Обзор статьи «Neural Ordinary Differential Equations»

1 Введение

Для начала вспомним принцип работы нейросетей, какие они бывают и как их «обучать». Итак, имеется:

- $\mathcal{D} = X \times Y = \{x_i, y_i\}_{i=1}^n$ – набор данных (выборка).
- $\mathcal{L}(y_{pred}, y_{true})$ – функция потерь, зависящая от y_{pred} – предсказанного значения целевой переменной и истинного значения y_{true}.

Цель состоит в том, чтобы найти параметры θ нейросети $NN(x, \theta) : X \to Y$ такие, что они минимизируют общие потери на выборке:

$$\sum_{i=1}^{n} \mathcal{L}(NN(x_i, \theta), y_i) \to \min_{\theta}. \tag{1}$$

Принцип работы нейросетей состоит в том, чтобы итеративно преобразовывать входные данные из X и переводить их пространство Y. Можно представить нейросеть как конечное число последовательных блоков, каждый из которых получает данные на вход и выдает на выход преобразованные данные. Пусть имеется T штук таких «блоков», тогда эту концепцию можно представить следующими формулами:

1)$h_1 = g_0(x, \theta_0)$

2)$h_2 = g_1(h_1, \theta_1)$

...

T) $h_T = g_{T-1}(h_{T-1}, \theta_{T-1}) = y_{pred}$

Каждая функция преобразования $g_i(h_i, \theta_i)$ задает блок нейросети. Например, блок может быть полносвязным с какой-нибудь функцией активации. Это значит, что $g(h_i, \theta_i) = \phi(Wh_i + b)$, где $h_i \in \mathbb{R}^{k\times 1}$, $W \in \mathbb{R}^{m\times k}$, $b \in \mathbb{R}^{m\times 1}$, а активация ϕ, к примеру, это $ReLU$. Параметры блока в данном случае: $\theta_i = \{W, b\}$.

Таким образом, указанные шаги преобразования данных x задают сложную функцию $NN(x, \theta)$, где $\theta = \{\theta_0, \theta_1..., \theta_T\}$. Поиск локального минимума функционала 1 проводится с помощью градиентного спуска. Для этого необходимо уметь находить следующий градиент:

$$\nabla_\theta \mathcal{L}(NN(x_i, \theta), y_i). \tag{2}$$

Пусть для каждого x заранее известно истинное значение y, тогда будем обозначать $\mathcal{L}(\cdot\,, y)$ как $\mathcal{L}(\cdot)$. Итак, требуется найти

$$\nabla_\theta \mathcal{L}(NN(x, \theta)). \tag{3}$$

Запишем схему взятия градиентов для некоторых элементов θ.

$$\nabla_{\theta_{T-1}} \mathcal{L}(NN(x, \theta)) = \frac{\partial \mathcal{L}}{\partial h_T} \frac{\partial h_T}{\partial \theta_{T-1}} = \frac{\partial \mathcal{L}}{\partial h_T} \frac{\partial g_{T-1}(h_{T-1}, \theta_{T-1})}{\partial \theta_{T-1}} \tag{4}$$

$$\nabla_{\theta_i} \mathcal{L}(NN(x, \theta)) = \frac{\partial \mathcal{L}}{\partial h_T} \frac{\partial h_T}{\partial \theta_i} = \frac{\partial \mathcal{L}}{\partial h_T} \frac{\partial h_T}{\partial h_{T-1}} \frac{\partial h_{T-1}}{\partial \theta_i} = \frac{\partial \mathcal{L}}{\partial h_T} \frac{\partial g_{T-1}}{\partial h_{T-1}} \frac{\partial g_{T-2}}{\partial h_{T-2}} \cdot \ldots \cdot \frac{\partial g_i}{\partial \theta_i} \tag{5}$$

Такую схему еще называют обратным распространением ошибки (backpropagation).

