# Построение нейродифференциальных уравнений с применением методов обратных задач динамики

Шорохов С.Г.

The 9th International Conference on Deep Learning in Computational Physics

SINP MSU, Moscow, Russia
4 июля, 2025

RUDN university

# Нейродифференциальные уравнения

Рассмотрим систему обыкновенных дифференциальных уравнений (ОДУ)

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}\left(\mathbf{x}, t; \boldsymbol{\theta}\right), \mathbf{x} = \left(x_1, ..., x_n\right) \in \mathbb{R}^n, t \in \left[t_0, t_N\right] \tag{1}$$

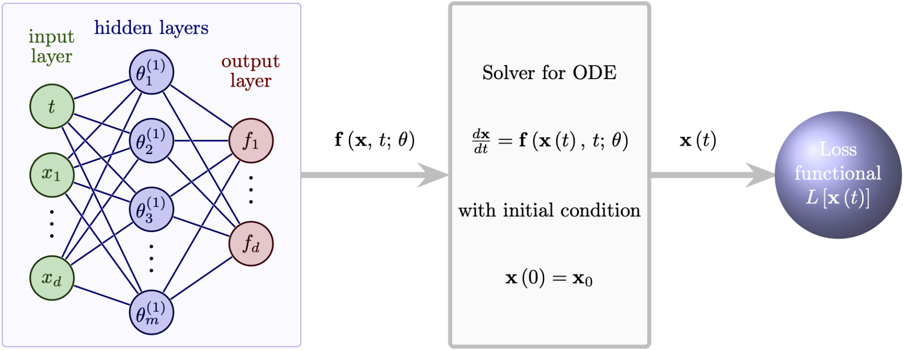
с начальным условием

$$\mathbf{x}\left(t_0\right) = \mathbf{x}_0 = \left(x_1^{(0)}, ..., x_n^{(0)}\right) \tag{2}$$

с правой частью, задаваемой нейронной сетью с параметрами (весами) $\boldsymbol{\theta}$, входом $(\mathbf{x}, t) \in \mathbb{R}^{n+1}$ и выходом $\mathbf{f}\left(\mathbf{x}, t; \boldsymbol{\theta}\right) \in \mathbb{R}^n$.

Результат (траектория) нейродифференциальных уравнений (нейроДУ) $\mathbf{x}\left(t\right)$ в момент времени $t$ вычисляется как приближенное решение системы (1) с начальным условием (2) алгоритмом-решателем дифференциальных уравнений, который использует правые части (выход нейросети) $\mathbf{f}$ в необходимых точках для определения решения $\mathbf{x}\left(t\right)$ с требуемой точностью.

# Обучение нейродифференциальных уравнений




Модель нейроДУ включает 1) нейронную сеть с выходом $\mathbf{f}$, 2) решатель дифференциальных уравнений, возвращающий $\mathbf{x}\,(t)$, и 3) функционал потерь $\mathcal{L}\,[\mathbf{x}\,(t)]$. Для обучения нейронной сети, входящей в модель, нужно уметь вычислять градиент функционала потерь по параметрам $\boldsymbol{\theta}$ нейронной сети с выходом $\mathbf{f}\,(\mathbf{x}, t; \boldsymbol{\theta})$.

Поэтому возникает задача выполнения обратного дифференцирования (обратного распространения ошибки) через решатель (solver) системы ОДУ.

# Идея нейродифференциальных уравнений

Основные идеи нейронных дифференциальных уравнений были предложены исследователями из Университета Торонто Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt и David Duvenaud в работе «Neural Ordinary Differential Equations» на 32й конференции Neural Information Processing Systems (NeurIPS) в 2018 году. Авторы подчеркнули, что идеи для нейронных ОДУ они получили из работ Л.С. Понтрягина по созданию принципа максимума.

Метод нейронных дифференциальных уравнений использует идею Л.С. Понтрягина о вычислении градиентов решения системы ОДУ по параметрам при помощи дополненной систему ОДУ, которая решается назад во времени (в обратном времени).

