Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«Московский физико-технический институт (государственный университет)»
Факультет Аэрофизики и Космических Исследований
Кафедра Логистические Системы и Технологии

КУЗЬМИНА Антонина Ильинична

Математическое моделирование конвейера принятия торговых решений трейдером фондовой биржи

ДИПЛОМНАЯ РАБОТА

Научный руководитель:

Москва, 2016

Содержание

Введение

В последнее время большое количество людей занимается торговлей на бирже — по некоторым оценкам, около 800 тысяч человек. О популярности этой сферы деятельности свидетельствует также и объем торгов на московской бирже, растущий день ото дня и составивший 4.65 трлн рублей 16 декабря 2014 года.

В то же время, игра на бирже является очень рискованным видом деятельности — до 80% участников торгов терпят убытки. Поэтому трейдер никогда не принимает решений по наитию, а всегда использует холодный расчет. Вследствие этого, как правило, большинство торговых стратегий довольно легко формализуемы.

Автоматические торговые системы имеют большое количество преимуществ по сравнению с трейдером-человеком. Самые значительные из них заключаются в следующем:

- Торговый робот может торговать круглосуточно, не отвлекаясь на еду и сон.
- Торговый робот не отнимает ценное время человека — можно запустить одновременно несколько роботов и, в то же время, заниматься созданием новых стратегий.
- Торговый робот принимает все решения в строгом соответствии с логикой алгоритма, он готов терпеть просадки и не

берет на себя лишний риск, что очень важно в условиях высоковолатильного рынка.

- Торговый робот способен принимать решения гораздо быстрее любого человека — если трейдеру для совершения сделки требуется не менее 0.5 секунды, робот способен совершить сделку уже через 2-3 мкс.

Объектом исследования в данной работе является конвейер принятия решений трейдером фондовой биржи.

Основной целью данной работы является разработка математической модели конвейера принятия решений трейдером фондовой биржи, а также ее практическая реализация и проверка на реальных данных как всей моделирующей программы в целом, так и ее отдельных частей.

Для достижения цели настоящей работы поставлены следующие задачи:

- Исследование существующих методов моделирования работы трейдера.

- Исследование существующих открытых источников финансовой информации (котировок различных активов, а также торговых индикаторов) и принятие решения. об использовании этих данных для построения и тестирования модели

конвейера принятия торговых решений трейдером фондовой биржи

- Исследование существующих открытых источников информации о фигурах технического анализа для использования этих данных при построении модели.

- Изучение существующей научной литературы по вопросу поиска известных паттернов в истории котировок финансовых инструментов.

- Выбор метрик расстояния между свечами для использования при реализации алгоритма динамического искажения времени.

- Реализация алгоритма динамического искажения времени для поиска известного паттерна в истории котировок торгового инструмента в среде программирования Microsoft Visual Studio с использованием языка программирования C#.

- Тестирование программы поиска известного паттерна в истории котировок торгового инструмента на реальных исторических данных с использованием методов модульного и функционального тестирования.

- Определение качества работы программы поиска известного паттерна на реальных исторических данных посредством

проведения слепого исследования, а также сравнение достигнутого уровня качества работы при использовании различных метрик расстояния.

- Изучение существующих методов кластерного анализа.

- Реализация различных методов кластеризации для поиска новых паттернов в истории котировок финансовых инструментов в среде программирования Microsoft Visual Studio с использованием языка программирования C#.

- Тестирование программы поиска новых паттернов в истории котировок торгового инструмента с применением алгоритмов кластеризации на реальных исторических данных с использованием методов модульного и функционального тестирования.

- Определение качества работы программы поиска новых паттернов в истории котировок торгового инструмента с применением алгоритмов кластеризации на реальных исторических данных посредством проведения слепого исследования, а также сравнение качества работы различных методов кластеризации между собой.

- Реализация торговой системы, основанной на использовании найденных фигур технического анализа, в среде программи-

рования Microsoft Visual Studio с использованием языка программирования C#.

- Тестирование торговой системы, основанной на использовании найденных фигур технического анализа, на реальных исторических данных и определение важнейших параметров этой торговой системы (прибыль, просадка и тд.).

- Реализация алгоритма-советника для торговой платформы MetaTrader для проверки торговой системы на демо-счете в режиме реального времени.

Теоретической основой исследования явились положения и концепции, представленные в работах отечественных и зарубежных авторов по проблемам:

- Численной оптимизации.

- Поиска паттернов во временных рядах.

- Алгоритма динамического искажения времени.

- Классической и вероятностной постановки задач машинного обучения.

- Эвристических, статистических и иерархических методов кластеризации, в том числе, с использованием нейронных сетей.

- Переобучения и мультиколлинеарности, а также методам борьбы с этими проблемами: регуляризации, выделению главных компонент и др.

- Сравнения качества работы различных алгоритмов машинного обучения.

- Технического анализа.

Вопросы, рассматриваемые в данной работе, нашли отражение в трудах таких классических авторов, как Колмогоров, Вапник, Червоненкис, Закс и Стоун.

Среди современных ученых схожими проблемами занимаются Воронцов, Горбань, Халл, Алексис, Бушерон, Тибширани, Румельхарт, Носедал, Райт, Куликов.

Работа состоит из введения, трех глав, заключения и списка литературы.

В первой главе приведены экономические термины, необходимые для понимания работы, описаны этапы конвейера принятия решений трейдером фондовой биржи, а также приведена формальная постановка задач, рассматривающихся в работе.

Вторая глава представляет собой теоретическое введение к описанию экспериментов, проведенных в рамках данной работы. В нем описаны алгоритмы, примененные при написании программы: алгоритм динамического искажения времени (а также его модифи-

кация, алгоритм derivative dynamic time warping, и используемые метрики расстояния в пространстве историй котировок) и алгоритмы кластеризации (эвристические, статистические и иерархические методы кластеризации, а также самоорганизующиеся карты Кохонена). Приведены основные теоремы, связанные с гарантиями сходимости используемых методов.

В третьей главе описываются результаты, полученные в ходе практической реализации алгоритмов и тестирования их на реальных исторических данных.

Глава 1 Постановка задач

1.1 Необходимые термины

Ниже приведены определения финансовых терминов, использующиеся в работе.

Актив — некоторая сущность, которая может быть куплена или продана в любой момент времени по цене, соответствующей этому моменту времени. Цены покупки и продажи актива в один и тот же момент времени не обязаны совпадать.

Тик — сделка купли-продажи, произошедшая на бирже. Характеризуется моментом времени, ценой и объемом. Объем сделки — количество элементарных единиц актива, которые были проданы продавцом и куплены покупателем.

