МИНИСТЕРСТВО НАУКИ И ВЫСШЕГООБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования   
**«Национальный исследовательский   
Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

**Кафедра: Теории управления и динамики систем**

Направление подготовки: «Фундаментальная информатика и информационные технологии»

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА**

на тему:

**«Методы машинного обучения в задачах предсказания и управления динамикой сложных систем»**

**Выполнил(а):** студент группы 382006-2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Смирнов Н.В.

Подпись

**Научный руководитель:**

доцент,

кандидат физико-математических наук

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Смирнов Л.А.

Подпись

Нижний Новгород  
2024

Оглавление

[Введение 3](#_Toc168799443)

[I. Теоретическая часть 3](#_Toc168799444)

[1.1. Архитектура ESN (Echo State Network) 3](#_Toc168799445)

[1.2. Математическое описание 4](#_Toc168799446)

[1.3. Параметры резервуара 7](#_Toc168799447)

[1.4. Аттрактор Лоренца 8](#_Toc168799448)

[II. Практическая часть 9](#_Toc168799449)

[2.1. Программная реализация 9](#_Toc168799450)

[2.2. Механизм обучения 10](#_Toc168799451)

[2.3. Использование сети 12](#_Toc168799452)

[2.4. Исследовательская задача 16](#_Toc168799453)

[Заключение 17](#_Toc168799454)

[Литература 18](#_Toc168799455)

# Введение

В данной работе исследуется применение нейронной сети архитектуры Echo State Networks (ESN) в моделировании динамических систем. Задача анализа и прогнозирования различных физических и биологических систем, которые генерируют значения во времени, на сегодняшний день весьма актуальна. Одним из ключевых аспектов исследования является задача прогнозирования, где нейронные сети типа ESN демонстрируют способность выявлять и обобщать зависимости между входными и выходными данными, что позволяет им предсказывать будущие значения временных рядов на основе предыдущих данных, что имеет широкий спектр применений в различных областях.

Сеть Эхо-состояния (ESN) — это разновидность рекуррентных нейронных сетей (RNN), характеризующаяся большим набором рекуррентно связанных нейронов (резервуаром). В отличие от традиционных RNN, в ESN имеют место фиксированные случайные соединения нейронов в резервуаре, что делает сеть более легкой в обучении.

В рамках данной работы планируется изучить архитектуру ESN, попробовать её в анализе и прогнозировании поведения сложных систем, оптимизировать, проверить некоторые гипотезы, связанные с оптимизацией её работы.

# I. Теоретическая часть

## 1.1. Архитектура ESN (Echo State Network)

ESN состоит из трех основных компонентов:

Резервуар:

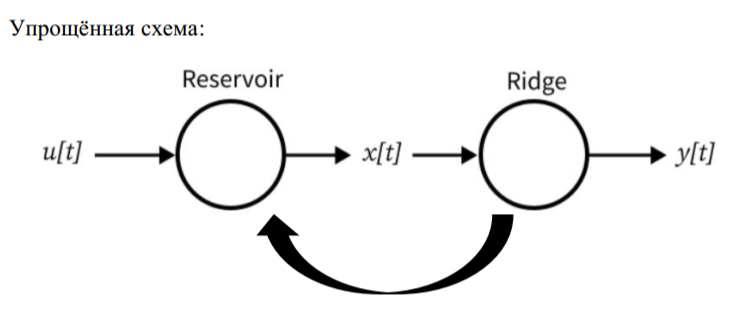
Резервуар является сложной динамической системой, которая преобразует входной сигнал в пространство более высокой размерности. Он состоит из множества нейронов, каждый нейрон резервуара имеет своё внутреннее состояние , – временной шаг, которое обновляется на основе входных данных и своего предыдущего внутреннего состояния. Связи между нейронами резервуара задаются случайным образом и остаются неизменными в процессе обучения сети. Получая входные данные, резервуар улавливает динамику изменения переменной во времени и генерирует модели поведения, описывающие её.

Входной линейный слой:

Он принимает входные данные и передает в резервуар. Входные данные являются временными рядами, которые могут быть одномерными или многомерными.

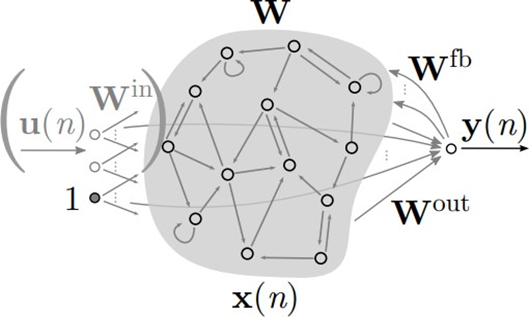
Выходной линейный слой:

Выходной слой получает состояния резервуара и применяет линейное преобразование к этим состояниям. Веса выходного слоя задаются в процессе обучения с использованием гребневой регрессии. Выходной слой генерирует предсказания на основе полученных состояний резервуара.



