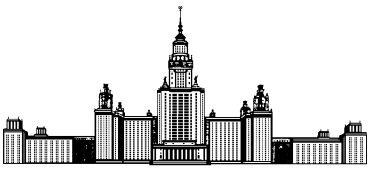
Московский государственный университет имени М. В. Ломоносова



Факультет Вычислительной Математики и Кибернетики

ОБЗОР НА СТАТЬЮ

«Neural Ordinary Differential Equations»

Москва, 2019

Содержание

1 Обыкновенные дифференциальные уравнения

1.1 Метод Эйлера

Рассмотрим обыкновенное дифференциальное уравнение первого порядка с заданным начальным условием

$$\begin{cases} \frac{\partial z(t)}{\partial t} = f(z(t), t), \quad t_0 \leq t \leq t_1, \\ z(t_0) = z_0. \end{cases} \tag{1}$$

Физический смысл решения ОДУ: по заданному уравнению скорости $\frac{\partial z(t)}{\partial t}$ найти уравнение траектории $z(t)$. Дифференциальное уравнение (1) эквивалентно интегральному уравнению

$$z(t) = z_0 + \int_{t_0}^{t} f(z(\tau), \tau) d\tau. \tag{2}$$

Как правило, на практике искать аналитическое решение $z(t)$ затруднительно, поэтому (1) или (2) решают численно. Одним из наиболее простых методов численного решения ОДУ является метод Эйлера. От непрерывной на отрезке $[t_0, t_1]$ задачи переходят к дискретной задаче, определенной на равномерной сетке с фиксированным шагом δ. Производную аппроксимируют разностной схемой вида

$$\frac{\partial z(t)}{\partial t} \approx \frac{z(t+\delta) - z(t)}{\delta}$$

и переходят к решению задачи

$$\frac{z(t+\delta) - z(t)}{\delta} = f(z(t), t).$$

Тогда $z(t)$ в узлах сетки находится в явном виде $z(t+\delta) = z(t) + \delta * f(z(t), t)$.

1.2 Связь метода Эйлера с нейросетевыми архитектурами

Пусть $z(t)$ — это t-й слой нейросети, а $f(z(t), t)$ — некоторое нелинейное преобразование t-го слоя. Во многих нейросетевых архитектурах (residual networks, recurrent neural network decoders, normalizing flows) следующий (t+1)-й слой получают как сумму предыдущего слоя и нелинейного преобразования:

$$z(t+1) = z(t) + f(z(t), t). \tag{3}$$

Данная схема в точности совпадает со схемой решения ОДУ методом Эйлера при $\delta = 1$. Таким образом, перечисленные выше архитектуры, фактически, реализуют дискретный аналог некоторой непрерывной «траектории» $z(t)$. Нейросетевые ОДУ предоставляют способ работы с непрерывными траекториями $z(t)$ без перехода к их дискретным аналогам.

2 Нейросетевые ОДУ

Пусть задано параметризованное семейство функций $f(z(t), t, \theta)$, и пусть при каждом фиксированном значении параметров θ данные функции задают ОДУ вида

$$\frac{\partial z(t)}{\partial t} = f(z(t), t, \theta), \quad t_0 \leq t \leq t_1. \tag{4}$$

На практике $f(z(t), t, \theta)$ — нейросеть с параметрами θ. Пусть задана функция потерь $L(z(t_1))$, которую мы хотим минимизировать. Искать $z(t_1)$ можно численно как $z(t_1) = ODESolve(z(t_0), t_0, t_1, f, \theta)$, где $ODESolve()$ — некоторый метод численного решения ОДУ. Тогда $L\left(z\left(t_1\right)\right) = L\left(ODESolve\left(z\left(t_0\right), t_0, t_1, f, \theta\right)\right) = L\left(z\left(t_0\right), t_0, t_1, \theta\right)$. Таким образом, свободными параметрами, по которым мы можем вести оптимизацию, являются: начальное условие $z(t_0)$, начальный момент времени t_0, конечный момент времени t_1, параметры нейросети θ.

