НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ
(повна назва інституту/факультету)

кафедра БІОМЕДИЧНОЇ КІБЕРНЕТИКИ
(повна назва кафедри)

«На правах рукопису»
УДК 612.12

«**До захисту допущено**»
Завідувач кафедри БМК

_____ Є.А. Настенко
(підпис) (ініціали, прізвище)

“___” _____2018р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 122 «Комп’ютерні науки та інформаційні технології»
(код і назва)

на тему: «Програмна система оцінки впливу зовнішніх втручань на організм базуючись на хаотичності ЕКГ»

Виконав (-ла): студент (-ка) **VI** курсу, групи БС-61м
(шифр групи)

ВАХОВСЬКИЙ ІВАН ВОЛОДИМИРОВИЧ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник *проф. каф. БМК, проф., д.т.н. Файнзільберг Л.С.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант з розділів МД _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент *доц. каф. БМІ, доц., к.т.н., Зубчук В.І* _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____
(підпис)

Київ – 2018 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут (факультет) БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ
(повна назва)

Кафедра БІОМЕДИЧНОЇ КІБЕРНЕТИКИ
(повна назва)

Рівень вищої освіти – другий (магістерський) за освітньо-науковою програмою
спеціальність 122 «Комп'ютерні науки та інформаційні технології»
(спеціалізація) (Інформаційні технології в біології та медицині)
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри БМК
_____ _Є.А. Настенко_
(підпис) (ініціали, прізвище)
«___»_____2018 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

ВАХОВСЬКОМУ ІВАНУ ВОЛОДИМИРОВИЧУ
(прізвище, ім'я, по батькові)

1. Тема дисертації «Програмна система оцінки впливу зовнішніх втручань на організм базуючись на хаотичності ЕКГ»

науковий керівник дисертації
Файнзільберг Леонід Соломонович, д.т.н., професор
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «_29_»_березня_ 2018 р. №_1041-с

2. Термін подання студентом дисертації ***11-12 травня 2018 року***
3. Об'єкт дослідження Методи синергетики в медицині

4. Предмет дослідження Фазовий портрет ентропії ЕКГ сигналу у часі

5. Перелік завдань, які потрібно розробити
Провести аналіз оцінювання хаотичності динамічних рядів, программно реалізувати алгоритми обчислення ентропій різного типу, программно новий метод оцінювання зміни у часі ентропії показників ЕКГ на основі

її фазового портрету та алгоритму обчислення його опуклої оболонки.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу

Графік ентропії 1 порядку, графік ентропії 2 порядку, графік фазового портрету, графік випуклої оболонки

7. Орієнтовний перелік публікацій 1. Fainzilberg L., Orikhovska K., Vakhovskyi I. Analysis of subtle changes in biomedical signals based on entropy phase portrait // Биомедицинская инженерия и электроника : електрон. версія журн. 2017. No. 3. P. 44-66. URL : biofbe.esrae.ru/212-1154. 2. Ваховський І.В. Аналіз тонких змін біомедичних сигналів на основі фазового портрету ентропії // Актуальные научные исследования в современном мире // Сб. научных трудов - Переяслав- мельницкий, 2017. - Вып. 12(32), ч. 7. С.122-125.

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Магістерської дисертації			

9. Дата видачі завдання ***19 березня 2018 р.***

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Отримати завдання на МД	19 березня 2018р..	
2	Огляд літератури	30 березня 2018р.	
3	Побудова структури програмної системи	10 березня 2018р.	
4	Проведення експериментів	10 квітня 2018р.	
5	Написання МД	1травня 2018р.	
6	Предзахист МД та допуск до захисту дисертації	3 травня 2018р..	
7	Подання МД рецензенту. Отримання рецензії.	4-7 травня 2018р.	
8	Подання в електронному вигляді МД та анотації до неї на сайт кафедри.	11-12 травня 2018р.	
9	Подання пакету документів по МД до захисту в ЕК	11-12 травня 2018р.	
10	Захист МД в ЕК	18-19 травня 2018р..	