*Рисунок 1. Упрощённая схема ESN*

## 1.2. Математическое описание



*Рисунок 2. Схема работы ESN*

**1) Уравнение обновления состояний резервуара:**

, (1)

– временной шаг

– активации нейронов резервуара

– скорость утечки

, (2)

[.;.] – означает вертикальную конкатенацию векторов

– входные данные

– выходные данные

– матрица обратных связей

– матрица внутренних связей между нейронами резервуара

**2) Уравнение выходных значений (предсказания) имеет вид:**

,(3)

– матрица скрытых состояний резервуара

Размерности:

– количество временных точек в целевом временно ряду

– ширина резервуара (количество строк матрицы )

**3) Обучение выходного слоя . Регрессия:**

Входной временной ряд (входные данные)

Матрица скрытых состояний резервуара

Вектор выходного слоя

Их размерности:

– количество временных точек в целевом временно ряду

– размерность системы (у нас =1)

– ширина резервуара (количество строк)

Введём квадратичную функцию потерь:

, (4)

Наша задача – минимизировать функцию потерь по аргументу .

= , (5)

Гребневая регрессия (ridge-регрессия):

+ , (6)

= , (7)

– параметр регуляризации , при его увеличении оценки коэффициентов стремятся к нулю.

Таким образом, гребневая регрессия стремится найти модель, которая не только хорошо описывает данные (минимизирует RSS), но и имеет коэффициенты небольшой величины (регуляризация). Это позволяет избежать переобучения и получить более устойчивые и обобщающие модели.

– диагональная единичная матрица.

## 1.3. Параметры резервуара

1) Units – Размер резервуара

Количество нейронов в резервуаре.

2) Leak rate – Скорость утечки α

Скорость утечки α определяет долю предыдущего состояния , которая сохранится в новом состоянии .

Если скорость утечки слишком маленькая, то узлы резервуара будут сохранять информацию о предыдущих данных слишком длительное время, что может привести к переобучению, если скорость утечки большая, то резервуар не сможет сохранять нужное количество информации о прошлых данных.

3) Spectral radius – Спектральный радиус

Спектральный радиус матрицы связей резервуара ρ(𝑊) – это наибольшее по модулю собственное значение.

Если ρ(𝑊) слишком большой, возникает вероятность переобучения, если он слишком маленький, то сигнал, проходящий через резервуар, будет затухать слишком быстро.

4) Input scaling – Масштабирование входного слоя

5) Сonnectivity – Разреженность матрицы 𝑊

Большая разреженность ускоряет вычисления, но снижает точность решения задачи. Более плотные связи повышают точность, но происходит значительное замедление вычислений.

6) Input connectivity – Разреженность входного слоя

7) Regularization – Регуляризация

Это параметр помогает уменьшить чрезмерное влияние отдельных весов на результаты.

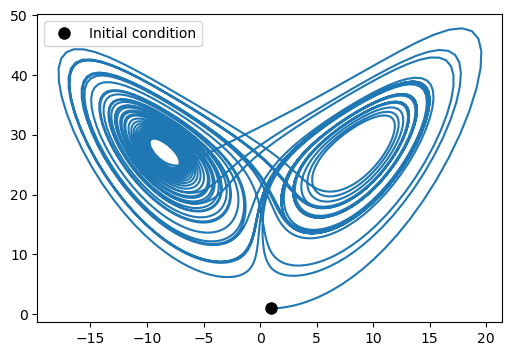
Таковы самые основные параметры резервуара.

## 1.4. Аттрактор Лоренца

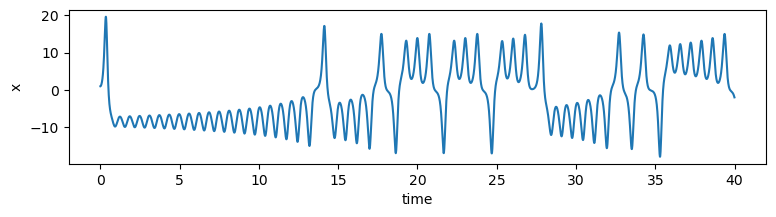
Для проведения экспериментов и оптимизации нейронной сети используются временные ряды, порождённые системой Лоренца:

= 10.0, = 8/3, = 28.0

Система имеет несколько вариантов поведения в зависимости от значений параметров: устойчивая точка, периодический режим, хаотический режим.



