ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В. ЛОМОНОСОВА»
МЕХАНИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ
КАФЕДРА ОБЩИХ ПРОБЛЕМ УПРАВЛЕНИЯ

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(ДИПЛОМНАЯ РАБОТА)
специалиста

ПРИМЕНЕНИЕ НЕЙРОННЫХ СЕТЕЙ ДЛЯ РЕШЕНИЯ СИСТЕМ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

Выполнил студент
632 группы
Дехович Александр Эдуардович

_____
подпись студента

Научный руководитель:
д.ф.-м.н., профессор
Локуциевский Лев Вячеславович

_____
подпись научного руководителя

Москва
2019

Содержание

1 Введение

В работе рассматривается метод численного решения обыкновенных дифференциальных уравнений и систем обыкновенных дифференциальных уровнений с использованием искусственных нейронных сетей с последующей корректировкой полученных результатов на основе алгоритма Калмана.

Приведем краткий обзор работ по теории аппроксимации функций нейронными сетями в целом, а также о приближении решений дифференциальных уравнений в частности.

В 1900 году Гильберт предложил список из 23 нерешённых задач, которые, по его мнению, должны были стать вызовом для математиков XX века. Тринадцатая проблема заключалась в следующем: возможно ли произвольную непрерывную функцию n аргументов представить в виде суперпозиции функций меньшего числа аргументов. Ответ был дан А. Н. Колмогоровым в [1].

Теорема 1 (Колмогоров, 1957) *Любая непрерывная функция n аргументов на единичном кубе* $[0,1]^n$ *представима в виде суперпозиции непрерывных функций одного аргумента и операции сложения:*

$$f(x^1,...,x^n) = \sum_{k=1}^{2n+1} h_k \sum_{i=1}^{n} \varphi_{ik}(x^i),$$

где h_k, φ_{ik} *— непрерывные функции, причём* φ_{ik} *не зависят от выбора* f*.*

Теперь дадим определение нейронной сети прямого распространения с n скрытыми слоями.

Определение 1 *Определим искусственную нейронную сеть с n скрытыми слоями как нелинейную функцию* $N : \mathbb{R}^{m_0} \to \mathbb{R}^{m_{n+1}}$*, заданную следующим образом. Имеется* $I \subseteq \mathbb{R}^{m_0}$ *- входное множество(слой),* $O \subseteq \mathbb{R}^{m_{n+1}}$*-выходное множество(слой), и n множеств (скрытых слоёв) размерности* $m_1, m_2, ..., m_n$*. Нейронная сеть имеет параметры* $A^{(l)} \in \mathbb{R}^{m_{l-1} \times m_l}$ *,* $b^{(l)} \in \mathbb{R}^{m_l}$ *для* $1 \le l \le n+1$ *и функции активации* $\sigma^{(m_k)} : \mathbb{R}^{m_k} \to \mathbb{R}^{m_k}, \quad 1 \le k \le n$*, применяемые покомпонентно, т.е.* $\sigma^{(m_k)}(x) = (\sigma_1^{(m_k)}(x_1), ..., \sigma_{m_k}^{(m_k)}(x_{m_k})), \quad \sigma_i^{(m_k)} : \mathbb{R} \to \mathbb{R}$*. Для входного вектора* $x \in \mathbb{R}^{m_0}$ *нейронная сеть вычисляет последовательно*

$$x^{(0)} := x$$

$$x^{(l)} := \sigma^{(m_l)}(A^{(l-1)}x^{(l-1)} + b^{(l)}) \qquad 1 \leq l \leq n$$
$$x^{(n+1)} := A^{(n+1)}x^{(n)} + b^{(n+1)}$$

Вектор $x^{(n+1)}$ является результатом работы нейронной сети, т.е. $x^{(n+1)} = N(x)$ по определению.

Нетрудно видеть, что выражение, записанное в теореме Колмогорова имеет структуру схожую со структурой нейронной сети с одним скрытым слоем из $2n+1$ нейронов. Таким образом, двух слоёв уже достаточно, чтобы вычислять произвольные непрерывные функции, и не приближённо, а точно. К сожалению, представление Колмогорова не является полностью моделью нейронной сети: функции φ_{ik} нелинейны, а функции h_k зависят от f, и в общем случае не являются дифференцируемыми.

В 1989 году Джорджем Цыбенко была доказана теорема, получившая название Универсальной теоремы аппроксимации [2], которая утверждает, что искусственная нейронная сеть прямой связи с одним скрытым слоем может аппроксимировать любую непрерывную функцию многих переменных с любой точностью. Однако, в теореме нет условия на параметры сети и количество нейронов в скрытом слое.

В 1993,1994 годах Эндрю Бэррон в серии статей [3], [4] ввел некоторый широкий класс функций от одной переменной, для которого также была доказана теорема аппроксимации нейронной сетью с одним скрытым слоем с любой точностью, а также дана оценка количества нейронов в скрытом слое. А в 2017 году появилось обобщение этой теоремы на функции многих переменных [5].

Одни из первых работ по решению разных типов обыкновенных дифферференциальных уравнений и уравнений в частных производных нейронными сетями прямого распространения появились в начале 1990-х годов, например в работах Lee, H. и Kang, I. Но универсальный подход к решений этой задачи появился только в 1997 в работе I.E.Lagaris, A.Likas и D.I.Fotiadis [6]. Позднее появились приложения этого метода к задачам оптимального управления, при решении гамильтоновой системы, возникающей из принципа максимума Понтрягина[7]. В последние годы стали изучать применение не только нейронных сетей прямого распространения, но и реккурентные нейронные сети, имеющих более сложную архитектуру.