2 Идея

А теперь рассмотрим частный случай блока, задаваемый следующим образом:

$$h_{t+1} = h_t + f(h_t, t, \theta) \tag{6}$$

Заметим, что в данное преобразование переводит данные h_t во множество той же размерности. На практике это может быть частью так называемой Residual или рекуррентной нейросети. Далее перепишем формулу в следующем виде:

$$\frac{h_{t+1} - h_t}{1} = f(h_t, t, \theta) \tag{7}$$

Данная запись есть не что иное как разностная схема Эйлера (aka метод Рунге-Кутта 1-ого порядка) для равномерной сетки на [t, t+1], состоящей из двух точек (концов отрезка) для некоторой задачи Коши. А задача Коши здесь ставится, например, следующим образом:

$$\begin{cases} h'(\widetilde{t}) = f(h, \widetilde{t}, \theta), \widetilde{t} \in [t, t+1] \\ h(t) = h_t \end{cases}$$

В связи с этим появляется следующая идея. Теперь будем иметь блок, который будет преобразовывать данные так: входные данные блока есть начальная точка траектории $h(\widetilde{t})$, задаваемой обыкновенным дифференциальным уравнением. Преобразованием входа будем считать реализацию траектории в точке $t+1$. Характер траектории в рамках функции f определяется настраиваемыми параметрами θ.

Можно представить, что блок состоит из континуума слоев, номера которых взяты из отрезка $[t, t+1]$. Все эти слои имеют общие веса θ. А непрерывное преобразование данных этими весами задает траекторию $h(\widetilde{t})$. К t можно относится как ко времени. Ранее блоки задавали дискретное временя, а введя ОДУ мы фактически задали непрерывный во времени процесс преобразования данных.

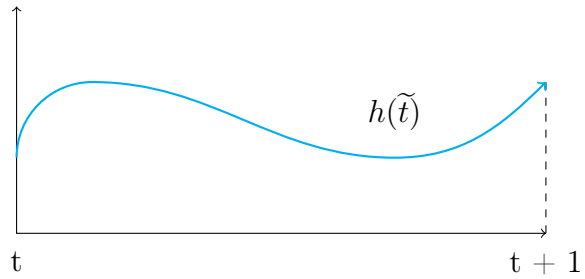

Рис. 1. Блок ОДУ.

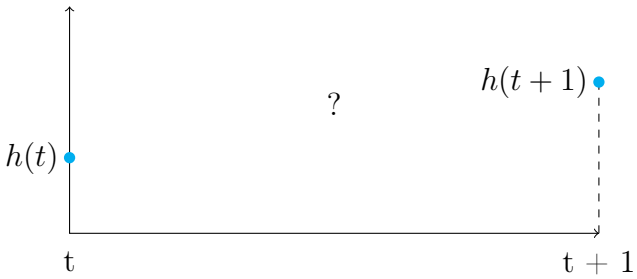

Рис. 2. Без ОДУ.

Далее возникают вопросы:

1) Как искать $h(t+1) = h_{t+1}$?

2) Как настраивать параметры θ?

3) Какие преимущества указанного расширения?

3 Как искать $h(t+1) = h_{t+1}$?

Задачу Коши предлагается решить любым из существующих методов решения ОДУ. Например, той же схемой Эйлера. Напомним, что метод состоит в построении равномерной сетки на $[t, t+1]$. Причём чем чаще узлы сетки, тем точнее итоговые точки траектории. Результат работы разностной схемы есть набор точек, близких к истинной траектории

$$\{h(t), h(t+\frac{1}{n}), ..., h(t+1)\}.$$

Подсчет точек происходит итеративно:

$$h(t+\frac{i+1}{n}) = h(t+\frac{i}{n}) + \frac{1}{n} f(h(t+\frac{i}{n}), t+\frac{i}{n}, \theta), i = \overline{0, n} \tag{8}$$

Значение $h(t+1)$ будем считать выходом блока. Поиск этого числа может быть осуществлен и с помощью другой разностной схемы (симметричной, Рунге-Кутта), но важны следующие пункты:

1) Методы подсчитывают выход блока с любой заранее указанной точностью.

2) Результат этих методов есть набор промежуточных точек траектории между моментами t и $t+1$.

4 Как настраивать параметры θ?

