Нейронные дифференциальные уравнения

Neural Ordinary Differential Equations

NeurIPS 2018

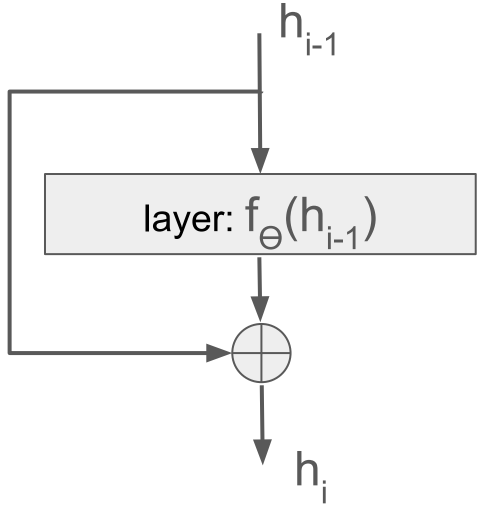
Neural Ordinary Differential Equations

Ricky T. Q. Chen*, Yulia Rubanova*, Jesse Bettencourt*, David Duvenaud
University of Toronto, Vector Institute
{rtqichen, rubanova, jessebett, duvenaud}@cs.toronto.edu

Lev Semenovich Pontryagin, EF Mishchenko, VG Boltyanskii, and RV Gamkrelidze. The mathematical theory of optimal processes. 1962.

Краткий обзор и обозначения

Технически, нейронные дифференциальные уравнения получены как предельный случай так называемых *residual networks* (ResNets). Выход i-го слоя ResNets определяется как:



$$h_i = h_{i-1} + f_\theta^i(h_{i-1}).$$

Краткий обзор и обозначения

Представим глубокую сеть:

$$h_i = h_{i-1} + \varepsilon f_\theta^i(h_{i-1}), \quad i = 1, \ldots, N,$$

$$\varepsilon \sim 1/N$$

Это формула хорошо известного метода Эйлера для решения дифференциального уравнения. Если N стремится к бесконечности, последовательность значений hi стремится к решению дифференциального уравнения:

$$\dot{x}(t) = f_\theta(t, x(t)),$$

$$x(t) \in \mathbb{R}^n \quad x_{input} \in \mathbb{R}^n$$

Краткий обзор и обозначения

Пусть решением диф. уравнения

$$\dot{x}(t) = f_\theta(t, x(t)),$$

с траекторий проходящей через точку:

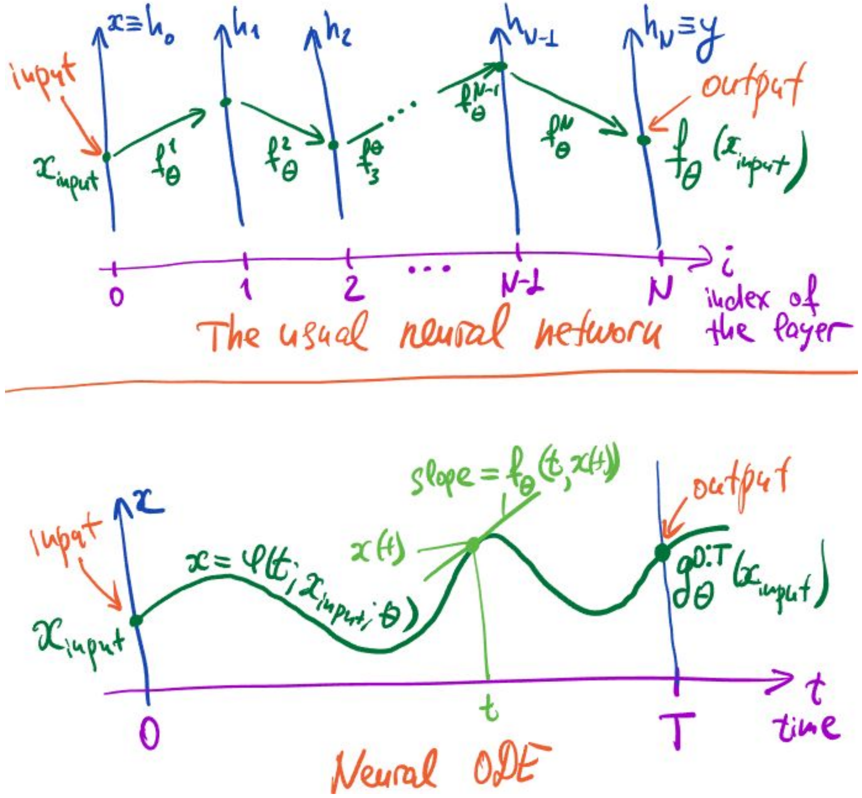
$$t = 0, x = x_{input}$$

будет: $\varphi(t; x_{input}; \theta)$

По определению: $\varphi(0; x_{input}; \theta) = x_{input}$

Выходной вектор нашей сети, это решение в точке T. Если поменять входной вектор, изменится решение и выходной вектор. Т.е. наша сеть определяет отображение:

$$g_\theta^{0:T} : \mathbb{R}^n \to \mathbb{R}^n,$$
$$g_\theta^{0:T}(x_{input}) = \varphi(T; x_{input}; \theta),$$



Функция потерь и градиент по параметрам

Как обычно у нас должна быть функция потерь:

$$\mathcal{L}(\theta) = L(g_\theta^{0:T}(x_{input}), y_{true})$$

И ее градиент по параметрам:

$$\nabla_\theta \mathcal{L}(\theta) = \nabla_y L \cdot \frac{\partial g_\theta^{0:T}(x_{input})}{\partial \theta}$$

Второй множитель в правой части мы перепишем как якобиан решения диф.уравнения:

$$\frac{\partial g_\theta^{0:T}(x_{input})}{\partial \theta} = \frac{\partial \varphi(x_{input}; T; \theta)}{\partial \theta}$$