Свеча — элемент данных, представляющий собой консолидированную информацию об изменении цены актива в некоторый промежуток времени. Как правило, свеча включает в себя 4 величины: цену открытия интервала (цена первого тика из временного интервала), цену закрытия (цена последнего тика из временного интервала), а также максимальную и минимальную цены тиков из рассматриваемого временного интервала. Нередко также в состав свечи включают общий объем всех сделок, произошедших в течение рассматриваемого промежутка времени, однако в данной работе эта величина не используется.

Торговая система — алгоритм, совершающий сделки на бирже по определенным математическим правилам. Может иметь параметры, влияющие на поведение системы.

Сделка покупки актива — сделка по покупке-продаже актива, в которой рассматриваемая торговая система выступает в качестве покупателя.

Сделка продажи актива — сделка по покупке-продаже актива, в которой рассматриваемая торговая система выступает в качестве продавца.

Закрытая сделка — пара сделок с совпадающими объемами, состоящая из сделки по покупке актива и сделки по продаже актива.

Ряд данных — последовательность данных о цене актива за определенный промежуток времени. Как правило, включает в себя

информацию обо всех свечах этого актива за данный промежуток времени.

Прибыль закрытой сделки — разность цен сделок продажи и покупки этой закрытой сделки, умноженная на объем этих сделок.

Прибыль торговой системы за некоторый интервал времени. Обозначим n общее число закрытых сделок торговой системы за рассматриваемый период. Обозначим $p_i,\ i = 1, ..., n$ прибыль $i-$ой закрытой сделки. Тогда прибылью торговой системы называется величина $profit = \sum_{i=1}^{n} p_i$.

Просадка торговой системы за некоторый интервал времени. Обозначим n общее число закрытых сделок торговой системы за рассматриваемый период. Обозначим $p_i,\ i = 1, ..., n$ прибыль $i-$ой закрытой сделки. Тогда просадкой торговой системы называется величина

$$drawdown = \max_{i=1,...,n} \left(\max_{k=1,...,i} \sum_{j=1}^{k} p_j - \sum_{j=1}^{i} p_j \right)$$

Функционал качества торговой системы — некоторая функция, характеризующая качество торговой системы. Как правило, для ее вычисления используется последовательность закрытых сделок торговой системы. Типичные примеры функционала качества — $profit$ и $profit/drawdown$.

1.2 Этапы конвейера принятия решений трейдером фондовой биржи

В данной работе рассматривается трейдер фондовой биржи, принимающий торговые решения на основе фигур технического анализа. Конвейер принятия решений в этом случае включает в себя следующие этапы:

1. Поиск закономерностей фондового рынка
    (a) Выделение типичных фигур технического анализа
    (b) Определение информативности каждой фигуры технического анализа
2. Создание торговой стратегии
    (a) Поиск фигур технического анализа в биржевых данных в режиме реального времени
    (b) Принятие торгового решения и совершение сделки
    (c) Оптимизация торговой стратегии
    (d) Запуск автоматической торговой системы

В данной работе рассматриваются все этапы этого конвейера, однако основное внимание уделяется трем задачам: задаче поиска известного паттерна в истории котировок, задаче кластеризации в

пространстве фрагментов историй котировок и задаче автоматизированного построения эффективной торговой стратегии.

1.3 Постановка задач

1.3.1 Задача поиска известного паттерна в истории котировок

Введем следующие обозначения:

$X^l = \{x_i\}_{i=1}^l$ - набор данных об истории котировок торгового инструмента, то есть,

- $x_i.open$ — цена открытия i-ой свечи;
- $x_i.high$ — цена максимума i-ой свечи;
- $x_i.low$ — цена минимума i-ой свечи;
- $x_i.close$ — цена закрытия i-ой свечи.

$P^n = \{p_i\}_{i=1}^n$ — паттерн для поиска в истории котировок, в виде последовательности свечей.

$Y \in \{0,1\}^{l \times l}$ — набор известных ответов (ground truth). $y_{ij} = 1$ обозначает, что подстрока котировок с i-ой по j-ую свечу включительно является вхождением искомого паттерна, а $y_{ij} = 0$ - что не является.

$a : \{1,\ldots,l\} \times \{1,\ldots,l\} \to \{0,1\}$ — построенный алгоритм. $a(i,j) = 1$ обозначает, что подстрока котировок с i-ой по j-ую свечу

включительно является вхождением искомого паттерна, а $a(i,j) = 0$ - что не является.

$L(a,y) = \begin{cases} 0, a = y; \\ 1, a \neq y \end{cases}$ — функция потерь, хакактеризующая величину ошибки алгоритма, выдавшего ответ a, на объекте с верным ответом y.

$Q(a, X^l, Y^{l \times l}) = \frac{1}{l^2} \sum_{i=1}^{l} \sum_{j=1}^{l} L(a(i,j), y_{ij})$ — функционал качества алгоритма a.

Задачу поиска паттерна в истории котировок можно сформулировать следующим образом:

$$\mu(X^l, Y^{l \times l}) = \underset{a \in A}{\operatorname{argmin}} Q(a, X^l, Y^{l \times l}). \tag{1.3.1}$$

1.3.2 Задача кластеризации в пространстве фрагментов историй торгов

1.3.3 Задача построения эффективной торговой стратегии

Глава 2 Теоретическое введение

2.1 Алгоритм динамического искажения времени

Динамическое искажение времени (DTW) является широко известной техникой поиска оптимального соответствия между двумя временными последовательностями (рис. 2.1.1). Грубо говоря, последовательности искажаются нелинейным образом для достижения максимального соответствия. Впервые DTW был применен для сравнения различных паттернов при распознавании голоса [1]. Помимо анализа данных [2, 3, 4], DTW также был успешно применен в таких областях, как распознавание жестов [5], робототехника [6], распознавание речи [7], производство [8] и медицина [9].

В этой главе вводятся и обсуждаются основные идеи классического алгоритма DTW, а также приводятся его различные модификации, касающиеся как локального, так и глобального поведения алгоритма.

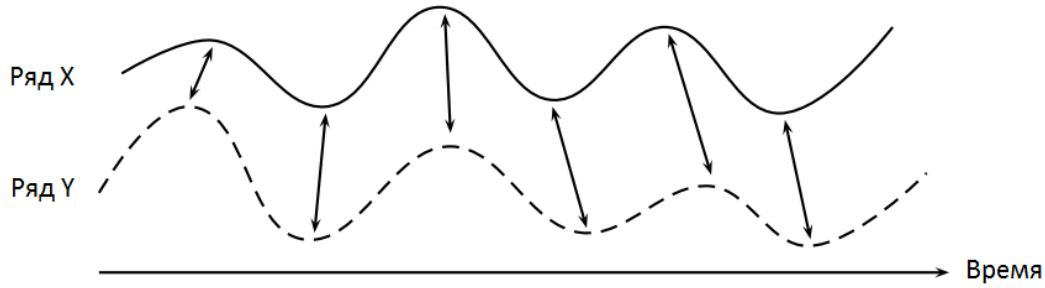

Рис. 2.1.1: Сопоставление двух временных рядов. Соответствующие пары точек указаны стрелками.