Рассмотрим случай, когда блок ОДУ находится последним в нейросети. Тогда её выходом является $\mathcal{L}(h_T) = \mathcal{L}(h(T))$. В этой части перейдем к нотации статьи:

- $z(t) = h(\widetilde{t})$
- $t_1 = T - 1, t_2 = T$
- $\mathcal{L}(z(t_1)) = \mathcal{L}(h(T))$
- $\theta = \theta_{T-1}$

Как было получено $z(t_1)$? Мы поставили задачу Коши:

$$\begin{cases} z'(t) = f(z, t, \theta), t \in [t_1, t_2] \\ z(t_0) = z_0 \end{cases} \quad (*)$$

Затем нашли $z(t_1)$ с помощью некоторого «решателя», который обозначим как $ODESolve$: $z(t_1) = ODESolve(z(t_0), f, t_0, t_1, \theta)$. Итак,

$$\mathcal{L}(z(t_1)) = \mathcal{L}(ODESolve(z(t_0), f, t_0, t_1, \theta)),$$

и теперь нужно найти градиент по θ от этого выражения. При этом результат не должен зависеть от конкретного вида «решателя».

В статье предлагается это сделать с помощью так называемого метода сопряженного состояния (ориг.: adjoint sensitivity method, википедия: adjoint state method). Пусть $a(t) = \frac{\partial \mathcal{L}}{\partial z}(t)$ – сопряженные переменные. Поиск $a(t)$ проведем решив задачу Коши:

$$\begin{cases} a'(t) = -a(t)^T \frac{\partial f(z(t), t, \theta)}{\partial z}, t \in [t_0, t_1] \\ a(t_1) = \frac{\partial \mathcal{L}}{\partial z}(t_1) \end{cases} \quad (**)$$

Наконец, искомое значение $\frac{d\mathcal{L}}{d\theta}$ выражается через $a(t)$:

$$\frac{d\mathcal{L}}{d\theta} = -\int_{t_1}^{t_0} a(t)^T \frac{\partial f(z(t), t, \theta)}{\partial \theta} dt = -\int_{t_1}^{t_0} \frac{\partial \mathcal{L}}{\partial z}(t)^T \frac{\partial f(z(t), t, \theta)}{\partial \theta} dt \quad (9)$$

Заметим, что выражение 9 очень похоже на процедуру backpropagation 4, только там определенный интеграл берётся от константы $\frac{\partial \mathcal{L}}{\partial h_T} \frac{\partial g_{T-1}(h_{T-1}, \theta_{T-1})}{\partial \theta_{T-1}}$ по t от $T-1$ до T.

Еще одно важное замечание заключается в том, что 9 можно тоже рассматривать как результат решения некоторой задачи Коши:

$$\begin{cases} (\frac{d\mathcal{L}}{d\theta})'_t = -a(t)^T \frac{\partial f(z(t),t,\theta)}{\partial \theta}, t \in [t_0, t_1] \\ \frac{\partial \mathcal{L}}{\partial \theta}(t_1) = 0 \end{cases} \quad (***)$$

Тогда

$$\frac{d\mathcal{L}}{d\theta}(t) = -\int_{t_1}^{t} a(t)^T \frac{\partial f(z(t),t,\theta)}{\partial \theta} dt$$

и

$$\frac{d\mathcal{L}}{d\theta} = \frac{d\mathcal{L}}{d\theta}(t_0).$$

В статье было предложено обоснование указанного способа поиска $\frac{d\mathcal{L}}{d\theta}$, однако оно кажется искусственным и не очень информативным (например, не указано, как была получена интегральная формула для $\frac{d\mathcal{L}}{d\theta}$). Альтернативное доказательство приведено в секции 6.1.

Итак, мы показали теоретически, как можно найти градиент по параметрам. Теперь покажем, как численно найти все необходимые величины.

Пусть мы уже посчитали прямой проход по блоку, то есть решателем ОДУ нашли точки траектории $\{z_0, z_{\frac{1}{n}}, ...z_{1-\frac{1}{n}}, z_1\}$ для равномерной сетки $T = \{0, \frac{1}{n}, \frac{2}{n}, ..., 1-\frac{1}{n}, 1\}$ (без потерии общности заменим t_0 на 0 и t_1 на 1 для удобства выкладок).

