**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**
**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**
**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук
Образовательная программа бакалавриата «Программная инженерия»

УДК 004.8

СОГЛАСОВАНО
профессор департамента анализа данных и искусственного интеллекта факультета компьютерных наук НИУ ВШЭ, д-р физ.-мат. наук

_____ В. А. Громов
«___» _____ 2022 г.

УТВЕРЖДАЮ
Академический руководитель образовательной программы «Программная инженерия»
профессор департамента программной инженерии, канд. техн. наук

_____ В. В. Шилов
«___» _____ 2022 г.

**Выпускная квалификационная работа**
(академическая)
на тему: **Прогнозирование хаотических временных рядов: алгоритм self-healing для прогнозирования с помощью кластеризации**
по направлению подготовки 09.03.04 «Программная инженерия»

Выполнила студентка
образовательной программы
09.03.04 «Программная инженерия»
группы БПИ182

_____ А.Т. Антонова

«___» _____ 2022 г.

Москва – 2022

# Аннотация

В данной работе обсуждается развитие алгоритма прогнозирования хаотических временных рядов на много шагов вперед с помощью кластеризации. Прогнозирование с помощью кластеризации — один из методов прогнозирования хаотических временных рядов на много шагов вперед, который избегает экспоненциального роста ошибки за горизонтом предсказуемости. Для точки, которая прогнозируется алгоритмом, вычисляется множество возможных прогнозных значений относительно большого размера. На их основе алгоритм определяет, является ли точка прогнозируемой или непрогнозируемой, и вычисляет единое прогнозное значение для прогнозируемых точек.

Исследуется новая стратегия: алгоритм self-healing. Алгоритм self-healing — это итеративный алгоритм, который в качестве входных данных принимает прогнозы, полученные базовым алгоритмом прогнозирования. На каждой итерации алгоритм self-healing находит новые возможные прогнозные значений, обновляет статус точек с прогнозируемых на непрогнозируемые или наоборот и вычисляет новые единые прогнозные значение для прогнозируемых точек.

В работе предложено несколько новых алгоритмов вычисления единого прогнозного значения и алгоритмов определения непрогнозируемых точек. Проведены исследования по подбору гиперпараметров для алгоритма self-healing, по его оценке качества прогнозирования и по сравнению с существующим алгоритмом прогнозирования. Вычислительные эксперименты показали, что для временного ряда Лоренца с использованием алгоритма self-healing средняя ошибка прогнозирования почти не возрастает, однако количество непрогнозируемых точек значительно сокращается по сравнению с существующим алгоритмом прогнозирования с помощью кластеризации.

Работа содержит 41 страницу, 4 главы, 7 рисунков, 5 таблиц, 18 источников, 3 приложения.

**Ключевые слова:** хаотические временные ряды, прогнозирование на много шагов вперед, прогнозирование с помощью кластеризации, алгоритм self-healing.

# Abstract

The paper discusses a further development of predictive clustering approach for multi-step-ahead prediction of chaotic time series. Predictive clustering approach is one of multi-step-ahead prediction methods which avoids the exponential growth of prediction error up to the prediction horizon. For a point to be predicted a set of possible predicted values can be calculated. Using it the algorithm evaluates whether the point is predictable and calculates a unified predicted value for the predictable one.

A novel strategy of the self-healing algorithm is suggested in the present paper. The self-healing algorithm is an iterative algorithm which takes predictions obtained by the base prediction algorithm and iteratively tries to "heal" the part of the time series to be predicted by finding new possible predicted values, updating the status of points from predictable to non-predictable or backwards and calculating new unified predictions for predictable points.

In the present paper a number of novel algorithms of identifying non-predictable points and algorithms of calculating unified prediction are suggested. Experiments for hyperparameters selection for the self-healing algorithm, evaluation of prediction quality and comparison with the existing prediction algorithm were conducted. They showed that for the Lorenz time series average prediction errors increase slightly, however at the same time the number of non-predictable points decreases drastically compared to the existing prediction algorithm.

The work contains 41 pages, 4 chapters, 7 figures, 5 tables, 18 bibliography items, 3 appendices.



**Keywords:** chaotic time series; multi-step-ahead prediction; predictive clustering; self-healing algorithm.

## Термины и определения

**Горизонт прогнозирования** — количество шагов, на которое алгоритм прогнозирует временной ряд.

**Единое прогнозное значение** — значение, которое вычисляется одним из алгоритмов, используя множество возможных прогнозных значений. Является конечным прогнозом для некоторой точки временного ряда.

**Мотив** — похожие последовательности значений временного ряда, которые встречаются в наблюдаемой его части. Более подробно, мотивы — это центры кластеров, которые получились в процессе кластеризации обобщенных z-векторов, соответствующих некоторому паттерну.

**Множество возможных прогнозных значений** — множество значений прогнозируемой точки, которые получены из похожих мотивов. Используется для определения непрогнозируемых точек и вычисления единого прогнозного значения.

**Непрогнозируемая точка** — точка, которая либо имеет пустое множество возможных прогнозных значений, либо была определена как непрогнозируемая в соответсвии с одним из алгоритмов определения непрогнозируемых точек.

**Обобщенный z-вектор** — вектор, который состоит из наблюдений временного ряда, которые не обязательно последовательны и взяты в соответствии с расстояниями некоторого паттерна. Множество z-векторов для паттерна $k_1, k_2, \ldots, k_{L-1}$ имеет вид $\{(y_m, y_{m+k_1}, y_{m+k_1+k_2}, \ldots, y_{m+k_1+\ldots+k_{L-1}}) \mid 1 \leq m < t\}$, где $t = |Y_1|$.

**Паттерн** — вектор расстояний между наблюдениями временного ряда $k_1, k_2, \ldots, k_{L-1}$, где $L$ — длина паттерна.

**Хаотический временной ряд** — изменение со временем некоторой фиксируемой величины в хаотической динамической системе.

**Z-вектор** — $m$-мерный вектор, получаемых из наблюдений временного ряда и имеющий вид $z_i = (y_i, y_{i+1}, \ldots, y_{i+m-1})$ для временного ряда $Y = \{y_1, y_2, \ldots, y_i, \ldots\}$.

## Оглавление

# Введение

Хаотические системы встречаются в медицине, механике, химии, экономике и многих других научных областях. Например, признаки хаоса наблюдаются в процессе турбулентного движения жидкости, в данных метеонаблюдений, в биржевых курсах финансовых активов. Хаотическим временным рядом называется изменение со временем некоторой фиксируемой величины в хаотической динамической системе.

Одной из характеристик хаотической динамической системы является высокая чувствительность к начальным данным. Это означает, что малое начальное отличие в траекториях некоторого параметра системы ведет к экспоненциальному росту этого отличия с течением времени. Скорость роста отличия траекторий, а также горизонт предсказуемости $T$ оценивается с помощью старшего показателя Ляпунова. Горизонт предсказуемости характеризует среднее время, на которое можно предсказывать поведение системы. Считают, что $T \sim \frac{1}{\lambda_1}$, где $\lambda_1$ — старший показатель Ляпунова [1, с. 238]. В терминах прогнозирования хаотических временных рядов на много шагов вперед горизонт предсказуемости $T$ — теоретическая верхняя граница для горизонта прогнозирования $h$ для любого алгоритма. Горизонт прогнозирования — это количество шагов, на которое алгоритм прогнозирует временной ряд. Обычно горизонт прогнозирования много меньше горизонта предсказуемости: $T \gg h$. Однако подход прогнозирования с помощью кластеризации, который рассматривается в данной работе, позволяет прогнозировать временные ряды за горизонтом предсказуемости.

Прогнозирование с помощью кластеризации использует понятие мотивов: похожих последовательностей значений временного ряда, которые встречаются в наблюдаемой его части. Идея состоит в том, что если временной ряд перед прогнозируемой точкой похож на начало некоторого мотива, то высока вероятность, что прогнозируемая точка будет близка по значению с последней точкой мотива (Рисунок 1). Мотивы получают из наблюдаемой части временного ряда с помощью кластеризации векторов, которые состоят из наблюдений прогнозируемого временного ряда.

Вычислительные эксперименты показывают, что близких мотивов для одной прогнозируемой точки может быть много, так формируется множество возможных прогнозных значений. С другой стороны, не гарантируется, что для каждой прогнозируемой

точки найдется хотя бы один похожий мотив. Так возникает концепция непрогнозируемых точек.

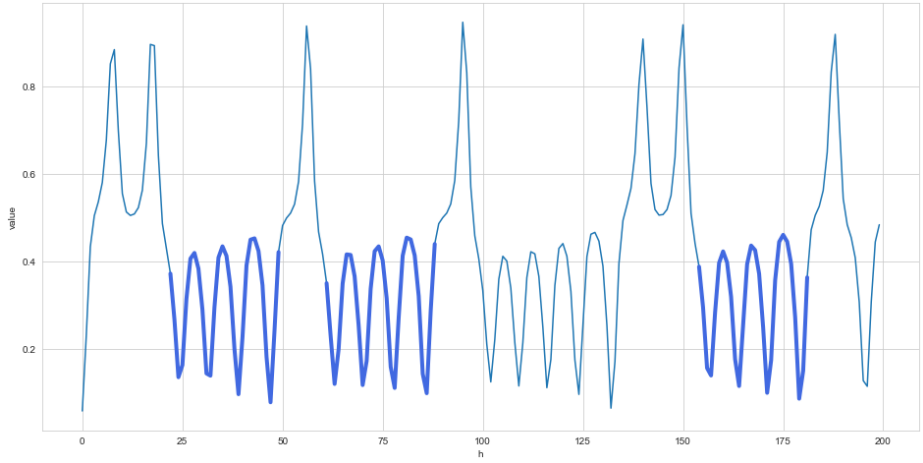


*Рисунок 1. Ряд Лоренца. Ярко-синим выделены похожие части временного ряда, которые могут быть использованы в прогнозировании*

Непрогнозируемые точки — это точки, для которых алгоритм либо не может вычислить прогнозное значение, так как не нашлось близких мотивов, либо вычисленное прогнозное значение имеет заведомо высокую ошибку. Примером второго случая может быть множество возможных прогнозных значений, состоящее из двух кластеров точек приблизительно одного размера, но с сильно отличающимися центрами. В алгоритме уже спрогнозированные точки используются в прогнозировании последующих. Преимущество определения прогнозируемая точка или нет состоит в том, что заведомо ошибочные прогнозные значения не используются в вычислении последующих точек. Более того, в терминах средней ошибки прогнозирования гораздо лучше, если алгоритм определяет точку как непрогнозируемую, чем он вынужденно вычисляет все промежуточные точки до горизонта прогнозирования, несмотря на высокую ошибку.

Достаточно большие множества возможных прогнозных значений, которые были получены статистически независимо друг от друга, позволяют использовать различные алгоритмы определения единого прогнозного значения. Например, это может быть среднее по всему множеству или центр наибольшего кластера.

Задача прогнозирования хаотического временного ряда в данных терминах может быть сформулирована как задача оптимизации: минимизации количества непрогнозируемых точек и минимизации средней по множеству единых прогнозных значений ошибки прогнозирования.

Цель данной работы — разработка и тестирование алгоритма self-healing, который должен улучшить качество прогнозирования в терминах средней ошибки прогнозирования и количества непрогнозируемых точек. Алгоритм self-healing — это итеративный алгоритм, в задачи которого входят поиск новых прогнозируемых точек и уточнение существующих прогнозов. В качестве входных данных он имеет результат работы базового алгоритма прогнозирования: множества возможных прогнозных значений и единые прогнозные значения для прогнозируемых точек.

Основная идея алгоритма self-healing заключается в следующем. Базовый алгоритм прогнозирует точку, используя только значения "слева", это либо известные значения временного ряда, либо полученные единые прогнозные значения. Алгоритм self-healing предлагает прогнозирование с использованием не только этих классов точек, но и точек "справа" — это полученные в базовом алгоритме или на предыдущих итерациях единые прогнозные значения, которые находятся за прогнозируемой точкой во временном ряде. Определение непрогнозируемых точек и здесь играет важную роль, так как алгоритм self-healing опирается на значения прогнозов, которые с высокой вероятностью близки к реальным значениям временного ряда, но не на заведомо ошибочные прогнозы.