2.1.1 Базовый алгоритм динамического искажения времени

Целью DTW является сравнение двух последовательностей данных $X = (x_i)_{i=1}^{n}$ длины n и $Y = (y_i)_{i=1}^{m}$ длины m. Эти последовательности данных могут быть как дискретными сигналами (временными рядами), так и, в более общем случае, последовательностями любых объектов, расположенных через одинаковые промежутки времени друг от друга. Обозначим пространство признаков F. Тогда $x_i \in F$, для $i \in [1, \ldots, n]$, и $x_i \in F$, для $i \in [1, \ldots, m]$. Для сравнения двух различных элементов $x, y \in F$, требуется наличие локальной меры стоимости, также называемой локальной мерой расстояния, определяющейся функцией

$$c : F \times F \to \mathbb{R}_{\geq 0}. \tag{2.1.1}$$

Обычно, c такова, что ее значения невелики, если x и y близки друг к другу (или в некотором смысле похожи), и велики в про-

тивном случае. Посредством вычисления метрики расстояния для каждой пары элементов двух данных последовательностей получается матрица стоимости (рис. 2.1.2), определенная следующим образом:

$$C \in \mathbb{R}^{n \times m},\ c_{ij} = c(x_i, y_j). \tag{2.1.2}$$

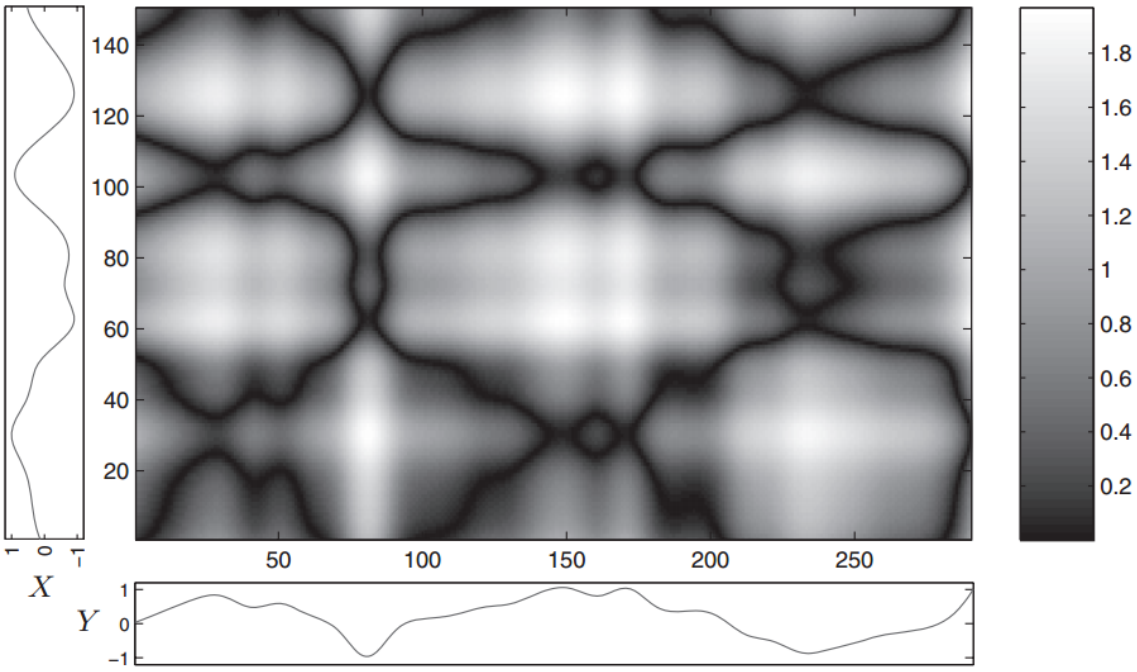

Рис. 2.1.2: Матрица стоимости для двух веществennозначных последовательностей X (вертикальная ось) и Y (горизонтальная ось) с использованием манхеттенского расстояния в качестве локальной меры расстояния. Регионы с меньшей стоимостью показаны черным цветом, с наибольшей - белым.

Таким образом, цель работы алгоритма DTW заключается в нахождении соответствия между последовательностями X и Y,

имеющего минимальную суммарную стоимость. Интуитивно, такое оптимальное соответствие должно проходить вдоль "долин"низкой стоимости в матрице стоимости (рис. 2.1.4). Следующие определения формализуют понятие соответствия.

Определение 2.1.1. *(n, m)-искажающий путь (также называемый просто искажающим путем, если n и m очевидны из контекста) — это последовательность $p = (p_1, \ldots, p_l)$, где $p_k = (i_k, j_k) \in [1, \ldots, n] \times [1, \ldots, m]$ для $i \in [1, \ldots, l]$, удовлетворяющая следующим трем условиям.*

1. *Граничное условие: $p_1 = (1, 1)$ и $p_l = (n, m)$.*
2. *Условие монотонности: $i_1 \leq i_2 \leq \cdots \leq i_l, j_1 \leq j_2 \leq \cdots \leq j_l$.*
3. *Условие размера шага: $p_{k+1} - p_k \in \{(0, 1), (1, 0), (1, 1)\}$ для $k \in [1, \ldots, l - 1]$.*

Условие монотонности является прямым следствием условия размера шага, однако, все же упомянуто для наглядности. (n, m)-искажающий путь $p = (p_1, \ldots, p_l)$ определяет соответствие между двумя последовательностями $X = (x_1, \ldots, x_n)$ и $Y = (y_1, \ldots, y_m)$. Элементу x_{i_k} здесь соответствует элемент y_{j_k}. Смысл граничного условия заключается в требовании соответствия между первыми и последними элементами последовательностей, другими словами, соответствие устанавливается между целыми последовательностя-

ми X и Y, а не между некоторыми их составными частями. Условие монотонности отражает требование реалистичности времени: если некоторый элемент x_i идет перед элементом x_j, то же самое должно выполняться для соответствующих им элементов последовательности Y, и наоборот. Наконец, условие размера шага является, в некотором роде, требованием непрерывности: каждый элемент каждой из последовательностей должен входить хотя бы в одну пару оптимального пути, и, в то же время, путь не должен иметь повторений в смысле одновременного равенства обеих компонент. Рисунок 2.1.3 иллюстрирует эти три условия.