Таким образом, на данный момент являются актуальными исследования как в теоретическом значении нейронных сетей в теории аппроксимации, с обоснованием существования нейронных сетей, приближающих функции от многих переменных с любой точностью, и оценкой количества слоев и нейронов в каждом слое, так и в прикладных задачах, при

решении которых используется нейросетевой подход, имеющий сопоставимый успех с классическими численными методами.

Во второй части работы выдвигается предположение о возможности применения фильтра Калмана для получения более точного численного решения после преобразования исходного сигнала. Подобная идея была предложена в [8], где алгоритм Калмана применялся для улучшения точности некоторого классификационного алгоритма картинок. Фильтр Калмана — эффективный рекурсивный алгоритм, оценивающий вектор состояния динамической системы, используя зашумленные измерения состояния системы, повсеместно применяющийся для решения как физических, так и экономических задач, например, в трейдинге. Поэтому идея применения этого алгоритма для корректировки численного решения уравнения кажется вполне естественной.

Далее, изложенные алгоритмы применяются в решению хаотической системы обыкновенных дифференциальных уравнений возникающих, возникающий в одной важной задаче субримановой геометрии (которая сформулирована в параграфе 5 в виде задачи оптимального управления). Важной особенностью является то, что начальные условия заданы частично в левом конце, частично в правом, что делает решение решения задачи методом стрельбы очень затруднительным ввиду хаотичности системы, доказанной в работе [9].

2 Описание метода

Далее будут рассматриваться нейронные сети с одним скрытым слоем ($n = 1$), зависящие только от скалярной величины $t \in \mathbb{R}$, выдающие скалярные значения, т.е. $m_0 = m_{n+1} = 1$. Таким образом, можно записать явный вид такой нейронной сети:

$$N(t; p) = \sum_{j=1}^{h} w_j \sigma(v_j t + b_j) + d \tag{1}$$

где h - число нейронов в скрытом слое, а $\vec{p} = (\vec{b}, \vec{v}, \vec{w}, d)$ - параметры сети. В качестве активационной функции будет использоваться логистическая функция, т.е. $\sigma(z) = \frac{1}{1+e^{-z}}$

Опишем подход, представленный в статье "Artificial Neural Networks for Solving Ordinary and Partial Differential Equations"написанной I.E.Lagaris, A.Likas и D.I.Fotiadis в 1997 году.

2.1 Математическая модель

В общем случае рассматривается дифференциальное уравнение n-го порядка, но для иллюстрации метода ограничимся уравнением первого порядка:

$$G(t, x(\vec{t}), \nabla x(\vec{t})) = 0,\ t \in [t_0, T]$$

с начальным условием $x(t_0) = x_0$

Разобьём отрезок $[t_0, T]$ на $n+1$ точку $t_0, ..., t_n$ так, чтобы

$$t_0 < t_1 < t_2 < ... < t_n = T$$

И преобразуем задачу к нахождению численного решения уравнения:

$$G(t_i, x(\vec{t_i}), \nabla x(\vec{t_i})) = 0,\ i \in \{0, ..., n\}$$

.

Предлагается искать решение $\hat{x} = \hat{x}(t; p)$, где p - регулируемые параметры, находящиеся из условия:

$$\sum_{i=0}^{n} (G(t_i, x(t_i, \vec{p}), \nabla x(t_i, \vec{p}))^2 \longrightarrow min_{\vec{p}}$$

В общем случае решение $\hat{x}(t; \vec{p})$ ищется в виде $\hat{x}(t; \vec{p}) = A(x) + F(t, \hat{x}(t; \vec{p}))$ таким образом, чтобы слагаемое $A(x)$ удовлетворяло начальным условиям.

Таким образом, если имеется обыкновенное дифференциальное уравнение первого порядка

$$\dot{x} = f(t, x(t)),\ t \in [t_0, T]$$

$$x(t_0) = x_0$$

то его численное решение будет искаться в виде

$$\hat{x}(t, \vec{p}) = x_0 + (t - t_0) N(t; \vec{p}) \tag{2}$$

где $N(t; \vec{p})$ - нейронная сеть, имеющая вид (1). Тогда вектор параметров $\vec{p}$ ищется из условия:

$$Q(\vec{p}) = \sum_{i=0}^{n} (\dot{\hat{x}} - f(t_i, \hat{x}(t_i)))^2 \longrightarrow min_{\vec{p}},\ t_i \in [t_0, T] \tag{3}$$

Отметим, что в силу (2), производная $\dot{\hat{x}}$ находится явно и имеет вид

$$\dot{\hat{x}}(t,\vec{p}) = N(t;\vec{p}) + (t-t_0)\frac{dN(t;\vec{p})}{dt}$$

В случае системы обыкновенных дифференциальных уравнений

$$\dot{x}_k = f(t, x_1(t), ..., x_m(t)) \text{ , } k = 1, m$$

$$x_k(t_0) = x_{0k}$$

решение ищется в виде $\hat{x}_k(t,\vec{p_k}) = x_{0k} + (t-t_0)N_k(t;\vec{p_k})$. А минимизируемая функция принимает вид

$$Q(\vec{p}) = \sum_{k=1}^{m}\sum_{i=0}^{n}(\dot{\hat{x}_k} - f(t_i, \hat{x_k}(t_i)))^2 \longrightarrow min_{\vec{p}},\ t_i \in [t_0, T] \qquad (3')$$

2.2 Обучение нейронной сети

Процесс обучения заключается в минимизации функции потерь $Q(\vec{p})$ (формула 3 или 3′) на фиксированных точках $\{t_i\}_{i=0}^n$ - разбиении отрезка $[t_0, T]$. Было опробовано два метода минимизации - градиентный спуск(и стохастический градиентный спуск) и BFGS алгоритм. В ходе реализации было замечено, что для одного уравнения применимы оба метода, однако в случае системы уравнений градиентный спуск не сходился вовсе, в то время как BFGS показывал хороший результат.
Далее приведем основную идею BFGS алгоритма и его реализацию.