# Градиент функции потерь нейроДУ

Пусть $\mathbf{x}_0 \in \mathbb{R}^d$, $\boldsymbol{\theta} \in \mathbb{R}^p$ и функция $\mathbf{f} : \mathbb{R}^d \times [t_0, t_1] \times \mathbb{R}^p \to \mathbb{R}^d$ непрерывна по $t$ и класса $\mathcal{C}^1$ по $\mathbf{x}$ и $\boldsymbol{\theta}$. Обозначим через $\boldsymbol{\phi}\left(t; \mathbf{x}_0, t_0, \boldsymbol{\theta}\right) : \mathbb{R} \times \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^p \to \mathbb{R}^d$ решение задачи с начальными условиями (IVP)

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, t, \boldsymbol{\theta}\right), \mathbf{x}\left(t_0\right) = \mathbf{x}_0. \tag{3}$$

Пусть $\mathbf{x}_1 = \boldsymbol{\phi}\left(t_1; \mathbf{x}_0, t_0, \boldsymbol{\theta}\right)$. Для дифференцируемой скалярной функции $L : \mathbb{R}^d \to \mathbb{R}$ определим функцию потерь $\mathcal{L}\left(t_1, \mathbf{x}_0, t_0, \boldsymbol{\theta}\right) = L\left(\boldsymbol{\phi}\left(t_1; \mathbf{x}_0, t_0, \boldsymbol{\theta}\right)\right)$. При фиксированных значениях $t_1 \in \mathbb{R}$, $\mathbf{x}_1 \in \mathbb{R}^d$, $\boldsymbol{\theta} \in \mathbb{R}^p$ определим векторные функции $\mathbf{a}_{\mathbf{x}} : [t_0, t_1] \to \mathbb{R}^d$ и $\mathbf{a}_{\boldsymbol{\theta}} : [t_0, t_1] \to \mathbb{R}^p$ как решения следующей системы ОДУ в обратном времени с конечными (терминальными) условиями $(t < t_1)$:

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, t, \boldsymbol{\theta}\right), & \mathbf{x}\left(t_1\right) = \mathbf{x}_1, \\ \dot{\mathbf{a}}_{\mathbf{x}}^T = -\mathbf{a}_{\mathbf{x}}^T \partial_{\mathbf{x}} \mathbf{f}\left(\mathbf{x}, t, \boldsymbol{\theta}\right), & \mathbf{a}_{\mathbf{x}}\left(t_1\right) = \dfrac{\partial L}{\partial \mathbf{x}}\left(\mathbf{x}_1\right), \\ \dot{\mathbf{a}}_{\boldsymbol{\theta}}^T = -\mathbf{a}_{\mathbf{x}}^T \partial_{\boldsymbol{\theta}} \mathbf{f}\left(\mathbf{x}, t, \boldsymbol{\theta}\right), & \mathbf{a}_{\boldsymbol{\theta}}\left(t_1\right) = \mathbf{0}, \end{cases}$$

тогда градиент функции потерь равен

$$\partial_{\boldsymbol{\theta}} \mathcal{L}\left(t_1, \mathbf{x}_0, t_0, \boldsymbol{\theta}\right) = \mathbf{a}_{\boldsymbol{\theta}}\left(t_0\right)^T. \tag{4}$$

# Базовая задача для нейроДУ

Рассматривается динамическая система, которая подчиняется системе ОДУ вида

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, t\right), \mathbf{x}\left(t_0\right) = \mathbf{x}_0$$

с неизвестной правой частью $\mathbf{f}\left(\mathbf{x}, t\right)$. Пусть в моменты времени $t_k, k = \overline{1, N}$, такие, что $t_0 < t_1 < ... < t_k < ... < t_N$, осуществляются, вообще говоря, зашумленные наблюдения положения динамической системы $\widetilde{\mathbf{x}}_k = \mathbf{x}\left(t_k\right)$.

Требуется найти аппроксимацию $\mathbf{f}\left(\mathbf{x}, t; \boldsymbol{\theta}\right)$ правых частей системы ОДУ $\mathbf{f}\left(\mathbf{x}, t\right)$, имея набор наблюдений $\left\{\left(\widetilde{\mathbf{x}}_1, t_1\right), ..., \left(\widetilde{\mathbf{x}}_N, t_N\right)\right\}$.

Допустим, что функционал потерь $\mathcal{L}$ представляет собой сумму оценок отклонений траектории нейродифференциальных уравнений $\mathbf{x}\left(t; \boldsymbol{\theta}\right)$ от наблюдений $\widetilde{\mathbf{x}}_k$ в моменты времени $t_k$, определяемых скалярной функцией вида $L\left(\mathbf{x}\left(t_k; \boldsymbol{\theta}\right), \widetilde{\mathbf{x}}_k\right)$:

$$\mathcal{L}\left[\mathbf{x}\left(t; \boldsymbol{\theta}\right)\right] = \sum_{k=1}^{N} L\left(\mathbf{x}\left(t_k; \boldsymbol{\theta}\right), \widetilde{\mathbf{x}}_k\right)$$