Итеративный алгоритм позволяет изучать новые подходы к определению непрогнозируемых точек и к вычислению единых прогнозных значений. Например, существует гипотеза о том, что единые прогнозные значения, полученные основным алгоритмом, более достоверны, чем полученные на итерациях алгоритма self-healing. Она может быть выражена в алгоритме вычисления единого прогнозного значения с использованием весов у возможных прогнозных значений. Вес значения максимален у значений, полученных базовым алгоритмом, а для алгоритма self-healing уменьшается с номером итерации.

Цель работы: разработка и тестирование алгоритма self-healing для прогнозирования с помощью кластеризации

Задачи работы:

1. Изучить алгоритм прогнозирования хаотических временных рядов с использованием кластеризации обобщенных z-векторов.
2. Изучить алгоритмы кластеризации обобщенных z-векторов по набору паттернов.
3. Изучить алгоритмы определения непрогнозируемых точек и вычисления единого прогнозного значения.
4. Разработать и протестировать алгоритм self-healing.

5. Провести исследование алгоритмов определения непрогнозируемых точек и алгоритмов вычисления единого прогнозного значения для базового алгоритма и алгоритма self-healing с целью найти наиболее подходящие в терминах средней ошибки прогнозирования и количества непрогнозируемых точек.

6. Сделать выводы об эффективности использования алгоритма self-healing для прогнозирования хаотических временных рядов.

Отчет организован следующим образом. В главе 1 приведён обзор и анализ источников. Глава 2 содержит постановку задачи, описание алгоритма прогнозирования с помощью кластеризации, алгоритма self-healing, алгоритмов определения непрогнозируемых точек и алгоритмов вычисления единого прогнозного значения. В главе 3 обоснован выбор средств реализации программы для проведения исследований и описаны методы проведения вычислительных экспериментов. В главе 4 представлены результаты вычислительных экспериментов и сравнение с существующими алгоритмами прогнозирования. В заключении перечислены выполненные задачи, обсуждаются результаты экспериментов и пути дальнейшей работы.

# Глава 1. Обзор источников

**1. Прогнозирование на один шаг или несколько шагов вперед**

Часть исследований в области прогнозирования хаотических временных рядов рассматривает проблему прогнозирования на один или несколько шагов вперед, в пределах 20. Для ее решения используются различные методы машинного обучения. Например, LSTM нейронные сети [2], метод опорных векторов [3] или бустинг, использующий рекуррентные нейронные сети в качестве базовых алгоритмов [4]. Алгоритмы прогнозирования на много шагов вперед сложнее в разработке, возможно, из-за описанного ранее экспоненциального роста ошибки с увеличением горизонта прогнозирования.

**2. Стратегии прогнозирования на много шагов вперед**

Одно из направлений исследования прогнозирования временных рядов на много шагов вперед — это разработка и сравнение стратегий алгоритмов прогнозирования. В обзорной статье Souhaib Ben Taieb и др. [5] рассматриваются пять стратегий: итеративная, прямая, стратегии DirRec, MIMO и DIRMO. Итеративная (или рекурсивная) стратегия [6] заключается в том, что алгоритм прогнозирования на один шаг вперед запускается последовательно $h$ раз, где $h$ — горизонт прогнозирования, при этом значение на предыдущей итерации используется для вычисления прогноза на следующих шагах. Прямая стратегия [6] предполагает прогнозирование сразу для $h$-го шага без вычисления промежуточных прогнозных значений. Стратегия DirRec [6] (Direct + Recursive) — это комбинация прямой и итеративной стратегий: прогнозируемые значения вычисляются алгоритмом независимо, как в прямой, но входные данные для него расширяются с помощью приближенных прогнозных значений с предыдущих итераций. Согласно стратегии MIMO [7] (Multi-Input Multi-Output) алгоритм предсказывает не одно значение для горизонта прогнозирования $h$, а множества значений для всех точек между последними известными значениями ряда и горизонтом прогнозирования. Стратегия DIRMO [8] (DIRect + miMO) предлагает разделить предсказываемую часть ряда на блоки и применить алгоритм на основе стратегии MIMO для каждого блока, таким образом, DIRMO сочетает в себе преимущества обеих стратегий. Авторы статьи [5] провели эксперименты на большом объеме данных временных рядов и пришли к выводу, что алгоритмы на основе стратегии MIMO наиболее эффективны, а также предложили несколько подходов к обработке данных перед прогнозированием, увеличивающих точность алгоритмов.

### 3. Реконструкция странного аттрактора

Еще один подход к прогнозированию хаотических временных рядов — реконструкция странного аттрактора на основе частичных данных временного ряда и прогнозирование с использованием его восстановленной структуры. Аттрактор — это подмножество, к которому стремятся "почти все траектории" динамической системы [1, сс. 76-77], понятие странный аттрактор используется в случае неустойчивых траекторий. В работе Ф. Такенса [12, 1, с. 244] был математически обоснован анализ временного ряда для получения удовлетворительной геометрической картины странного аттрактора с использованием z-векторов — m-мерных векторов, получаемых из наблюдений временного ряда как в моделях авторегрессионного анализа: $z_i = (x_i, x_{i+1}, \ldots, x_{i+m-1})$.

Одно из недавних исследований — работа William Gilpin 2020 года [10], в ней описан алгоритм реконструкции странного аттрактора с помощью нейронной сети на основе автокодировщика с предложенной автором функцией потерь. Алгоритм с высокой точностью и высокой устойчивостью к шуму во входных данных восстанавливает структуры странных аттракторов на основе смоделированных и существующих в реальном мире временных рядов. Для прогнозирования используется cross-mapping алгоритм [11]. В этом алгоритме для каждой точки реконструированного аттрактора строится симплекс из ее ближайших соседей, для получения прогноза вычисляется среднее, взвешенное по дисперсии. Автор статьи [10] утверждает, что на горизонте прогнозирования $h = 50$ качество предложенной модели лучше, чем у других моделей, основанных на этом же принципе прогнозирования с использованием реконструкции аттрактора.

### 4. Методы глубокого обучения в прогнозировании хаотических временных рядов

В последнее время алгоритмы глубокого обучения получили широкое распространение, они применялись и для прогнозирования хаотических временных рядов на много шагов вперед. В статье Rohitash Chandra и др. [9] сравниваются такие алгоритмы, как нечеткие, рекуррентные и сверточные нейронные сети, LSTM сеть и ее улучшенные версии, регрессия на основе метода опорных векторов и другие. Они применены для прогнозирования различных временных рядов, в том числе хаотических. Авторы заключили, что Encoder-Decoder LSTM нейронная сеть и двунаправленная LSTM нейронная сеть показали наилучшую точность и на смоделированных, и на существующих в реальном мире временных рядах.

## 5. Прогнозирование с помощью кластеризации

Прогнозирование с помощью кластеризации — один из методов прогнозирования хаотических временных рядов на много шагов вперед, который избегает экспоненциального роста ошибки за горизонтом предсказуемости. В прогнозировании метод использует мотивы — похожие последовательности значений наблюдаемой части временного ряда. В терминах хаотической динамической системы похожие мотивы происходят из близких областей странного аттрактора. V.A. Gromov, E.A. Borisenko [13] предложили концепцию обобщенных z-векторов для получения набора мотивов. Для обобщенных z-векторы, соответствующих одному паттерну, проводится кластеризация, и центры кластеров называются мотивами. В статье V.A. Gromov, P.S. Baranov, A. Tsybakin [14] рассматривается множество подходов к определению непрогнозируемых точек и вычислению единых прогнозных значений. Вычислительные эксперименты показали лучшую точность в терминах средней ошибки прогнозирования для алгоритма траекторного прогнозирования со случайным возмущением. В то время, как количество непрогнозируемых точек растет почти экспоненциально с увеличением горизонта прогнозирования, благодаря алгоритмам определения непрогнозируемых точек ошибка прогнозирования остается почти на одном уровне.

Данная работа во многом основана на этом исследовании, алгоритм self-healing — это развитие существующего алгоритма прогнозирования с помощью кластеризации. В задачи данной работы входят разработка и тестирование алгоритма self-healing, подбор для него подходящих алгоритмов определения непрогнозируемых точек и вычисления единых прогнозных значений и разработка новых, так как объем исследуемых данных для вычисления прогноза увеличивается и существует возможность для исследования других подходов.

# Глава 2. Описание выбранных методов, моделей, алгоритмов исследования и решения задач

## 1. Постановка задачи

В данном разделе прогнозирование хаотического временного ряда на много шагов вперед сформулировано в терминах задачи оптимизации, введены основные понятия и обозначения.

Пусть $Y = \{y_1, y_2, \dots, y_t, \dots\}$ — хаотический временной ряд, все переходные процессы в динамической системе завершены, значения нормализованы. Задача: прогнозирование хаотического временного ряда на $h$ шагов вперед, $h \in \mathbb{N}$. Ряд разделен на обучающую $Y_1$ и тестовую $Y_2$ части: $Y = Y_1 \cup Y_2, Y_1 \cap Y_2 = \emptyset$. Обучающую часть ряда не стоит путать с обучающей выборкой, которая в свою очередь состоит из множеств обобщенных z-векторов.

Алгоритм прогнозирования вычисляет множество возможных прогнозных значений $\hat{S}_{t+h} = \{\hat{y}_{t+h}^{(1)}, \dots, \hat{y}_{t+h}^{(N_{t+h})}\}$, где $t$ — последняя позиция ряда с известным значением, $h$ — горизонт прогнозирования, $\hat{y}_{t+h}^{(i)}$ — $i$-ое возможное прогнозное значение, найденное алгоритмом, а $N_{t+h}$ — количество возможных прогнозных значений для прогнозируемой точки на позиции $t + h$.

Пусть оператор $\zeta$ показывает, является ли точка прогнозируемой:

$$\zeta(\hat{S}_{t+h}) = \begin{cases} 0, \text{ если точка непрогнозируемая} \\ 1, \text{ если точка прогнозируемая} \end{cases}$$

Оператор $g$ сопоставляет множеству возможных прогнозных значений единое прогнозное значение: $\hat{y}_{t+h} = g(\hat{S}_{t+h})$.

В этих терминах задача прогнозирования может быть сформулирована как задача оптимизации двух функций:

$$I_1 = min \sum_{t+h \in Y_2} (1 - \zeta(\hat{S}_{t+h})) \qquad (1)$$

$$I_2 = min \frac{1}{|Y_2|} \sum_{t+h \in Y_2} \zeta(\hat{S}_{t+h}) \parallel g(\hat{S}_{t+h}) - y_{t+h} \parallel^2 \qquad (2)$$

Функция $I_1$ минимизирует количество непрогнозируемых точек, а функция $I_2$ — среднюю ошибку единых прогнозных значений. Данный подход сочетает в себе итеративную, так как уже предсказанные значения используются в прогнозировании

следующих точек, и MIMO стратегии, так как алгоритм использует множества возможных прогнозных значений для формирования единого прогноза.

## 2. Алгоритм прогнозирования с помощью кластеризации

### 2.1. Мотивы

Для формирования обучающей выборки для кластеризации вводится понятие обобщенных z-векторов [13] — это z-векторы, состоящие из наблюдений, которые не обязательно последовательны и взяты в соответствии с расстояниями некоторого паттерна. Паттерн — это вектор расстояний между наблюдениями $k_1, k_2, \ldots, k_{L-1}$, где $L$ — длина паттерна. Таким образом, множество z-векторов для паттерна $k_1, k_2, \ldots, k_{L-1}$ имеет вид $\{(y_m, y_{m+k_1}, y_{m+k_1+k_2}, \ldots, y_{m+k_1+\ldots+k_{L-1}}) \mid 1 \leq m < t\}$, где $t = |Y_1|$. Например, для паттерна (3, 2, 4) множество z-векторов будет $(y_1, y_4, y_6, y_{10}), (y_2, y_5, y_7, y_{11}), \ldots, (y_{t-10}, y_{t-7}, y_{t-5}, y_{t-1})$ (Рисунок 2).

