U}(t)$ и их сравнение со значениями $u_{exact}(\tau, X^m(t))$. На рисунке 1 справа приведены 5 траектории для процесса $\widetilde{V}(t)$ и их сравнение со значениями

$v_{exact}(\tau, X^m(t))$. Решения $\left(\widetilde{U}(\tau), \widetilde{V}(\tau)\right)$ приведены в точке $(\tau, x) = (0.6, 0.5)$. Относительная погрешность решений составляет, соответственно, 0,34% и 0,42%.

Использовалась нейросеть размером 4x128, состоящая из 4 скрытных слоев ($L = 4$) и 128 нейронов в каждом скрытом слое (одинаковый размер каждого слоя $S = 128$). В качестве функции активации использовался гиперболический тангенс (tanh). Для реализации метода Монте-Карло использовались пакеты (батчи) по $M = 100$ траекторий процесса $X(t)$. Для дискретизации по времени промежуток $[\tau; T] = [0{,}6; 1]$ разбивался на $N = 10$ частей. Для обучения нейронной сети понадобилось $K = 1000$ итераций при постоянной скорости обучения $\eta = 0{,}01$. Достигнутое значение функции потерь (24) составляет $\hat{L}(\theta; N, M) = 0{,}0001$.

ЗАКЛЮЧЕНИЕ

В данной работе был применен подход, основанный на методе дифференциального продолжения, позволяющий свести задачу Коши для полностью нелинейного параболического уравнения (1) относительно функции $u(\tau, x)$ к задаче Коши для системы квазилинейных параболических уравнений (5), (10). *Предположения С 1 и С 2* обеспечивают возможность применения этого подхода. Для квазилинейной системы (5), (10) построено вероятностное представление решение, основанное на решении системы прямого-обратного стохастических дифференциальных уравнений (ПОСДУ) (15)–(17). Выполнение *Условия С 3* обеспечивает существование и единственность решения ПОСДУ (15)–(17) $\left(X(t), \left(U(t), V(t)\right), \left(Z(t), G(t)\right)\right)$, при этом $u(\tau, x) = U(\tau)$.

Решение ПОСДУ было сведено к решению оптимизационной задачи (18), которая численно решается с помощью глубокой нейронной сети типа FBSNNs [16]. Для этого был разработан фреймворк FBSNN_tf1_FnL, написанный на языке Python с использованием библиотеки TensorFlow и основанный на оригинальном фреймворке FBSNNs [22]. Использовалась нейросеть прямого распространения с полносвязной архитектурой. Предложена модифицированная функция потерь (18), предназначенная для решения ПОСДУ (15)–(17).

Для иллюстрации применимости описанных методов рассмотрено полностью нелинейное уравнение, описывающее цену оптимального портфеля на рынке Блэка-Шоулса, выписаны соответствующая ему квазилинейная система и ПОСДУ.

Численное решение было апробировано для функций полезности $h(x)$ специальных видов: логарифмического, экспоненциального и степенного, для которых существует точное решение. В продемонстрированном примере относительная погрешность численного решения составляет 0,34% после $K = 1000$ итераций обучения при постоянной скорости обучения $\eta = 0{,}01$.