К недостаткам BFGS алгоритма стоит отнести его относительную сложность реализации по сравнению с градиентным спуском. Поиск параметра a_k из условий Вольфа может сильно замедлять процесс, учитывая, что для достаточно сложных минимизируемых функций от многих переменных нет никаких условий сходимости.

В качестве выходного условия может выступать либо достижение значения ошибки меньше наперед заданного значения ε (в случае, если мы знаем, что минимальное значение равно 0), либо окончание определенного числа итераций.

В данной задаче аргумент t_i выступает в роли признака для нашей модели, поэтому может возникнуть проблема, что обучающая выборка $\{t_i\}_{i=0}^n$ имеет сильно отличающиеся по модулю величины, что является существенной сложностью при обучении нейронных сетей. В этом случае, если рассматриаемый отрезок $[t_0, T]$ так велик, что не обеспечивается необходимая точность ε, можно разбить отрезок на k частей

$$[t_0, T] = \bigcup_{i=1}^{k}[T_{i-1}, T_i] \text{ , где } T_0 = t_0, T_k = T$$

Algorithm 1 BFGS algorithm

procedure BFGS(f, x_0, ε) ▷ f - минимизируемая функция, x_0 - начальное приближение, выбранное случайно, ε - необходимая точность

    Вычисление обратного гессиана H_0 в точке x_0, на практике берут $H_0 = I$ - единичная матрица

    while условие не выполнено **do**

        Вычисление направления поиска: $p_k = -H_k \nabla f(x_k)$

        Следующее приближение: $x_{k+1} = x_k + \alpha_k p_k$, где α_k ищется из условий Вольфа: $f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f_k^T p_k$, $\nabla f(x_k + \alpha_k p_k)^T p_k \geq c_2 \nabla f_k^T p_k$

        Определяем вектора: $s_k = x_{k+1} - x_k$ и $y_k = \nabla f_{k+1} - \nabla f_k$

        Обновление оценки гессиана: $H_{k+1} = (I - \rho_k s_k y_k^T) H_k (I - \rho_k y_k s_k^T) + \rho_k s_k s_k^T$

и последовательно решать k задач

$$\dot{x} = f(t, x(t)), t \in [T_{i-1}, T_i],$$

$$x(T_{i-1}) = \hat{x}(T_{i-1}; \vec{p}), 1 < i \leq k \text{ и } x(T_0) = x_0$$

3 Примеры

Приведем два примера, иллюстрирующие метод. Первый пример - это одно обыкновенное дифференциальное уравнение первого порядка, обучение нейронной сети будет проведено двумя способами - градиентным спуском и BFGS алгоритмом. Во втором примере будет рассмотрена задача оптимального управления, при решении которой вознкнет система из нескольких уравнений, также будет показана неприменимость градиентного спуска в этом случае.

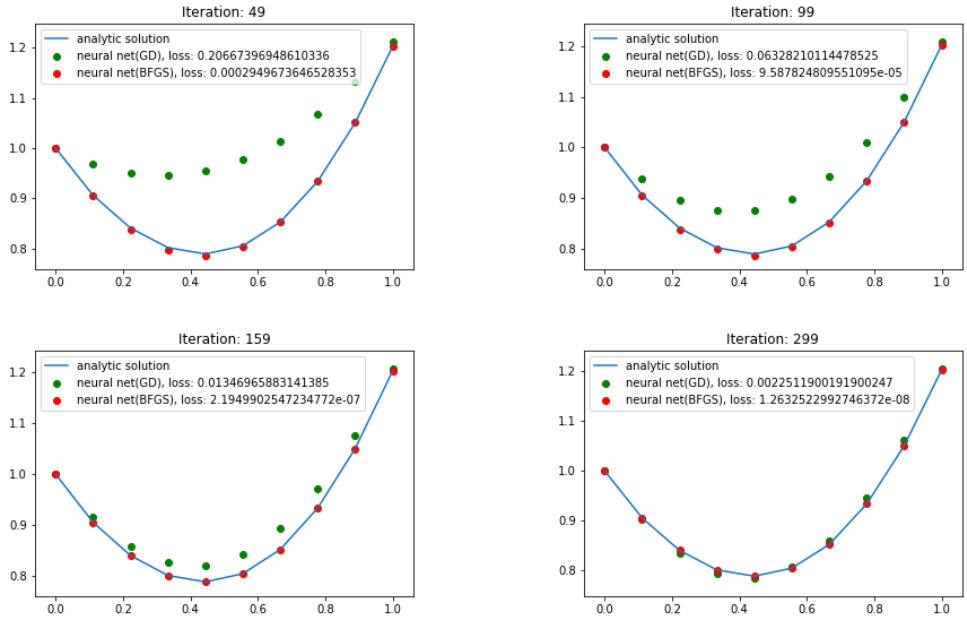
3.1 Случай одного уравнения

Рассмотрим следующую задачу:

$$\frac{dx}{dt} + (t + \frac{1 + 3t^2}{1 + t + t^3})x = t^3 + 2t + t^2 \frac{1 + 3t^2}{1 + t + t^3}$$

с начальным условием $x(0) = 1$ и $t \in [0, 1]$.