*Рисунок 3. График зависимости аргумента x от аргумента z*



*Рисунок 4. График зависимости аргумента x от t*

Система Лоренца была выбрана потому, что она является классическим примером нелинейной динамической системы, которая может демонстрировать сложное поведение и хаотическую динамику.

# II. Практическая часть

## 2.1. Программная реализация

В данной работе используется модуль ReservoirPy языка Python. Python является популярным и широко используемым языком программирования в области машинного обучения и научных вычислений. Он отличается простотой синтаксиса, большим количеством библиотек и удобством использования. ReservoirPy – это библиотека для создания сетей резервуарной архитектуры. В ReservoirPy все данные хранятся в массивах модуля Numpy. ReservoirPy использует только модули Numpy и Scipy для всех вычислений.

Вот пример программы, где создаются параметры резервуара, резервуар и выходной слой, затем они соединяются в сеть ESN:

*units = 100*

*leak\_rate = 0.3*

*spectral\_radius = 1.25*

*input\_scaling = 1.0*

*connectivity = 0.1*

*input\_connectivity = 0.2*

*regularization = 1e-8*

*seed = 1234*

*reservoir = Reservoir(units, input\_scaling=input\_scaling, sr=spectral\_radius,*

*lr=leak\_rate, rc\_connectivity=connectivity,*

*input\_connectivity=input\_connectivity, seed=seed)*

*readout = Ridge(1, ridge=regularization)*

*esn = reservoir >> readout*

В дальнейшем эта модель будет использоваться для предсказания временных рядов, генерируемых различными динамическими системами. В данной работе рассматривается система Лоренца.

## 2.2. Механизм обучения

Получим массив значений, порождённых системой Лоренца:

*length = 6500*

*train\_train = 4960*

*forecast = 10*

*dt = 0.01*

*x0 = [0.62225717, -0.08232857, 30.60845379]*

*x = data.lorenz(length=length, sample=dt, x0=x0,*

*sigma=16.0, beta=4.0, rho=45.92)[1]*

Мы получаем массив на *length* значений переменной системы Лоренца на *length* временных шагах с заданными начальными условиями, под временным шагом имеется в виду интервал *dt = 0.01*. Задаётся размер обучающей выборки.

Распределим данные для обучения сети, зададим тренировочные и тестовые выборки:

*time = copy.deepcopy(x)*

*X = time[:,0]*

*X = X.reshape(length,1)*

*X = 2 \* (X - X.min()) / (X.max() - X.min()) - 1*

*x, y = to\_forecasting(X, forecast=forecast)*

*X\_train1, y\_train1 = x[:train\_train], y[:train\_train]*

*X\_test1, y\_test1 = x[train\_train:], y[train\_train:]*

Берём начальный массив значений иксов системы Лоренца (X), получаем из него два массива, первый (x) — это массив (X), укороченный с конца на количество значений, которые мы собрались предсказать, например 10. Второй (y) — это массив (X), укороченный на столько же значений сначала. Каждый из них разбиваем на тренировочную и тестовую выборки:

X\_train1, y\_train1,

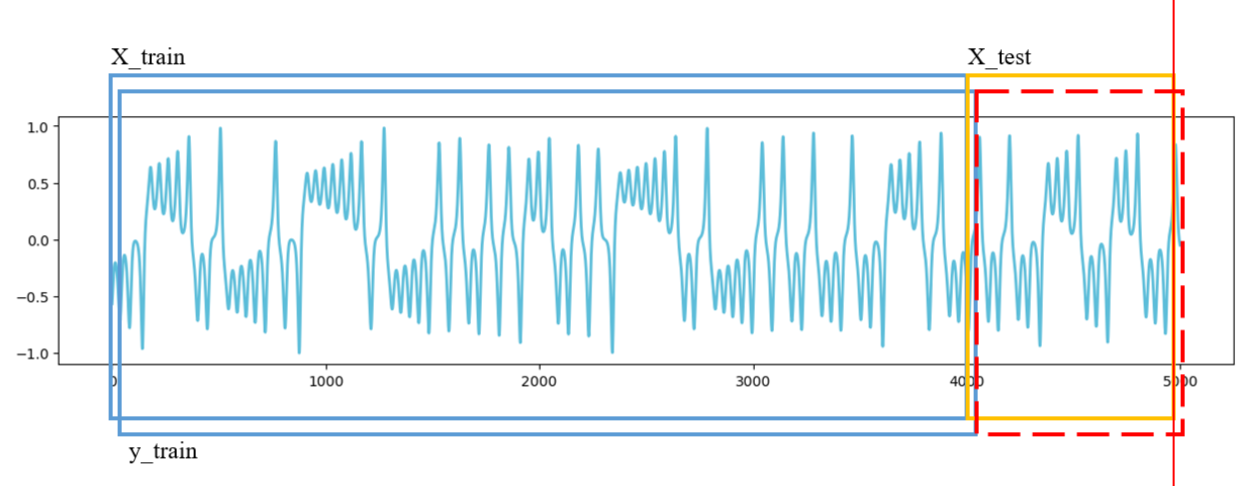
X\_test1, y\_test1.