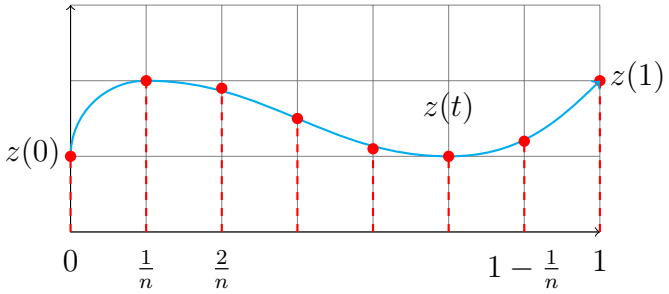

Рис. 3. Прямой проход.

Теперь нам нужно найти $\frac{d\mathcal{L}}{d\theta} = \frac{d\mathcal{L}}{d\theta}(t_0)$ для обновления весов θ и $\frac{\partial \mathcal{L}}{\partial z}(t_0) = a(t_0)$ для подсчета градиентов по весам ранних блоков. Для этого воспользуемся тем же решателем, что и для прямого прохода, и решим две задачи Коши $(**)$ и $(***)$. Для определенности будем пользоваться формулами пересчета для метода Эйлера[1] (8).

Будем запускать пересчет траектории не слева направо от t_0 до t_1 (как при прямом проходе), а справа налево от t_1 до t_0, то есть в обратную сторону. Сделаем первый шаг пересчета. Сначала для задачи (**):

$$a(1-\frac{1}{n}) = a(1) + \frac{1}{n} a(1)^T \frac{\partial f(z(1),1,\theta)}{\partial z}$$

[1]Можно воспользоваться и другими решателями для разностных схем.

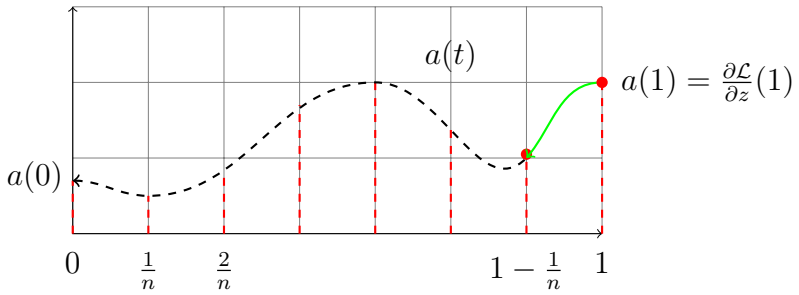

Рис. 4. Обратный проход для $a(t)$ (1-ый шаг).

Компоненты для осуществления первого шага уже известны заранее: $a(1) = \frac{\partial \mathcal{L}}{\partial z}(1)$ из начального условия для $(**)$; функциональный вид $\frac{\partial f}{\partial z}$ получен путем автоматического дифференцирования f по z, а $z(1)$ из уже просчитанной траектории прямым проходом.

Затем проведем шаг для $(***)$. Формула пересчёта для первого шага:

$$\frac{dL}{d\theta}(1 - \frac{1}{n}) = \frac{dL}{d\theta}(1) + \frac{1}{n}a(1)^T \frac{\partial f(z(1), 1, \theta)}{\partial \theta}$$
$$= \frac{1}{n}a(1)^T \frac{\partial f(z(1), 1, \theta)}{\partial \theta}$$

Слагаемое $\frac{dL}{d\theta}(1)$ заранее известно из начального условия и равно нулю, $a(1)$ – начальное условие $(**)$, функциональный вид $\frac{\partial f}{\partial \theta}$ опять же может быть найден с помощью методов библиотеки автоматического дифференцирования, а z(1) – результат прямого прохода.