Аналитическое решение этой задачи имеет вид $x(t) = t^2 + \frac{e^{-\frac{t^2}{2}}}{1+t+t^3}$. Согласно методу, описанному ранее, будем искать решение в виде $\hat{x}(t;\vec{p}) = 1+tN(t;\vec{p})$. Нейронную сеть будем обучать по 10 равноудаленным точкам из отрезка $[0, 1]$.



Из графиков видно, что и градиентный спуск, и BFGS в целом справляются с задачей минимизации и находят подходящие параметры, хотя градиентному спуску требуется значительно больше итераций для достижения хорошей точности. К тому же, к недостаткам градиентного спуска стоит отнести подбор скорости минимизации, что также усложняет задачу.

3.2 Случай системы уравнений

Рассмотрим задачу оптимального управления из [9]

$$\int_0^1 [x^2(t) + u^2(t)]dt \longrightarrow min_{u(t)\in\mathbb{R}}$$

$$\dot{x} = u(t)$$

$$x(0) = 1$$

Сформулируем принцип максимума Понтрягина для такой задачи [10]:

Теорема 2 *Пусть для задачи*

$$\dot{x}_i = f_i(\vec{x}, \vec{u}), x \in X \subset \mathbb{R}^n, u \in U \subset \mathbb{R}^m, i = 1...n$$

$$\int_0^T f_0(x(t), u(t))dt \longrightarrow min_{u(t)\in U}$$

$\hat{u}(t), \hat{x}(t)$ *- оптимальный процесс,* $0 \leq t \leq T$*(*T *- фиксировано). Тогда существует ненулевая непрерывная вектор-функция* $\psi(t)$*, удовлетворяющая системе уравнений*

$$\dot{\psi}_0 = 0$$
$$\psi_i = -\frac{\partial H}{\partial x_i}, i = 1..n$$

где $H(\psi, x, u) = \sum_{i=0}^{n} \psi_i f_i(x, u)$
А также следующим условиям:

1. Условие максимума: $maxH(\varphi(t), \hat{x}(t), u(t)) = H(\varphi(t), \hat{x}(t), \hat{u}(t)), \psi_0 \geq 0$

2. Условие трансверсальности: $\sum_{i=1}^{n} \psi_i(t)\xi_i = 0$, $\forall \xi \in N$

где N *- терминальное множество (множество точек окончания траектории)*

Таким образом, в нашем примере гамильтониан H имеет вид $H(t, p(t), x(t), u(t)) = x^2(t) + u^2(t) + p(t)u(t)$ (без ограничения общности считаем, что $\psi_0 = 1$)
Тогда условия теоремы записываются в виде:

$$\begin{cases} \dot{p}(t) = -2x(t) \\ \dot{x}(t) = u(t) \\ 2u(t) + p(t) = 0 \end{cases}$$

$x(0) = 1$ по условию, $p(1) = 0$ из условия трансверсальности, т.к. $x(1)$ не закреплен. Решая это систему, получаем точное решение:

$$\begin{cases} x(t) = \frac{e^2}{1+e^2}e^{-t} + \frac{1}{1+e^2}e^t \\ p(t) = \frac{2e^2}{1+e^2}e^{-t} - \frac{2}{1+e^2}e^t \\ u(t) = \frac{1}{1+e^2}e^t - \frac{e^2}{1+e^2}e^{-t} \end{cases}$$

Теперь получим численные решения, которые будем искать в виде:

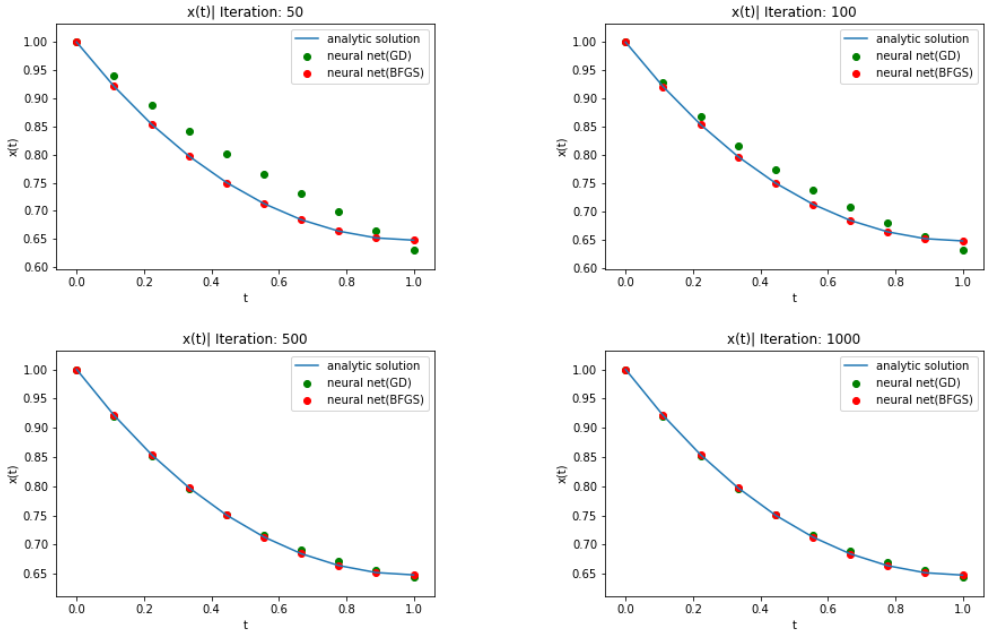
$$\hat{x}(t; p) = 1 + tN_1(t; p_1)$$

$$\hat{p}(t;p) = (t-1)N_2(t;p_2)$$
$$\hat{u}(t;p) = N_3(t;p_3)$$

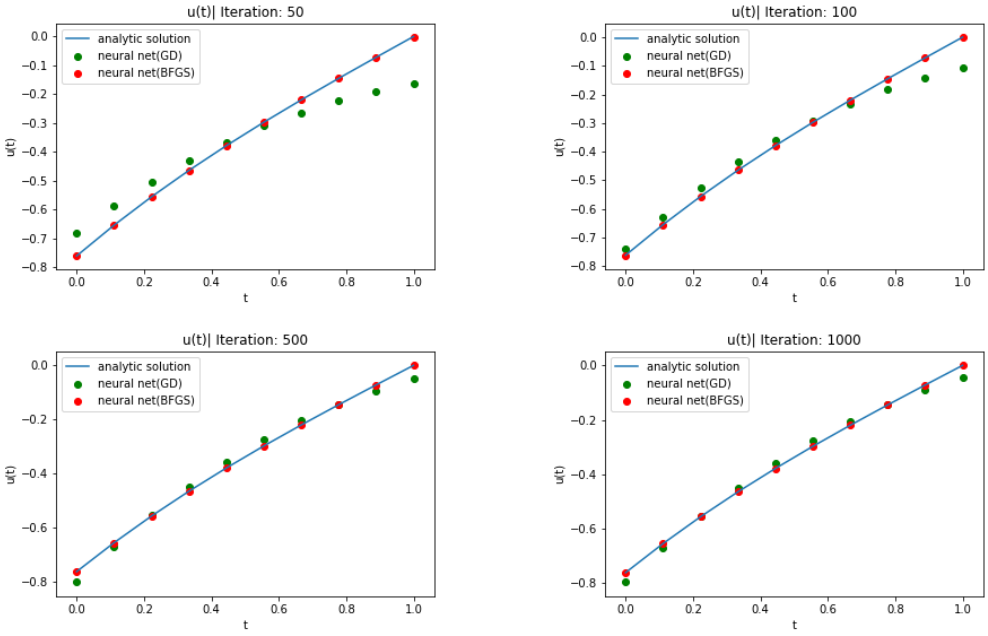
В таблице приведены значения среднего квадрата отклонения численного решения от аналитического в равноудаленных узлах $t_i \in [0,1]$

	$x(t)$	$u(t)$
BFGS	$6.96 \cdot 10^{-11}$	$3.47 \cdot 10^{-10}$
Градиентный спуск	$1.29 \cdot 10^{-5}$	0.00047

Оптимальная траектория, полученная аналитически и двумя методами обучения сети:



Оптимальное управление имеет вид:



Как видно из полученных результатов, BFGS алгоритм справился с задачей за 50 итераций, в то время как градиентный спуск не показывает сопоставимый результат и за 1000. К тому же, из грфиков видно, что ошибка в некоторых отдельных точках очень значительна, несмотря на средний показатель. Отсюда мы делаем вывод, что градиентный спуск неприменим для решения систем уравнений, как указывалось в [6][7].

3.3 Вывод

Приведенные выше примеры показывают, что нейросетевой метод может быть очень эффективен в численном решении дифференциальных уравнений. Также стоит отметить, что важным является выбор метода минимизации функции ошибки: если в случае одного уравнения применимы оба рассматриваемых метода, хотя и градиентному спуску необходимо большее количество итераций по сравнению с BFGS алгоритмом, то в случае системы уравнений градиентный спуск не сходится к минимуму за адекватное число шагов. Заметим, что во втором примере решалась задача оптимального управления, что позволяет предположить, что данный метод может быть применен для большого класса экономических задач, решаемых теорией управления, аналитическое решение которых либо трудно, либо невозможно найти в силу сложности возникающей гамильтоновой системы.

4 Использование Фильтра Калмана

Попробуем проверить гипотезу о корректировки результата работы нейронной сети, принимая результат работы алгоритма за некий измеритель состояний динамической системы, а дифференциальное уравнение за саму динамическую модель. Приведем предпосылки использования Алгоритма Калмана, его идею и численный пример, подтверждающий гипотезу о его возможном применении для данной задачи.

4.1 Мотивация применения

Обычно в качестве измермерителя выступает некоторый датчик, способный с некоторой погрешностью измерить физические параметры. Например, одной из возможностей фильтра является получение оптимальных, непрерывно обновляемых оценок положения и скорости некоторого объекта по результатам временного ряда неточных измерений его местоположения. Так, например, в радиолокации стоит задача сопровождения цели, определения её местоположения, скорости и ускорения, при этом результаты измерений поступают постепенно и сильно зашумлены. Фильтр Калмана использует вероятностную модель динамики цели, задающую тип вероятного движения объекта, что позволяет снизить воздействие шума и получить хорошие оценки положения объекта в настоящий, будущий или прошедший момент времени.

Мы будем исследовать возможность использование фильтра в случае, когда имеется численное решение после применение некоторого алгоритма. В данном случае, будет рассматриваться процедура получения решения, представленная в пункте 3 нейронной сетью. В качестве динамической системы будет выступать исходное дифференциальное уравнение, приведенное к дискретной форме.

4.2 Алгоритм

В многомерном случае уравнение для состояния системы имеет вид:

$$x_{k+1} = Fx_k + Bu_k + Q \tag{4}$$

F – матрица перехода между состояниями, описывающая динамическую модель системы, B - матрица применения управляющего воздействия, Q - матрица ковариаций шума процесса. Уравнение на показания датчика имеет вид:

$$z_k = Hx_k + R$$

H – матрица измерений, отображающая отношение измерений и состояний; R - матрица ковариаций шума измерений.