Определение 2.1.2. *Полной стоимостью искажающего пути* $p = (p_1, \ldots, p_l)$ *между последовательностями* $X = (x_1, \ldots, x_n)$ *и* $Y = (y_1, \ldots, y_m)$ *относительно меры стоимости* c *называется величина*

$$c_p(X, Y) = \sum_{k=1}^{l} c(x_{i_k}, y_{j_k}); \tag{2.1.3}$$

Определение 2.1.3. *Оптимальным искажающим путем* $p^*(X, Y)$ *называется искажающий путь, имеющий наименьшую полную стоимость.*

Определение 2.1.4. *Расстоянием динамического искажения времени между последовательностями* $X = (x_1, \ldots, x_n)$ *и* $Y = (y_1, \ldots, y_m)$

относительно меры стоимости c называется величина

$$DTW(X, Y, c) = c_{p^*}(X, Y) = \min\{c_p(X, Y) | p - (n, m) - \textit{искажающий путь}\}. \quad (2.1.4)$$

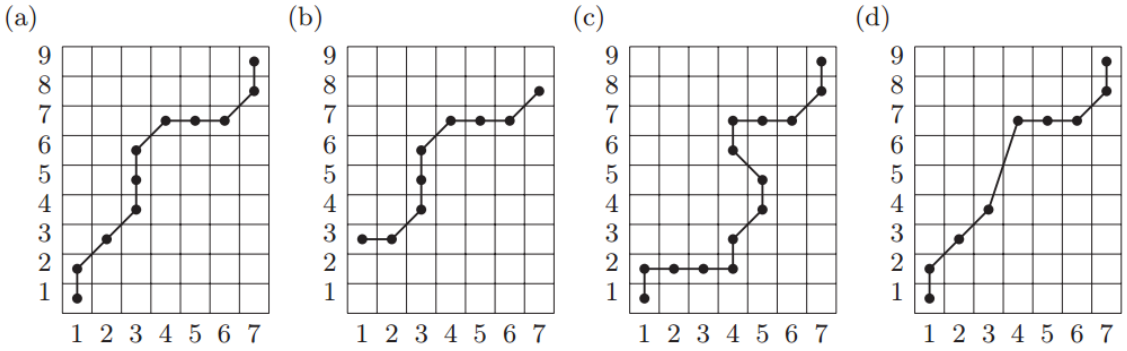

Рис. 2.1.3: Иллюстрация некоторых вариантов искажающего пути для двух последовательностей: X из 9 элементов и Y из 7 элементов. **(a)** Допустимый искажающий путь, удовлетворяющий условиям 1 - 3. **(b)** Граничное условие нарушено. **(c)** Условие монотонности нарушено. **(d)** Условие размера шага нарушено.

Заметим, что расстояние DTW является корректно определенным даже в случае наличия нескольких искажающих путей минимальной полной стоимости. Кроме того, легко доказать, что расстояние DTW является симметричным, если локальная мера c является симметричной. Однако, расстояние DTW может не являться положительно определенным даже в случае, если для исходной меры это так. Более того, расстояние DTW может не удовлетво-

рять неравенству треугольника даже в случае, если c является метрикой. Этот факт иллюстрируется следующим примером.

Пример 2.1.1. *Пусть $F = (\alpha, \beta, \gamma)$ — пространство признаков из трех элементов, $c(x, y) = 0$, если $x = y$ и $c(x, y) = 1$, если $x \neq y$ — локальная мера расстояния. Очевидно, c является метрикой над F и удовлетворяет неравенству треугольника. Теперь рассмотрим $X = (\alpha, \beta, \gamma)$, $Y = (\alpha, \beta, \beta, \gamma)$ и $Z = (\alpha, \gamma, \gamma)$. Тогда $DTW(X, Y, c) = 0$, $DTW(X, Z, c) = 1$, но $DTW(Y, Z, c) = 2$.*

Оптимальный искажающий путь p^* может быть найден перебором всех возможных искажающих путей, однако такой метод имеет экспоненциальную сложность. Далее в этом разделе будет приведен алгоритм, основанный на методе динамического программирования, и имеющий сложность $O(nm)$.

Обозначим $X(1 : k) = (x_1, \ldots, x_k)$ для $k \in [1, \ldots, n]$ и $Y(1 : k) = (y_1, \ldots, y_k)$ для $k \in [1, \ldots, m]$ префиксы последовательностей X и Y.

Определение 2.1.5. *Аккумулированной матрицей стоимости называется матрица, определенная следующим образом:*

$$D \in \mathbb{R}^{n \times m},\ d_{ij} = DTW(X(1 : i), Y(1 : j)). \tag{2.1.5}$$

Очевидно, $d_{nm} = DTW(X, Y)$. Следующая теорема показывает, что матрица D может быть эффективно вычислена.

Теорема 2.1.1. *Матрица аккумулированной стоимости удовлетворяет следующим равенствам:*

$$d_{i1} = \sum_{k=1}^{i} c(x_k, y_1) \text{ для } i \in [1, \ldots, n]; \tag{2.1.6}$$

$$d_{1i} = \sum_{k=1}^{i} c(x_1, y_k) \text{ для } i \in [1, \ldots, m]; \tag{2.1.7}$$

$$d_{ij} = \min\{d_{i-1j-1}, d_{ij-1}, d_{i-1j}\} + c(x_i, y_j)$$
$$\text{для } i \in [1, \ldots, n] \text{ и } j \in [1, \ldots, m]. \tag{2.1.8}$$

В частности, $DTW(n, m)$ может быть вычислено за $O(nm)$ операций.

Доказательство.

1. Пусть $i = 1$ и $j \in [1, \ldots, m]$. Тогда существует единственный искажающий путь для $X(1 : i)$ и $Y(1 : j)$, имеющий полную стоимость $\sum_{k=1}^{j} c(x_1, y_k)$. Формула (2.1.6) доказана.

2. Аналогично доказывается формула (2.1.7).

3. Положим $i > 1$ и $j > 1$. Пусть $q = (q_1, \ldots, q_l)$ — оптимальный искажающий путь для префиксов $X(1 : i)$ и $Y(1 : j)$.

4. Тогда, в соответствии с граничным условием, $q_l = (i, j)$.

5. Обозначим $(a, b) = q_{l-1}$. В соответствии с условием размера шага, $(a, b) \in \{(i-1, j-1), (i, j-1), (i-1, j)\}$.

6. Кроме того, $(q_1, \ldots, q_{l-1})$ должна быть оптимальным искажающим путем для $X(1:a)$ и $Y(1:b)$.

7. Поскольку $d_{ij} = c_{q_1,\ldots,q_{l-1}}(X(1:a), Y(1:b)) + c(x_i, y_j)$, из оптимальности искажающего пути q следует истинность (2.1.8).