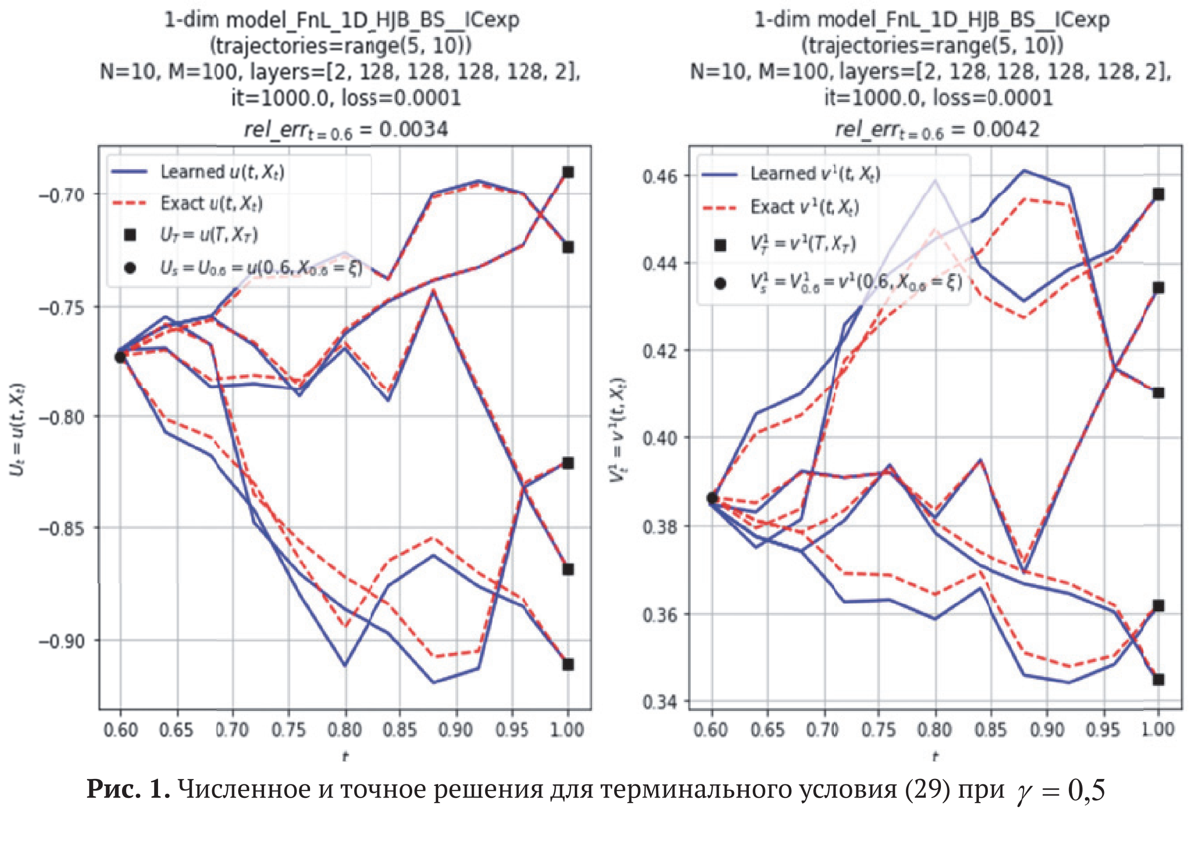

Рис. 1. Численное и точное решения для терминального условия (29) при $\gamma = 0{,}5$

БЛАГОДАРНОСТИ

Выражаю особую благодарность д. ф.-м. н., проф. Белопольской Яне Исаевне за значимые замечания и важнейшие советы при проведении исследования и подготовке данной статьи.