Таким образом, нам надо найти производную решения диф.уравнения по параметрам

Градиент функции потерь

Theorem. For each $t \in [0, T]$ and some fixed θ, let us denote

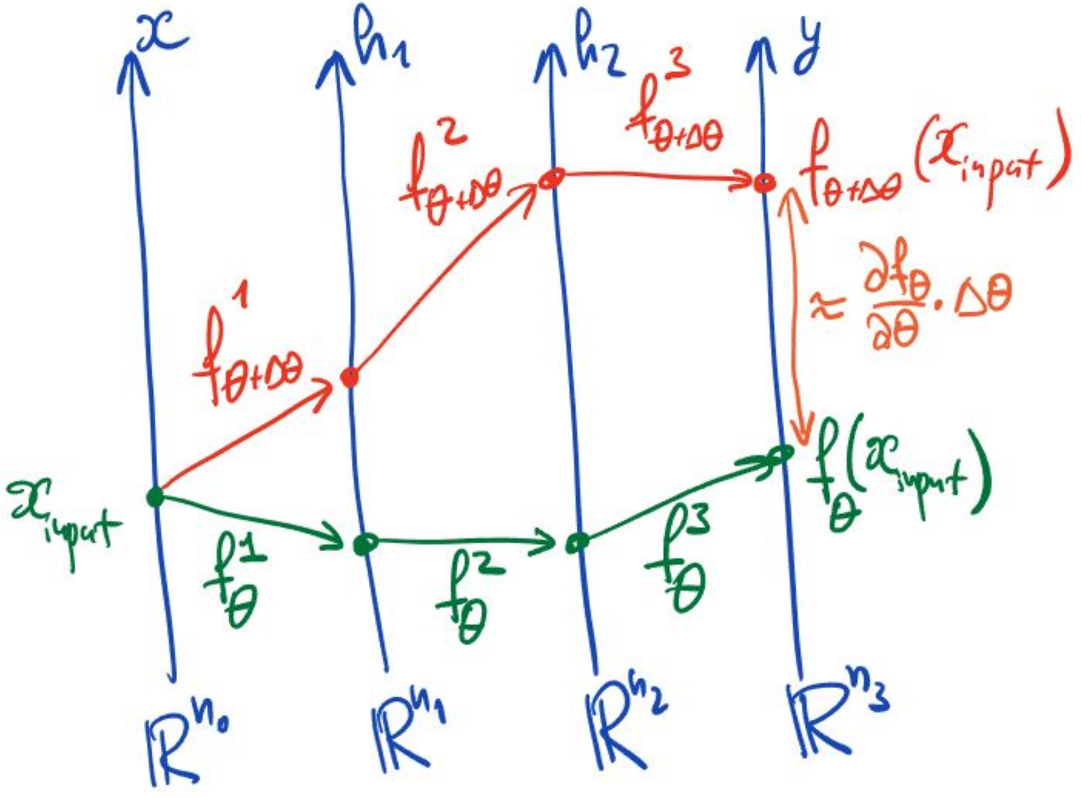
$$v(t) := \frac{\partial \varphi(x_{input}; t; \theta)}{\partial \theta}.$$

So $v(t)$ measures how the solution of our equation at point t depends on the parameters θ. Then (under reasonable assumptions) v satisfies the following differential equation:

$$\dot{v} = \frac{\partial f_\theta(t, x(t))}{\partial \theta} + \frac{\partial f_\theta(t, x(t))}{\partial x} v, \qquad (16)$$

where $x(t) = \varphi(t; x_{input}; \theta)$.

Градиент функции потерь (нейросеть)



Градиент функции потерь (нейросеть)

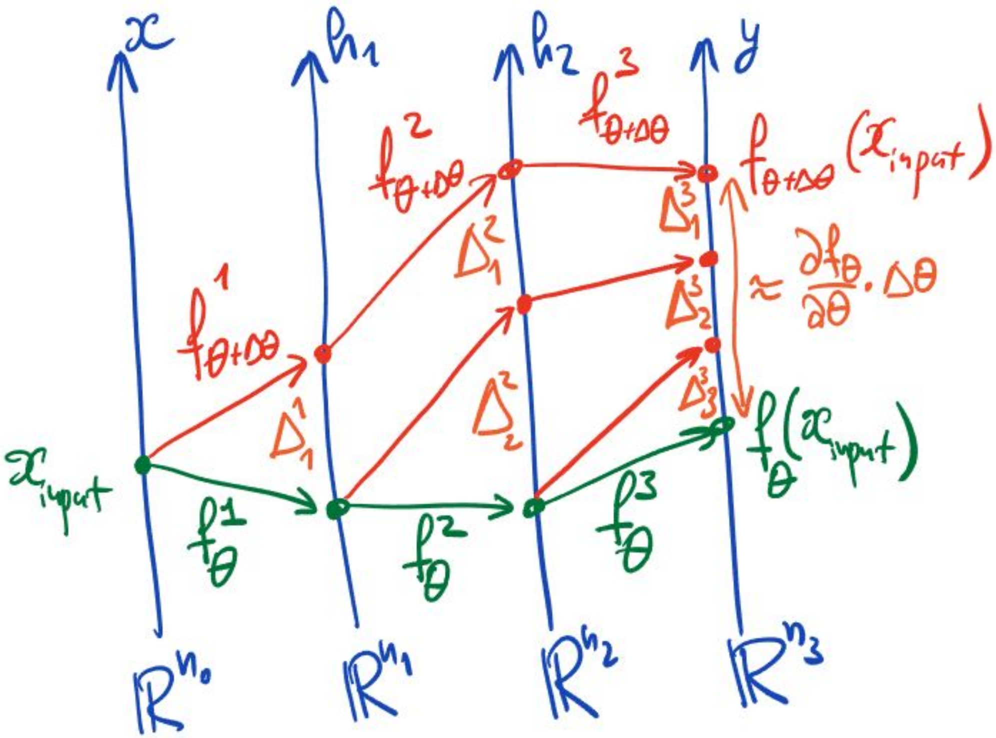
$$\Delta_3^3 \approx \frac{\partial f_\theta^3}{\partial \theta}\Delta\theta.$$

$$\Delta_2^2 \approx \frac{\partial f_\theta^2}{\partial \theta}\Delta\theta.$$

$$\Delta_2^3 \approx \frac{\partial f_{\theta+\Delta\theta}^3}{\partial h_2}\Delta_2^2 \approx \frac{\partial f_{\theta+\Delta\theta}^3}{\partial h_2}\frac{\partial f_\theta^2}{\partial \theta}\Delta\theta.$$

$$\Delta_1^3 \approx \frac{\partial f_{\theta+\Delta\theta}^3}{\partial h_2}\Delta_1^2 \approx \frac{\partial f_{\theta+\Delta\theta}^3}{\partial h_2}\frac{\partial f_{\theta+\Delta\theta}^2}{\partial h_1}\Delta_1^1$$

$$\approx \frac{\partial f_{\theta+\Delta\theta}^3}{\partial h_2}\frac{\partial f_{\theta+\Delta\theta}^2}{\partial h_1}\frac{\partial f_\theta^1}{\partial \theta}\Delta\theta.$$



Градиент функции потерь (нейросеть)

$$\frac{\partial f_\theta}{\partial \theta}\Delta\theta \approx \Delta_3^3 + \Delta_2^3 + \Delta_1^3 \approx$$
$$\left(\frac{\partial f_\theta^3}{\partial \theta} + \frac{\partial f_{\theta+\Delta\theta}^3}{\partial h_2}\frac{\partial f_\theta^2}{\partial \theta} + \frac{\partial f_{\theta+\Delta\theta}^3}{\partial h_2}\frac{\partial f_{\theta+\Delta\theta}^2}{\partial h_1}\frac{\partial f_\theta^1}{\partial \theta}\right)\Delta\theta$$

В пределе $\Delta\theta \to 0$ приближенное равенство становится точным:

$$\frac{\partial f_\theta}{\partial \theta} = \frac{\partial f_\theta^3}{\partial \theta} + \frac{\partial f_\theta^3}{\partial h_2}\frac{\partial f_\theta^2}{\partial \theta} + \frac{\partial f_\theta^3}{\partial h_2}\frac{\partial f_\theta^2}{\partial h_1}\frac{\partial f_\theta^1}{\partial \theta}$$