Для того чтобы иметь возможность искать минимум функции потерь градиентными методами, необходимо научиться считать градиенты по свободным параметрам. В первую очередь, рассмотрим схему вычисления $\frac{dL}{d\theta}$. Можно показать, что

$$\frac{dL}{d\theta} = -\int_{t_1}^{t_0} a(t)^T \frac{\partial f(z(t), t, \theta)}{\partial \theta} dt, \tag{5}$$

где $a(t) = \frac{\partial L}{\partial z(t)}$ — так называемое сопряженное состояние. Производная $\frac{\partial f(z(t), t, \theta)}{\partial \theta}$ считается аналитически. Интеграл (5) можно вычислить численно при условии, что мы знаем $a(t) = \frac{\partial L}{\partial z(t)}$.

В общем случае посчитать $a(t) = \frac{\partial L}{\partial z(t)}$ аналитически невозможно. Однако известно, что справедливо равенство

$$\frac{da(t)}{dt} = -a(t)^T \frac{\partial f(z(t), t, \theta)}{\partial z}. \tag{6}$$

Производную $\frac{\partial f(z(t), t, \theta)}{\partial z}$ можно вычислить аналитически. Производная $\frac{\partial f(z(t), t, \theta)}{\partial z}$ является функцией от $z(t)$, t, θ. Таким образом, для численного решения ОДУ (6) достаточно знать

значения $z(t)$ в узлах сетки и конечное состояние $a(t_1) = \frac{\partial L}{\partial z(t_1)}$. Значения $z(t)$ в узлах сетки мы можем вычислить, решая ОДУ (4). А конечное состояние $a(t_1) = \frac{\partial L}{\partial z(t_1)}$ считается аналитически (известно, как функция потерь зависит от ответа модели). Таким образом, можно численно вычислить значение $\frac{dL}{d\theta}$.

Для вычисления остальных градиентов рассматривают аугментированное ОДУ, в котором

$$z_{aug} = [z, \theta, t]\,, \quad f_{aug} = [f, 0, 1]\,.$$

К аугментированному ОДУ применяют метод, описанный выше при нахождении $\frac{dL}{d\theta}$, и численно получают все необходимые градиенты, схема вычисления которых представлена ниже.

Algorithm 2 Complete reverse-mode derivative of an ODE initial value problem

Input: dynamics parameters θ, start time t_0, stop time t_1, final state $\mathbf{z}(t_1)$, loss gradient $\partial L / \partial \mathbf{z}(t_1)$

$\frac{\partial L}{\partial t_1} = \frac{\partial L}{\partial \mathbf{z}(t_1)}^{\mathsf{T}} f(\mathbf{z}(t_1), t_1, \theta)$ ▷ Compute gradient w.r.t. t_1

$s_0 = [\mathbf{z}(t_1), \frac{\partial L}{\partial \mathbf{z}(t_1)}, \mathbf{0}_{|\theta|}, -\frac{\partial L}{\partial t_1}]$ ▷ Define initial augmented state

def aug_dynamics($[\mathbf{z}(t), \mathbf{a}(t), \cdot, \cdot], t, \theta$): ▷ Define dynamics on augmented state

  return $[f(\mathbf{z}(t), t, \theta), -\mathbf{a}(t)^{\mathsf{T}} \frac{\partial f}{\partial \mathbf{z}}, -\mathbf{a}(t)^{\mathsf{T}} \frac{\partial f}{\partial \theta}, -\mathbf{a}(t)^{\mathsf{T}} \frac{\partial f}{\partial t}]$ ▷ Compute vector-Jacobian products

$[\mathbf{z}(t_0), \frac{\partial L}{\partial \mathbf{z}(t_0)}, \frac{\partial L}{\partial \theta}, \frac{\partial L}{\partial t_0}] = \text{ODESolve}(s_0, \text{aug\_dynamics}, t_1, t_0, \theta)$ ▷ Solve reverse-time ODE

return $\frac{\partial L}{\partial \mathbf{z}(t_0)}, \frac{\partial L}{\partial \theta}, \frac{\partial L}{\partial t_0}, \frac{\partial L}{\partial t_1}$ ▷ Return all gradients

Рис. 1: Схема нахождения градиентов функции потерь по свободным параметрам.