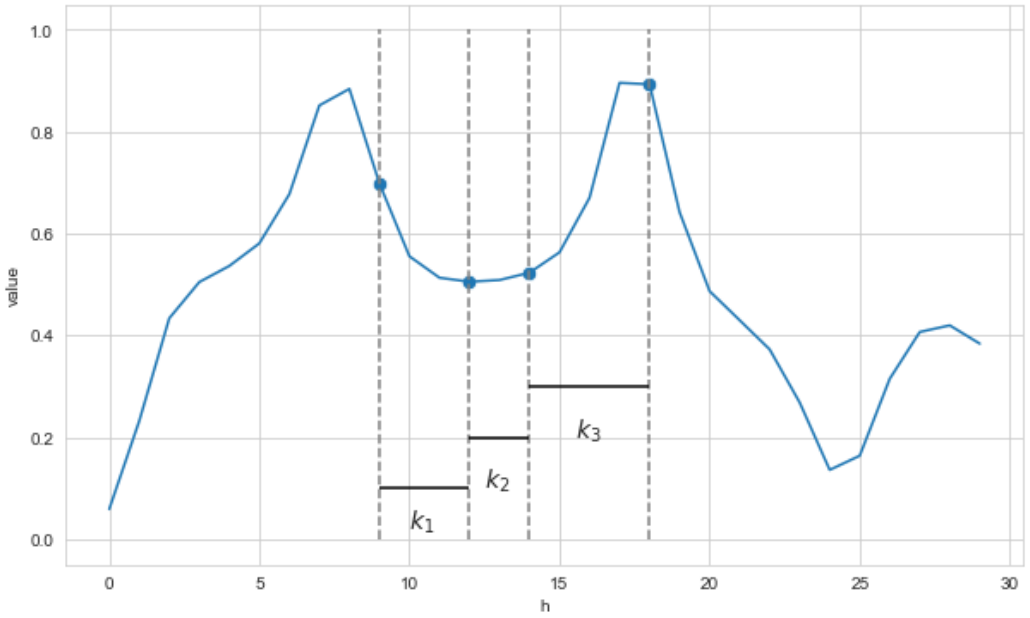


*Рисунок 2. Паттерн* (3, 2, 4) *для ряда Лоренца.*

Пусть $\aleph(L, k_{max})$ — множество всех комбинаторно возможных паттернов длины $L$, для которых максимальное расстояние между наблюдениями не более $k_{max}$. Обозначим как $\beta$ процент паттернов, которые будут использованы в алгоритме прогнозирования, паттерны выбираются случайно. Для каждого паттерна $\alpha \in \beta\aleph(L, k_{max})$ вычисляется множество z-векторов и выделяются мотивы — центры кластеров.

Используемые в работе алгоритмы кластеризации — DBSCAN и Wishart. Преимущество этих алгоритмов в том, что они не требуют количества кластеров во входных параметрах.

DBSCAN (density-based spatial clustering of applications with noise) — алгоритм кластеризации, основанный на плотности. Впервые алгоритм был предложен в 1996 году [18] и сейчас является одним из наиболее часто используемых алгоритмов кластеризации.

Алгоритм кластеризации Wishart [15], который был улучшен А. В. Лапко и С. В. Ченцовым [16], использует концепции теории графов и непараметрическую оценку функции плотности вероятности *r* ближайших соседей.

**2.2. Формирование множества возможных прогнозных значений**

Обозначим $\Xi_\alpha$ множество мотивов для паттерна $\alpha = (k_1^{(\alpha)}, \ldots, k_{L-1}^{(\alpha)})$. Для прогнозируемой точки на позиции $t + h$ для каждого паттерна $\alpha \in \beta\aleph(L, k_{max})$, для каждого мотива $C_\alpha \in \Xi_\alpha$, $C_\alpha = (\eta_1, \ldots, \eta_L)$ алгоритм прогнозирования формирует вектор $C = (y_{t+h-k_{L-1}^{(\alpha)}-\ldots-k_1^{(\alpha)}}, y_{t+h-k_{L-1}^{(\alpha)}-\ldots-k_2^{(\alpha)}}, \ldots, y_{t+h-k_{L-1}^{(\alpha)}})$. Все точки вектора *C* либо известны, либо предсказаны алгоритмом на предыдущих шагах. Если расстояние Евклида между вектором *C* и усеченным мотивом $Trunc(C_\alpha) = (\eta_1, \ldots, \eta_{L-1})$ меньше, чем некоторое малое число $\varepsilon$, то последнее значение мотива $C_\alpha$ — $\eta_L$ — добавляется в множество возможных прогнозных значений.

Другими словами, мотив "прикладывается" к ряду так, что последняя его точка совпадает с прогнозируемой. Если мотив достаточно похож на наблюдаемые значения, то высока вероятность, что и последнее значение мотива будет достаточно близко к настоящему значению прогнозируемой точки ряда.

**2.3. Траекторное прогнозирование**

В ходе вычислительных экспериментов, описанных в статье [14], хорошую точность показал алгоритм траекторного прогнозирования.

Вводится понятие возможной прогнозной траектории — это последовательность возможных прогнозных значений для точки на позиции горизонта прогнозирования и для предшествующих ей, не обязательно последовательных. Так алгоритм прогнозирования формирует не последовательность множеств возможных прогнозных значений, а несколько возможных прогнозных траекторий.

Один из методов вычисления возможных прогнозных траекторий — это траектории со случайным возмущением. К каждому полученному единому прогнозному значению траектории прибавляется случайный шум $\sigma(\Delta)$ с нулевым математическим ожиданием и дисперсией $\Delta > 0$, то есть $\hat{y}_{t+h}^{(p)} = g_0(\hat{S}_{t+h}^{(p)}) + \sigma(\Delta)$ для некоторого алгоритма вычисления

единого прогнозного значения $g_0(\hat{S}_{t+h}^{(p)})$ для $p$-ой траектории. Единое прогнозное значение вычисляется для каждой траектории отдельно. Количество возможных прогнозных траекторий $S_{max}$ — параметр алгоритма.

Единое прогнозное значение вычисляется как среднее по всем траекториям:

$$\hat{y}_{t+h} = \frac{1}{S_{max}} \sum_{p=1}^{S_{max}} \hat{y}_{t+h}^{(p)}.$$

### 2.4. Алгоритмы определения непрогнозируемых точек

Алгоритмы определения непрогнозируемых точек используют в качестве входных данных множество возможных прогнозных значений $\hat{S}_{t+h}$, а в случае траекторного прогнозирования — множество возможных прогнозных траекторий. Здесь и далее формулы приведены для позиции $t + h$, она отвечает точке, которая находится ровно на горизонте прогнозирования $h$. Для промежуточных точек алгоритмы определения непрогнозируемых точек и вычисления единых прогнозных значений такие же. В названии алгоритма в скобках дано название на английском языке и принятое сокращение. Полный список сокращений можно найти в приложении А.

1. Принудительное прогнозирование (forced prediction, *fp*)

Алгоритм прост: все точки с непустым множеством возможных прогнозных значений являются прогнозируемыми:

$$\zeta(\hat{S}_{t+h}) = \begin{cases} 0, |\hat{S}_{t+h}| = 0 \\ 1, |\hat{S}_{t+h}| > 0 \end{cases}$$

Этот алгоритм используется в основном для оценки работы других алгоритмов определения непрогнозируемых точек как граничный случай.

2. Большой разброс (large spread, *ls*)

Этот алгоритм предполагает, что если разница между максимальным и минимальных значениями множества возможных прогнозных значений достаточно велика, то точка не может считаться прогнозируемой.

Для работы алгоритма необходима валидационная часть временного ряда $Y_3$, $Y_1 \cap Y_3 = \emptyset$, $Y_2 \cap Y_3 = \emptyset$. Для каждого значения $Y_3$ вычисляется множество возможных прогнозных значений $\hat{S}_i$, для них вычисляется средняя разница между минимальным и максимальным значением множества (разброс): $\overline{L} = \frac{1}{|Y_3|} \sum_{i=1}^{|Y_3|} \max(\hat{S}_i) - \min(\hat{S}_i)$. Если

разница между максимальным и минимальным значениями больше, чем средний разброс, умноженный на некоторый коэффициент $\nu > 0$, то точка является непрогнозируемой:

$$\zeta(\hat{S}_{t+h}) = \begin{cases} 0, \ \max(\hat{S}_{t+h}) - \min(\hat{S}_{t+h}) \geq \nu\overline{L} \\ 1, \ \max(\hat{S}_{t+h}) - \min(\hat{S}_{t+h}) < \nu\overline{L} \end{cases}.$$

3. Быстрый рост разброса (rapid growth, *rg*)

Было замечено, что могут существовать такие отрезки прогнозируемого временного ряда, где разница между минимальным и максимальным значением множеств возможных прогнозных значений последовательных точек монотонно растет. Алгоритм основан на предположении, что если разброс возрастет монотонно на трех или более точках подряд, то точки после третьей включительно являются непрогнозируемыми. Обозначим $L_i = \max(\hat{S}_i) - \min(\hat{S}_i)$. Тогда

$$\zeta(\hat{S}_{t+h}) = \begin{cases} 0, \ L_{t+h-2} < L_{t+h-1} < L_{t+h} \\ 1, \text{ иначе} \end{cases}.$$

4. Быстрый рост числа кластеров DBSCAN (rapid growth DBSCAN, *rd*)

Алгоритм похож на предыдущий, но он наблюдает за ростом количества кластеров, которые получились с помощью кластеризации DBSCAN множества возможных прогнозных значений. Обозначим количество кластеров точки на $i$-ой позиции как $N_i^D$. Тогда

$$\zeta(\hat{S}_{t+h}) = \begin{cases} 0, \ N_{t+h-2}^D < N_{t+h-1}^D < N_{t+h}^D \\ 1, \text{ иначе} \end{cases}$$

5. Быстрый рост числа кластеров Wishart (rapid growth Wishart, *rw*)

Аналогичен предыдущему алгоритму, но в качестве кластеризации — алгоритм Wishart. Обозначим количество кластеров точки на $i$-ой как $N_i^W$. Тогда

$$\zeta(\hat{S}_{t+h}) = \begin{cases} 0, \ N_{t+h-2}^W < N_{t+h-1}^W < N_{t+h}^W \\ 1, \text{ иначе} \end{cases}$$

6. Ограничение на минимальный размер наибольшего кластера и на максимальное количество кластеров (limit cluster size, *lcs*)

Данный алгоритм применяется с алгоритмами единого прогнозного значения, которые вычисляют среднее по наибольшему кластеру. Алгоритм состоит в том, что если наибольший кластер содержит некоторую долю точек всего множества возможных прогнозных значений и количество кластеров не превышает установленное, то она является прогнозируемой.

Минимальный процент точек в наибольшем кластере $\gamma$ может быть как 5%, что отсеет множества возможных прогнозных значений, где кластеризация детектирует много

выбросов; так и 20-50%, что позволит считать точки прогнозируемыми только в том случае, если значительная часть точек множества возможных прогнозных значений попадает в этот наибольший кластер.

Максимальное количество кластеров $N_{max}$ также может варьироваться. Например, можно предположить, что точки с двумя и более кластерами не являются прогнозируемыми, так как нельзя однозначно сказать, что наибольший кластер — верный.

**2.5. Алгоритмы вычисления единого прогнозного значения**

Для вычисления единого прогнозного значения $\hat{y}_{t+h}$ используется множество возможных прогнозных значений $\hat{S}_{t+h}$: $\hat{y}_{t+h} = g_0(\hat{S}_{t+h})$ для некоторого алгоритма вычисления единого прогнозного значения $g_0$. Обозначим $i$-ое значение множества $\hat{S}_{t+h}$ как $\hat{y}^{(i)}_{t+h}$.