Градиент функции потерь (нейросеть)

$$\nabla_\theta \mathcal{L} = \nabla_y L \cdot \frac{\partial f_\theta}{\partial \theta} =$$

$$\nabla_y L \cdot \left(\frac{\partial f_\theta^3}{\partial \theta} + \frac{\partial f_\theta^3}{\partial h_2} \frac{\partial f_\theta^2}{\partial \theta} + \frac{\partial f_\theta^3}{\partial h_2} \frac{\partial f_\theta^2}{\partial h_1} \frac{\partial f_\theta^1}{\partial \theta} \right) =$$

$$\nabla_y L \frac{\partial f_\theta^3}{\partial \theta} + \nabla_y L \frac{\partial f_\theta^3}{\partial h_2} \frac{\partial f_\theta^2}{\partial \theta} + \nabla_y L \frac{\partial f_\theta^3}{\partial h_2} \frac{\partial f_\theta^2}{\partial h_1} \frac{\partial f_\theta^1}{\partial \theta}.$$

$$\begin{aligned}\nabla_\theta \mathcal{L}(\theta) = \nabla_y L \frac{\partial f_\theta^3}{\partial \theta} &+ \left(\nabla_y L \frac{\partial f_\theta^3}{\partial h_2} \right) \frac{\partial f_\theta^2}{\partial \theta} \\ &+ \left(\nabla_y L \frac{\partial f_\theta^3}{\partial h_2} \right) \frac{\partial f_\theta^2}{\partial h_1} \frac{\partial f_\theta^1}{\partial \theta}\end{aligned}$$

Градиент функции потерь (нейросеть)

Еще немного преобразований:

$$\nabla_y L \frac{\partial f_\theta^3}{\partial h_2} = \nabla_{h_2}(L \circ f_\theta^3) \qquad \nabla_y L \frac{\partial f_\theta^3}{\partial h_2} \frac{\partial f_\theta^2}{\partial h_1} = \nabla_{h_1}(L \circ f_\theta^3 \circ f_\theta^2)$$

Введем новое обозначение: $h_3 \equiv y$

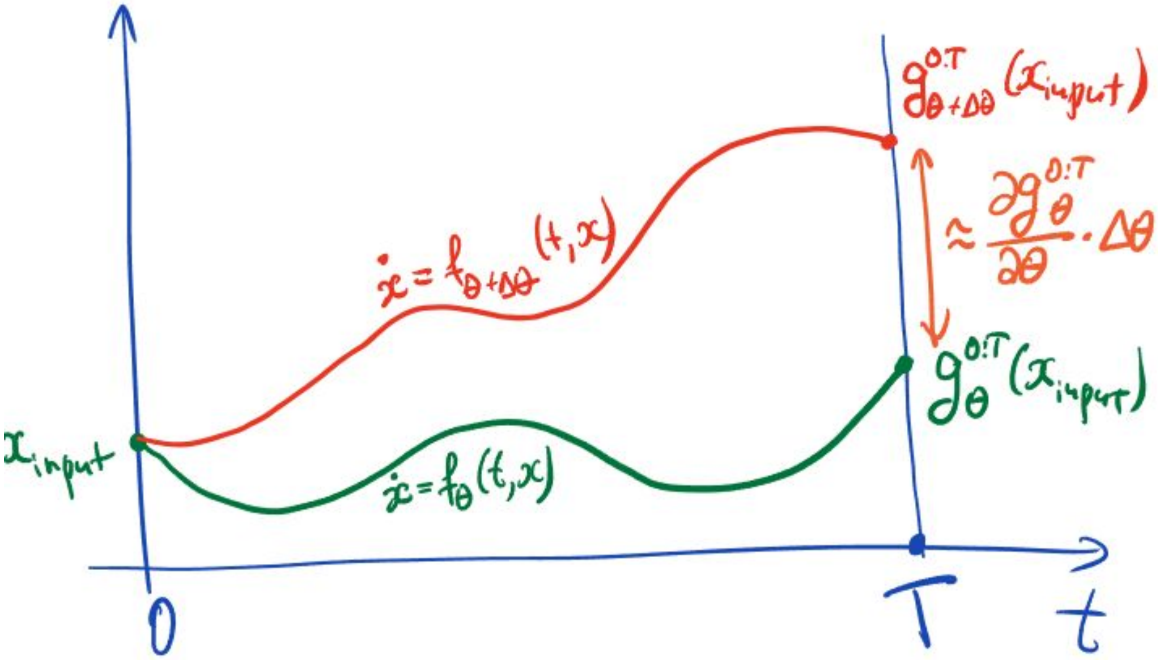
$$\nabla_{h_2} L = \nabla_{h_3} L \cdot \frac{\partial f_\theta^3}{\partial h_2}, \qquad \nabla_{h_1} L = \nabla_{h_2} L \cdot \frac{\partial f_\theta^2}{\partial h_1}.$$

$$\nabla_\theta \mathcal{L}(\theta) = \nabla_{h_3} L \frac{\partial f_\theta^3}{\partial \theta} + \nabla_{h_2} L \frac{\partial f_\theta^2}{\partial \theta} + \nabla_{h_1} L \frac{\partial f_\theta^1}{\partial \theta}$$

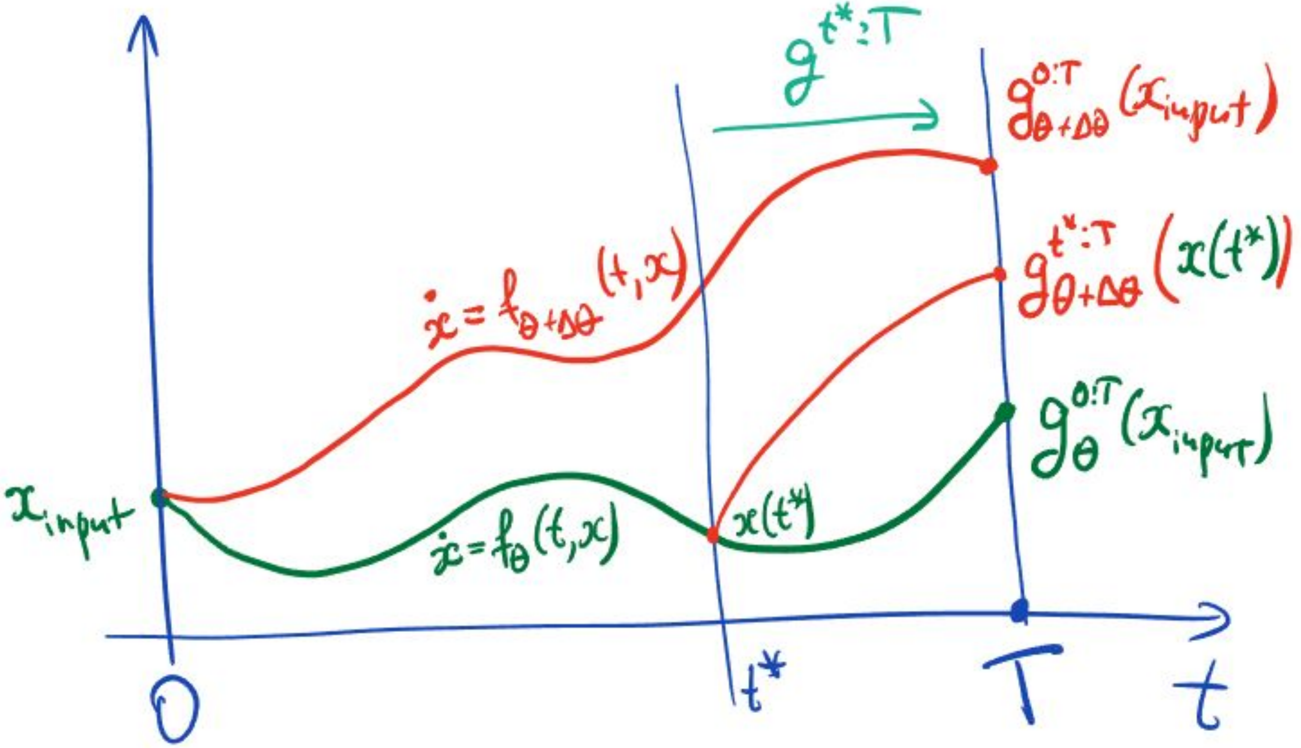
$$\nabla_\theta \mathcal{L}(\theta) = \sum_{i=N}^{1} \nabla_{h_i} L \frac{\partial f_\theta^i}{\partial \theta} \qquad \nabla_{h_i} L = \nabla_{h_{i+1}} L \cdot \frac{\partial f_\theta^{i+1}}{\partial h_i}, \quad i = N, \ldots, 1.$$