Очевидно, что чем точнее мы хотим искать минимум функции потерь, тем точнее нам надо искать производные. Таким образом, возникает некоторый компромисс между точностью решения ODESolve и между вычислительной эффективностью.

Во многих приложениях начальное условие $z(t_0)$ может быть задано. Например, пусть дана обучающая выборка $\{(x, y)\}$, где x — начальные данные, а y — некоторые целевые значения. Тогда в качестве начальных условий ОДУ задаются входные данные $z(t_0) = x$. В этом случае оптимизация по $z(t_0)$, очевидно, не ведется.

3 Приложения нейросетевых ОДУ

3.1 Обучение с учителем

Решается задача классификации изображений рукописных цифр (датасет MNIST). В подобных задачах себя зарекомендовала архитектура ResNet, в которой используются преобразования слоев вида (3). Последовательность подобных преобразований авторы статьи заменяют на нейросетевое ОДУ.

Как видно на рисунке 2, качество классификации нейросетевого ОДУ и ResNet сети практически совпадает, однако ОДУ-сеть (ODE-Net) хранит примерно в три раза меньше параметров, что дает существенный выигрыш в памяти.

	Test Error	# Params	Memory	Time
1-Layer MLP†	1.60%	0.24 M	-	-
ResNet	0.41%	0.60 M	$\mathcal{O}(L)$	$\mathcal{O}(L)$
RK-Net	0.47%	0.22 M	$\mathcal{O}(\tilde{L})$	$\mathcal{O}(\tilde{L})$
ODE-Net	0.42%	0.22 M	$\boldsymbol{\mathcal{O}(1)}$	$\mathcal{O}(\tilde{L})$

Рис. 2: Сравнение работы ResNet и ОДУ-сети.

3.2 Нормализующие потоки

Пусть плотность распределения случайной величины z известна и равна $p_z(z)$. Пусть $x = g(z)$ — биективное отображение. Обратное ему отображение $z = f(x)$. Значение x — будет являться случайной величиной (функция от случайной величины — случайная величина), распределенной по закону $p_x(x)$. Функции плотностей распределения случайных величин z и x связаны формулами

$$p_x(x) = p_z\left(f\left(x\right)\right)\left|det\left(\frac{\partial f(x)}{\partial x^T}\right)\right|, \quad \log p_x(x) = \log p_z\left(f\left(x\right)\right) + \log\left|det\left(\frac{\partial f(x)}{\partial x^T}\right)\right| \tag{7}$$

где $\frac{\partial f(x)}{\partial x^T}$ — матрица Якоби однозначного отображения $f(x)$.

Данный факт используют следующим образом. Пусть дана обучающая выборка $X \sim p_x(x)$. Мы хотим найти неизвестное распределение $p_x(x)$, для того чтобы иметь возможность генерировать новые объекты, которых нет в обучающей выборке. При этом распределение $p_x(x)$ довольно сложное и мы не можем оценить его стандартными статистическими методами. Тогда фиксируют некоторое известное распределение $p_z(z)$ (например,

нормальное) и пытаются обучить обратимое отображение $z = f(x)$ согласно формуле (7), которое будет преобразовывать неизвестно распределенную случайную величину x в известно распределенную случайную величину z. Если это удается сделать, то генерировать новые объекты x можно, сэмплируя $z \sim p_z(z)$ и вычисляя $x = g(z)$, где $g(z)$ — обратная к $f(x)$ функция.