Из теоремы 2.1.1 следует возможность построения алгоритма рекурсивного вычисления аккумулированной матрицы расстояний за вреия $O(nm)$. Инициализация, производящаяся в алгоритме, может быть упрощена, если положить $d_{0i} = \infty$ для $i \in [1, \ldots, m]$, $d_{i0} = \infty$ для $i \in [1, \ldots, n]$ и $d_{00} = 0$.

Более того, вычисления могут производиться построчно (или по столбцам), при этом, для вычисления очередной строки (столбца) матрицы D требуется знание только одной предыдущей строки (столбца). Таким образом, вычисление $DTW(X, Y)$ требует $O(nm)$ времени и $O(\min(n, m))$ памяти. Кроме того, если требуется восстановление оптимального искажающего пути, эта оптимизация неприменима, и потребуется $O(nm)$ памяти.

Ниже приводятся два алгоритма: алгоритм вычисления матрицы кумулятивного расстояния и алгоритм восстановления оптимального искажающего пути по матрице кумулятивного расстояния.

Алгоритм 2.1.1. AccumulatedCostMatrix

Исходные данные: *Последовательности X и Y, а также мера локального расстояния c.*

Результат работы: *Матрица кумулятивного расстояния D.*

1. **Инициализация:**
2. $d_{00} = 0$;
3. *Для* $i = 1, \ldots, n$:
4. $d_{i0} = \infty$
5. *Для* $i = 1, \ldots, m$:
6. $d_{0i} = \infty$
7. **Ход алгоритма:**
8. *Для* $i = 1, \ldots, n$:
9. *Для* $j = 1, \ldots, m$:
10. $d_{ij} = \min\{d_{i-1j-1}, d_{ij-1}, d_{i-1j}\} + c(x_i, y_j)$.

На рисунке 2.1.4 изображен оптимальный искажающий путь для последовательностей с рисунка 2.1.2.

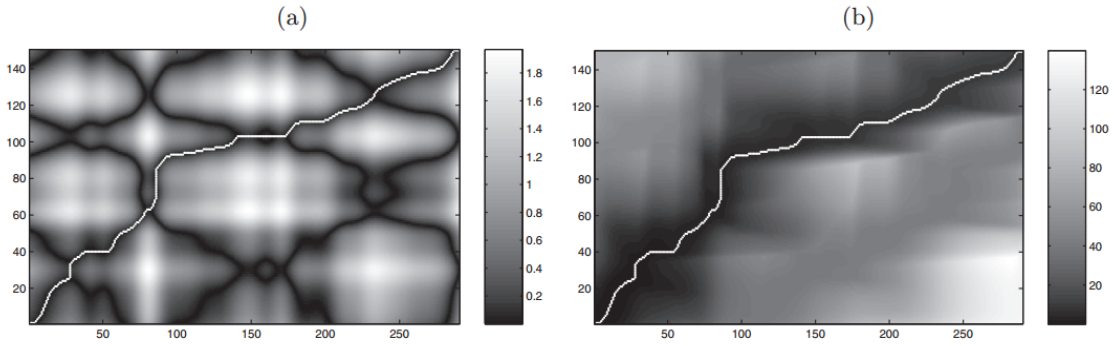

Рис. 2.1.4: Матрица стоимости, взятая с рис. 2.1.2, и кумулятивная матрица расстояния с отмеченным на них оптимальным путем

Алгоритм 2.1.2. OptimalWarpingPath

Исходные данные: *Матрица кумулятивного расстояния* D.

Результат работы: *Оптимальный искажающий путь* p^*.

1. **Инициализация:**
2. $p = (n, m)$;
3. $(a, b) = (n, m)$;
4. **Ход алгоритма:**
5. *Пока* $(a, b) \neq (1, 1)$:
6. *Если* $a = 1$:
7. $(a, b) = (a, b - 1)$;
8. *Иначе, если* $b = 1$:
9. $(a, b) = (a - 1, b)$;
10. *Иначе:*

11. $(a, b) = \arg\min d_{a-1b-1}, d_{ab-1}, d_{a-1b}$;

12. $p = \{(a, b), p\}$;

Ниже будут рассмотрены различные модификации алгоритма Dynamic Time Warping.

2.1.2 Изменение условия на размер шага

Напомним, что условие размера шага из определения 2.1.1 является, в некотором смысле, условием непрерывности построенного искажающего пути, именно оно дает гарантию того, что каждому элементу последовательности $X = \{x_1, \ldots, x_n\}$ ставится в соответствие элемент последовательности $Y = \{y_1, \ldots, y_m\}$, и наоборот. Однако, тот вид условия, который приведен в определении 2.1.1, имеет один существенный недостаток: одному и тому же элементу одной из последовательностей может быть поставлено в соответствие много элементов другой последовательности. Таким образом, на искажающем пути могут появиться вертикальные или горизонтальные участки, как изображено на рисунке 2.1.5а. Интуитивно, искажающий путь может "застрять"в некотором элементе, что приводит к значительному замедлению одной из последовательностей (и соответствующему ускорению второй).

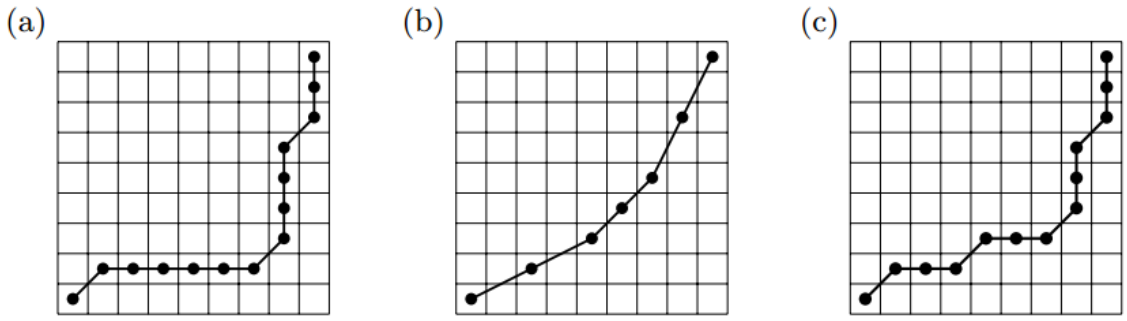

Рис. 2.1.5: Искажающие пути, которые могут быть получены с применением условий размера шага, представленных на рис. 2.1.6: **(a)** части искажающего пути вырождены, **(b)** некоторым элементам последовательностей на соответствует ничего, **(c)** Искажающий путь, лишенный этих недостатков.