Градиент функции потерь (нейронные диф.уравнения)

$$\varphi(T;\theta+\Delta\theta)-\varphi(T;\theta)=g^{0:T}_{\theta+\Delta\theta}(x_{input})-g^{0:T}_{\theta}(x_{input})\approx\frac{\partial g^{0:T}_{\theta}}{\partial\theta}\Delta\theta.$$



Градиент функции потерь (нейронные диф.уравнения)

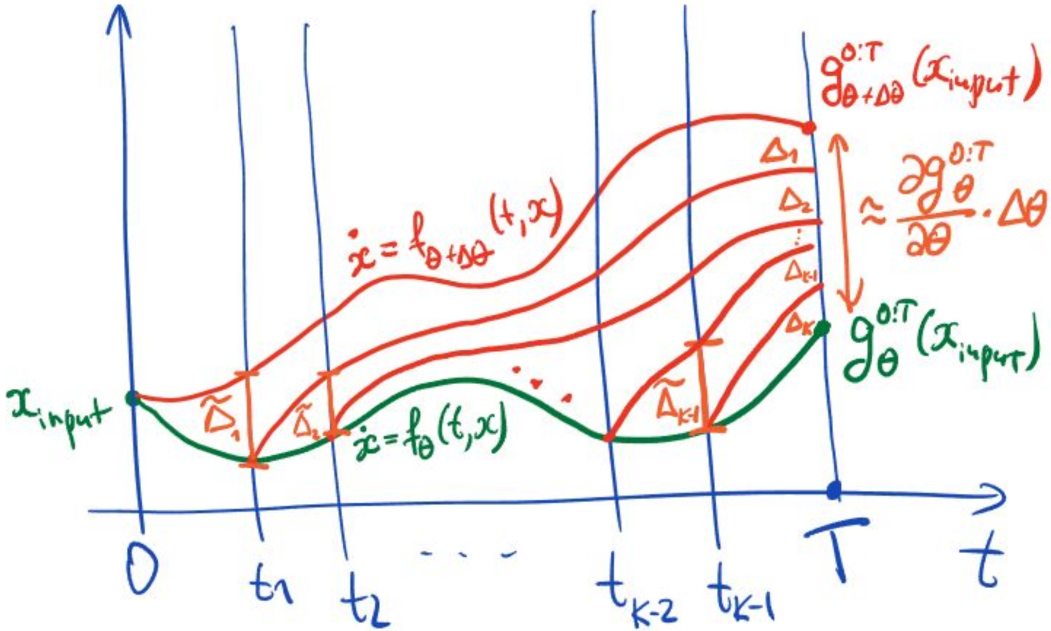


Градиент функции потерь (нейронные диф.уравнения)

Разобьем сегмент [0, T] на K сегментов равной длины и обозначим конечные точки t0, t1, …tk-1, где t0 = 0, tk = T

$$\frac{\partial g_\theta^{0:T}(x_{input})}{\partial \theta} \Delta\theta \approx \sum_{j=1}^{K} \Delta_j.$$

$$\Delta_j \approx \frac{\partial g_{\theta+\Delta\theta}^{t_j:T}(x(t_j))}{\partial x} \tilde{\Delta}_j.$$

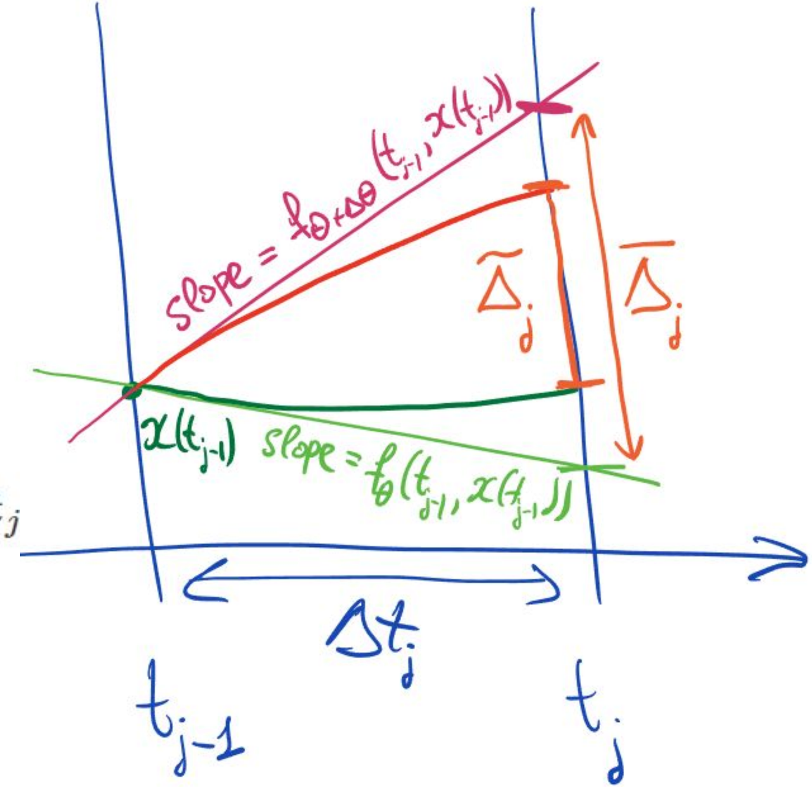


Градиент функции потерь (нейронные диф.уравнения)

$$\bar{\Delta}_j = (f_{\theta+\Delta\theta}(t_{j-1}, x(t_{j-1})) - f_\theta(t_{j-1}, x(t_{j-1})))\Delta t_j$$

$$\tilde{\Delta}_j \approx \frac{\partial f_\theta(t_{j-1}, x(t_{j-1}))}{\partial \theta} \Delta\theta \Delta t_j$$

$$\Delta_j \approx \frac{\partial g^{t_j:T}_{\theta+\Delta\theta}(x(t_j))}{\partial x} \frac{\partial f_\theta(t_{j-1}, x(t_{j-1}))}{\partial \theta} \Delta\theta\, \Delta t_j$$