Основная проблема данного метода заключается в том, что в общем случае сложность вычисления определителя в (7) — кубическая. Поэтому на практике рассматривают функции $f(x)$ специального вида, при которых сложность вычисления определителя линейная. Кроме того, на практике обычно применяют несколько последовательных преобразований. В качестве примера подобных преобразований можно рассмотреть планарный нормализующий поток

$$z(t+1) = z(t) + uh(w^T z(t) + b),$$

где $h()$ — некоторая нелинейная функция, а u и w — векторы соответствующих размерностей.

В статье «Neural Ordinary Differential Equations» предлагается заменить «дискретный» нормализующий поток непрерывным аналогом вида

$$\frac{dz(t)}{dt} = \sum_{n=1}^{M} f_n\left(z\left(t\right)\right).$$

В данном случае плотность распределения определяется следующим выражением

$$\frac{d \log p\left(z\left(t\right)\right)}{dt} = \sum_{n=1}^{M} tr\left(\frac{\partial f_n}{\partial z}\right). \tag{8}$$

Сложность вычисления правой части (8) — линейная, то есть применение подобной схемы является вычислительно эффективным.

В качестве примера рассматриваются три распределения и нормализующие потоки (дискретные и непрерывные), восстанавливающие эти распределения. Как видно, в двух случаях из трех непрерывный поток восстанавливает распределение лучше, чем его дискретный аналог.

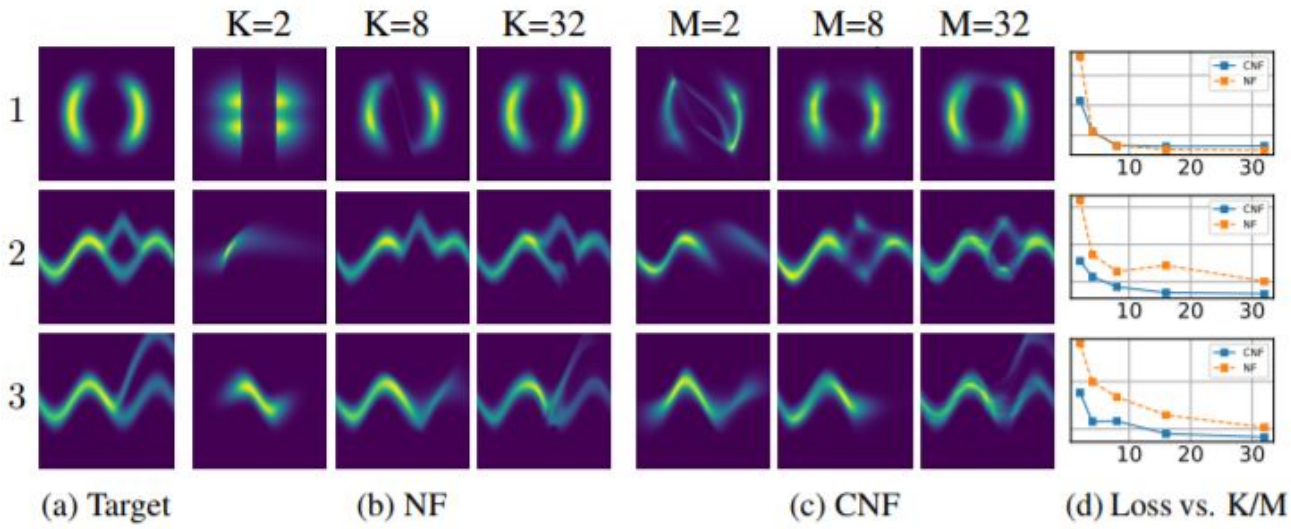


Рис. 3: Сравнение работы дискретных и непрерывных нормализующих потоков.

3.3 Генеративные модели временных рядов

Пусть имеется временной ряд $x_{t_1}, \ldots, x_{t_N}, \ldots, x_{t_M}$. Причем значения $x_{t_1}, \ldots, x_{t_N}$ мы наблюдаем — это обучающая выборка, а значения $x_{t_{N+1}}, \ldots, x_{t_M}$ предсказываем — тестовая выборка.