Поэтому этап предсказания состоит из двух частей:
1. Предсказание состояния системы:

$$\hat{x}_{k+1}^{-} = F\hat{x}_k + Bu_k$$

2. Предсказание ошибки ковариации:

$$P_{k+1}^{-} = FP_kF^T + Q$$

Этап корректировки состоит из вычисления коэффициента Калмана и обновления состояния системы и ошибки ковариации:
1. $K_{k+1} = P_{k+1}^{-}H^T(HP_{k+1}^{-}H^T + R)^{-1}$
2. $\hat{x}_{k+1} = \hat{x}_{k+1}^{-} + K_{k+1}(z_{k+1} - H\hat{x}_{k+1}^{-})$
3. $P_{k+1} = (I - K_{k+1}H)P_{k+1}^{-}$

4.3 Пример

Рассмотрим уравнение:

$$\frac{dx}{dt} + \frac{1}{5}x = e^{-\frac{t}{5}}\cos t$$

$$x(0) = 0 \ , \ t \in [0, 2]$$

Аналитическое решение имеет вид: $x(t) = e^{-\frac{t}{5}}\sin t$

Найдем численное решение в виде $\hat{x}(t;\vec{p}) = t \cdot N(t,\vec{p})$. Нейронную сеть будем обучать градиентным спуском на 10 равноудаленных точках на отрезке $[0, 2]$.

Теперь найдем матрицы F, H. Так как мы принимаем наш алгоритм на датчик, то истинное решение будет равно сумме ответа алгоритма и некоторой ошибки r:

$$z_k = x_k + r, \text{ то есть } H = I \text{ - единичная матрица}$$

Теперь преобразуем исходное уравнение к дискретному виду:

$$\frac{dx}{dt} + \frac{1}{5}x = e^{-\frac{t}{5}}\cos t$$

$$\frac{x(t+\Delta t) - x(t)}{\Delta t} = -\frac{1}{5}x + e^{-\frac{t}{5}}\cos t + q(\Delta t)$$

$$x(t+\Delta t) = (1 - \frac{\Delta t}{5})x(t) + \Delta t \cdot e^{-\frac{t}{5}}\cos t + \Delta t \cdot q(\Delta t)$$

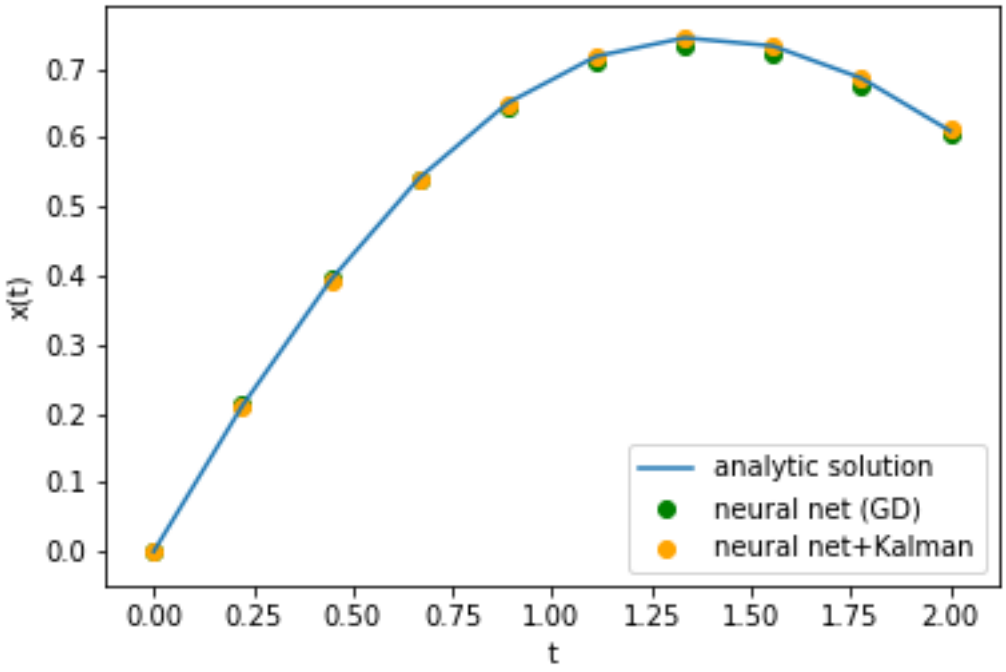
$$x_{k+1} = (1 - \frac{\Delta t}{5})x_k + \Delta t \cdot e^{-\frac{t_k}{5}} \cos t_k + \Delta t \cdot q(\Delta t)$$

Таким образом,

$$F = (1 - \frac{\Delta t}{5}),$$

где $\Delta t = \frac{2}{n}$

Авторы работы [9] предлагают выбирать матрицы ошибок Q и R в виде диагональных матриц с числами q и r на диагонали соответсвенно. Сами числа q и r подбираются вручную так, чтобы увеличивалась точность работы алгоритма. Для оценки точности рассмотрим две метрики: средний квадрат отклонения от аналитического решения(MSE) и максимальный модуль отклонения от аналитического решения(MaxAE). Применяя сначала алгоритм численного решения, а потом алгоритм Калмана, получаем следующий результат:



	MSE	**MaxAE**
Нейронная сеть	$5.5585 \cdot 10^{-5}$	0.0063
Нейронная сеть и фильтрация	$3.3024 \cdot 10^{-6}$	0.0012

Как видно из таблицы, получилось улучшить точность по обеим метрикам. Применение такой последовательности(предсказание с последующей фильтрацией) имеет ряд вычислительных преимуществ: во-первых, позволяет использовать менее трудоёмкие методы минимизации функции потерь, а во-вторых позволяет использовать меньшее количество итераций для работы конкретного алгоритма, оставляя часть работы фильтру Калмана, работающему за линейное время.