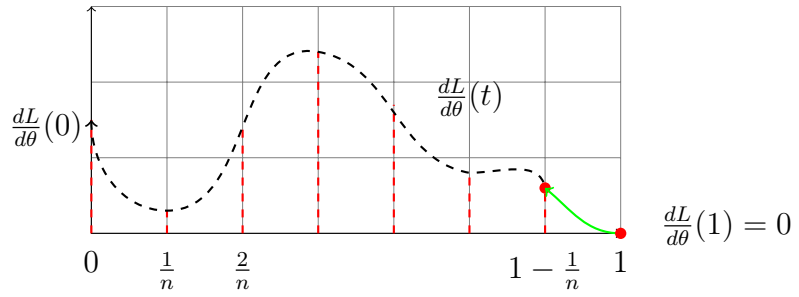

Рис. 5. Обратный проход для $\frac{dL}{d\theta}(t)$ (1-ый шаг).

Теперь рассмотрим i-ый шаг. Его формулы пересчета:

$$a(1 - \frac{i}{n}) = a(1 - \frac{i-1}{n}) + \frac{1}{n}a(1 - \frac{i-1}{n})^T \frac{\partial f(z(1 - \frac{i-1}{n}), 1 - \frac{i-1}{n}, \theta)}{\partial z}$$

$$\frac{dL}{d\theta}(1 - \frac{i}{n}) = \frac{dL}{d\theta}(1 - \frac{i-1}{n}) + \frac{1}{n}a(1 - \frac{i-1}{n})^T \frac{\partial f(z(\frac{i-1}{n}), \frac{i-1}{n}, \theta)}{\partial \theta}$$

Видно, что для пересчета шага нам всего лишь нужно знать значения $z(1-\frac{i-1}{n})$, $a(1-\frac{i-1}{n})$ и $\frac{dL}{d\theta}(1-\frac{i-1}{n})$ с предыдущего шага. И тогда возникает следующая мысль: нам не нужно хранить всю предыдущую траекторию. Для оптимального использования памяти будем хранить только последний шаг. А траекторию прямого прохода пересчитаем повторно, решив задачу Коши $(*)$, но с другим начальным условием:

$$\begin{cases} z'(t) = f(z,t,\theta), t \in [t_1, t_2] \\ z(t_1) = z_1 \end{cases} \quad (***)$$

Схему i-ого шага вычислений можно наблюдать на Рис. 6.