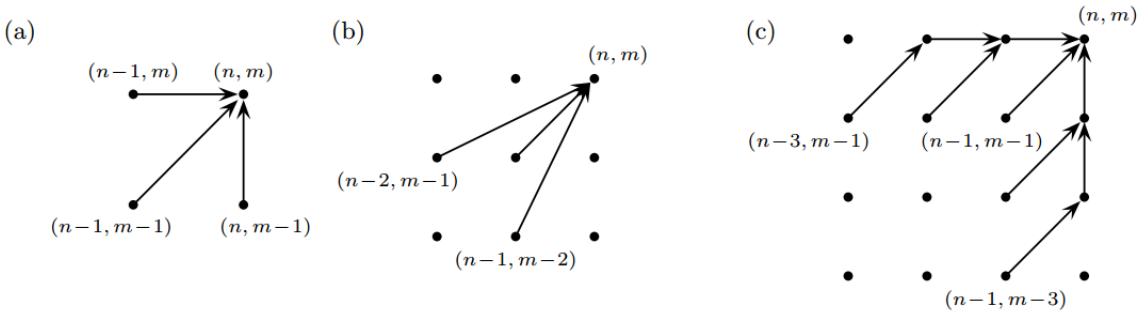

Рис. 2.1.6: Иллюстрация различных условий размера шага. **(a)** соответствует условию из определения 2.1.1.

Один из возможных способов избежать подобного вырождения заключается в изменении условия на размер шага для ограничения наклона рассматриваемых искажающих путей. Например, условие можно задать в виде

$$p_i - p_{i-1} \in \{(2,1),(1,2),(1,1)\} \text{ для } i \in [2,\ldots,l]. \tag{2.1.9}$$

Такой вид ограничения размера шага приводит к наклону искажающего пути от $\frac{1}{2}$ до 2 (см. рис. 2.1.5b). В этом случае матрица кумулятивного расстояния может быть вычислена с использованием рекуррентного соотношения

$$d_{ij} = \min\{d_{i-1\,j-1}, d_{i-2\,j-1}, d_{i-1\,j-2}\} + c(x_i, y_j) \tag{2.1.10}$$

для $i \in [2,\ldots,n]$ и $j \in [2,\ldots,m]$ и соответствующих начальных значений. При использовании такого ограничения на размер шага, искажающий путь между двумя последовательностями X и Y будет существовать только в случае, если их длины отличаются не более, чем вдвое. Кроме того, не для каждого элемента первой последовательности будет найден парный ему элемент второй последовательности (и наоборот). Эта ситуация показана на рисунке 2.1.6b.

Рисунок 2.1.5c иллюстрирует другой пример ограничения на размер шага, лишенного этих недостатков: это ограничение ограничивает наклон искажающего пути, но, в то же время, запрещает пропуск элементов последовательностей. Рекуррентное соотношение для матрицы кумулятивного расстояния в этом случае задается как

$$d_{ij} = \min \begin{cases} d_{i-1\,j-1} + c(x_i, y_j), \\ d_{i-2\,j-1} + c(x_{i-1}, y_j) + c(x_i, y_j), \\ d_{i-1\,j-2} + c(x_i, y_{j-1}) + c(x_i, y_j), \\ d_{i-3\,j-1} + c(x_{i-2}, y_j) + c(x_{i-1}, y_j) + c(x_i, y_j), \\ d_{i-1\,j-3} + c(x_i, y_{j-2}) + c(x_i, y_{j-1}) + c(x_i, y_j) \end{cases} \tag{2.1.11}$$

для $(i, j) \in [1, \ldots, n] \times [1, \ldots, m] \backslash \{(1, 1)\}$. Здесь в качестве начальных значений можно использовать $d_{11} = c(x_1, y_1), d_{i\,-2} = d_{i\,-1} = d_{i\,0} = \infty$ для $i \in [-2, \ldots, n]$ и $d_{-2\,i} = d_{-1\,i} = d_{0\,i} = \infty$ для $i \in [-2, \ldots, m]$.

Такой вид ограничения на размер шага ограничивает наклон искажающего пути значениями $\frac{1}{3}$ и 3. Рисунок 2.1.5 показывает отличия оптимального искажающего пути при использовании рассмотренных здесь условий размера шага.

2.1.3 Добавление локальных весов

Для того, чтобы поощрить использование алгоритмом горизонтальных, вертикальных или диагональных ходов, можно ввести вектор локальных весов $(w_d, w_h, w_v) \in \mathbb{R}^3$. В этом случае рекуррентные соотношения могут быть записаны в виде

$$d_{ij} = \min \begin{cases} d_{i-1\,j-1} + w_d \cdot c(x_i, y_j), \\ d_{i-1\,j} + w_h \cdot c(x_i, y_j), \\ d_{i\,j-1} + w_v \cdot c(x_i, y_j), \end{cases} \tag{2.1.12}$$

где $i \in [2, \ldots, n]$ и $j \in [2, \ldots, m]$. Кроме того, $d_{i\,1} = \sum_{k=1}^{i} w_h \cdot c(x_k, y_1)$ для $i \in [2, \ldots, n]$ и $d_{1\,i} = \sum_{k=1}^{i} w_v \cdot c(x_1, y_k)$ для $i \in [2, \ldots, m]$.

Случай $(w_d, w_h, w_v) = (1, 1, 1)$ соответствует классическому DTW. В этом случае алгоритм, вероятно, будет отдавать предпочтение диагональным ходам, поскольку один диагональный ход соответствует комбинации горизонтального и вертикального ходов. Аналогично могут быть введены веса при использовании других видов ограничения на размер шага.

2.1.4 Добавление глобальных ограничений

Другим широко известным вариантом алгоритма DTW является введение глобальных ограничений на допустимые искажающие пути. Такие ограничения не только ускоряют вычисление DTW, но также предотвращают построение странных искажающих путей.

Более формально, назовем $R \subseteq [1, \ldots, n] \times [1, \ldots, m]$ глобальным ограничивающим регионом. Тогда искажающий путь относительно R — это искажающий путь, полностью лежащий внутри региона R. Оптимальный искажающий путь относительно R, обо-

значающийся p_R^* — это искажающий путь относительно R, имеющий минимальную полную стоимость.

Двумя самыми распространенными глобальными ограничивающими регионами являются полоса Сакоэ-Шиба [10] и параллелограмм Итакура [11], представленные на рисунке 2.1.7. Здесь в качестве элементов искажающего пути могут быть выбраны только ячейки, закрашенные серым цветом.