Градиент функции потерь (нейронные диф.уравнения)

$$\frac{\partial g_\theta^{0:T}(x_{input})}{\partial \theta}\Delta\theta \approx \sum_{j=1}^{K} \frac{\partial g_{\theta+\Delta\theta}^{t_j:T}(x(t_j))}{\partial x} \frac{\partial f_\theta(t_{j-1}, x(t_{j-1}))}{\partial \theta} \Delta\theta\, \Delta t_j$$

Это выражение становится интегральной суммой при $\Delta t \to 0$:

$$\frac{\partial g_\theta^{0:T}(x_{input})}{\partial \theta}\Delta\theta \approx \int_0^T \frac{\partial g_{\theta+\Delta\theta}^{t:T}(x(t))}{\partial x} \frac{\partial f_\theta(t, x(t))}{\partial \theta} \Delta\theta\, dt = \left(\int_0^T \frac{\partial g_{\theta+\Delta\theta}^{t:T}(x(t))}{\partial x} \frac{\partial f_\theta(t, x(t))}{\partial \theta}\, dt\right)\Delta\theta.$$

И при $\Delta\theta \to 0$:

$$\frac{\partial g_\theta^{0:T}(x_{input})}{\partial \theta} = \int_0^T \frac{\partial g_\theta^{t:T}(x(t))}{\partial x} \frac{\partial f_\theta(t, x(t))}{\partial \theta}\, dt$$

Градиент функции потерь (нейронные диф.уравнения)

$$\begin{aligned}\nabla_\theta \mathcal{L}(\theta) &= \nabla_y L \cdot \int_0^T \frac{\partial g_\theta^{t:T}(x(t))}{\partial x} \frac{\partial f_\theta(t, x(t))}{\partial \theta}\, dt \\ &= \int_0^T \left(\nabla_y L \frac{\partial g_\theta^{t:T}(x(t))}{\partial x} \right) \frac{\partial f_\theta(t, x(t))}{\partial \theta}\, dt\end{aligned}$$

$$\nabla_y L \frac{\partial g_\theta^{t:T}(x(t))}{\partial x} = \nabla_x (L \circ g_\theta^{t:T}(x)),$$

$$\nabla_\theta \mathcal{L}(\theta) = \int_0^T \nabla_{x(t)} L \frac{\partial f_\theta(t, x(t))}{\partial \theta}\, dt.$$

Сопряженное уравнение

Обозначим $\nabla_{x(t)} L$ как $a(t)$

Л.В.Понтрягин показал, что $a(t)$ удовлетворяет следующему диф.уравнению:

$$\dot{a}(t) = -a(t)\frac{\partial f(t, x)}{\partial x} \qquad a(T) = \nabla_y L(y, y_{true}), \quad y = x(T)$$

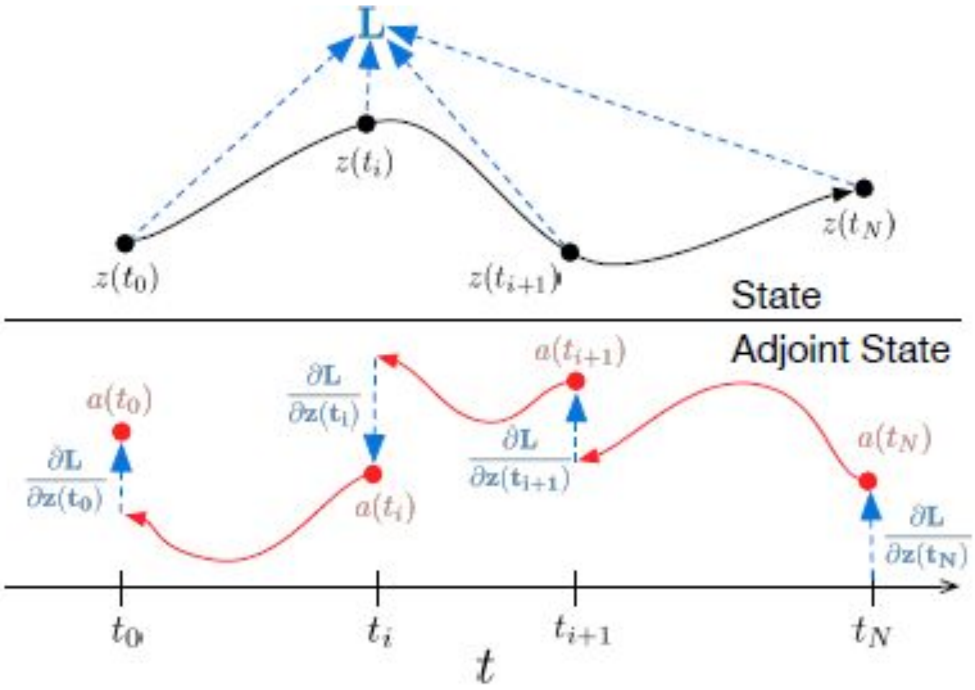
Backpropagation в нейронных диф.уравнениях

$$\begin{cases} \dot{x} = f_\theta(t, x), \\ \dot{a} = -a \cdot \frac{\partial f_\theta(t,x)}{\partial x}, \\ \dot{u} = -a \cdot \frac{\partial f_\theta(t,x)}{\partial \theta}. \end{cases} \quad \begin{cases} x(T) = x_{output} \\ a(T) = \nabla_y L(y, y_{true}), \quad y = x_{output} \\ u(T) = 0 \end{cases}$$

$$u(t) = -\int_T^t a(\tau) \cdot \frac{\partial f_\theta(\tau, x(\tau))}{\partial \theta} d\tau = \int_t^T a(\tau) \cdot \frac{\partial f_\theta(\tau, x(\tau))}{\partial \theta} d\tau.$$

$$u(0) = \nabla_\theta \mathcal{L}(\theta)$$

Обратное дифференцирование



Алгоритм

Algorithm 1 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 ${}^{\partial L}/_{\partial \mathbf{z}(t_1)}$

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

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

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

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

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

Сравнение с NN

Table 1: Performance on MNIST. †From LeCun et al. (1998).