1. Среднее множества возможных прогнозных значений

$$\hat{y}_{t+h} = \frac{1}{|\hat{S}_{t+h}|} \sum_{i=1}^{|\hat{S}_{t+h}|} \hat{y}^{(i)}_{t+h}$$

2. Среднее, взвешенное по дистанции до мотива

$$\hat{y}_{t+h} = \frac{1}{\sum_{j=1}^{|\hat{S}_{t+h}|} \omega_j} \sum_{i=1}^{|\hat{S}_{t+h}|} \omega_i \hat{y}^{(i)}_{t+h}, \text{ где } \omega_j = \|C - Trunc(C_\alpha)\|_2$$

3. Центр наибольшего кластера DBSCAN (*db*)

Кластеризация DBSCAN для множества возможных прогнозных значений возвращает некоторое количество кластеров, в числе которых кластер под меткой $-1$, который содержит выбросы. Обозначим множество точек, которые попадают в кластер номер $j$ как $Q_j$. Пусть $j^{max} = \arg\max_j(|Q_j|)$ для кластеров, не являющихся выбросами. Тогда

$$\hat{y}_{t+h} = \frac{1}{|Q_{j^{max}}|} \sum_{\hat{y} \in Q_{j^{max}}} \hat{y}.$$

В данном алгоритме неявно содержится алгоритм определения непрогнозируемых точек: если кластеризация DBSCAN не идентифицировала ни один кластер, все точки попали в кластер выбросов, то точка является непрогнозируемой.

4. Центр наибольшего кластера Wishart (*wi*)

Алгоритм аналогичен алгоритму под пунктом 4, но в качестве кластеризации используется алгоритм Wishart.

5. Центр наибольшего кластера OPTICS (*op*)

Алгоритм аналогичен алгоритму под пунктом 4, но в качестве кластеризации используется алгоритм OPTICS. Основная идея алгоритма OPTICS похожа на DBSCAN, но он лучше решает проблемы обнаружения содержательных кластеров в данных, имеющих различные плотности. Есть предположение, что данный алгоритм больше подходит для данных возможных прогнозных значений, так как более плотный кластер может считаться более достоверным кластером для вычисления центра как единого прогнозного значения.

В ходе вычислительных экспериментов в основном был использован алгоритм центра наибольшего кластера DBSCAN.

### 2.6. Псевдокод базового алгоритма прогнозирования

**Algorithm 1** Base prediction algorithm

**procedure** BASE PREDICTION($Y_1, h$)
  $\varepsilon \leftarrow 0.01$
  normalize observations $Y_1$
  $t \leftarrow$ index of last known observation
  **for** $i \leftarrow 1, h$ **do**
    **for** $\alpha \in \beta\aleph(L, k_{max})$ **do** ▷ here $\alpha = (k_1^{(\alpha)}, ..., k_{L-1}^{(\alpha)})$
      **for** $l \leftarrow 1, L$ **do**
        $\eta_l^{(\alpha)} \leftarrow y_{t-k_{L-1}^{(\alpha)}-...-k_{L-i+1}^{(\alpha)}}$
      **end for**
      $C_\alpha \leftarrow (\eta_1^{(\alpha)}, ..., \eta_L^{(\alpha)})$
      $C \leftarrow (y_{t+h-k_{L-1}^{(\alpha)}-...-k_1^{(\alpha)}}, y_{t+h-k_{L-1}^{(\alpha)}-...-k_2^{(\alpha)}}, ..., y_{t+h-k_{L-1}^{(\alpha)}})$
      $Trunc(C_\alpha) \leftarrow (\eta_1^{(\alpha)}, ..., \eta_{L-1}^{(\alpha)})$
      **if** $\rho(C, Trunc(C_\alpha)) < \varepsilon$ **then** ▷ $\rho$ - Euclidian distance
        add $\eta_L^{(\alpha)}$ to $\hat{S}_{t+i}$
      **end if**
    **end for**
    **if** point $t+i$ is predictable **then** ▷ by corresponding algorithm
      calculate $\hat{y}_{t+h}$ ▷ by corresponding algorithm
    **end if**
  **end for**
**end procedure**

## 3. Алгоритм self-healing

### 3.1. Описание алгоритма self-healing

Алгоритм self-healing так же, как базовый алгоритм прогнозирования, решает задачу оптимизации функций (1) и (2). Алгоритм итеративный. На каждой итерации он выполняет следующие функции:

- поиск новых возможных прогнозных значений для каждой точки до горизонта прогнозирования,
- обновление статуса точек с прогнозируемой на непрогнозируемую или наоборот,
- вычисление новых единых прогнозных значений для прогнозируемых точек.

Критерий останова алгоритма:

- на последних трех итерациях набор прогнозируемых точек остается таким же;
- расстояние Евклида между векторами, состоящими из единых прогнозных значений, на последних двух итерациях должно быть не больше, чем некоторое малое число $\varepsilon_h$.

Возможные прогнозные значения вычисляются почти так же, как и в базовом алгоритме. Однако здесь прогнозируемая точка может быть на любой позиции в мотиве, с которым сравнивается отрезок ряда, содержащих прогнозируемую точку. То есть для каждого паттерна $\alpha = (k_1^{(\alpha)}, \ldots, k_{L-1}^{(\alpha)})$, для каждого мотива $C_\alpha \in \Xi_\alpha$, $C_\alpha = (\eta_1, \eta_2, \ldots, \eta_L)$, для каждой позиции в мотиве $j = (1,2,...,L)$ составляется вектор $C' = (y_{t+h-k_{L-1}^{(\alpha)}-\ldots-k_1^{(\alpha)}}, \ldots, y_{t+h-k_{L-1}^{(\alpha)}-\ldots-k_{j-1}^{(\alpha)}}, y_{t+h-k_{L-1}^{(\alpha)}-\ldots-k_{j+1}^{(\alpha)}}, \ldots, y_{t+h})$. Все значения вектора $C'$ либо известны, либо были вычислены на предыдущих итерациях или основным алгоритмом. Если расстояние Евклида между $C'$ и $Trunc'(C_\alpha) = (\eta_1, \ldots, \eta_{j-1}, \eta_{j+1}, \ldots, \eta_L)$ меньше, чем некоторое малое число $\varepsilon$, то компонента мотива $\eta_j$ считается новым возможным прогнозным значением для точки на позиции $t + h - k_{L-1}^{(\alpha)} - \ldots - k_p^{(\alpha)}$.

Изменение статуса точки с прогнозируемой на непрогнозируемую и наоборот происходит в соответствии с алгоритмами определения непрогнозируемых точек, которые описаны в соответствующем разделе работы. Однако для некоторых экспериментов полезно фиксировать статус точек, которые были прогнозируемыми в результате работы базового алгоритма прогнозирования.

### 3.2. Новые алгоритмы вычисления единого прогнозного значения

В алгоритме self-healing помимо множества возможных прогнозных значений для вычисления единого прогнозного значения можно использовать еще один параметр: на какой итерации было получено возможное прогнозное значение.

Например, можно предположить, что нужно больше доверять прогнозам, полученным базовым алгоритмом, а с ростом номера итерации возможные прогнозные значения все менее и менее заслуживают доверия. Или напротив, считать, что прогнозы, полученным базовым алгоритмом, изначально содержат значительную ошибку, которая может вести к экспоненциальному росту ошибки на следующих позициях. То есть считать возможные прогнозные значения, полученные алгоритмом self-healing, более значимыми и исправляющими ошибки базового прогнозирования.

В работе предложено несколько алгоритмов вычисления единого прогнозного значения с учетом итераций, на которой были получены возможные прогнозные значения. Введем следующие обозначения. $\hat{S}_{t+h}^{[i]}$ — множество возможных прогнозных значений, полученных на $i$-ой итерации, $\hat{S}_{t+h}^{[i]}$ является подмножеством множества возможных прогнозных значений $\hat{S}_{t+h}$. $i_{max}$ — количество итераций, которое работал алгоритм, пока не был выполнен критерий останова, а нулевая итерации — это базовый алгоритм прогнозирования, то есть $\hat{S}_{t+h}^{[0]}$ — множество возможных прогнозных значений, полученное базовым алгоритмом.

1. Двойная кластеризация (double clustering, *dc*)

Для каждой итерации единое прогнозное значение $\hat{y}_{t+h}^{[i]}$ вычисляется независимо с использованием только значений множества $\hat{S}_{t+h}^{[i]}$ для $0 \leq i \leq i_{max}$. Затем выполняется кластеризация полученных значений алгоритмом DBSCAN с параметрами $\varepsilon = 0.1$, $min\_samples = 2$. То есть кластер может содержать от 2 элементов, и минимальное расстояние для элементов внутри кластера значительно больше, чем при кластеризации множества возможных прогнозных значений. Это позволяет с высокой вероятностью получить хотя бы один кластер, отличный от кластера выбросов. Единое прогнозное значение вычисляется как среднее наибольшего кластера.

Алгоритм вычисления $\hat{y}_{t+h}^{[i]}$ может быть любым, но в основном использовался алгоритм, вычисляющий центр наибольшего кластера DBSCAN, отсюда и название двойная кластеризация.

2. Взвешенное по итерации среднее (weighted average, *wa*)

Для каждой итерации единое прогнозное значение $\hat{y}_{t+h}^{[i]}$ вычисляется независимо с использованием только значений множества $\hat{S}_{t+h}^{[i]}$ для $0 \leq i \leq i_{max}$. Единое прогнозное значение вычисляется как взвешенное среднее: $\hat{y}_{t+h} = \frac{1}{\sum_{i=0}^{i_{max}} \omega_i} \sum_{i=0}^{i_{max}} \omega_i \hat{y}_{t+h}^{[i]}$. Веса определяются с помощью фактора $f > 0$, вес промежуточного прогноза $\hat{y}_{t+h}^{[i]}$ устанавливается как $\omega_i = f^i$.

3. Центр кластера, который является наибольшим с учетом веса точек в нем

Точка множества возможных прогнозных значений наделяется весом в зависимости от итерации, на которой она была получена, для каждой точки $\hat{y} \in \hat{S}_{t+h}^{[i]}$ вес равен $\omega(\hat{y})$.

Выполняется кластеризация множества возможных прогнозных значений $\hat{S}_{t+h}$. Размер кластера вычисляется с учетом весов точек, которые в него входят. Пусть $Q_j$ — множество точек, которые попадают в кластер номер $j$. Тогда размер кластера равен $K_j = \sum_{\hat{y} \in Q_j} \omega(\hat{y})$. Тогда

$$\hat{y}_{t+h} = \frac{1}{|Q_{j^{max}}|} \sum_{\hat{y} \in Q_{j^{max}}} \hat{y}, \text{ где } j^{max} = \arg\max_j(K_j).$$

Веса могут быть различны по структуре:

- Выбирается фактор $f > 0$, вес точки, полученной на $j$-ой итерации, устанавливается как $\omega(\hat{y}) = f^j$. Если $f < 1$, то значения, полученные на ранних итерациях имеют бóльший вес, чем на поздних. Если $f > 1$, то наоборот. Данный метод позволяет проверить гипотезы о значимости для минимизации ошибки итерации, на которой было получено значение. (*factor*)
- Вес вычисляется как обратное расстоянию до мотива, из которого было получено значение, то есть $\omega(\hat{y}) = \frac{1}{d(\hat{y})}$, где $d(\hat{y})$ — расстояние до соответствующего мотива. Точки, более близкие к мотиву, из которого они были получены, имеют бóльший вес. (*dist*)
- Вес вычисляется как обратное длине паттерна, из которого было получено значение, то есть $\omega(\hat{y}) = \frac{1}{k_1^{(\hat{y})} + .. + k_{L-1}^{(\hat{y})}}$, где $k_1^{(\hat{y})}, \ldots, k_{L-1}^{(\hat{y})}$ — расстояния в паттерне. Данный алгоритм может использоваться для проверки гипотезы о том, что чем ближе точки мотива к прогнозируемой, тем больше нужно доверять прогнозу, полученному с его помощью. (pattern length, *pl*)

### 3.3. Новые алгоритмы определения непрогнозируемых точек

Как и для новых алгоритмов вычисления единого прогнозного значения, новые алгоритмы определения непрогнозируемых точек используют дополнительный один параметр: на какой итерации было получено возможное прогнозное значение.