5 Задача оптимального управления с хаотической системой принципа максимума Понтрягина

5.1 Постановка задачи

Рассмотрим следующую задачу: построить кривую $L = \{x = x(t), y = y(t) | t \in [0, T]\}$ минимальной длины l так, чтобы

$$x(0) = 0, y(1) = 1, x(T) = 1, y(T) = 0$$

при этом

$$\int_\sigma xydxdy = 0,$$

где σ - фигура, ограниченная кривой L.

Перепишем задачу в виде задачи оптимального управления:

$$\begin{cases} \dot{x}(t) = u \\ \dot{y}(t) = v \\ \dot{z(t)} = \frac{1}{2}x^2yv \\ x(0) = 0, y(0) = 1, z(0) = 0 \\ x(0) = 1, y(0) = 0, z(0) = 0 \end{cases}$$

$$T \longrightarrow \min_{u,v:u^2+v^2\leq 1}$$

5.2 Принцип максимума Понтрягина для задачи

Запишем гимильтониан для данной задачи:

$$H = \lambda + pu + qv + \frac{r}{2}x^2yv = pu + v(q + \frac{r}{2}x^2y)$$

Обозначим

$$\alpha := p, \beta := q + \frac{r}{2}x^2 y, \text{ где}(\alpha, \beta) \neq (0,0) \tag{5}$$

Тогда из условия

$$\alpha u + \beta v \longrightarrow \max_{u,v:u^2+v^2\leq 1}$$

Получаем, что $u^* = \frac{\alpha}{\sqrt{\alpha^2+\beta^2}}$, $v^* = \frac{\beta}{\sqrt{\alpha^2+\beta^2}}$.
Переобозначим u и v:

$$u = \alpha, v = \beta, \text{ с условием } \alpha^2 + \beta^2 = 1$$

. Тогда введя новую функци $\theta = \theta(t)$ получим:

$$\begin{cases} u = \cos\theta \\ v = \sin\theta \\ \theta = \arctan\frac{v}{u} \end{cases}$$

Следовательно,

$$\begin{cases} \dot{x}(t) = \cos\theta \\ \dot{y}(t) = \sin\theta \\ \dot{\theta}(t) = u\dot{v} - v\dot{u} \end{cases}$$

В силу Принципа максимума:

$$\begin{cases} \dot{p} = -H_x = -rxyv \\ \dot{q} = -H_y = rxyu \\ \dot{r} = -H_z = 0 \end{cases}$$

Из (5) получаем, что $\dot{\alpha} = -rxyv$, $\dot{\beta} = rxyu$. Заметем, что из $\alpha^2+\beta^2 = 1$ следует, что

$$\frac{1}{2}\frac{d}{dt}(\alpha^2 + \beta^2) = \alpha\dot{\alpha} + \beta\dot{\beta} = -\alpha rxyv + \beta rxyu = 0$$

Таким образом,

$$u\dot{v} - v\dot{u} = \alpha rxyu + \beta rxyv = (\alpha u + \beta v)rxy = rxy$$

В итоге, получаем следующую систему уравнений

$$\begin{cases} \dot{x}(t) = \cos\theta(t) \\ \dot{y}(t) = \sin\theta(t) \\ \dot{\theta}(t) = rx(t)y(t) \\ x(0) = 0, y(0) = 1, \theta(0) = \theta_0 \end{cases} \tag{6}$$

Неизестные параметры системы r, θ_0 ищатся и условия минимизации времени T, за которое кривая должна дойти до точки $(0, 1)$.

5.3 Численные результаты

Решение ищется в виде:

$$\hat{x} = tN_1(t; \vec{p_1})$$
$$\hat{y} = 1 + tN_2(t; \vec{p_2})$$
$$\hat{\theta} = \theta_0 + tN_3(t; \vec{p_3})$$

Решение искалось последоватьельно на отрезках $[t_{k-1}, t_k], t_0 = 0$; параметры $\vec{p_1}, \vec{p_2}, \vec{p_3}$ находятся из условия:

$$Q(\vec{p}) = \frac{1}{3n}\sum_{i=1}^{n}[(\dot{\hat{x}}(s_i; \vec{p_1}) - \cos\hat{\theta}(s_i; \vec{p_3})))^2 + (\dot{\hat{y}}(s_i; \vec{p_2}) - \sin\hat{\theta}(s_i; \vec{p_3}))^2 +$$

$$(\dot{\hat{\theta}}(s_i; \vec{p_3}) - r\hat{x}(s_i; \vec{p_1})\hat{y}(s_i; \vec{p_2}))^2] \longrightarrow \min_{p_1, p_2, p_3},$$

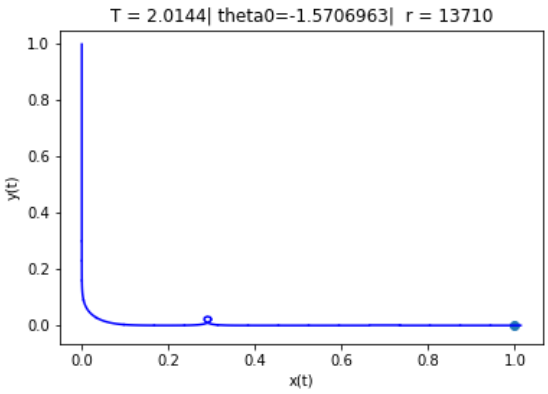
где $\{s_1, ..., s_n\}$ разбиение отрезка $[t_{k-1}, t_k]$