	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	$\boldsymbol{\mathcal{O}(1)}$	$\mathcal{O}(\tilde{L})$

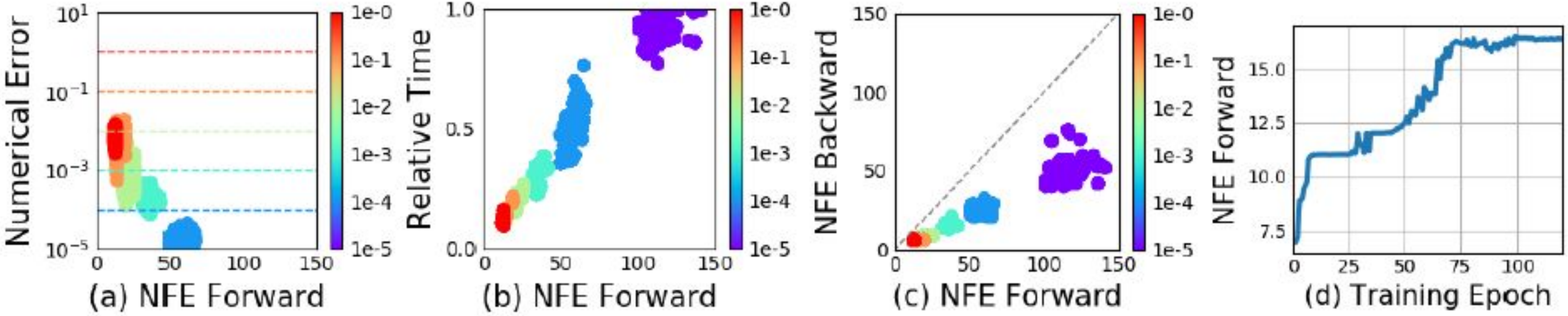
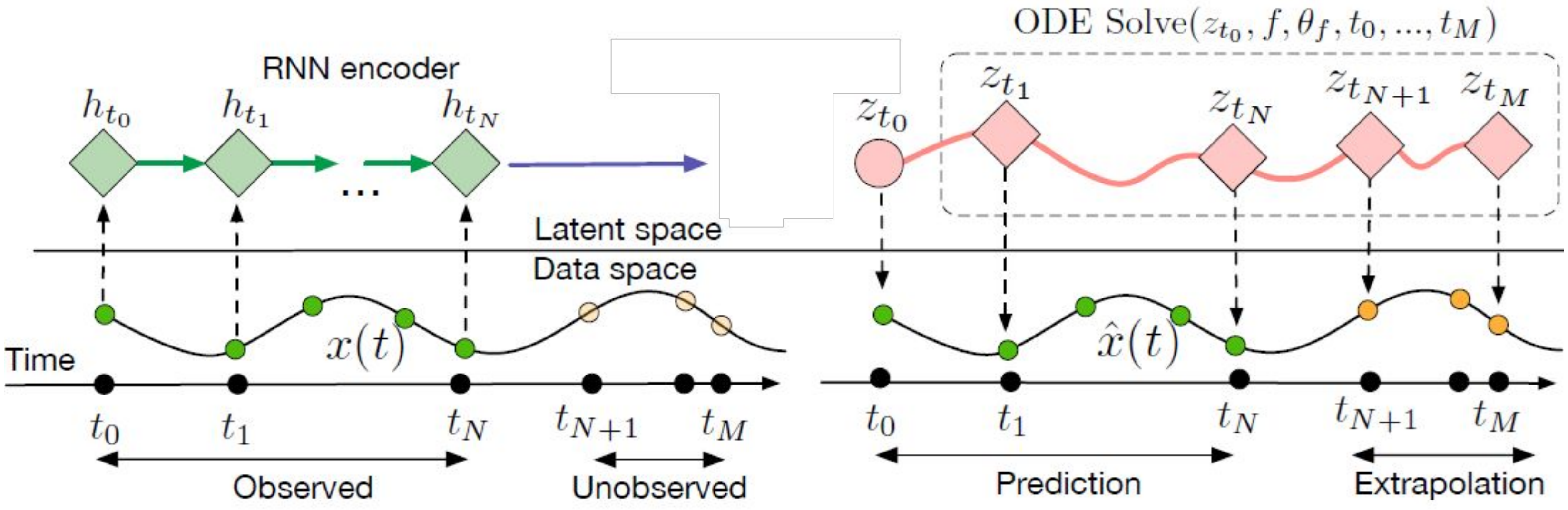


Figure 3: Statistics of a trained ODE-Net. (NFE = number of function evaluations.)

Временные ряды с нерегулярными интервалами



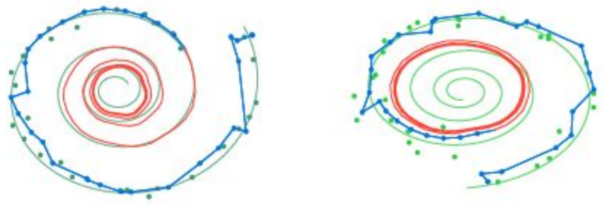
Временные ряды с нерегулярными интервалами

Bi-directional spiral dataset We generated a dataset of 1000 2-dimensional spirals, each starting at a different point, sampled at 100 equally-spaced timesteps. The dataset contains two types of spirals: half are clockwise while the other half counter-clockwise. To make the task more realistic, we add gaussian noise to the observations.

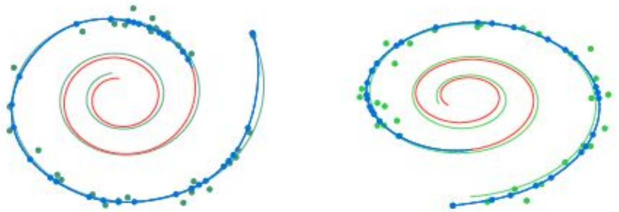
Table 2: Predictive RMSE on test set

# Observations	30/100	50/100	100/100
RNN	0.3937	0.3202	0.1813
Latent ODE	**0.1642**	**0.1502**	**0.1346**

Временные ряды с нерегулярными интервалами



(a) Recurrent Neural Network



(b) Latent Neural Ordinary Differential Equation

Примеры применения в биологии

Predicting metabolomic profiles from microbial composition through neural ordinary differential equations, March 2023, Nature Machine Intelligence

Tong Wang,[1] Xu-Wen Wang,[1] Kathleen A. Lee-Sarwar,[1,2] Augusto A. Litonjua,[3] Scott T. Weiss,[1] Yizhou Sun,[4] Sergei Maslov,[5,6] and Yang-Yu Liu[1,5,*]