Таким образом получили две тренировочные и две тестовые выборки, сдвинутые на значение 10 друг относительно друга.

Для обучения модели используются две тренировочные выборки: X\_train1, y\_train1, и одна тестовая, X\_test1, которая направляется в резервуар для предсказания себя самой, но со сдвигом (y\_test1).

Таким образом, сеть обучается на двух массивах значений, начальном и массиве со сдвигом. Если после обучения подать на вход сети массив данных, который следует за начальным массивом, мы получим его же, но со сдвигом. Этот сдвиг и будет данными, которые сеть, ничего о них не зная, предсказала. Наша цель заключается в наиболее точном их предсказании.

После подачи на вход резервуара X\_test1 и прохождения полученного от него отклика через выходной слой мы получаем предсказание (y\_pred1) – это массив, который по месту расположения соответствует y\_test1, то есть включает в себя 10 новых значений.



*Рисунок 5. Схема разделения данных*

Вертикальной красной чертой на схеме отделён отрезок, который будет предсказан, изначально он будет неизвестен для сети. Синими прямоугольниками обозначены обучающие выборки, на них будет обучен выходной слой, рыжим прямоугольником обозначена тестовая выборка, которая подаётся на вход резервуара. Красной пунктирной линией обозначена область, которая будет результатом работы системы.

## 2.3. Использование сети

Обучим резервуарную сеть и воспользуемся ей для предсказания временного ряда:

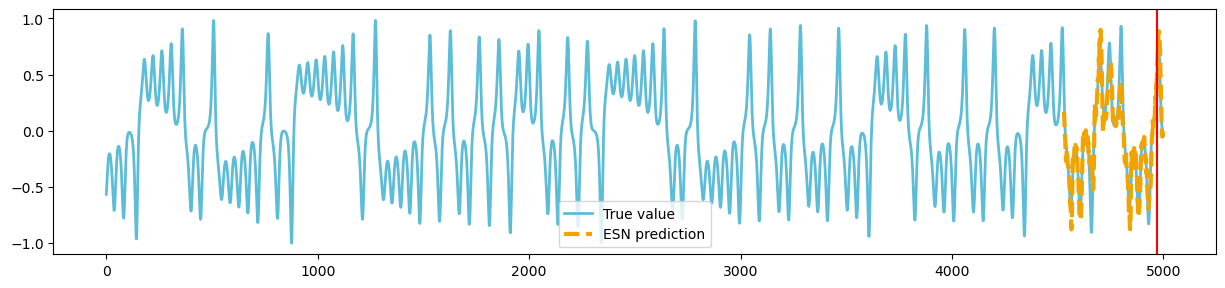
*esn = esn.fit(X\_train1, y\_train1)*

*y\_pred1 = esn.run(X\_test1)*

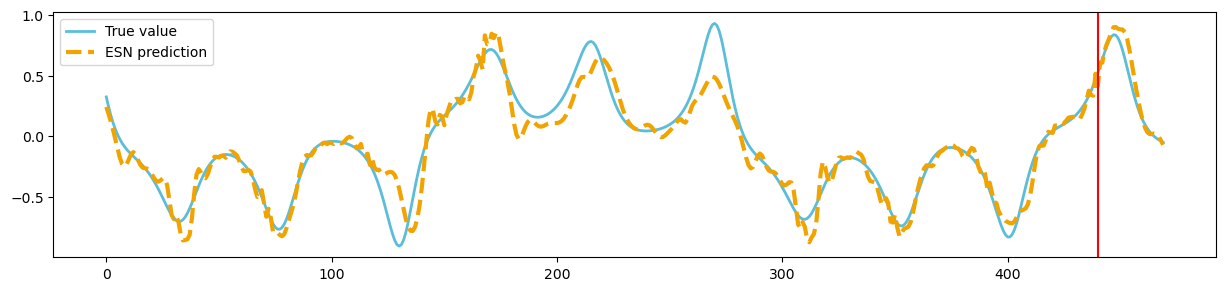
Для получения хорошего результата проведём гиперпараметрическую оптимизацию методом grid search. Гиперпараметрическая оптимизация методом grid search (сеточного поиска) заключается в последовательном переборе всех возможных комбинаций значений на сетке гиперпараметров для нахождения оптимальной модели. Использовалась метрика качества RMSE.