Студент _____ Ваховський І.В.
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ Файнзільберг Л.С.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ЕКГ – електрокардіограма

ПЗ – програмне забезпечення

ПП – програмний продукт

DFD – Data Flow Diagram

ВСТУП

Актуальність дослідження.

У наш час основну небезпеку для здоров'я населення у всьому світі і проблему для охорони здоров'я стали являти собою хвороби серцево-судинної системи, які є провідною причиною захворюваності, інвалідизації та смертності населення всього світу [52]. Саме тому пошук і розробка нових методів їх діагностики та лікування — залишається одним з найбільш актуальних завдань медицини.

При вивченні динаміки поведінки складних медико-біологічних систем все більшу увагу привертають методи теорії хаосу і синергетики [1], які дозволяють більш повно розкрити і проаналізувати механізми функціонування живої складноорганізованої системи.

Синергетика зіграла важливу роль і в осмисленні функціонування нашого організму. Для стабільного стану всіх систем людського організму потрібен певний режим між порядком і безпорядком, так званий режим детермінованого хаосу [52].

Наприклад, аритмія серця є небезпечною, але не менш небезпечним буде вважатися занадто регламентоване серце, що буде свідчити про якусь патологію. Строго регламентована робота серця призводить до нездатності гнучко реагувати на мінливі умови середовища, тим самим зменшуючи його пристосувальні можливості [52].

Кожна система нашого організму направлено прагне до гомеостазу - тимчасової сталості організму. Але жодна система не може постійно бути стабільна. Відсутність змін в системі, тобто її стабільність і врівноваженість, так чи інакше призводять її до застою, а в подальшому і до смерті. Розвиток і функціонування системи являє собою хвилеподібний рух з підйомами і спадами [52].

Методи синергетики знайшли застосування і в кардіології для оцінки хаотичності серцевого ритму, який несе інформацію про функціональний

стан всіх ланок регулювання життєдіяльності людини, як в нормі, так і при різних патологіях [2-4]. Така оцінка зазвичай проводиться по динамічному ряду тривалостей серцевих циклів ($R-R$ інтервалів), які визначаються в процесі реєстрації ЕКГ [5].

Проте, подальші дослідження, результати яких викладені у роботі [9] показали, що зміна ентропії у часі несе діагностичну цінність. Тому було запропоновано обчислити фазовий портрет, а потім на основі його випуклої оболнки зробити висновок про доцільність методу, провівши дослідження на модельних та реальних даних.

Мета роботи – розробити програмний модуль для оцінювання впливу зовнішніх втручань на організм людини на основі аналізу хаотичності фрагментів ЕКГ.

У відповідності до мети сформовано наступні **задачі**:

- провести аналіз оцінювання хаотичності динамічних рядів;

- програмно реалізувати алгоритми обчислення ентропій різного типу;

- програмно реалізувати новий метод оцінювання зміни у часі ентропії показників електрокардіограми на основі її фазового портрету та алгоритму обчислення його опуклої оболонки;

- провести експериментальні дослідження на модельних та реальних сигналах.

Об’єкт дослідження – методи синергетики в медицині.

Предмет дослідження – фазовий портрет ентропії ЕКГ сигналу у часі.

Для досягнення мети дослідження використано наступні **методи**:

- для реалізації програмного продукту використано Microsoft Visual Studio 2017 з використанням мови програмування C#.

Реалізація результатів роботи.

Робота виконана на замовлення Міжнародного науково-навчального центру інформаційних технологій і систем Національної Академії наук України та Міністерства освіти і науки України. Одержані результати дослідження впроваджені в Міжнародному науково-навчальному центрі інформаційних технологій і систем Національної Академії наук України та Міністерства освіти і науки України (акт впровадження від 25.04.18).

Апробація результатів роботи.

Основні положення та результати магістерської дисертації були викладені в наступних роботах:

1. Fainzilberg L., Orikhovska K., Vakhovskyi I. Analysis of subtle changes in biomedical signals based on entropy phase portrait // Биомедицинская инженерия и электроника : електрон. версія журн. 2017. No. 3. P. 44-66. URL : biofbe.esrae.ru/212-1154.
2. Ваховський І.В. Аналіз тонких змін біомедичних сигналів на основі фазового портрету ентропії // Актуальные научные исследования в современном мире // Сб. научных трудов - Переяслав- мельницкий, 2017. - Вып. 12(32), ч. 7. С.122-125.