Прогнозирование метаболомного профиля

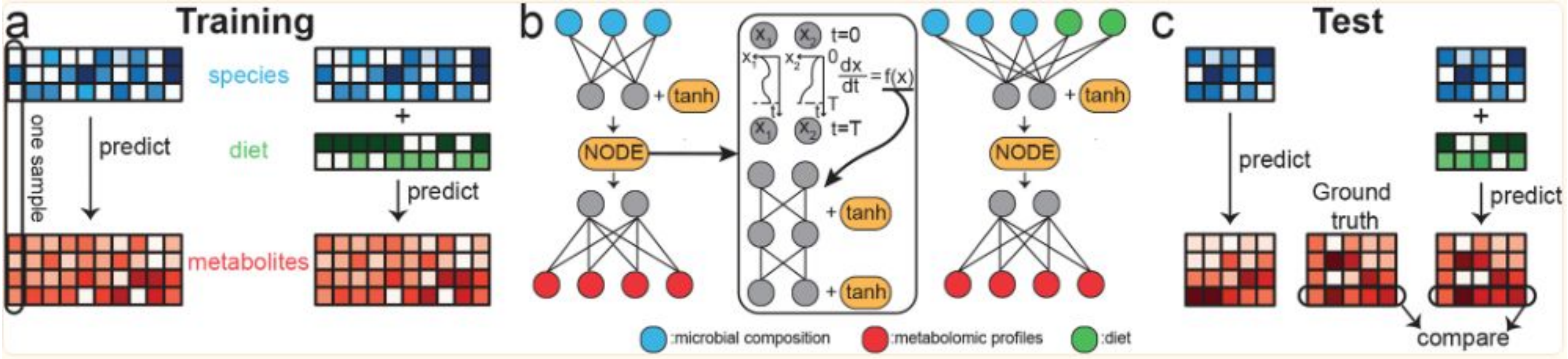
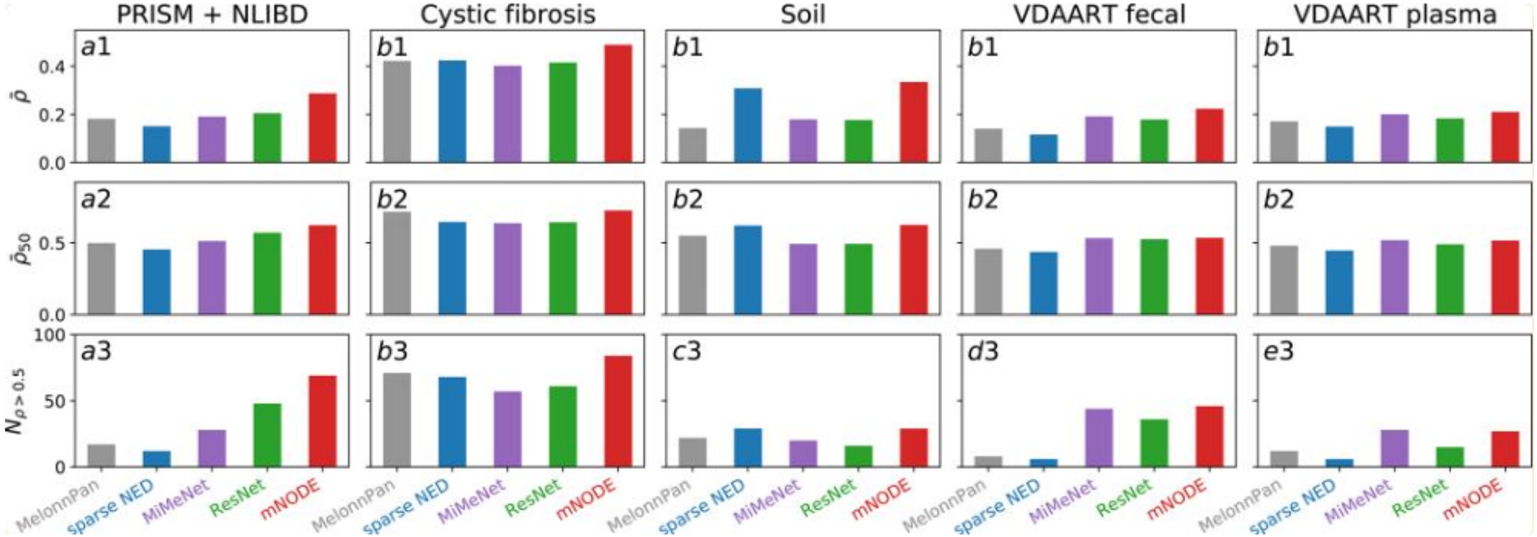


Схема работы mNODE для прогнозирования метаболомных профилей на основе видового микробного состава и информации о питании.

На всех панелях массивы синего цвета представляют микробный состав. Массивы зеленого цвета представляют информацию о питательных веществах. Массивы красного цвета представляют метаболомные профили. Ни в одном из массивов нет недостающих данных, белые квадраты во всех массивах означают небольшие значения. Для иллюстрации идеи используются в общей сложности 15 гипотетических образцов с 3 видами, 2-мя питательными средами и 4 метаболитами. Выборки разделены на обучающую и тестовую с соотношением 2/1. 10 образцов в обучающем наборе используются для обучения модели. Существует два способа прогнозирования метаболомных профилей: один без включения питательных веществ во входные данные, а другой с питательными веществами, включенными в дополнение к микробным композициям. б Архитектура mNODE для двух подходов к обучению. Нейронное ОДУ представляет собой модуль в середине архитектуры и вычисляет временную эволюцию ОДУ, чьи производные по времени первого порядка аппроксимируются с помощью MLP с одним скрытым слоем. Серые узлы представляют нейроны в скрытых слоях mNODE. с Полностью обученный mNODE может генерировать прогнозы для метаболомных профилей в тестовом наборе.ф

Прогнозирование метаболомного профиля



Сравнение качества прогноза mNODE и существующих методов на наборах данных реальных микробных сообществ. Для сравнения характеристик модели используются три показателя: среднее SCC $\bar{\rho}$, среднее SCC $\bar{\rho}50$ для топ-50 и количество метаболитов с SCC больше 0,5 $N_{\rho>0,5}$. Все наборы данных случайным образом разделены на обучающие и тестовые с соотношением 80/20, за исключением набора данных PRISM и NLIBD. a1-a3. Результаты после обучения на PRISM и тестирования на NLIBD. б1-б3 Результаты на данных образцов легких больных муковисцидозом. c1-c3 Выполнение методов на данных образцов биокорки почвы после 5-ти увлажнений. d1-d3 Эффективность метода на данных образцов кала детей в возрасте 3 лет. e1-e3 Эффективность методов на данных образцов плазмы крови детей в возрасте 3 лет. SCC - Коэффициент корреляции Спирмена

Ассимиляция данных культивирования

Предположим у нас есть нестерильный процесс культивирования сообщества микроорганизмов. В сообществе есть основная культура и спутники. Концентрация основной культуры много больше концентрации спутников. Подобная ситуация имеет место в крупнотоннажной биотехнологии. Поскольку процесс не стерильный состав сообщества может дрейфовать и хотелось бы иметь недорогой метод оценки “усредненного” состава. Еще одной особенностью является отсутствие steady state из-за случайных отклонений протока, состава питательной среды, ошибок в измерениях оптической плотности, растворенного кислорода, подачи воздуха и т.д.. Т.е. алгоритмы автоматического контроля удерживают процесс в определенных рамках, но о состоянии равновесия говорить не приходится. Актуальное знание состава сообщества и позволит нам повысить стабильность культивирования и качество контроля

У нас есть данные измерений ряда параметров процесса культивирования, например оптической плотности и концентрации питательной среды. Нам необходимо регулярно уточнять параметры или даже структуру модели описывающей процесс.

Идея подхода с использованием технологии ODE-Net заключается в следующем. Функция f в правой части ODE необязательно должна быть нейросетью. Это может быть мат.модель процесса культивирования и в процессе тренировки такой ODE+Net мы будем уточнять параметры нашей модели. В общем случае если мы составим f в обобщенном варианте, включая n различных спутников взаимодействующих по m метаболитам с основной культурой и между собой, то в идеале после тренировки мы должны получить модель которую можно упростить убрав незначительные части.

Для начала я взял модель (1) в качестве точной модели процесса. Чтобы уйти от стационарности к протоку D = 0.25 добавил низкочастотный шум. Рассчитал точный процесс и отобрал точки со значениями x и s через каждый 15 мин на интервале 3-х месяцев. Тренировка модели выполняется на случайных выборках длительностью 12 часов (48 точек) из отобранных s и x.