, (8)

**Результат: 30 предсказанных шагов**

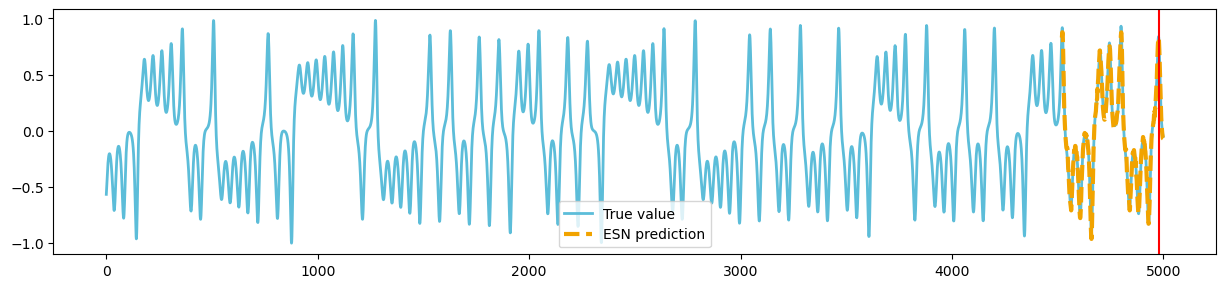


*Рисунок 6. Результат: 30 предсказанных шагов*

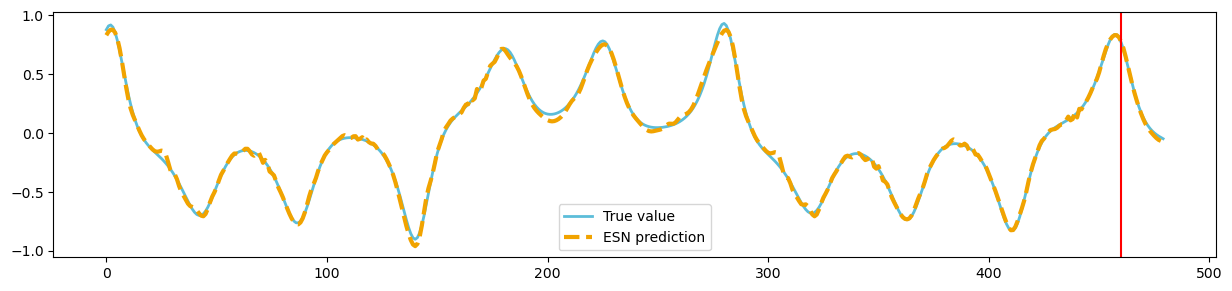


*Рисунок 7. Последние 500 значений предыдущего графика*

**Результат: 20 предсказанных шагов**

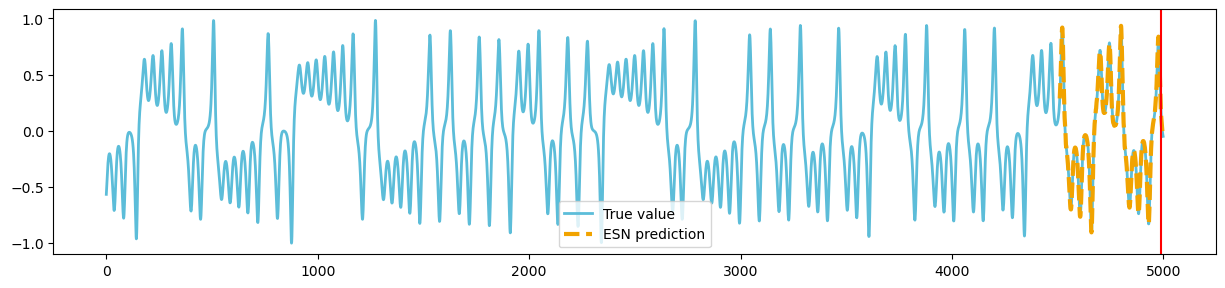


*Рисунок 8. Результат: 20 предсказанных шагов*

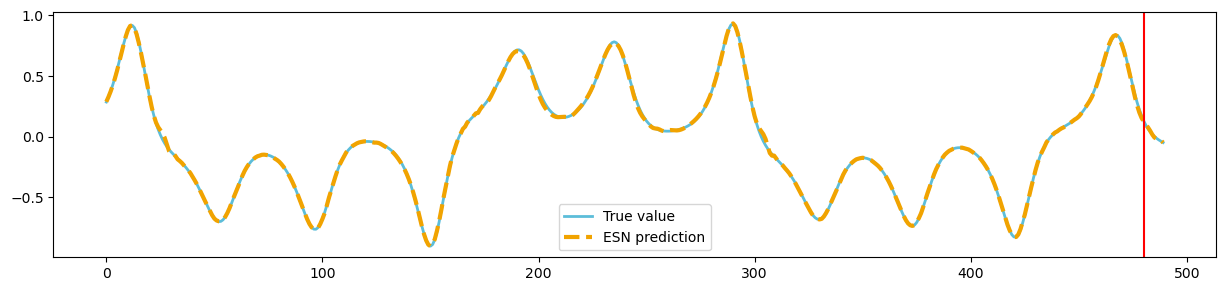


*Рисунок 9. Последние 500 значений предыдущего графика*

**Результат: 10 предсказанных шагов**

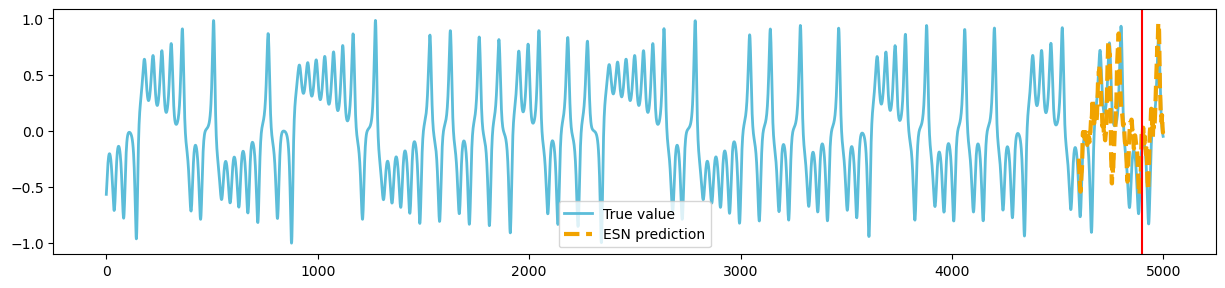