СПИСОК ЛИТЕРАТУРЫ

1. *Pardoux E., Peng S.* Adapted solution of a backward stochastic differential equation // Systems Control Letters. 1990. V. 14. No. 1. Pp. 55–61.
2. *Pardoux E., Peng S.* Backward stochastic differential equations and quasilinear parabolic partial differential equations // Lecture Notes in CIS. 1992. V. 176. Pp. 200–217.
3. Second order backward stochastic differential equations and fully non-linear parabolic PDEs / P. Cheridito, H. M. Soner, N. Touzi, N. Victoir // Comm. Pure Appl. Math. 2007. V. 60. No. 7. Pp. 1081–1110.
4. *Belopolskaya Ya.I. and Woyczynski W.A.* SDEs, FBSDEs and fully nonlinear parabolic systems // Rendiconti del Seminario Matematico Univ. Politec. Torino. 2013. V. 71. No. 2. Pp. 209–217.
5. *Belopolskaya Ya.I.* Probabilistic counterparts of nonlinear parabolic PDE systems // Springer Optimization and Its Applications. 2014. Ch. in "Modern Stochastics and Applications". Pp. 71–94.
6. *Belopolskaya Ya.* Probabilistic interpretations of quasilinear parabolic equations // AMS. Contemporary Mathematics. 2019. V. 734. Pp. 39–56,
7. *Abraham R., Rivi´ere O.* Forward-backward stochastic differential equations and PDE with gradient dependent second order coefficients // ESAIM: Probability and Statistics. 2006. V. 10. Pp. 184-205.
8. *Ma J., Yong J.* Forward-Backward stochastic differential equations and their applications. Springer. 2007. 270 p.
9. Three algorithms for solving high-dimensional fully-coupled FBSDEs through deep learning / S. Ji, S. Peng, Y. Peng, X. Zhang // IEEE Intelligent Systems. 2020. V. 35. No. 3. Pp. 71–84.
10. *Hornik K., Stinchcombe M., White H.* Multilayer feedforward networks are universal approximators // Neural Networks. 1989. V. 2. No. 5. Pp. 359–366.
11. *Hornik K., Stinchcombe M., White H.* Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks // Neural Networks. 1990. V. 3. No. 5. Pp. 551–560.
12. *E W., Han J., Jentzen A.* Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations // Communications in Mathematics and Statistics. 2017. V.5. No. 4. Pp. 349–380.
13. *Beck C., E W., Jentzen A.* Machine learning approximation algorithms for high-dimensional fully nonlinear partial differential equations and second-order backward stochastic differential equations // Journal of Nonlinear Science. 2019. V. 29. No. 4. Pp. 1563–1619.
14. *Pham H., Warin X., Germain M.* Neural networks-based backward scheme for fully nonlinear PDEs // SN Partial Differ. Eq. Appl. 2021. V. 2. No. 16. Pp. 1–27.
15. An overview on deep learning-based approximation methods for partial differential equations / C. Beck, M. Hutzenthaler, A. Jentzen, B. Kuckuck // Discrete and Continuous Dynamical Systems – B. 2023. V. 28. No. 6. Pp. 3697-3746.
16. *Raissi M.* Forward-backward stochastic neural networks: Deep learning of high-dimensional partial differential equations // arXiv preprint arXiv:1804.07010 (2018). URL: https://arxiv.org/abs/1804.07010 (Available 1.06.2024).
17. *Goodfellow I., Bengio Y., Courville A.* Deep learning. 2016. URL: http://www.deeplearningbook.org (Available 1.06.2024).
18. *M. Hutzenthaler, A. Jentzen, T. Kruse, T. Anh Nguyen, P. von Wurstemberger* //Overcoming the curse of dimensionality in the numerical approximation of semilinear parabolic partial differential equations: Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, Dec 2020. V. 476. No. 2244.
19. *Hutzenthaler M., Jentzen A., Kruse T.* Overcoming the curse of dimensionality in the numerical approximation of parabolic partial differential equations with gradient-dependent nonlinearities // Foundations of Computational Mathematics. 2021. V. 22. Pp. 1–62.
20. *Belopolskaya Ya., Dalecky Yu.* Stochastic equations and differential geometry. Dordrecht: Kluwer Academic Publishers. 1990. 260 p.
21. Automatic differentiation in machine learning: a survey / A.G. Baydin, B.A. Pearlmutter, A.A. Radul, J.M. Siskind // arXiv preprint arXiv:1502.05767 (2015). URL: https://arxiv.org/abs/1502.05767 (Available 1.06.2024).
22. *Raissi M.* FBSNN framework. 2018. URL: https://github.com/maziarraissi/FBSNNs (Available 1.06.2024).
23. *Zariphopoulou Th.* Optimal Asset Allocation in a Stochastic Factor Model – An Overview and Open Problems //Adv. Financ. Model. Radon Ser. Comput. Appl. Math. 2009. V. 8. Pp. 1–29.
24. *Kingma D.P., Ba J.* Adam: A Method for Stochastic Optimization // arXiv preprint arXiv:1412.6980 (2017). URL: https://arxiv.org/pdf/1412.6980 (Available 1.06.2024).
25. *Glorot X., Bengio Y.* Understanding the difficulty of training deep feedforward neural networks // Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. 2010. V. 9. Pp. 249–256.

A NUMERICAL ALGORITHM FOR SOLVING FULLY NONLINEAR PARABOLIC EQUATIONS BASED ON FORWARD-BACKWARD STOCHACTIC DIFFERENCIAL EQUATIONS AND NEURAL NETWORKS

Sirius University of Science and Technology, federal territory “Sirius”, Sochi, Russia

The article presents a numerical method for solving the Cauchy problem for a fully nonlinear parabolic equation. Considered equation is reduced to a system of quasilinear parabolic equations. A probabilistic representation of this system solution was constructed based on the solution of the system of forward-backward stochastic differential equations (FBSDE). The FBSDE solution is reduced to solving an optimization problem solved numerically using a neural network. An example of applying this method to an equation describing the price of an optimal portfolio in the Black-Scholes market is considered. The numerical solution was tested while choosing utility functions for which exact solutions exist.
Keywords: Fully nonlinear parabolic equations, Cauchy problem, stochastic differential equations (SDE), optimization problem, deep learning, forward-backward stochastic neural networks (FBSNNs).

DOI: 10.37313/1990-5378-2024-26-4-161-169
EDN: FQJDSR

Andrey Chubatov, Junior Researcher, Scientific Center for Information Technology and Artificial Intelligence.
E-mail: chaa@inbox.ru