Структура дисертації

Дисертація побудована за класичним типом та викладена на 140 сторінках машинописного тексту. Складається з вступу, 6 розділів, висновків, списку використаних літературних джерел, який містить 53 найменувань, 41 – на кирилиці, 12 – на латиниці. У роботі представлено 53 рисунків і 13 таблиць.

РОЗДІЛ 1
ЗАГАЛЬНІ ВІДОМОСТІ З ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Опис предметної області

Діагностика серцево-судинної системи людини належить до числа найважливіших завдань кардіології. Основні причини смертності людей працездатного віку пов'язані з серцево-судинними захворюваннями. Цим обумовлена необхідність розробки і вдосконалення засобів моніторингу для об'єктивного оцінювання і прогнозування стану серцево-судинної системи. На даний момент електрокардіограма (ЕКГ) є найпоширенішим методом діагностики роботи серцево-судинної системи людини [53].

Центральний елемент системи кровообігу ссавців - серце - порожнистий м'язовий орган, здатний до ритмічних скорочень, що забезпечує безперервний рух крові всередині судин. Скорочення серця супроводжуються його електричною активністю. Останню, можна реєструвати з поверхні тіла людини методом електрокардіографії [53].

Дамо коротку характеристику інформативним фрагментами циклу ЕКГ [53].

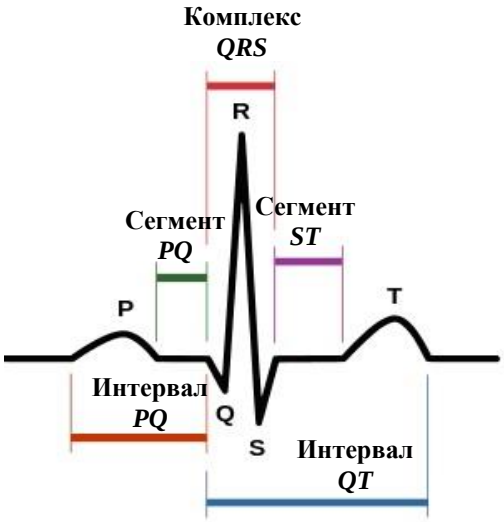


Рисунок 1.1. Ідеалізована форма циклу ЕКГ здорової людини.

Кожне биття серця складається з п'яти стандартних зубців, позначених буквами P, Q, R, S, T. Ці зубці вказують на деполяризацію і реполяризації фази серцевого м'яза. Крім того, існує п’ять сегментів PR, QRS, QT, ST. Ці інтервали показані на рис. 1.1. Тепер дамо коротку характеристику ролі цих клінічних ознак:

P хвилі. P хвилі, як правило, є низькоамплітудними ознаками, які відповідають скороченням правого і лівого передсердь. Їх важко виявити, але вони є важливими для виявлення різних серцевих аритмій [53].

QRS комплекс. Комплекс QRS відповідає систолі шлуночків. Хвиля збудження поширюється по шлуночках в різних напрямках в різні моменти часу, при цьому на ЕКГ формуються зубці Q, R і S. Початковий зубець Q реєструється під час збудження лівої частини міжшлуночкової перегородки. Зубець R (найчастіше найбільш виражений зубець ЕКГ), обумовлений збудженням основної маси міокарда лівого і правого шлуночків. Зубець S здебільшого обумовлений збудженням основи лівого шлуночка. Це найбільш значима хвиля ЕКГ через велику м'язову масу шлуночків. Таким чином, комплекс може бути легко виявлений і часто використовується для визначення частоти серцевих скорочень [53].

Зубець T. T хвиля представляє реполяризацію шлуночків. Це фаза відновлення серцевого м'яза. Форма цієї хвилі несе в собі багато інформації про порушення серцевої діяльності. Тому важливо проаналізувати його геометричні властивості, такі як симетрія, асиметрія і нахил [53].

Інтервал PR. Це час, що минув між початком зубця P і початком наступного QRS комплексу [53].