1. Большой скачок (big leap, *bl*)

Данный алгоритм выполняется после завершения очередной итерации. Он использует валидационную часть ряда $Y_3$. Для нее вычисляется минимальное и максимальное отличие соседних значений временный рядов ($d_{min}$, $d_{max}$). Если отличие от значения предыдущей точки $d_{i-1,i}$ меньше минимального или больше максимального (с некоторыми коэффициентами $\kappa_{min}$, $\kappa_{max}$), то точка непрогнозируемая:

$$\zeta(\hat{S}_{t+h}) = \begin{cases} 0, \, d_{h-1,h} < \kappa_{min} d_{min} \quad \text{или} \quad d_{h-1,h} > \kappa_{max} d_{max} \\ 1, \text{ иначе} \end{cases}.$$

2. Большой скачок между итерациями (big leap between iterations, *blbi*)

Предположение алгоритма состоит в том, что единые прогнозные значения изменяются незначительно от итерации к итерации. Если единое прогнозное значение, которое было вычислено на текущей итерации, значительно отличается от предыдущего, то точка является непрогнозируемой. Если на предыдущей итерации точка была непрогнозируемой, то алгоритм рассматривает не более трех предыдущих итераций.

3. Странные паттерны (weird patterns, *wp*)

Вычислительные эксперименты показали, что на промежуточных итерациях возможны последовательности наблюдений, которые точно не могут встретиться в настоящем временном ряде. Алгоритм использует мотивы для некоторых паттернов (например, {(1,1,1), (1,2,1)}). Если после завершения очередной итерации в полученном временном ряде есть обобщенные z-векторы, для которых не найдено ни одного достаточно близкого мотива, то все точки обобщенного z-вектора, кроме первой, являются непрогнозируемыми.

### 3.4. Псевдокод алгоритма self-healing

**Algorithm 2** Self-healing algorithm

**procedure** SELF-HEALING($Y_1, h, \hat{Y}_2, \hat{S}$)
  **for** $i \leftarrow 1, h$ **do**
    **for** $\alpha \in \beta\aleph(L, k_{max})$ **do** ▷ here $\alpha = (k_1^{(\alpha)}, ..., k_{L-1}^{(\alpha)})$
      **for** $j \leftarrow 1, L$ **do**
        **for** $l \leftarrow 1, L$ **do**
          $\eta_l^{(\alpha)} \leftarrow y_{t-k_{L-1}^{(\alpha)}-...-k_{L-i+1}^{(\alpha)}}$
        **end for**
        $C_\alpha \leftarrow (\eta_1^{(\alpha)}, ..., \eta_L^{(\alpha)})$
        $C' \leftarrow (y_{t+h-k_{L-1}^{(\alpha)}-...-k_1^{(\alpha)}}, ..., y_{t+h-k_{L-1}^{(\alpha)}-...-k_{j-1}^{(\alpha)}},$
          $y_{t+h-k_{L-1}^{(\alpha)}-...-k_{j+1}^{(\alpha)}}, ..., y_{t+h})$
        $Trunc'(C_\alpha) \leftarrow (\eta_1, ..., \eta_{j-1}, \eta_{j+1}, ..., \eta_L)$
        **if** $\rho(C', Trunc'(C_\alpha)) < \varepsilon$ **then** ▷ $\rho$ - Euclidian distance
          add $\eta_L^{(\alpha)}$ to $\hat{S}_{t+i}$
        **end if**
      **end for**
    **end for**
    **if** point $t+i$ is predictable **then** ▷ by corresponding algorithm
      calculate $\hat{y}_{t+h}$ ▷ by corresponding algorithm
    **end if**
  **end for**
**end procedure**

# Глава 3. Выбор средств реализации программы для проведения исследований, описание методов проведения вычислительных экспериментов

## 1. Выбор средств реализации программы для проведения исследований

Технологии, которое были использованы в ходе проведения исследования:

- Python 3.9
- Библиотеки Python для анализа данных, машинного обучения, визуализации и сериализации (Pandas, Numpy, Scikit-learn, Seaborn, Pickle и другие)
- Среды разработки Jupyter Notebook и Visual Studio Code
- "cHARISMa" (Computer of HSE for Artificial Intelligence and Supercomputer Modelling) — суперкомпьютер НИУ ВШЭ для ресурсозатратных вычислений [17]

## 2. Данные для вычислительных экспериментов

Хаотический временной ряд, который используется в проведенном исследовании, — это ряд Лоренца. Хаотический временной ряд Лоренца порождается динамической системой Лоренца, поведение которой определяется системой обыкновенных дифференциальных уравнений:

$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(r - z) - y. \\ \dot{z} = xy - bz \end{cases} \qquad (3)$$

Параметры, использованные в данной работе: $\sigma = 10$, $r = 28$, $b = 8/3$.

Для получения численного решения системы обыкновенных дифференциальных уравнений (3) используется метод Рунге-Кутты 4 порядка, в качестве одномерного временного ряда взято изменение компоненты $x$.

## 3. Вычислительные эксперименты

### 3.1. Оценка качества работы алгоритма прогнозирования

Использованные паттерны, соответствующие им мотивы вычисляются с помощью обучающей части временного ряда $Y_1$, производятся расчеты для алгоритма определения непрогнозируемых точек на валидационной части временного ряда $Y_3$, если это предусмотрено алгоритмом.

Эксперимент проводится на тестовой части временного ряда $Y_2 = (y_1^{(2)}, y_2^{(2)} \ldots, y_n^{(2)})$, где $n$ — размер тестовой части. Для каждого горизонта прогнозирования $1 \leq h \leq h_{max}$ вычисляются средние ошибки и среднее количество непрогнозируемых точек следующим образом. Каждая точка $y_i^{(2)} \in Y_2$ обозначается для алгоритма как точка для прогнозирования,

позиция которой совпадает с текущим горизонтом прогнозирования $h$. Другими словами, $Y_2$ разделяется на известную и прогнозируемую части — $(y_1^{(2)}, \ldots, y_{i-h}^{(2)})$ и $(y_{i-h+1}^{(2)}, \ldots, y_i^{(2)})$ соответственно. Алгоритм прогнозирования запускается для того, чтобы получить прогноз для $i$-ой точки тестовой части временного ряда для горизонта прогнозирования $h$ — $\hat{y}_i^{(2)}(h)$.

Все прогнозы, соответствующие одному горизонту прогнозирования объединяются в множество $\hat{Y}_2(h) = \{\hat{y}_i^{(2)}(h) \mid 1 \leq i \leq n\}$. Вычисляются метрики качества: среднеквадратичная ошибка (RMSE — root mean squared error), средняя абсолютная ошибка (MAPE — mean absolute percentage error) и среднее количество точек, которые оказались непрогнозируемыми для данного горизонта прогнозирования.

### 3.2. Эксперимент для временного ряда с выкинутыми точками

Для того чтобы проверить, восстанавливает ли алгоритм self-healing промежуточные точки и с какой точностью он это делает, используется эксперимент с выкинутыми точками. Из тестовой части ряда, значения которого известны, "выкидывается" некоторое количество точек: они объявляются непрогнозируемыми, и у них пустое множество возможных прогнозных значений. Оставшиеся точки — прогнозируемые, для корректной работы алгоритма множество возможных прогнозных значений — это кластер из 20 наблюдений, каждое из которых — это истинное значение ряда со случайным шумом $\sigma(\Delta)$ с нулевым математическим ожиданием и дисперсией $\Delta > 0$.

Один из вариантов эксперимента использует фиксированный статус точек, которые не были выкинуты. В таком случае в оценку качества работы алгоритма включается только восстановление промежуточных точек, а не исправление ошибок предыдущих итераций.

Данный эксперимент может проводиться для визуальной оценки восстановления промежуточных точек и вычисления ошибок прогнозирования в частных случаях количества выкинутых точек, например, для пяти выкинутых точек. Более подробное исследование включает вычисление ошибок прогнозирования (среднеквадратичной и средней абсолютной ошибок), количества непрогнозируемых точек и количества итераций, которое потребовалось алгоритму self-healing, для количества выкинутых точек $1 \leq k \leq h$, где $h$ — исследуемый горизонт прогнозирования.

# Глава 4. Обработка результатов вычислительных экспериментов

Исследование выполнено с использованием суперкомпьютерного комплекса НИУ ВШЭ [17]. Обозначения, используемые в описаниях графиков, перечислены в Приложении А.

## 1. Базовый алгоритм прогнозирования: проверка корректонсти реализации и подбор гиперпараметров

### 1.1. Исследование алгоритмов определения непрогнозируемых точек для базового алгоритма

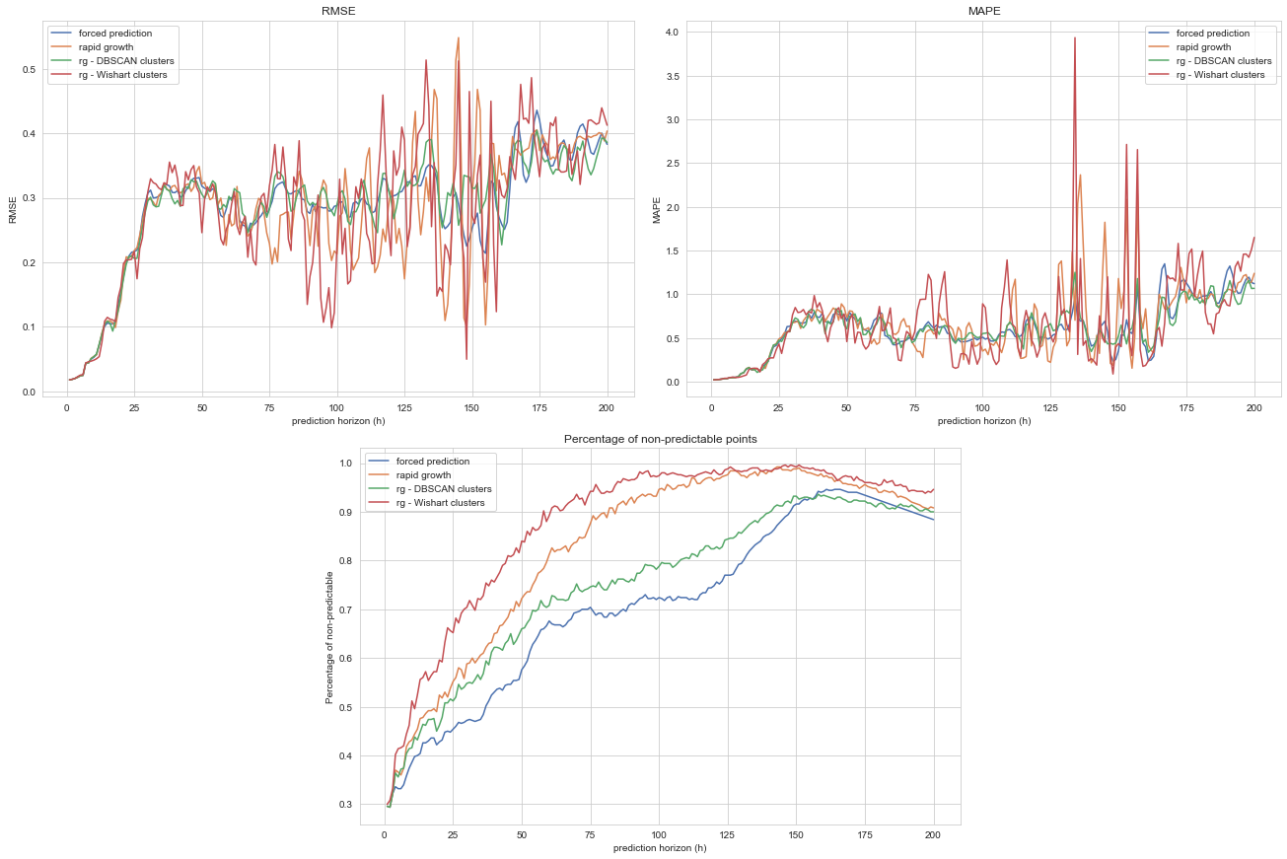


*Рисунок 3. Графики зависимости RMSE, MAPE и количества непрогнозируемых точек от горизонта прогнозирования. Горизонт прогнозирования h=200. Тестовая выборка 500. Кластеризация мотивов — db, 20%. $\varepsilon = 0.01$. Алгоритм вычисления единого прогнозного значения — db. Алгоритмы определения непрогнозируемых точек: fp (синий), rg (желтый), rd (зеленый), rw (красный).*

На рисунке 3 представлены графики зависимости ошибок RMSE, MAPE и количества непрогнозируемых точек от горизонта прогнозирования, которые получены в эксперименте оценки качества работы модели для размера тестовой выборки 500 и максимального горизонта прогнозирования $h = 200$. Для алгоритма вынужденного прогнозирования (fp) непрогнозируемые точки — это точки, для которых алгоритм вычисления единого прогнозного значения (центр наибольшего кластера DBSCAN) не может быть определено.