(1)

$$\frac{dS}{dt} = \left(S_0 - S\right)D - \gamma\mu_X X$$

$$\frac{dX}{dt} = \left(\mu_X - D\right)X$$

$$\frac{dY}{dt} = \left(\mu_Y - D\right)Y$$

$$\frac{dA}{dt} = v_A X - DA - q_A Y$$

$$\frac{dB}{dt} = v_B X - DB - q_B Y$$

$$q_A = g_A \frac{1}{1+\frac{A}{K_A}+\frac{B}{K_B}} \frac{\mu_{Amax}A}{K_A}$$

$$q_B = g_B \frac{1}{1+\frac{A}{K_A}+\frac{B}{K_B}} \frac{\mu_{Bmax}B}{K_B}$$

$$\mu_Y = \frac{1}{1+\frac{A}{K_A}+\frac{B}{K_B}}\left(\frac{\mu_{Amax}A}{K_A} + \frac{\mu_{Bmax}B}{K_B}\right)$$

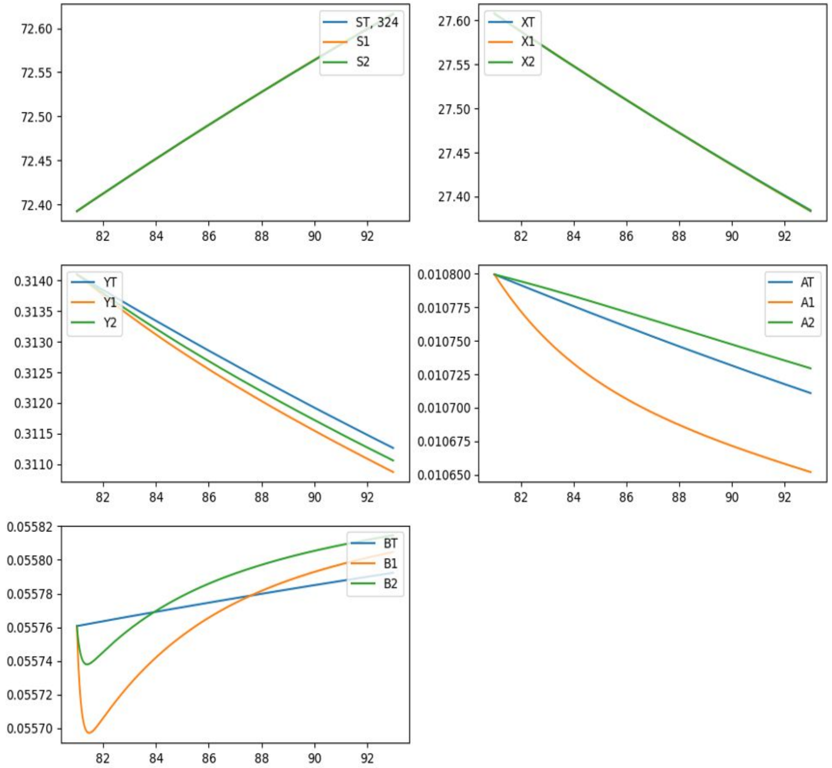
$$\mu_X = \mu_0 \frac{I_A}{I_A+A} \frac{I_B}{I_B+B} \frac{S}{K_s+S}$$

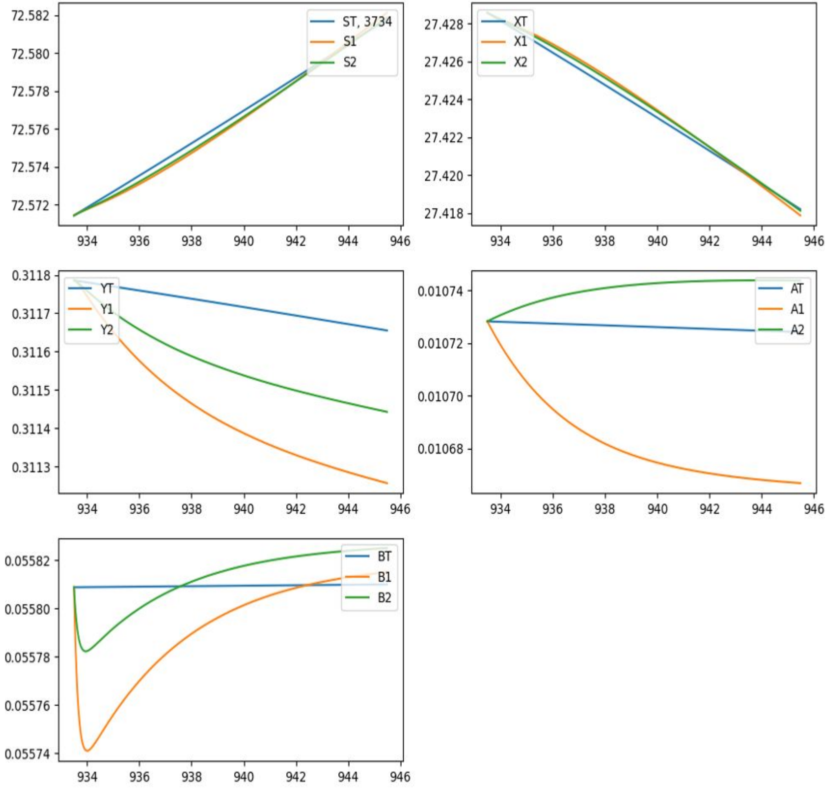
Параметры точного решения:

$S_0 = 100;\ \gamma = 1;\ v_{A'} = 0.0001;\ v_B = 0.009,\ g_A = 0.3;\ g_B = 3.0;\ \mu_0 = 0.3;\ \mu_{A\,max} = 0.1;\ \mu_{B\,max} = 0.7$
$K_A = 1.0;\ K_B = 0.1;\ I_A = 100.0;\ I_B = 1.0$

Эти значения соответствуют сообществу в котором основная культура X потребляет субстрат S и выделяет продукты метаболизма A и B. Оба метаболита ингибируют рост X. Метаболит B ингибирует рост X много больше чем A. Усредненный спутник Y “подъедает” A и B, но быстрее растет на B.

Ассимиляция данных культивирования





Ассимиляция данных культивирования

Пока лучший результат:

ga: 0.391015 gb: 5.693592 va: 0.000108 vb: 0.016499 mua: 0.213601 mub: 1.365904 Ka: 0.749377 Ia: 101.617818 Kb: 0.249022 Ib: 0.999999

Точные значения:

ga: 0.300000 gb: 3.000000 va: 0.000100 vb: 0.009000 mua: 0.100000 mub: 0.700000 Ka: 1.000000 Ia: 100.000000 Kb: 0.100000 Ib: 1.000000

Начальные значения:

ga: 1.0 gb: 5.0 va: 0.1 vb: 1.0 mua: 0.2 mub: 0.9 Ka: 1.0 Ia: 10.0 Kb: 0.5 Ib: 5.0

Основная проблема на данный момент это разница в несколько порядков градиентов различных параметров. Если параметр значимо влияет на результат его градиент много больше. Значение такого параметра сходится к точному, но градиент по прежнему остается огромным и осциллирует около нуля резко замедляя схождение остальных.