*Рисунок 10. Результат: 10 предсказанных шагов*

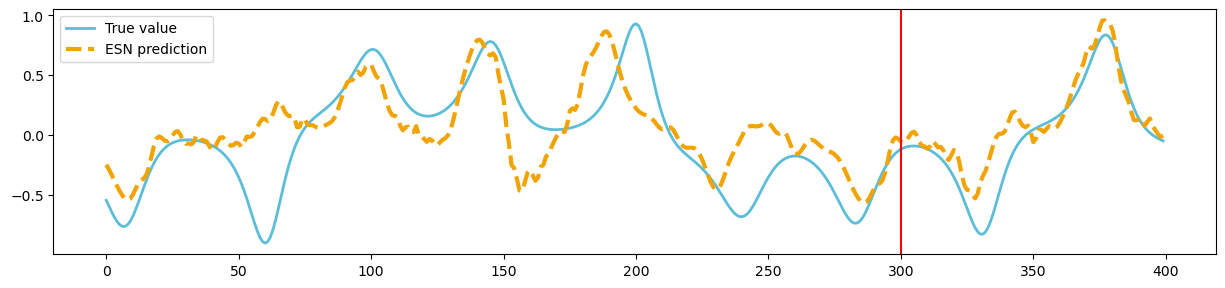


*Рисунок 11. Последние 500 значений предыдущего графика*

**Результат: 100 предсказанных шагов**



*Рисунок 12. Результат: 100 предсказанных шагов*



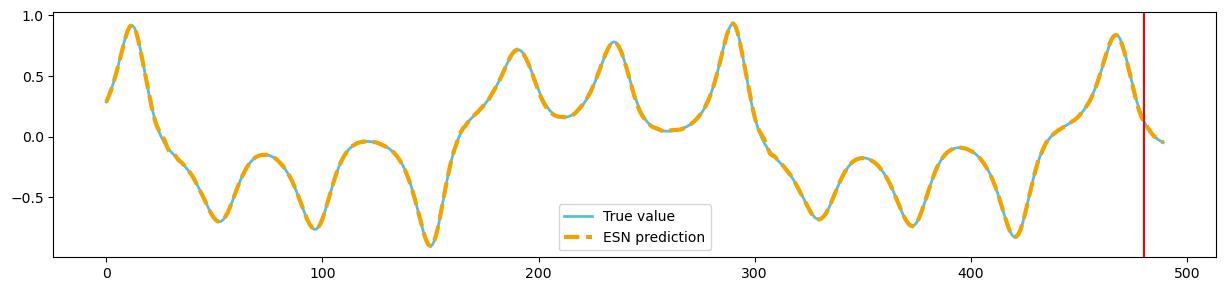
*Рисунок 13. Последние 500 значений предыдущего графика*

Как видим, даже предсказывая такой интервал как 100 временных шагов, резервуарная сеть улавливает динамику системы.

**Проверим обобщающую способность модели:**

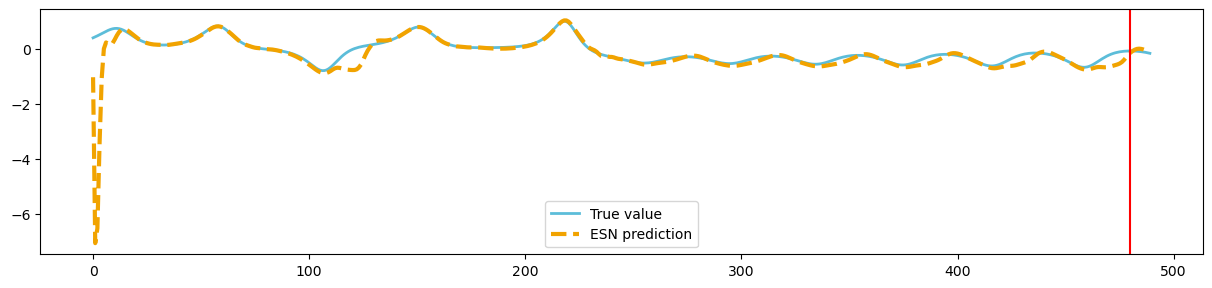