PR-сегмент. Як правило, це ізоелектричний сегмент між кінцем зубця P і початком комплексу QRS [53].

ST сегмент. Це період з кінця шлуночкової деполяризації до початку шлуночкової реполяризації. Зміна рівню ST-сегменту є маркером серцевих аномалій [53].

QT інтервал. Являє собою час між початком шлуночкової деполяризації і кінцем реполяризації. Інтервал QT обернено пропорційний частоті серцевих скорочень: скорочується на більш високій частоті серцевих скорочень і подовжується на більш повільній [53].

Органічні ураження і функціональні порушення серцево-судинної системи відображаються відповідними змінами параметрів і форми фрагментів ЕКГ. Тому головною метою морфологічного аналізу ЕКГ є розпізнавання і інформативних фрагментів ЕКГ і визначення амплітудно-часових параметрів зубців P, Q, R, S, T, інтервалів PQ, QT, і зміщення сегмента ST вниз (депресія) або вгору (елевація) щодо нульової (ізоелектричної) лінії [53].

Для комп'ютерної обробки важлива якість сигналу. Активний рух пацієнта або випробуваного може привести до того, що практично неможливо визначити R-зубці навіть візуально [53]. В такому випадку автоматизована обробка сигналу дасть неправдиві результати. Основною характеристикою якості сигналу для нас є те, які деталі можемо ми візуально визначити в сигналі, що містить шуми . Чим менше ми можемо визначити візуально положення R зубців, тим досліджуваний сигнал гірше.

Для ЕКГ існує кілька джерел перешкод і шумів:

- вплив мережевих наведень, 50 тобто (60) Гц шумів;
- рух пацієнта під час зняття ЕКГ;
- міографічна перешкода;
- дрейф нульової лінії.

Найбільш істотним з них є мережеві наведення. Це пов'язано з тим, що приміщення, де відбувається моніторинг ЕКГ, підпадає під вплив електромагнітних полів, що створюються електромережею. Перешкоди з'являються в результаті дії двох механізмів: через електричне і магнітне поля, що діють як окремо, так і разом [53].

Шуми від руху виникають в основному внаслідок зміни опору на межі електрод-шкіра. Якщо пара електродів знаходиться в контакті зі шкірою і один з них починає рухатися, то різниця потенціалів між ними почне змінюватися. Для поліпшення контакту застосовують марлеві серветки, змочені розчином якого-небудь нешкідливого електроліту. При тривалому моніторингу розчин може висохнути, що призведе до різкого збільшення опору. Щоб уникнути цього іноді навіть надійніше застосовувати голчасті електроди. Причиною перешкод може бути і електрична активність тканин скелетних м'язів, тому при знятті електрокардіограми пацієнт повинен максимально розслабитися. Коливання, викликані м'язовими біопотенціалами, іноді важко відрізнити від справжнього зубця ЕКГ [53].

Звісно, навіть у здорової людини форма циклу ЕКГ відрізняється від ідеальної, що ускладнює побудову ефективних алгоритмів обробки ЕКГ-сигналу. Навіть вирішення, здавалося б, зовсім простого завдання поділу ЕКГ на окремі цикли вимагає залучення досить складних алгоритмів виявлення QRS-комплексів [25].

Ще більші проблеми викликає завдання морфологічного аналізу ЕКГ для визначення значень діагностичних ознак, зосереджених на інформативних фрагментах. Наприклад, завдання визначення тривалості інтервалів PQ і QT потребує точного обчислення моментів початку і закінчення відповідних зубців, які через спотворення циклів реальних ЕКГ не мають чітких меж [26]. Тому визначення таких моментів на основі диференціювання сигналу і використання порогових функцій [27] стає ненадійним вже при порівняно невисокому рівні шуму.

1.2. Синергетика як наука про хаос

Синергетика – вчення про самоорганізацію матерії, міждисциплінарний напрямок науки, що вивчає загальні закономірності і процеси в складних системах [9].

Синергетика грає важливу роль в медицині. Виникнення життя на землі, передбіологічний розвиток, формування і поява імунності - в усіх цих сферах синергетика показала себе в ролі концепції, яка пояснює суть цих явищ. Більш того, в нашому житті зустрічаються безладні, неврівноважені системи [9].