# Общая схема обучения нейроДУ

### Набор данных

Набор из $N$ точек $\{(\widetilde{\mathbf{x}}_k, t_k)\}_{k=1}^N$

Наблюдаемые точки: $\{\widetilde{\mathbf{x}}_k\}_{k=1}^N$

### Прямой проход

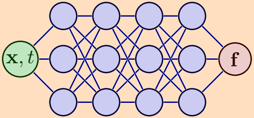
Системы ОДУ в прямом времени $(k = 1, ..., N)$

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, t, \boldsymbol{\theta}\right), \mathbf{x}\left(t_{k-1}\right) = \mathbf{x}_{k-1},$$

где $t \in [t_{k-1}, t_k], \mathbf{x}_k = \mathbf{x}\left(t_k; \boldsymbol{\theta}\right)$

Фактические точки: $\{\mathbf{x}_k = \mathbf{x}\left(t_k; \boldsymbol{\theta}\right)\}_{k=1}^N$

### Нейронная сеть




$\mathbf{f}$

$\mathbf{f}, \partial_{\mathbf{x}}\mathbf{f}, \partial_{\boldsymbol{\theta}}\mathbf{f}$

### Функционал потерь $\mathcal{L}$

$$\mathcal{L}\left[\mathbf{x}\left(t; \boldsymbol{\theta}\right)\right] = \sum_{k=1}^N L\left(\mathbf{x}\left(t_k; \boldsymbol{\theta}\right), \widetilde{\mathbf{x}}_k\right)$$

Фактические точки: $\{\mathbf{x}_k = \mathbf{x}\left(t_k; \boldsymbol{\theta}\right)\}_{k=1}^N$

$\partial_{\boldsymbol{\theta}}\mathcal{L}$ для обновления параметров $\boldsymbol{\theta}$

$\partial_{\boldsymbol{\theta}}\mathcal{L}$

### Обратный проход

Системы ОДУ в обратном времени (k=N,...,1)

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, t, \boldsymbol{\theta}\right), & \mathbf{x}\left(t_k\right) = \mathbf{x}_k, \\ \dot{\mathbf{a}}_{\mathbf{x}}^T = -\mathbf{a}_{\mathbf{x}}^T \partial_{\mathbf{x}}\mathbf{f}\left(\mathbf{x}, t, \boldsymbol{\theta}\right), & \mathbf{a}_{\mathbf{x}}\left(t_k\right) = \dfrac{\partial L}{\partial \mathbf{x}}\left(\mathbf{x}_k\right), \\ \dot{\mathbf{a}}_{\boldsymbol{\theta}}^T = -\mathbf{a}_{\mathbf{x}}^T \partial_{\boldsymbol{\theta}}\mathbf{f}\left(\mathbf{x}, t, \boldsymbol{\theta}\right), & \mathbf{a}_{\boldsymbol{\theta}}\left(t_k\right) = \mathbf{0} \end{cases}$$

где $t \in [t_{k-1}, t_k], k = N, ..., 1$

$\{\mathbf{a}_{\boldsymbol{\theta}}^T\left(t_{k-1}\right)\}_{k=1}^N$

### Градиент $\partial_{\boldsymbol{\theta}}\mathcal{L}$

$$\partial_{\boldsymbol{\theta}}\mathcal{L} = \sum_{k=1}^N \mathbf{a}_{\boldsymbol{\theta}}^T\left(t_{k-1}\right)$$

# Обратные задачи динамики

«Обратными задачами динамики называются задачи определения активных сил и моментов, приложенных к механической системе, параметров системы и дополнительно наложенных на нее связей, при которых движение с заданными свойствами является одним из возможных движений рассматриваемой механической системы» (А.С. Галиуллин).


- Галиуллин А.С., Мухаметзянов И.А., Мухарлямов Р.Г., Фурасов В.Д. Построение систем программного движения. М.: Наука, 1971
- Галиуллин А.С. Обратные задачи динамики. М.: Наука, 1981.
- Галиуллин А.С. Методы решения обратных задач динамики. М., Наука, 1986


Основным математическим инструментом решения обратных задач динамики является метод построения систем дифференциальных уравнений по заданному интегральному многообразию, предложенный Н.П. Еругиным в 1952 г.