Сеть ESN была обучена (fit) на тренировочном интервале, это значит, была сформирована матрица . На вход резервуара был подан (run) тестовый интервал, и были получены его скрытые состояния . Было получено предсказание .

Попробуем на вход резервуара уже обученной сети с готовым подать другие траектории, из других начальных состояний, будем предсказывать 10 временных шагов:

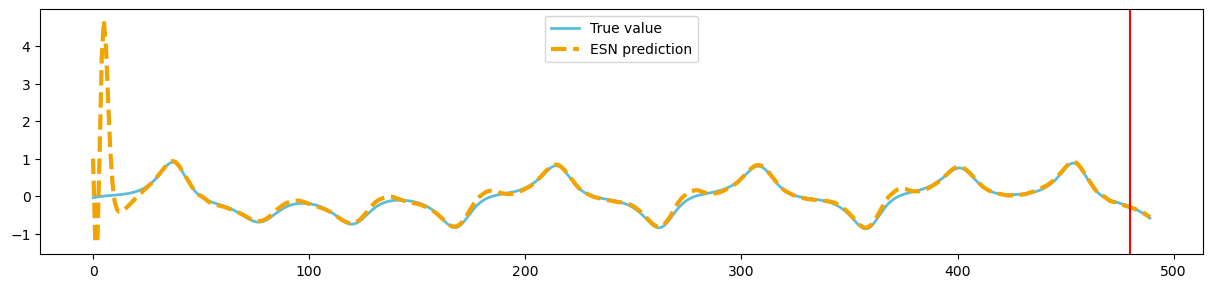


*Рисунок 14. Последние 500 значений предсказания траектории с начальным состоянием x0 = [0.1, -0.1, 1.0]. На ней сеть обучалась*

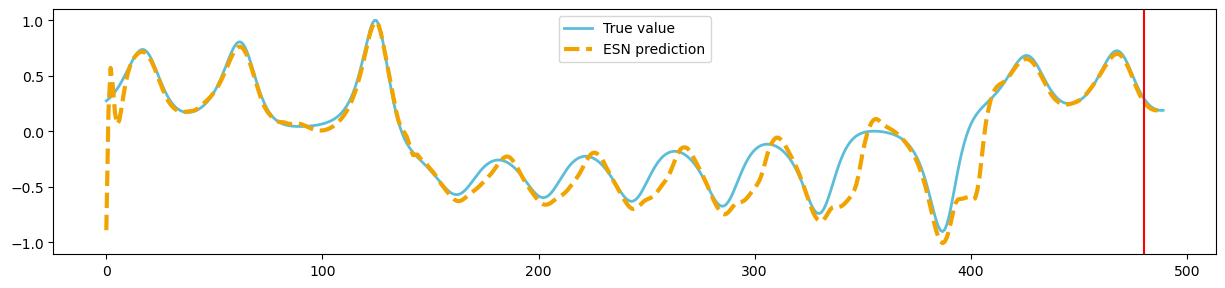
Новые траектории:



*Рисунок 15. Последние 500 значений предсказания траектории с начальным состоянием x0 = [4.0, -5.1, -10.0]*



*Рисунок 16. Последние 500 значений предсказания траектории с начальным состоянием x0 = [-1.0, 0.1, 0.0]*



*Рисунок 17. Последние 500 значений предсказания траектории с начальным состоянием x0 = [1.0, -500.1, 1.90]*

Сеть имеет обобщающую способность и может уловить динамку других траекторий.

## 2.4. Исследовательская задача

Проверить гипотезу о том, что оптимальные наборы гиперпараметров сети для разных траекторий, порождённых одной системой близки друг к другу. Для этого будем устанавливать различные начальные состояния, получать соответствующие траектории системы и проводить гиперпараметрическую оптимизацию методом grid search для каждой траектории, затем сравним результаты.

Результаты приведены в таблице. Строки – траектории (разные начальные условия), столбцы – гиперпараметры:

units (1)

leak\_rate (2)

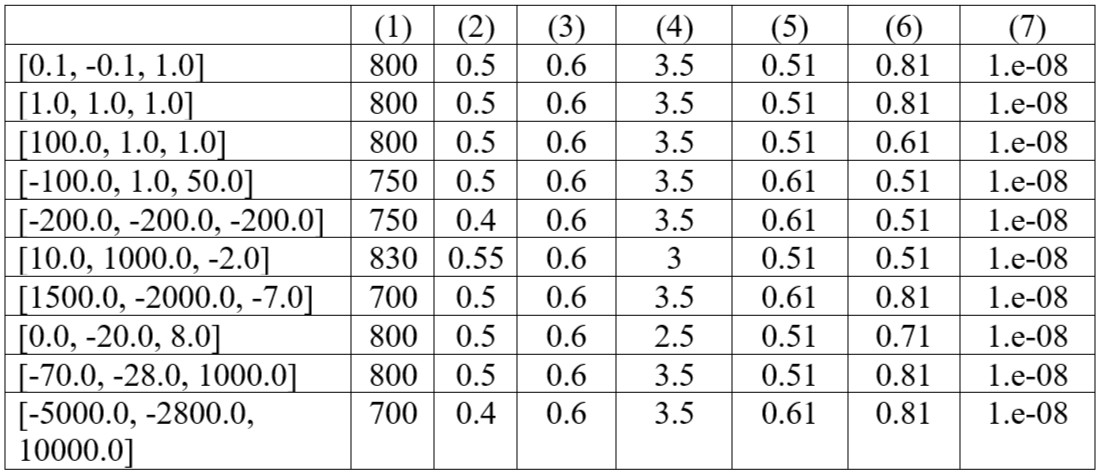
spectral\_radius (3)

input\_scaling (4)

connectivity (5)

input\_connectivity (6)

regularization (7)



Исходя из опытов, гипотеза верна.

# Заключение

В рамках данной работы была изучена архитектура ESN. Она была использована в анализе и прогнозировании поведения системы Лоренца. Была проведена гиперпараметрическая оптимизация сети, были достигнуты успехи в прогнозировании неизвестных значений системы на новых временных шагах. Была проверена обобщающая способность сети и было проведено исследование, близки ли друг к другу оптимальные наборы гиперпараметров для разных траекторий системы.

# Литература

1. DNN - Tutorial 2 Part I: Physics inspired Machine Learning

2. deeptime-ml.github.io

3. reservoirpy.readthedocs.io

4. Кузнецов С. П., Лекция 3. Система Лоренца; Лекция 4. Динамика системы Лоренца. // Динамический хаос (курс лекций). — М.: Физматлит, 2001.

5. Беляев В. А. Нейросетевые вычисления динамических параметров: магистерская диссертация. Санкт-Петербург: СПбГУ, 2020. 34 с.

6. Яшина Д.С. Прогнозирование временных рядов с помощью нейронной сети эхо – состояний на основе резервуарных вычислений: выпускная квалификационная работа бакалавра. Нижний Новгород: ННГУ, 2023. 33 с.