Кожна система нашого організму направлено прагне до гомеостазу - тимчасової сталості організму. Але жодна система не може постійно бути стабільна. Відсутність змін в системі, тобто її стабільність і врівноваженість, так чи інакше призводять її до застою, а в подальшому і до смерті. Розвиток і функціонування системи являє собою хвилеподібний рух з підйомами і спадами [9].

Науковці різних медико-біологічних наук в кінцевому рахунку приходять до висновку про те, що здоров'я – це тонкий баланс між хаосом і порядком. У цьому зв'язку вчені, застосовуючи концепцію нелінійних систем, посилено формують уявлення динамічного захворювання. Хвороба не є щось стале, а являє собою «рухомий» процес, що розвивається. Людський організм являє собою саморегулюючу систему. Теорія хаосу в нелінійній динаміці може зіграти значну, позитивну роль в діагностиці та усуненні відповідного захворювання, в розумінні епідемічного перебігу захворювання. Питання полягає в тому, яка кількість хаосу потрібна людському організму, щоб він був в здоровому стані і яка кількість хаосу здатна витримати людина, щоб не захворіти? Відповіді на всі ці запитання може дати синергетика, застосування нелінійних методів [9].

Динамічний (детермінований) хаос - складна непередбачувана поведінка детермінованої нелінійної системи. Виявилося, що прості

системи, що складаються з незначної кількості компонентів і детерміновані правилами, що не включають елементів випадковості, можуть проявляти випадкову поведінку, досить складну і непередбачувану, причому випадковість носить принциповий, непереборний характер. Такого роду випадковість, непередбачуваність розвитку системи розуміється як хаос [10].

Детермінований хаос поєднує детермінованість і випадковість, обмежену передбачуваність і непередбачуваність та проявляється в таких різних явищах як кінетика хімічних реакцій, турбулентність рідини і газу, геофізичні, зокрема, погодні зміни, фізіологічні реакції організму, динаміка популяцій, епідемії, соціальні явища (наприклад, курс акцій) [10].

Спершу поділяли детерміновані системи, для яких був можливий прогноз на будь-який відрізок часу (подібно прогнозу затемнень сонця) і стохастичні системи, які можна охарактеризувати лише статистично. Тепер же з'явився новий клас об'єктів, формально детермінованих, але з поведінкою, прогнозованою лише на обмежений відрізок часу. Обидва полюси - порядок і хаос - не існують в чистому вигляді, якщо розуміти впорядковані системи як повністю регулярні, детерміновані, передбачувані, а невпорядковані системи як абсолютно нерегулярні, випадкові, непередбачувані. Прикладом систем з високим ступенем порядку і стабільності служать кристали; на протилежному полюсі розташовуються такі хаотичні системи як гази [10].

Теорія динамічного хаосу знищила розрив між класичною динамікою і статистичною фізикою: регулярний рух стає стохастичним внаслідок завжди присутніх невеликих флуктуацій. Розвиток теорії динамічного хаосу пов'язано з іменами А. Пуанкаре (H. Poincare), А.М. Ляпунова, А.А. Андронова, Е. Хопфа (E. Hopf), А.Н. Колмогорова, В.І. Арнольда [52].

Еволюція системи математично описується векторним полем в фазовому просторі - абстрактному просторі динамічних змінних системи,