# Метод построения ОДУ по заданной кривой

Н.П. Еругин рассмотрел задачу определения множества правых частей системы обыкновенных дифференциальных уравнений

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}\right), \mathbf{x} = \left(x_1, x_2\right), \mathbf{f} = \left(f_1, f_2\right), \tag{5}$$

имеющей заданную интегральную кривую, определенную в неявном виде соотношением

$$\omega\left(\mathbf{x}\right) = 0, \left\|\partial_{\mathbf{x}}\omega\right\|^2 = \left(\partial_{x_1}\omega\right)^2 + \left(\partial_{x_2}\omega\right)^2 \neq 0, \tag{6}$$

и показал, что любая система дифференциальных уравнений вида (5), допускающая заданную интегральную кривую (6), представляется в виде

$$\mathbf{f}\left(\mathbf{x}\right) = \Phi\left(\mathbf{x}\right)\mathbf{n}\left(\mathbf{x}\right) + \Psi\left(\mathbf{x}\right)\mathbf{s}\left(\mathbf{x}\right), \tag{7}$$

где $\mathbf{n} = \frac{1}{\|\partial_{\mathbf{x}}\omega\|}\begin{pmatrix}\partial_{x_1}\omega \\ \partial_{x_2}\omega\end{pmatrix}$ – единичный вектор нормали, $\mathbf{s} = \frac{1}{\|\partial_{\mathbf{x}}\omega\|}\begin{pmatrix}\partial_{x_2}\omega \\ -\partial_{x_1}\omega\end{pmatrix}$ – единичный вектор касательной к кривой $\omega\left(\mathbf{x}\right) = 0$, $\Phi\left(\mathbf{x}\right), \Psi\left(\mathbf{x}\right)$ – произвольные скалярные функции, причем

$$\Phi\left(\mathbf{x}\right)\Big|_{\omega(\mathbf{x})=0} \equiv 0. \tag{8}$$

# Сравнение нейроДУ и обратных задач динамики

## Обратные задачи динамики

Траектория задана аналитически в неявном виде

$$\omega(\mathbf{x}) = 0$$

Решение представляется в аналитическом виде

$$\mathbf{f}(\mathbf{x}) = \Phi(\mathbf{x})\,\mathbf{n}(\mathbf{x}) + \Psi(\mathbf{x})\,\mathbf{s}(\mathbf{x})$$

Обеспечивают устойчивость траектории, не содержат средств для определения касательной компоненты к траектории

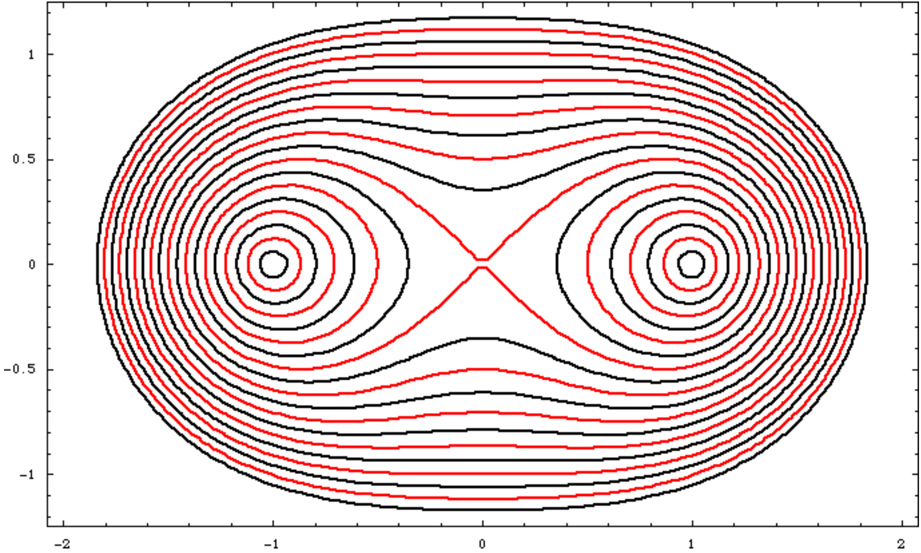
## НейроДУ

Задан набор точек траектории вида $\{(\widetilde{\mathbf{x}}_1, t_1), ..., (\widetilde{\mathbf{x}}_N, t_N)\}$

Решение представляется в виде выхода $\mathbf{f}(\mathbf{x}, t; \boldsymbol{\theta})$ обученной нейронной сети

Вообще говоря, не обеспечивают устойчивости движения

# Овалы Кассини



# Система ОДУ для овала Кассини

Для овала Кассини с параметрами $a$ и $c$ имеем

$$\omega\left(\mathbf{x}\right)=\left(x_1^2+x_2^2\right)^2-2c^2\left(x_1^2-x_2^2\right)+c^4-a^4=0$$