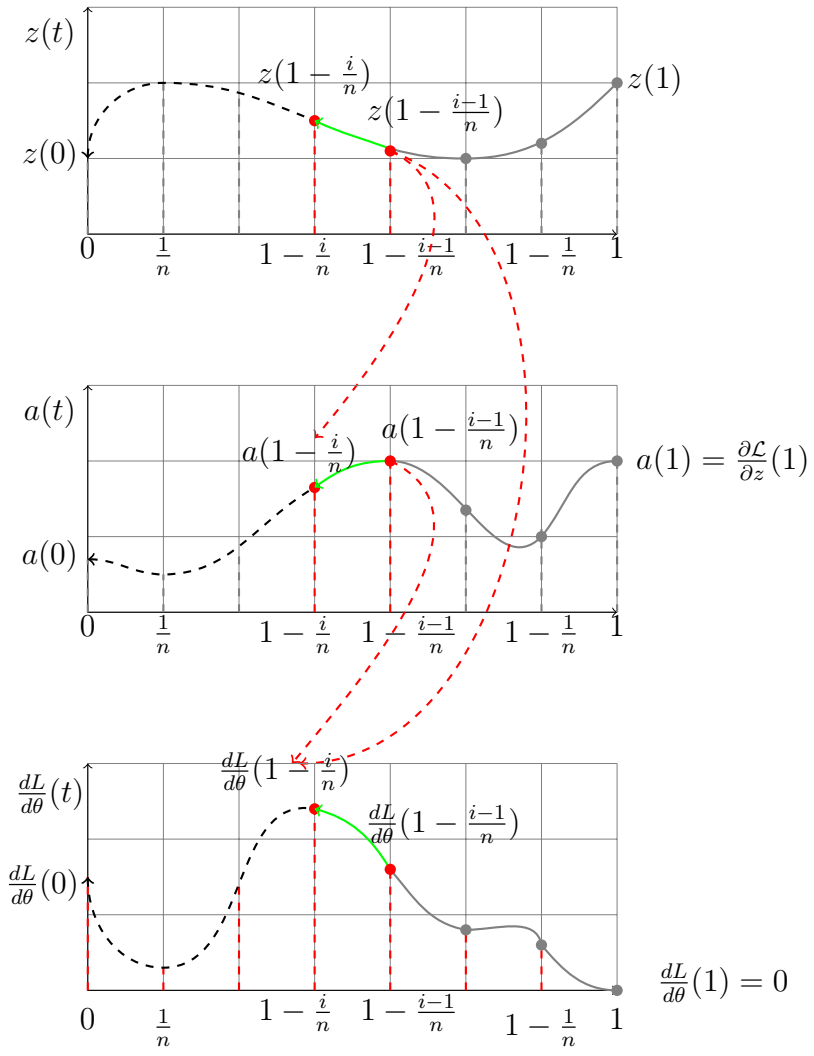

Рис. 6. Иллюстрация i-ого шага. Серым показаны точки, просчитанные ранее, но уже не хранящиеся в памяти. Красными точками обозначены моменты траектории, хранящиеся в памяти и ожидающие расчета. Пунктирные стрелки иллюстрируют зависимости следующих точек траектории и порядок вычисления траекотрий решателем.

Всю вышеизложенную процедуру авторы называют reverse-mode differentiation.

5 Какие преимущества указанного расширения?

Во-первых, таким образом мы более эффективно используем память. Рассмотрим случай: хотим классифицировать рукописные цифры из датасета MNIST. При этом хотим для этого обучить нейросеть с L подряд идущими residual блоками. Тогда на каждой итерации градиентного спуска нам нужно хранить L значений активаций h_t. А зачем вводить L блоков, если их можно заменить одним блоком ОДУ? Действительно, путем варьирования равномерной сетки мы получаем сколь угодно много residual блоков с общими весами. При этом количество вычислений растет с числом элементов сетки, а количество выделяемой памяти неизменно. Это следствие того, что нам не нужно хранить при прямом проходе всю траекторию h_t достаточно запустить решатель при обратном проходе с другим начальным условием.

Как при этом меняется качество классификации? Оказывается, что ошибка ниже именно на нейросети с блоком NeuralODE (Рис. 7)

Table 1: Performance on MNIST.

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

Рис. 7. L – число residual блоков в нейросети (авторы стати взяли 6 шт.), $\widetilde{L}$ – число resiual блоков с общими весами для NeuralODE. Интересно, что качество лучше даже при меньшем числе параметров.

Во-вторых, это способность к экстраполяции. Представим, что последний блок в сети – блок ОДУ, а точки траектории – это значения звукового сигнала в разные моменты времени. Иными словами, нейросеть выдает некий аудиосигнал. А что нам мешает взять и участить сетку, то есть повысить качество звукового сигнала? Или не останавливаться на $h(t_1)$, и далее просчитать точки траектории? Все это вписывается в концепцию Neural ODE.

6 Дополнение

6.1 Обоснование алгоритма поиска $\frac{d\mathcal{L}}{d\theta}$

Итак, идея альтернативного доказательства состоит в постановке исходной задачи как задачи оптимизации с ограничениями. Ограничения будут необычными:

$$\begin{cases} \mathcal{L}(h(t_1)) \to \min\limits_{\theta} \\ h' = f(h(t), t, \theta) \quad (****) \\ h(t_0) = h_0 \end{cases}$$

При этом

$$h(t_1) = h(t_0) + \int\limits_{t_0}^{t_1} f(h(t), t, \theta) dt.$$

Здесь $h' = f(h(t), t, \theta)$ и $h(t_0) = h_0$ будем считать ограничениями (траектория $h(t)$ удовлетворяет ограничениям, если является решением соответствующей задачи Коши).