векторному полі в координатах змінних. Точка фазового простору задає стан системи, вектор в цій точці вказує напрямок зміни системи. Криві послідовних станів процесу, створювані зміною положення точки в фазовому просторі, називаються фазовими траєкторіями, а їх сукупність - фазовим портретом системи. Траєкторії поля, що притягуються до одного з центрів тяжіння, утворюють область, звану областю дії (басейном) цього центру тяжіння (Р. Том, 1968) [52]. Фазовий простір - зручний засіб для наочного уявлення поведінки динамічної системи. На рис. 1.1 показані фазові портрети (нижній ряд) для системи з затухаючими коливаннями (траєкторія, яка прагне до стану рівноваги), з постійними коливаннями (замкнута крива) і більш складний випадок системи, що коливається в позбавленому суворої періодичності режимі. Сталі режими руху, іншими словами, безліч точок (в найпростішому випадку - одна точка) в фазовому просторі системи, до яких прагнуть її траєкторії, отримали назву атракторів - вони як би залучають, притягують траєкторії у фазовому просторі. У першому випадку атрактором виявляється нерухома точка, в другому - граничний цикл, в третьому ж - так званий дивний, або хаотичний (стохастичний) атрактор (рис. 1.2, зліва направо). Таким чином, атрактори - геометричні структури, що характеризують поведінку системи в фазовому просторі після досить тривалого періоду часу. Хаотичні, дивні атрактори відповідають непередбачуваній поведінці систем, що не мають строго періодичної динаміки, це математичний образ детермінованих неперіодичних процесів. Дивні атрактори структуровані і можуть мати досить складні і незвичайні зміни у тривимірному просторі [10].

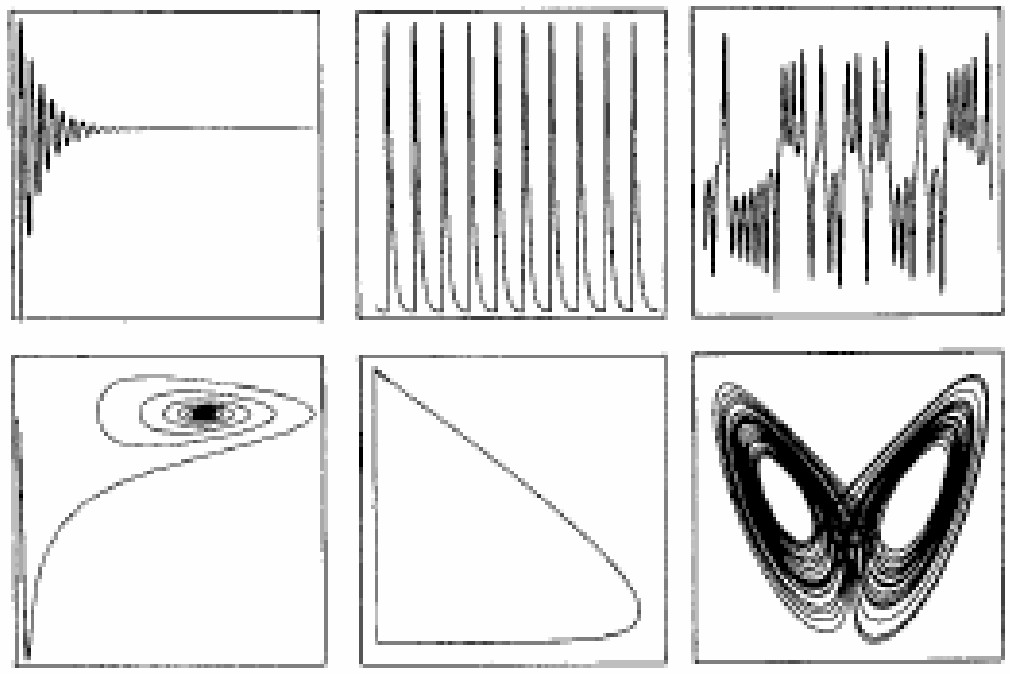


Рисунок 1.2. Послідовність змін в часі (верхній ряд) і фазові портрети (нижній ряд) для трьох різних систем (Глейк, 2001).

Синергетика зіграла важливу роль і в осмисленні функціонування нашого організму. Для стабільного стану всіх систем людського організму потрібен певний режим між «безладом» і порядком, так званий режим детермінованого хаосу [10].

Наприклад, аритмія серця є небезпечною, але не менш небезпечним буде вважатися занадто регламентоване серце, що буде свідчити про якусь патологію. Строго регламентована робота серця призводить до нездатності гнучко реагувати на мінливі умови середовища, тим самим зменшуючи його пристосувальні можливості [10].

Ентропія - це міра хаосу. Її величина дає нам уявлення про те, як далеко система знаходиться від упорядкованого, структурованого стану і як близько – до повністю хаотичного, безструктурного, однорідного виду [11].