Выберем $\Phi=-\alpha\omega\left(\mathbf{x}\right)$, $\Psi=\beta$, $\alpha,\beta\in\mathbb{R},\alpha>0$, тогда правые части системы ОДУ, допускающей движение по овалу Кассини, равны

$$\begin{aligned}
\mathbf{f}\left(x_1,x_2\right)&=\frac{-\alpha\omega\left(\mathbf{x}\right)}{\|\partial_{\mathbf{x}}\omega\|}\begin{bmatrix}\frac{\partial\omega}{\partial x_1}\\ \frac{\partial\omega}{\partial x_2}\end{bmatrix}+\frac{\beta}{\|\partial_{\mathbf{x}}\omega\|}\begin{bmatrix}\frac{\partial\omega}{\partial x_2}\\ -\frac{\partial\omega}{\partial x_1}\end{bmatrix}=\\
&=\frac{4}{\|\partial_{\mathbf{x}}\omega\|}\begin{bmatrix}-\alpha\omega\left(x_1,x_2\right)x_1\left(x_1^2+x_2^2-c^2\right)+\beta x_2\left(x_1^2+x_2^2+c^2\right)\\ -\alpha\omega\left(x_1,x_2\right)x_2\left(x_1^2+x_2^2+c^2\right)-\beta x_1\left(x_1^2+x_2^2-c^2\right)\end{bmatrix},\\
\omega\left(x_1,x_2\right)&=\left(x_1^2+x_2^2\right)^2-2c^2\left(x_1^2-x_2^2\right)+c^4-a^4,\\
\|\partial_{\mathbf{x}}\omega\|&=4\sqrt{x_1^2\left(x_1^2+x_2^2-c^2\right)^2+x_2^2\left(x_1^2+x_2^2+c^2\right)^2}.
\end{aligned}$$

# Параметры вычислительного эксперимента

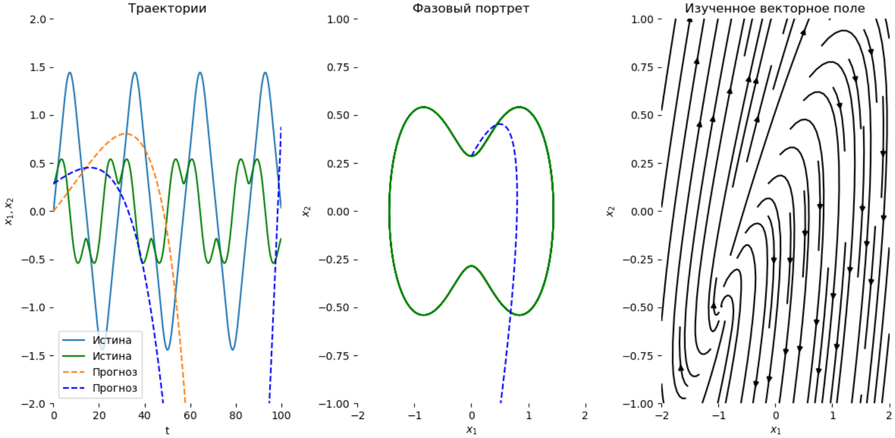
Вычислительный эксперимент выполняется для архитектуры, указанной ниже, со следующими гиперпараметрами:

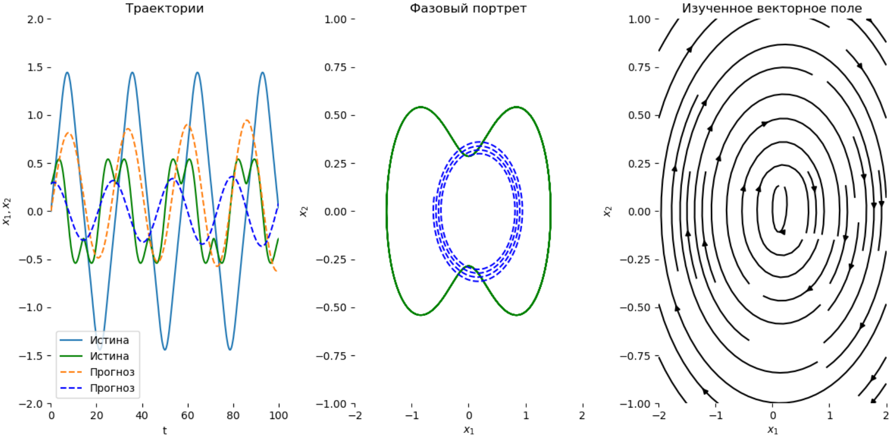
- нейронная сеть прямого распространения с плотными слоями
- один скрытый слой с функцией активации tanh
- число нейронов в скрытом слое равно 50
- два нейрона без активации в выходном слое
- выполняется 2000 эпох обучения
- используется оптимизатор Adam с начальной скоростью обучения
- пакет состоит из 100 точек, которые выбираются случайным образом из 5000 точек на кривой
- $a = 1.04, c = 1., \alpha = 0.1, \beta = 0.25, \mathbf{x}_0 = (0, 0.2856)$