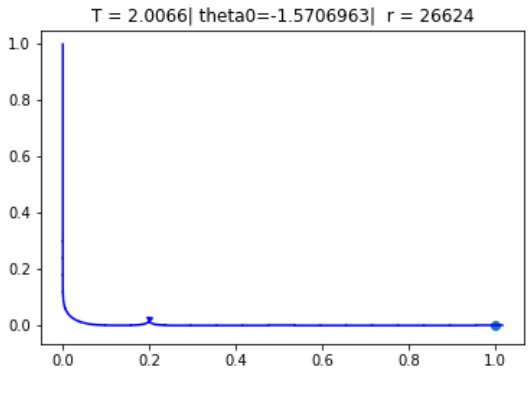
Условием окончания работы алгоритма минимизации являлось достижение

$$Q < 10^{-7},$$

$\max\left(|\dot{\hat{x}}(s_i; \vec{p_1}) - \cos\hat{\theta}(s_i; \vec{p_3})|, |\dot{\hat{y}}(t; \vec{p_2}) - \sin\hat{\theta}(s_i; \vec{p_3})|, |\dot{\hat{\theta}}(s_i; \vec{p_3}) - r\hat{x}(s_i; \vec{p_1})\hat{y}(s_i; \vec{p_2})|\right) < 10^{-3}$
$\forall i \in \{1...n\}$.

Ниже приведены налучшие варианты траекторий:





В обоих случаях $\theta_0 = -\frac{\pi}{2} + 0.0001$. При $r = 13710$ длина кривой получилась равной $T = 2.0144$, а при $r = 26624$ $T = 2.0066$.

6 Заключение

Рассмотренный метод численного решения обыкновенных дифференциальных уравнений с использованием искусственных нейронных сетей имеет ряд преимуществ, по сравнению с классическими численными методами. Во-первых, реализация метода не зависит от типа уравнения. Таким образом, поменяв только само уравнение, можно получить ответ без необходимости рассмотрения различных подходов. Во-вторых, даже обучив нейронную сеть на определенном разбиении необходимого отрезка, мы можем получить ответ в любой точке этого отрезка, так как в действительности мы аппроксимируем аналитическое решение функцией, заданной формулой (1), находя её параметры.

Далее была показана возможность применение фильтра Калмана для корректировки и улучшения точности аппроксимации аналитического решения. На данный момент не было найдено работ, где применялся бы такой метод для улучшения точности в подобных задачах, поэтому существует множество открытых областей исследования и вопросов. Например, оценка погрешностей q и r, а не выбор этих параметров произвольными малыми величинами. Также необходимо обобщить метод на случай системы уравнения и попытаться оценить уже матрицы Q и R. Но необходимо также отметить, что с помощью фильтрации Калмана можно исследовать вопрос о корректировке результата любого предсказателного алгоритма, например, регрессионных моделей, в эконометрике или другой области, где можно явно выписать и преобразовать к виду (4) исходную модель.

В параграфе 5 рассматривалась задача оптимального управлени, особенностью которой является экспоненциальный рост ошибки при численном решении гамильтоновой системы (6). Нейросетевой метод позволяет аппроксимировать решение на каждом из подотрезков, однако оптимальную траекторию построить проблематично, ввиду недостаточной точности алгоритма.

Список литературы

[1] Колмогоров А. Н. О представлении непрерывных функций нескольких переменных в виде суперпозиции непрерывных функций одного переменного // Докл. АН СССР. — 1958. — Т. 114, № 5. — С. 953–956.

[2] Воронцов К.В. Лекции по искусственным нейронным сетям, 2006 г. , 24 стр.

[3] A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. In: IEEE Transactions on Information Theory 39.3 (1993), pp. 930–945

[4] A. R. Barron. Approximation and estimation bounds for artificial neural networks. In: Machine Learning 14.1 (1994), pp. 115–133.

[5] Holden Lee, Rong Ge, Tengyu Ma, Andrej Risteski, Sanjeev Arora. On the ability of neural nets to express distributions, Proceedings of Machine Learning Research vol 65:1–26, 2017

[6] I.E. Lagaris, A.C. Likas, Artificial neural networks for solving ordinary and partial differential equations, 1997, arXiv:physics/9705023

[7] Effati, S. , Pakdaman, M. Optimal control problem via neural networks, Neural Computing and Applications (2013) 23: 2093. https://doi.org/10.1007/s00521-012-1156-2

[8] Joko Siswantoroa, Anton Satria Prabuwonoa, Aziz Abdullaha, Bahari Idrusa. A linear model based on Kalman filer for improving neural network classification performance. Expert Systems with Applications, Volume 49, 1 May 2016, Pages 112-122

[9] Л. В. Локуциевский, Ю. Л. Сачков, “Об интегрируемости по Лиувиллю субримановых задач на группах Карно глубины 4 и больше”, Матем. сб., 209:5 (2018), 74–119; L. V. Lokutsievskiy, Yu. L. Sachkov, “Liouville integrability of sub-Riemannian problems on Carnot groups of step 4 or greater”, Sb. Math., 209:5 (2018), 672–713

[10] М.И. Зеликин. Оптимальное управление и вариационное исчисление. УРСС, 2004