Как решать задачу оптимизации с ограничениями? Нужно записать Лагранжиан:

$$L(\theta, \lambda, \beta) = \mathcal{L}(h(t_1)) + \int\limits_{t_0}^{t_1} \lambda(t)^T (h'(t) - f(h(t), t, \theta)) dt + \beta^T (h(t_0) - h_0)$$

Где $\lambda(t)$ и β – двойственные (сопряженные) переменные. Очевидно, что $\frac{dL}{d\theta} = \frac{d\mathcal{L}}{d\theta}$, если h удовлетворяет ограничениям (слагаемые с сопр. переменными просто обнуляются). В этом случае двойств. переменные можно выбрать какими угодно.

Давайте запишем и преобразуем $\frac{dL}{d\theta}$:

$$\frac{dL}{d\theta} = \frac{dL}{dh}(t_1) \int\limits_{t_0}^{t_1} (\frac{\partial f}{\partial \theta} + \frac{\partial f}{\partial h}\frac{\partial h}{\partial \theta}) dt + \int\limits_{t_0}^{t_1} \lambda(t)^T (\frac{\partial h'}{\partial \theta} - \frac{\partial f}{\partial h}\frac{\partial h}{\partial \theta} - \frac{\partial f}{\partial \theta}) dt$$

Далее возьмем следующий интеграл по частям чтобы избавиться от h':

$$\int\limits_{t_0}^{t_1} \lambda(t)^T \frac{\partial h'}{\partial \theta} dt = \lambda^T(t_1)\frac{\partial h}{\partial \theta}(t_1) - \int\limits_{t_0}^{t_1} \lambda'(t)^T \frac{\partial h}{\partial \theta} dt$$

и перепишем $\frac{dL}{d\theta}$:

$$\frac{dL}{d\theta} = \int\limits_{t_0}^{t_1} \frac{dL}{dh}(t_1)\frac{\partial f}{\partial \theta} dt + \int\limits_{t_0}^{t_1} (\frac{dL}{dh}(t_1)\frac{\partial f}{\partial h} - \lambda'(t)^T - \lambda^T(t)\frac{\partial f}{\partial h})\frac{\partial h}{\partial \theta} dt - \int\limits_{t_0}^{t_1} \lambda(t)^T \frac{\partial f}{\partial \theta} dt - \lambda^T(t_1)\frac{\partial h}{\partial \theta}(t_1)$$

Теперь воспользуемся тем, что мы можем выбирать сопр. переменные произвольными. Пусть $\lambda(t)$ удовлетворяет задаче Коши:

$$\begin{cases} \lambda'(t)^T = \frac{dL}{dh}(t_1)\frac{\partial f}{\partial h} - \lambda^T(t)\frac{\partial f}{\partial h} = (\frac{dL}{dh}(t_1) - \lambda^T(t))\frac{\partial f}{\partial h} \\ \lambda(t_1) = 0 \end{cases} \quad (*****)$$

Тогда второе слагаемое-интеграл в $\frac{dL}{d\theta}$ обнуляется. И, наконец, сделаем замену вида

$$-a^T(t) = \frac{dL}{dh}(t_1) - \lambda^T(t)$$

. В итоге задача Коши $(*****)$ преобразуется в уже известную задачу для сопр. состояния $(**)$:

$$\begin{cases} a'(t) = -a(t)^T \frac{\partial f(h(t),t,\theta)}{\partial h}, t \in [t_0, t_1] \\ a(t_1) = \frac{\partial \mathcal{L}}{\partial h}(t_1) \end{cases} \quad (**)$$

Далее нетрудно показать, что, подставив замену в $\frac{dL}{d\theta}$, получим итоговую известную формулу:

$$\frac{dL}{d\theta}(t_1) = \frac{d\mathcal{L}}{d\theta}(t_1) = -\int_{t_1}^{t_0} a(t)^T \frac{\partial f(h(t),t,\theta)}{\partial \theta} dt$$

6.2 Реализация и эксперименты

В статье на сайте Хабр содержится авторский код с пояснениями для актуальной статьи. Автор сильно упростил оригинальный код с использованием библиотеки PyTorch для автоматического дифференцирования. Далее упрощать его сложно, так что читателю предлагается ознакомится с ним по ссылке. Также в этой статье были повторены некоторые эксперименты из оригинальной статьи.