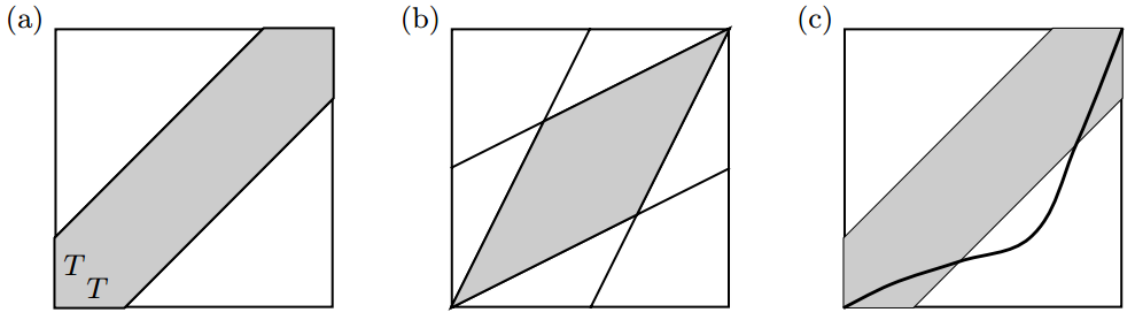

Рис. 2.1.7: **(a)** Полоса Сакоэ-Шиба ширины T; **(b)** Параллелограмм Итакура при $S = 2$; **(c)** Искажающий путь, частично лежащий вне глобального ограничивающего региона.

Полоса Сакоэ-Шиба проходит вдоль главной диагонали и имеет фиксированную ширину T. Это означает, что элементы x_i и y_j могут быть поставлены в соответствие только если $j \in [\frac{m-T}{n-T} \cdot i - T, \frac{m-T}{n-T} \cdot i + T] \cap [1, \ldots, m]$ (рис. 2.1.7a). Параллелограмм Итакура описывает регион, который ограничивает наклон искажающего пути. Более формально, для фиксированного $S \in \mathbb{R}_{>1}$ параллелограмм Итакура состоит из всех ячеек, являющихся элементами

каких-либо искажающих путей с наклоном от $\frac{1}{S}$ до S (рис. 2.1.7b). Заметим также, что локальное ограничение на размер шага также вводит некоторый вид глобального ограничения. Например, рассматривавшееся выше ограничение $p_i - pi - 1 \in \{(2,1),(1,2),(1,1)\}$ для $i \in [2,\ldots,l]$, является синонимом глобального ограничения в виде параллелограмма Итакуры с $S = 2$.

Для глобального ограничивающего региона R, путь p_R^* вычисляется практически таким же образом, как оптимальный искажающий путь в задаче без ограничений. Единственным отличием является то, что $c(x_i, y_j)$ полагается равным ∞ для всех пар $(i,j) \notin R$. Таким образом, требуется вычисление лишь тех ячеек матрицы кумулятивного расстояния, которые лежат в R. Это может значительно ускорить вычисление алгоритма DTW. Например, в случае полосы Сакоэ-Шиба фиксированной ширины T, потребуется лишь $O(T \cdot \max(n,m))$ времени вместо $O(nm)$ в классическом алгоритме DTW, что может привести к значительной экономии времени работы в случае $T \ll n, m$.

Однако, использование глобальных ограничений может быть нежелательным в некоторых случаях, поскольку оптимальный искажающий путь может выходить за рамки выбранного региона, и, таким образом, не совпадать с p_R^* (см. 2.1.7c). В некоторых случаях это может привести к нежелательным или полностью бесполезным результатам. Кроме того, существуют и другие способы увеличе-

ния скорости работы алгоритма DTW, см. [12].

2.1.5 Алгоритм derivative dynamic time warping

Если на вход DTW подаются две последовательности, которые похожи друг на друга, за исключением локальных ускорений и замедлений по временной оси, алгоритм обычно показывает неплохие результаты. Однако, DTW не столь успешно справляется с ситуацией, когда две последовательности также отличаются по оси y. Глобальные отличия, затрагивающие последовательности целиком, такие, как разные значения средних (сдвиг), различные масштабы или линейные тренды, могут быть эффективно устранены [13, 14].

Однако, два временных ряда могут быть искажены локально, например, вогнутость на одном из них может быть глубже, чем на другом. Например, на рисунке 2.1.8 две идентичные последовательности имели тривиальное соответствие, однако небольшое углубление вогнутости привело к образованию двух сингулярностей.

Проблема DTW заключается в признаках, которые он рассматривает. В рамках DTW для вычисления результата используются только значения объектов. Например, представим себе два элемента данных x_i и y_j, которые имеют одинаковые значения, но x_i является частью восходящего тренда, а y_j — нисходящего. С точки

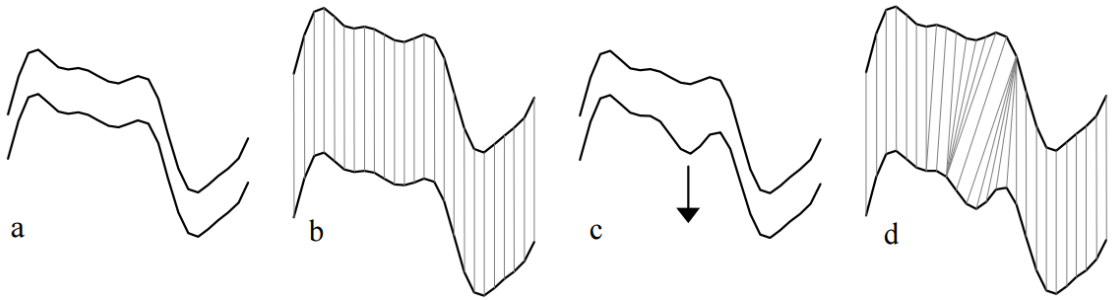

Рис. 2.1.8: При использовании DTW, две идентичных последовательности **(a)**, очевидно, приводят к тривиальному соответствию **(b)**. Однако, если в одной из последовательностей был немного изменен локальный признак (в данном случае — глубина вогнутости **(c)**), то DTW пытается объяснить расхождение в терминах искажения временной оси, что приводит к неожиданным результатам **(d)**.

зрения DTW соответствие между этими двумя точками идеально (расстояние равно 0), хотя интуитивно мы предпочли бы не сопоставлять элемент восходящего тренда с элементом нисходящего. Именно эту проблему решает модификация алгоритма DTW, рассматривающаяся в этом разделе. Такой модифицированный алгоритм использует не только значения элементов последовательностей, но также и информацию о "форме"последовательности. Поскольку в качестве носителя информации о "форме"используются производные рассматриваемого временного ряда, алгоритм получил название Derivative Time Warping (DDTW).