В среднем ошибки RMSE и MAPE одинаковы для всех алгоритмов, и они не растут экспоненциально с ростом горизонта прогнозирования. Доля непрогнозируемых точек не становится 1 с ростом горизонта прогнозирования. Это означает, что основной алгоритм может прогнозировать некоторые точки тестовой выборки на горизонте прогнозирования более 100. Данные результаты сопоставимы с результатами в статье [14], это говорит о том, что реализация основного алгоритма корректна.

**1.2. Выбор базового алгоритма для алгоритма self-healing**

Алгоритм self-healing имеет в качестве входных параметров множества возможных прогнозных значений и единые прогнозные значения базового алгоритма. Траекторное прогнозирование по сравнению с поточечным имеет множества возможных прогнозных значений большего размера и разброса. Это негативно сказывается на работе алгоритма self-healing, может возникнуть ситуация, похожая на переобучение модели машинного обучения. Больший объем информации о возможных прогнозных значениях, ошибка которых велика, приводит к некачественным результатам работы алгоритма self-healing.

Такие алгоритмы определения непрогнозируемых точек, как быстрый рост разброса (rg) и быстрый рост количества кластеров (rd, rw) были предложены именно для траекторного прогнозирования и показывает хорошие результаты прогнозирования для него. Поэтому было проведено исследование алгоритма определения непрогнозируемых точек, основанное на минимальной доле точек в наибольшем кластере $\gamma$ и максимальном количестве кластеров множества возможных прогнозных значений $N_{max}$ ($lcs$). Исследовались комбинации параметров $\gamma = 0.05, 0.1, 0.2, 0.5, 0.7$ и $N_{max} = 1, 2, 5, 10$, алгоритмы сравнивались с алгоритмами принудительного прогнозирования, быстрого роста разброса и быстрого роста числа кластеров, тестовая выборка 250 точек. Были сделаны следующие наблюдения:

- $N_{max} = 5$ и $N_{max} = 10$ не отличаются (приложение Б, рисунок 14);
- $N_{max} = 2$ и $N_{max} = 5$ почти не отличаются (приложение Б, рисунок 15);
- $\gamma = 0.05, 0.1, 0.2$ не отличаются при равных $N_{max}$ и почти не отличаются от принудительного прогнозирования, который не определяет непрогнозизуемые точки, но алгоритм вычисления единого прогнозного значения db не может его посчитать, если алгоритм кластеризации DBSCAN не находит ни одного кластера, кроме кластера выбросов (приложение Б, рисунок 16);

• если сравнить lcs $\gamma = 0.5,\ N_{max} = 1$ и алгоритм быстрого роста разброса (rg), то получается примерно одинаковое количество непрогнозируемых точек, но ошибка в среднем после $h = 50$ меньше у lcs $\gamma = 0.5,\ N_{max} = 1$ (рисунок 4);

• если сравнить lcs $\gamma = 0.5,\ N_{max} = 5$ и алгоритм принудительного прогнозирования (fp), то они почти не отличаются в терминах количества непрогнозируемых точек и средних ошибок прогнозирования (рисунок 4);

• lcs $\gamma = 0.7,\ N_{max} = 1$ дает слишком много непрогнозируемых на $h \geq 50$, в среднем около 96-99% (приложение Б, рисунок 17).

Исходя из этих наблюдений в финальном эксперименте по оценке качества работы исследуются два варианта: lcs с параметрами $\gamma = 0.5,\ N_{max} = 1$ и lcs с параметрами $\gamma = 0.7,\ N_{max} = 2$. Второй алгоритм по сравнению с первым будет выдавать меньше прогнозируемых точек в базовом алгоритме, но с меньшей ошибкой прогнозирования, таким образом проверяется гипотеза о том, что важнее: большее количество прогнозируемых точек или меньшая ошибка прогнозирования (Рисунок 4).

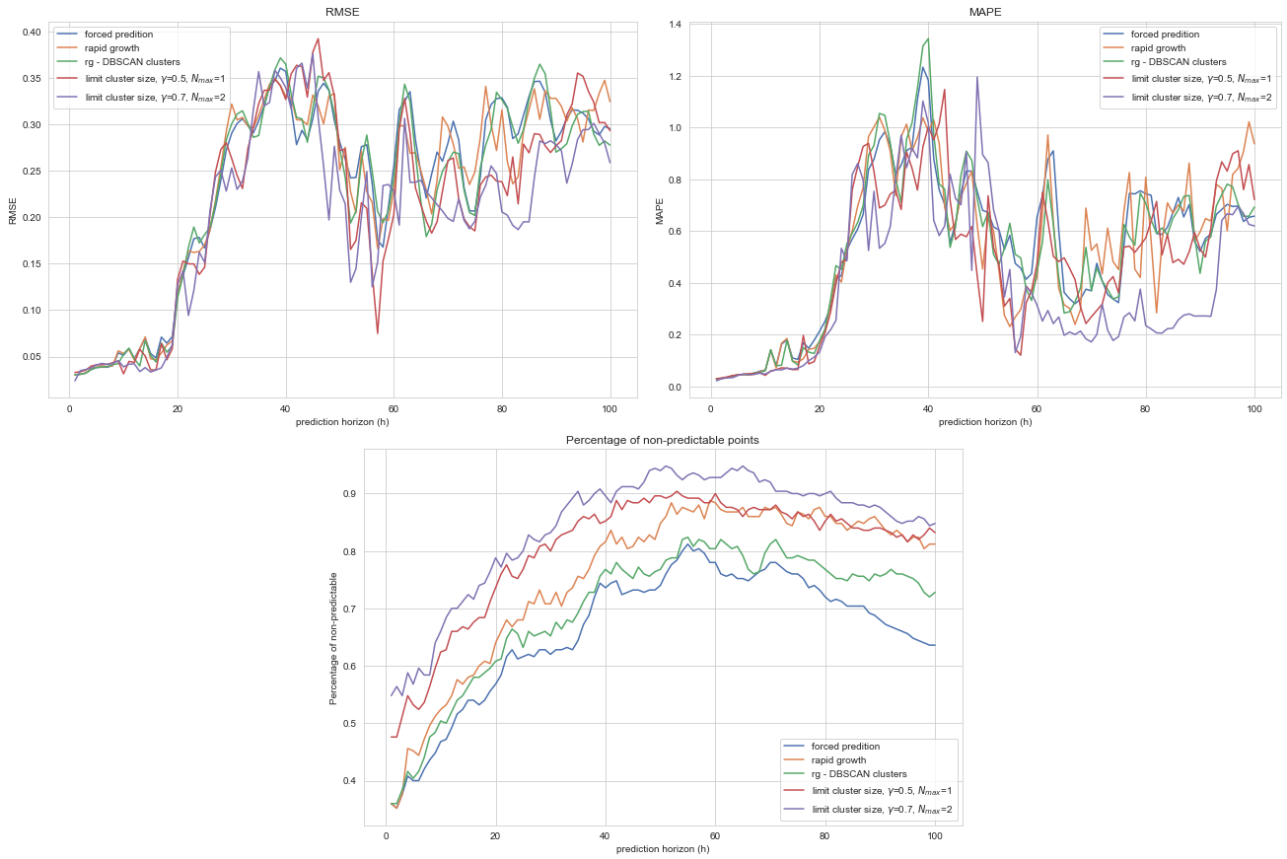


*Рисунок 4. Графики зависимости RMSE, MAPE и количества непрогнозируемых точек от горизонта прогнозирования. Горизонт прогнозирования h=100. Тестовая выборка 250. Кластеризация мотивов — db, 20%. $\varepsilon = 0.01$. Алгоритм вычисления единого прогнозного значения — db. Алгоритмы определения непрогнозируемых точек: fp (синий), rg (желтый), rd (зеленый), lcs_0.5_1 (красный), lcs_0.7_2 (фиолетовый).*

Дополнительные графики данного исследования приведены в приложении Б (рисунки 14-17).

## 2. Эксперимент для временного ряда с выкинутыми точками

### 2.1. Исследование алгоритмов определения непрогнозируемых точек


*Рисунок 5. Графики зависимости количества итераций, количества непрогнозируемых точек, RMSE и MAPE от количества выкинутых точек. Максимальный горизонт прогнозирования h=100. Шаг 5. Кластеризация мотивов — cl_db, $\beta = 20\,\%$. $\epsilon = 0.01$. Представлены следующие комбинации: fp + db (синий), ls + db (желтый), rg + db (зеленый), fp + factor (красный), ls + factor (фиолетовый), rg + factor (коричневый).*

На рисунке 5 представлены графики зависимости количества итераций, количества непрогнозируемых точек, RMSE и MAPE от алгоритма определения непрогнозируемых точек и алгоритма вычисления единого прогнозного значения. Статус всех точек может меняется с прогнозируемой на непрогнозируемую или наоборот. Количество выкинутых точек взято от 1 до 96 с шагом в 5.

Ожидается, что алгоритм self-healing восстановит некоторые выкинутые точки и незначительно изменит оставшиеся точки. То есть ожидается, что график количества непрогнозируемых точек будет линейно возрастать с ростом числа выкинутых точек. Ни один алгоритм этой гипотезе не удовлетворяет. Более того, алгоритм большого разброса определяет как непрогнозируемые 80-100% точек вне зависимости от количества выкинутых точек, из чего можно сделать вывод, что с этими параметрами он не подходит для алгоритма self-healing. Процент непрогнозируемых точек для алгоритма быстрого роста разброса остается на уровне примерно 20%, что опять же не отвечает гипотезе. Для 1, 6, 11 точек количество непрогнозируемых точек становится больше, хотя оставшиеся точки — это истинные значения ряда. Алгоритм взвешенного по итерации среднего (2 алгоритм в пункте 3.2. главы 2) в среднем не улучшает качество прогнозирования.

Можно сделать вывод, что данные алгоритмы определения непрогнозируемых точек не подходят для алгоритма self-healing.

### 2.2. Исследование новых алгоритмов вычисления единого прогнозного значения

Алгоритмы вычисления единого прогнозного значения, перечисленные в пункте 3.2 главы 2, были исследованы на временном ряде с 5 выкинутыми точками. Такой подход был выбран исходя из следующего предположения. Если алгоритм self-healing не восстанавливает с хорошей точностью промежуточные точки, используя настоящие точки временного ряда, то используя прогнозы, которые получены базовым алгоритмом и имеют ненулевую ошибку прогнозирования, промежуточные точки не будут восстановлены с хорошей точностью.

Для проверки восстановления выкинутых точек также были проведены эксперименты с фиксированным статусом начальных точек. Ошибки RMSE и MAPE приведены для всех прогнозируемых точек до горизонта прогнозирования $h = 100$.