Рассматривается следующая модель. Обучающая выборка подается на вход реккурентной нейросети (кодировщик). Кодировщик выдает параметры, как правило, нормального распределения: матожидание μ и стандартное отклонение σ. Из полученного распределения сэмплируется случайная величина $z_{t_0} \sim q(z_{t_0}|\mu, \sigma)$. Значение z_{t_0} подается на вход другой реккурентной нейросети (декодировщик), которая обучается восстанавливать значения $x_{t_1}, \ldots, x_{t_N}$, поданные на вход кодировщику. Затем генерируются оставшиеся «тестовые» значения. Реккурентная сеть использует дискретные преобразования вида (3), которые авторы статьи заменили на непрерывный аналог ОДУ-сетью. Схема работы ОДУ-сети изображена на рисунке 4. Как видно на рисунке 5, новый подход лучше восстанавливает рассматриваемые временные ряды. Кроме того, в силу непрерывности, имеется возможность достаточно просто получать значения временного ряда в любые моменты времени.

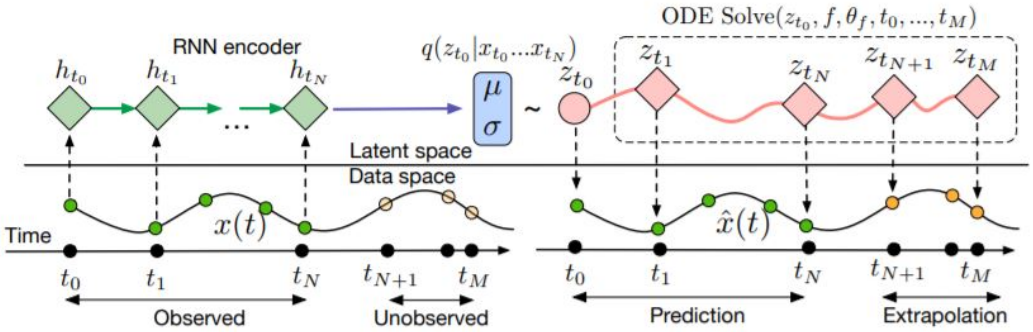
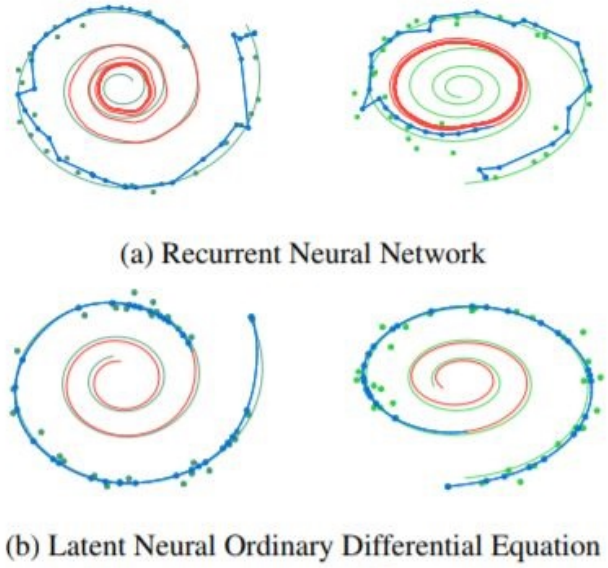


Рис. 4: Схема работы ОДУ-сети.



(a) Recurrent Neural Network

(b) Latent Neural Ordinary Differential Equation

Рис. 5: Восстановление траектории временного ряда.

4 Заключение

В настоящем обзоре рассмотрены основные теоретические результаты, полученные в оригинальной статье. Поставлена задача решения ОДУ, описан метод Эйлера и показана его связь с нейросетевыми архитектурами. Описан метод обучения ОДУ-сетей.

В обзоре большое внимание уделено возможным приложениям теоретических результатов. Рассмотрены такие задачи как обучение с учителем, восстановление плотности распределения (нормализующие потоки), предсказание временных рядов. Показано, в чем ОДУ-сети «выигрывают» у стандартных архитектур.