Как и в DTW, в DDTW строится матрица попарных расстояний между объектами последовательностей. Однако, вместо использования евклидовой метрики (или манхеттенского расстояния) в DDTW используется квадрат разности между оценками первых производных для x_i и y_j. Несмотря на наличие сложных методов для оценки производных, особенно в случае наличия информации о модели рассматриваемых данных, в статье [15] предлагается использовать следующую формулу:

$$D_t[x] = \frac{(x_i - x_{i-1}) + ((x_{i+1} - x_{i-1})/2)}{2} \tag{2.1.13}$$

Эта оценка является простым средним наклона линии, проходящей через текущую точку и ее левого соседа, и линии, проходящей через левого и правого соседей рассматриваемой точки. Эмпирически, эта оценка более устойчива к выбросам, чем любая другая оценка, использующая только две точки. Заметим, что оценка не определена для первой и последней точек интервала, поэтому DDTW строит соответствие между частями рассматриваемых последовательностей со второго по предпоследний элемент. Для сильно зашумленных данных предлагается [16] использовать экспоненциальное сглаживание последовательностей данных перед вычислением производных.

Временная сложность DDTW — $O(nm)$, такая же, как у DTW.

2.2 Алгоритмы кластеризации

Типичная задача кластеризации состоит из следующих этапов [17]:

- Представление данных (например, отбор и выделение признаков);
- Определение меры близости, имеющей смысл в исследуемой предметной области;
- Собственно кластеризация;
- Абстракция данных;
- Оценка полученных результатов.

Представление данных включает в себя выбор количества классов, а также количества, типа и масштаба признаков, доступных алгоритму кластеризации. Не всегда у исследователя есть свобода выбора всей этой информации. *Отбор признаков* — это процесс выбора "наилучшего"в некотором смысле подмножества из имеющихся признаков для использования при кластеризации. *Выделение признаков* — это процесс создания новых признаков на основе уже существующих. Одна (или обе) из этих техник может быть применена для получения информативного набора признаков для использования при кластеризации.

Мера близости обычно определяется как функция расстояния, определенная на множестве пар объектов. В различных задачах используется множество функций расстояния [18, 17, 19]. Простые меры расстояния, такие, как евклидово или манхеттенское расстояние, используются во многих задачах в качестве меры непохожести объектов, однако в некоторых случаях, если требуется подчеркнуть определенные свойства сравниваемых объектов, используются другие меры расстояния [20, 21].

Этап *группировки,* или, собственно, *кластеризации,* может производиться различными способами. Кластеризация может быть жесткой (каждому объекту ставится в соответствие идентификатор группы) или мягкая (для каждого объекта выдается вероятность его принадлежности каждой из групп). Иерархические алгоритмы кластеризации выдают систему вложенных разбиений множества всех объектов на кластеры, объединяя или разделяя объекты в зависимости от похожести кластеров.

Список литературы

[1] Wilpon JG, Juang BH, Rabiner LR. An investigation on the use of acoustic sub-word units for automatic speech recognition // Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'87. / IEEE. T. 12. 1987. C. 821–824.

[2] Keogh Eamonn J, Pazzani Michael J. Scaling up dynamic time warping for datamining applications // Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining / ACM. 2000. C. 285–289.

[3] Yi Byoung-Kee, Jagadish HV, Faloutsos Christos. Efficient retrieval of similar time sequences under time warping // Data Engineering, 1998. Proceedings., 14th International Conference on / IEEE. 1998. C. 201–208.

[4] Berndt Donald J, Clifford James. Using Dynamic Time Warping to Find Patterns in Time Series. // KDD workshop / Seattle, WA. T. 10. 1994. C. 359–370.

[5] Gavrila DM, Davis LS [и др.]. Towards 3-d model-based tracking and recognition of human movement: a multi-view approach // International workshop on automatic face-and gesture-recognition / Citeseer. 1995. C. 272–277.

[6] Schmill Matthew D, Oates Tim, Cohen Paul R. Learned models for continuous planning. // AISTATS. 1999.

[7] Rabiner Lawrence, Juang Biing-Hwang. Fundamentals of speech recognition. 1993.

[8] Gollmer Klaus, Posten Clemens. Detection of distorted pattern using dynamic time warping algorithm and application for supervision of bioprocesses // On-line fault detection and supervision in chemical process industries. 1995.

[9] Warped-average template technique to track on a cycle-by-cycle basis the cardiac filling phases on left ventricular volume / EG Caiani, A Porta, G Baselli [и др.] // Computers in Cardiology 1998 / IEEE. 1998. С. 73–76.

[10] Sakoe Hiroaki, Chiba Seibi. Dynamic programming algorithm optimization for spoken word recognition // Acoustics, Speech and Signal Processing, IEEE Transactions on. 1978. Т. 26, № 1. С. 43–49.

[11] Itakura Fumitada. Minimum prediction residual principle applied to speech recognition // Acoustics, Speech and Signal Processing, IEEE Transactions on. 1975. Т. 23, № 1. С. 67–72.

[12] Myers Cory, Rabiner Lawrence R, Rosenberg Aaron E. Performance tradeoffs in dynamic time warping algorithms

for isolated word recognition // Acoustics, Speech and Signal Processing, IEEE Transactions on. 1980. Т. 28, № 6. С. 623–635.

[13] Keogh Eamonn J, Pazzani Michael J. An Enhanced Representation of Time Series Which Allows Fast and Accurate Classification, Clustering and Relevance Feedback. // KDD. Т. 98. 1998. С. 239–243.

[14] Lin Rake, King-lp Agrawal, Shim Harpreet S Sawhney Kyuseok. Fast similarity search in the presence of noise, scaling, and translation in time-series databases // Proceeding of the 21th International Conference on Very Large Data Bases / Citeseer. 1995. С. 490–501.

[15] Keogh Eamonn J, Pazzani Michael J. Derivative Dynamic Time Warping. // Sdm / SIAM. Т. 1. 2001. С. 5–7.

[16] Mills Terence C. Time series techniques for economists. Cambridge University Press, 1991.

[17] Jain Anil K, Dubes Richard C. Algorithms for clustering data. Prentice-Hall, Inc., 1988.

[18] Anderberg Michael R. Cluster analysis for applications // Academic, New York. 1973.

[19] Diday Edwin, Simon JC. Clustering analysis // Digital pattern recognition. Springer, 1976. C. 47–94.

[20] Michalski Ryszard S, Stepp Robert E. Automated construction of classifications: Conceptual clustering versus numerical taxonomy // Pattern Analysis and Machine Intelligence, IEEE Transactions on. 1983. № 4. C. 396–410.

[21] Oates Tim, Firoiu Laura, Cohen Paul R. Clustering time series with hidden markov models and dynamic time warping // Proceedings of the IJCAI-99 workshop on neural, symbolic and reinforcement learning methods for sequence learning / Citeseer. 1999. C. 17–21.