*Таблица 1. RMSE, MAPE и количество непрогнозируемых точек в зависимости от алгоритма вычисления единого прогнозного значения. Кластеризация мотивов — db, 20%. $\varepsilon = 0.01$.*

| **Алгоритм определения непрогнозируемых точек** | **Алгоритм вычисления единого прогнозного значения** | **RMSE** | **MAPE** | **Количество непрогнозируемых точек** |
|---|---|---|---|---|
| rd | db | 0.11 | 0.38 | 4 |
| rd | wa | 0.14 | 0.18 | 11 |
| rd | dc | 0.13 | 0.15 | 9 |
| rd | db, factor = 0.5 | 0.12 | 0.41 | 7 |
| rd | db, factor = 0.6 | 0.13 | 0.41 | 9 |

| | | | | |
|---|---|---|---|---|
| rd | db, factor = 0.8 | 0.13 | 0.43 | 10 |
| rd | db, dist | 0.16 | 0.48 | 3 |
| rd | db, pl | 0.14 | 0.44 | 2 |
| Фиксированный статус начальных точек | | | | |
| rd | db | 0.12 | 0.39 | 0 |
| rd | db, factor = 0.4 | 0.17 | 0.18 | 1 |
| rd | db, factor = 0.5 | 0.12 | 0.38 | 0 |
| rd | db, factor = 0.9 | 0.12 | 0.38 | 0 |
| rd | db, dist | 0.12 | 0.39 | 2 |
| rd | db, pl | 0.12 | 0.39 | 1 |

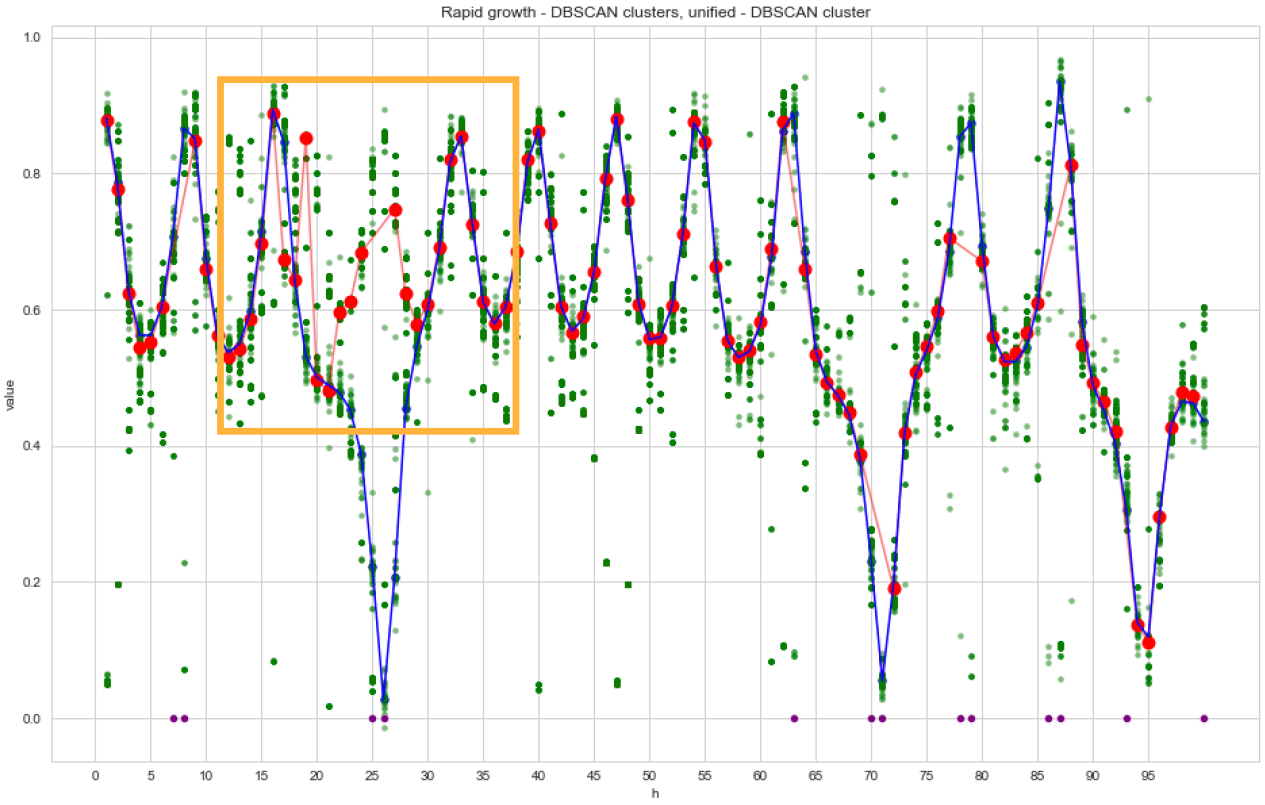


*Рисунок 6. Последняя итерация алгоритма self-healing для эксперимента с 5 выкинутыми точками. Кластеризация мотивов — db, 20%.* $\varepsilon = 0.01$. *Алгоритм вычисления единого прогнозного значения — db. Алгоритмы определения непрогнозируемых точек — rd. Синим обозначен настоящий временной ряд, зеленые точки — возможные прогнозные значения, красные точки и линии — единые прогнозные значения, фиолетовые точки на прямой* $y = 0$ *— непрогнозируемые точки. Оранжевым выделен участок, который не похож на мотивы, встречающиеся в настоящем временном ряде.*

Как можно заметить из результатов в таблице 1, новые алгоритмы вычисления единого прогнозного значения не улучшают качество прогнозирования. В некоторых случаях

уменьшается MAPE, но RMSE не улучшается по сравнению с алгоритмом без учета итерации в вычислении единого прогнозного значения. Это связано с неверной корректировкой настоящих точек на участке $h = [15, 30]$. Также было замечено наличие мотивов, которые не встречаются в настоящем временном ряде (Рисунок 6). Ни один исследованный алгоритм вычисления единого прогнозного значения не смог корректно спрогнозировать данный участок временного ряда.

В приложении Б на рисунках 18-20 находятся иллюстрации к исследованию конкретных отрезков настоящего временного ряда, которые составляют кластеры, центры которых близки к наблюдаемому ряду. Исследование было проведено с целью определить причину неверной корректировки прогнозов на участке $h = [15, 30]$ для точки 24. Кластер, близкий к настоящему значению 24-ей точки ( $\approx 0.4$) был существенно меньше кластера, центр которого находится возле $0.8$. Ошибок в работе алгоритма поиска похожих мотивов найдено не было, поэтому был сделан вывод о том, что такие точки должны являться непрогнозируемыми.

В эксперименте, описанном в пункте 3, будет использован алгоритм factor с параметром 0.7, несмотря на то, что он не дает серьезного улучшения качества прогнозирования на ряде с 5 выкинутыми точками. Это обосновано намерением протестировать гипотезу о том, что значения на более ранних итерациях должны учитываться с бóльшим весом, на выборке бóльшего размера.

### 2.3. Исследование новых алгоритмов определения непрогнозируемых точек

Алгоритмы определения непрогнозируемых точек, перечисленные в пункте 3.3 главы 2, так же сначала были исследованы на временном ряде с 5 выкинутыми точками. Основная проблема, которую должны решать эти алгоритмы, — определение точек на участке $h = [15, 30]$ как непрогнозируемых. Алгоритм большого скачка (*bl*) исследует первые разности временного ряда, если некоторая первая разность больше максимальной в настоящем временном ряде, вторая точка разности является непрогнозируемой. Алгоритм большого скачка между итерациями (*blbi*) не позволяет алгоритму резко корректировать прогнозы, полученные на предыдущих итерациях. Алгоритм странных паттернов (*wp*) исследует полученный после очередной итерации ряд на наличие мотивов, которые не встречаются в настоящем ряде.

*Таблица 2. RMSE, MAPE и количество непрогнозируемых точек в зависимости от алгоритма вычисления единого прогнозного значения. Кластеризация мотивов — db, 20%. $\epsilon = 0.01$.*

| **Алгоритм определения непрогнозируемых точек** | **Алгоритм вычисления единого прогнозного значения** | **RMSE** | **MAPE** | **Количество непрогнозируемых точек** |
|---|---|---|---|---|
| rd | db | 0.11 | 0.38 | 4 |
| rd, bl | db | 0.13 | 0.42 | 2 |
| rd, blbi | db | 0.04 | 0.04 | 7 |
| rd, wp | db | 0.06 | 0.04 | 16 |


*Рисунок 7. Графики зависимости количества итераций, количества непрогнозируемых точек, RMSE и MAPE от количества выкинутых точек. Максимальный горизонт прогнозирования h=100. Шаг 5. Кластеризация мотивов — cl_db, $\beta = 20\,\%$. Алгоритм вычисления единого прогнозного значения — factor = 0.7. Алгоритмы определения непрогнозируемых точек: fp (синий), ls (оранжевый), rg (зеленый), blbi (красный), wp (фиолетовый).*

В таблице 2 представлены результаты исследования новых алгоритмов определения непрогнозируемых точек на временном ряде с 5 выкинутыми точками. Алгоритм большого скачка между итерациями (*blbi*) и алгоритм странных паттернов (*wp*) значительно улучшили качество прогноза в терминах средних ошибок прогнозирования.

На рисунке 7 показано сравнение алгоритмов определения непрогнозируемых точек на эксперименте с выкинутыми точками от 1 до 96 с шагом в 5. В среднем при близком проценте прогнозируемых точек ошибка новых алгоритмов (*blbi* и *wp*) меньше, чем алгоритмов принудительного прогнозирования (*fp*) и быстрого роста разброса (*rg*).

Можно сделать вывод, что новые алгоритмы определения непрогнозируемых точек улучшают качество прогнозирования, поэтому алгоритм большого скачка между итерациями и алгоритм странных паттернов применены в следующем эксперименте.

**3. Исследование алгоритма self-healing при прогнозировании хаотического временного ряда**

Для оценки качества работы алгоритма self-healing был проведен эксперимент по оценке качества работы алгоритма прогнозирования, тестовая выборка — 20 наблюдений. В таблицах 3 и 4 представлены его результаты.

Два базовых алгоритма прогнозирования с алгоритмами определения непрогнозируемых точек lcs $\gamma = 0.5,\ N_{max} = 1$ и lcs $\gamma = 0.7,\ N_{max} = 2$ сравниваются с четырьмя алгоритмами прогнозирования с использованием self-healing: с теми же алгоритмами определения непрогнозируемых точек, но в алгоритме self-healing используются алгоритмы большого скачка между итерациями и странных паттернов.

*Таблица 3. Усредненные по тестовой выборке ошибки RMSE и MAPE, процент непрогнозируемых точек (np, %) для двух базовых алгоритмов и четырех алгоритмов, использующих self-healing. Кластеризация мотивов — db, 20%. $\varepsilon = 0.01$. Горизонт прогнозирования $h = 1, 5, 20$.*

| **Тип алгоритма** | **Алгоритм ОНТ** | **Алгоритм ВЕПЗ** | **h = 1** | | | **h = 5** | | | **h = 20** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | **RMSE** | **MAPE** | **np, %** | **RMSE** | **MAPE** | **np, %** | **RMSE** | **MAPE** | **np, %** |
| s | lcs_0.5_1 | db | 0.01 | 0.01 | 25 | 0.02 | 0.17 | 30 | 0.13 | 0.05 | 85 |
| s, sh | lcs_0.5_1, wp | db, factor = 0.7 | 0.01 | 0.01 | 5 | 0.02 | 0.02 | 5 | 0.06 | 0.05 | 10 |
| s, sh | lcs_0.5_1, blbi | db, factor = 0.7 | 0.01 | 0.01 | 5 | 0.02 | 0.02 | 5 | 0.06 | 0.05 | 10 |
| s | lcs_0.7_2 | db | 0.01 | 0.01 | 35 | 0.02 | 0.02 | 30 | 0.13 | 0.17 | 85 |
| s, sh | lcs_0.7_2, wp | db, factor = 0.7 | 0.01 | 0.01 | 5 | 0.02 | 0.02 | 5 | 0.06 | 0.14 | 15 |
| s, sh | lcs_0.7_2, blbi | db, factor = 0.7 | 0.01 | 0.01 | 5 | 0.02 | 0.02 | 5 | 0.06 | 0.05 | 15 |