Программный код реализован при помощи фреймворка PyTorch и библиотеки torchdiffeq.

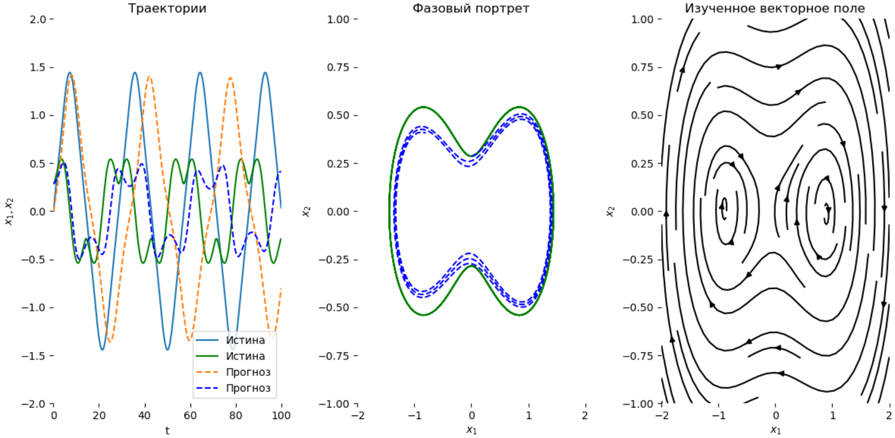
# Кривая обучения нейроДУ

# Промежуточные результаты обучения (эпоха 17)



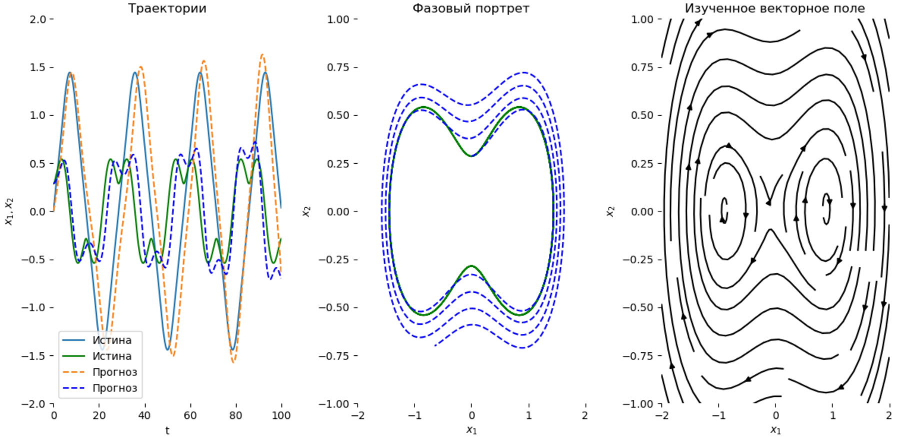


Траектории
$x_1, x_2$
t
Истина
Истина
Прогноз
Прогноз
Фазовый портрет
$x_2$
$x_1$
Изученное векторное поле
$x_2$
$x_1$



Траектории
$x_1, x_2$
t
Истина
Истина
Прогноз
Прогноз
Фазовый портрет
$x_1$
$x_2$
Изученное векторное поле
$x_1$
$x_2$

# Финальные результаты обучения (эпоха 1724)



# Заключение

- Обратные задачи динамики могут строить системы ОДУ для заданных плоских, пространственных и многомерных кривых самой различной геометрии и поставлять для обучения нейродифференциальных уравнений соответствующие наборы данных
- Нейродифференциальные уравнения смогли обучиться на точках нелинейной траектории, полученной при помощи решения обратной задачи динамики для овала Кассини
- Планируется провести исследования по построению и вычислительные эксперименты по обучению нейродифференциальных уравнений со структурой правых частей, соответствующей решению обратных задач динамики

# Вопросы?