*Таблица 4. Усредненные по тестовой выборке ошибки RMSE и MAPE, процент непрогнозируемых точек (np, %) для двух базовых алгоритмов и четырех алгоритмов, использующих self-healing. Кластеризация мотивов — db, 20%. $\varepsilon = 0.01$. Горизонт прогнозирования $h = 50, 70, 100$.*

| **Тип алгоритма** | **Алгоритм ОНТ** | **Алгоритм ВЕПЗ** | **h = 50** | | | **h = 70** | | | **h = 100** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | **RMSE** | **MAPE** | **np, %** | **RMSE** | **MAPE** | **np, %** | **RMSE** | **MAPE** | **np, %** |
| s | lcs_0.5_1 | db | 0.02 | 0.05 | 90 | 0.2 | 0.14 | 30 | 0.24 | 0.17 | 50 |
| s, sh | lcs_0.5_1, wp | db, factor = 0.7 | 0.07 | 0.07 | 30 | 0.26 | 0.2 | 0 | 0.26 | 0.21 | 0 |
| s, sh | lcs_0.5_1, blbi | db, factor = 0.7 | 0.25 | 0.18 | 5 | 0.26 | 0.2 | 0 | 0.26 | 0.21 | 0 |
| s | lcs_0.7_2 | db | — | — | 100 | 0.22 | 0.17 | 25 | 0.08 | 0.1 | 45 |
| s, sh | lcs_0.7_2, wp | db, factor = 0.7 | 0.2 | 0.15 | 20 | 0.26 | 0.2 | 0 | **0.26** | **0.2** | **0** |
| s, sh | lcs_0.7_2, blbi | db, factor = 0.7 | 0.27 | 0.2 | 5 | 0.26 | 0.2 | 0 | **0.26** | **0.2** | **0** |

На горизонтах прогнозирования $h = [1, 5]$ средние ошибки RMSE и MAPE прогнозирования базовых и self-healing алгоритмов не отличаются, однако количество непрогнозируемых точек существенно ниже. На горизонте прогнозирования $h = 20$ средние ошибки прогнозирования ниже в два раза, тогда как количество непрогнозируемых точек существенно сократилось. На горизонте прогнозирования $h = 50$ средняя ошибка

существенно больше, но количество непрогнозируемых точек сократилось. На горизонтах прогнозирования $h = [70, 100]$ средняя ошибка больше, однако количество непрогнозируемых точек для алгоритма self-healing равно нулю.

Дополнительно были проведены непараметрические статистические тесты для оценки различия между ошибками прогнозирования RMSE, MAPE и количеством непрогнозируемых точек. Использовался U-критерий Манна — Уитни, так как он подходит для малых выборок.

Нулевая гипотеза U-критерия Манна — Уитни — группы извлечены из одной совокупности, альтернативная гипотеза — наличие существенного различия между уровнем признака в рассматриваемых выборках.

*Таблица 5. p-значения для U-критерия Манна — Уитни, сравнение двух базовых алгоритмов прогнозирования с четырьмя алгоритмами self-healing.*

| **RMSE** | lcs_0.5_1, wp | lcs_0.5_1, blbi | lcs_0.7_2, wp | lcs_0.7_2, blbi |
|---|---|---|---|---|
| lcs_0.5_1 | 0.94 | 0.59 | 0.82 | 0.59 |
| lcs_0.7_2 | 1.0 | 1.0 | 1.0 | 1.0 |
| **MAPE** | lcs_0.5_1, wp | lcs_0.5_1, blbi | lcs_0.7_2, wp | lcs_0.7_2, blbi |
| lcs_0.5_1 | 0.94 | 0.59 | 0.94 | 0.59 |
| lcs_0.7_2 | 1.0 | 1.0 | 1.0 | 1.0 |
| **Процент непрогнозируемых точек** | lcs_0.5_1, wp | lcs_0.5_1, blbi | lcs_0.7_2, wp | lcs_0.7_2, blbi |
| lcs_0.5_1 | 0.009 | 0.002 | 0.002 | 0.002 |
| lcs_0.7_2 | 0.008 | 0.008 | 0.008 | 0.008 |

В таблице 5 приведены p-значения статистик, полученных при сравнении ошибок RMSE и MAPE При уровне значимости $\alpha = 0.05$ для ошибок RMSE и MAPE принимается нулевая гипотеза, то есть ошибки RMSE и MAPE статистически не отличаются при сравнении базового алгоритма и алгоритма self-healing. Однако для процента непрогнозируемых точек принимается альтернативная гипотеза, что означает статистически значимое различие процента непрогнозируемых точек в зависимости от типа алгоритма.

# Заключение

В данном исследовании разработан и проанализирован алгоритм self-healing для прогнозирования с помощью кластеризации. В ходе работы изучена предметная область, реализован базовый алгоритм прогнозирования с помощью кластеризации, разработан алгоритм self-healing и предложено несколько новых алгоритмов определения непрогнозируемых точек и алгоритмов вычисления единого прогнозного значения. Алгоритм self-healing применен для прогнозирования хаотического временного ряда Лоренца, на основе полученных результатов вычислительных экспериментов сделаны выводы об эффективности алгоритма self-healing и новых алгоритмов определения непрогнозируемых точек и алгоритмов вычисления единого прогнозного значения.

Для базового алгоритма наиболее подходящим алгоритмом определения непрогнозируемых точек с учетом дальнейшего применения алгоритма self-healing оказался алгоритм ограничения на минимальный размер наибольшего кластера и на максимальное количество кластеров с параметрами $\gamma = 0.5,\ N_{max} = 1$ и с параметрами $\gamma = 0.7,\ N_{max} = 2$. Первый алгоритм в среднем определяет больше прогнозируемых точек, чем второй, однако имеет бóльшую среднюю ошибку прогнозирования, но это не сильно сказывается на конечном качестве прогнозирования.

В данной работе предложены несколько новых алгоритмов вычисления единого прогнозного значения. Они не повлияли значительно на качество прогнозирования в эксперименте с 5 выкинутыми точками. Их исследование помогло выявить проблему неверной корректировки прогнозов, полученных на предыдущих итерациях. Для ее решения предложены несколько новых алгоритмов определения непрогнозируемых точек, хорошие результаты показали алгоритм большого скачка между итерациями и алгоритм странных паттернов.

В ходе проведения эксперимента по оценке качества работы алгоритма self-healing и его сравнении с базовым алгоритмом прогнозирования были получены следующие результаты. Средние ошибки прогнозирования RMSE и MAPE базового алгоритма и алгоритма self-healing для соответствующих горизонтов прогнозирования согласно U-критерию Манна — Уитни статистически не отличаются. Однако этот критерий показал статистически значимое различие процента непрогнозируемых точек в зависимости от типа алгоритма. Другими словами, в среднем больше точек являются прогнозируемыми при остающихся почти на том же уровне ошибках прогнозирования.

В качестве дальнейшей работы можно провести эксперименты на выборках большего размера и исследование гиперпараметров алгоритма self-healing и алгоритмов определения непрогнозируемых точек и алгоритмов вычисления единого прогнозного значения, которые уже показали хорошее качество прогнозирования. Также можно провести исследование качества прогнозирования алгоритма self-healing на других хаотических временных рядах, например, на финансовых.

## Список источников


1. Малинецкий, Г. Г., Потапов, А. Б. Современные проблемы нелинейной динамики / Г. Г. Малинецкий, А. Б. Потапов — М.: Эдиториал УРСС, 2000. — 336 с.
2. Meng X., Yang T. Entanglement-Structured LSTM Boosts Chaotic Time Series Forecasting. / Meng X., Yang T. // Entropy (Basel) — 2021 Nov 11 — 23(11) — 1491. URL: https://doi.org/10.3390/e23111491 (дата обращения: 11.04.2022)
3. Y. Bao, T. Xiong, Z. Hu, Multi-step-ahead time series prediction using multiple-output support vector regression / Y. Bao, T. Xiong, Z. Hu // Neurocomputing — 2014. — 129. — 482-493. URL: https://doi.org/10.1016/j.neucom.2013.09.010 (дата обращения: 11.04.2022)
4. Assaad M., Boné R., Cardot H. Predicting Chaotic Time Series by Boosted Recurrent Neural Networks. / Assaad M., Boné R., Cardot H. // Lecture Notes in Computer Science — 2006. — 4233. — 831-840. URL: https://link.springer.com/chapter/10.1007/11893257_92 (дата обращения: 11.04.2022)
5. Ben Taieb, Souhaib and Bontempi, Gianluca and Atiya, Amir, Sorjamaa, Antti. A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition / Ben Taieb, Souhaib and Bontempi, Gianluca and Atiya, Amir, Sorjamaa, Antti. // Expert Systems with Applications. — 2011. — 39. URL: https://doi.org/10.1016/j.eswa.2012.01.039 (дата обращения: 11.04.2022)
6. A. Sorjamaa, J. Hao, N. Reyhani, Y. Ji, A. Lendasse. Methodology for long-term prediction of time series. / A. Sorjamaa, J. Hao, N. Reyhani, Y. Ji, A. Lendasse. // Neurocomputing — October 2007. — 70(16-18) — 2861–2869. URL: https://doi.org/10.1016/j.neucom.2006.06.015 (дата обращения: 11.04.2022)
7. G. Bontempi. Long term time series prediction with multi-input multi-output local learning. / G. Bontempi. // In Proceedings of the 2nd European Symposium on Time Series Prediction (TSP), ESTSP08 — 2008. — pages 145–154. URL: https://www.researchgate.net/publication/228659313_Long_term_time_series_prediction_with_multi-input_multi-output_local_learning (дата обращения: 11.04.2022)
8. S. B. Taieb, G. Bontempi, A. Sorjamaa and A. Lendasse. Long-term prediction of time series by combining direct and MIMO strategies. / S. B. Taieb, G. Bontempi, A. Sorjamaa and A. Lendasse // 2009 International Joint Conference on Neural Networks — 2009. — pp. 3054-3061. URL: https://doi.org/10.1109/IJCNN.2009.5178802 (дата обращения: 11.04.2022)

9. Rohitash Chandra, Shaurya Goyal, Rishabh Gupta. Evaluation of deep learning models for multi-step ahead time series prediction. / Rohitash Chandra, Shaurya Goyal, Rishabh Gupta. // IEEE Access — 2021. — 9. — 83105-83123. URL: https://doi.org/10.1109/ACCESS.2021.3085085 (дата обращения: 11.04.2022)

10. William Gilpin. Deep reconstruction of strange attractors from time series / William Gilpin // NeurIPS (Neural Information Processing Systems) — 2020. URL: https://doi.org/10.48550/arXiv.2002.05909 (дата обращения: 12.04.2022)

11. Sugihara, G. & May, R. M. Nonlinear forecasting as a way of distinguishing chaos from measurement error in time series. Nature 344, 734–741 (1990).

12. Takens F. Detecting strange attractors in turbulence // Lecture Notes in Mathematics, Vol. 898, Springer-Verlag, Berlin, 1980, pp. 366–381; and in Dynamical System in Turbulence, Warlock, 1980, Eds. D. Rand and L. S. Young.

13. V.A. Gromov, E.A. Borisenko. Chaotic time series prediction and clustering methods / V.A. Gromov, E.A. Borisenko // Neural Computing and Appl. 2 — 2015 — pp. 307–315.

14. V.A. Gromov, P.S. Baranov, A. Tsybakin, Prediction After a Horizon of Predictability: Non-Predictable Points and Partial Multi- Step Prediction for Chaotic Time Series / V.A. Gromov, P.S. Baranov, A. Tsybakin // Neurocomputing — 2020. URL: https://www.researchgate.net/publication/347239842_Prediction_After_a_Horizon_of_Predictability_Non-Predictable_Points_and_Partial_Multi-Step_Prediction_for_Chaotic_Time_Series (дата обращения: 11.04.2022)

15. D. Wishart, A numerical classification methods for deriving natural classes / Nature 221 — 1969 — pp. 97–98.

16. A.V. Lapko, S.V. Chentsov. Nonparametric Information Processing Systems / Nauka, Novosibirsk — 2000.

17. Kostenetskiy P.S., Chulkevich R.A., Kozyrev V.I. HPC Resources of the Higher School of Economics // Journal of Physics: Conference Series — 2021 — Vol. 1740 — No. 1. P. 012050.

18. Ester, M., H. P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. / Ester, M., H. P. Kriegel, J. Sander, and X. Xu // In: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, OR — AAAI Press — pp. 226-231 — 1996.