Є різні визначення поняття «структура». В даному випадку під структурою розуміється характер організації елементів і сукупність

відносин між елементами системи. Зрозуміло, що структура задається природою зв'язків між елементами системи. Таке визначення структури не накладає ніяких обмежень на природу самої системи і її елементів. Це можуть бути системи атомів, літерні повідомлення, космічні, біологічні або соціальні системи. Кожна система може бути або високоорганізованою, тобто володіти розвиненою і складною структурою, або менш організованою, яка має досить просту структуру, або і зовсім хаотичною, тобто коли її елементи розподілені випадково і, в середньому, однорідно [11].

Як уже зазначалося, мірою хаосу є ентропія. Але в такій же мірі ентропія є і мірою структурної організованості систем, так як порядок і хаос - це не тільки протилежні, але, як зазначалося, і взаємодоповнюючі поняття. Це та сама єдність протилежностей, рівновагу або нерівновагу між якими визначають напрямок і темп розвитку або деградації структур в системі, яка розглядається [11].

1.3. Ентропія як міра хаосу

Ентропія як міра хаосу або порядку виявлялася в системах різної природи: це і ентропія Клаузіуса в термодинаміці, і ентропія Больцмана в статистичній фізиці, і ентропія Шеннона в теорії інформації, ентропія Колмогорова в теорії динамічних систем, і ентропія фон Неймана в квантовій механіці. Розуміння універсальності ентропії як міри хаотичності незалежно від природи системи приходило поступово. Наприклад, А. Н. Колмогоров писав в 1987 р. [12] з приводу ентропії Шеннона наступне: «Загальновідомо і те, що вираз $H\left(\xi\right)$ формально тотожний з виразом ентропії в фізиці. Цей збіг я вважаю цілком достатнім для виправдання найменування величини $H\left(\xi\right)$ і в теорії інформації «ентропією»: на таких математичних аналогіях слід завжди наголошувати,

оскільки зосередження уваги на них сприяє прогресу науки». «З розвитком фізики відкритих систем стало зрозуміло, що «серед різних макроскопічних функцій тільки ентропія S володіє сукупністю властивостей, що дозволяють використовувати її в якості міри невизначеності (хаотичності) при статистичному описі процесів в макроскопічних системах», відзначав у 2002 р. Ю. Л. Климонтович [13].

Максимальна ентропія заданої системи, відповідає нижчому ступеню її структурної організованості, тобто найбільшій хаотичності і невпорядкованості. Низька ентропія, навпаки, відповідає високій структурної впорядкованості. Ентропія цікава, перш за все, саме як міра структурної організованості системи. Щоб підкреслити цю обставину, далі вона буде згадуватись як структурна ентропія [52].

У роботі «Математична теорія зв'язку» (1984 г.) К. Шеннон [14], досліджуючи передачу повідомлень по шумливим лініях, ввів міру дискретного розподілу ймовірності P_i на безлічі альтернативних станів передавача і приймача повідомлень і вивів формулу, що стала основою формулою теорії інформації:

$$H = -\sum_{i=1}^{n} p_i \log_2 p_i \qquad (1.1)$$

Тут n - число символів, з яких може бути складено повідомлення (алфавіт), H - міра невизначеності. Якщо N - число всіх переданих і прийнятих символів в повідомленні, то $P_i = m_i / N$ - ймовірність появи і-го символу в повідомленні, m_i - число народження і-го символу в повідомленні. За аналогією з формулою Больцмана, він назвав величину H ентропією. Шеннон говорив, що назвати величину H ентропією йому порадив знаменитий математик Джон фон Нейман, який, жартуючи, обґрунтовував це тим, що мало хто з математиків і інженерів знає про

ентропії, і це дасть Шеннону велику перевагу в неминучих дискусіях. Але, швидше за все, це був не просто жарт, заснований на формальній аналогії. І Дж. Фон Нейман і К. Шеннон, ймовірно, були добре обізнані про інформаційний сенс ентропії Больцмана як про величину, що характеризує відсутню інформацію про систему [52].

На питання, що таке сама інформація, Шеннон дає геніально просту відповідь: це спад ентропії в результаті прийому повідомлень [52]. Отже, інформацію I Шеннон визначив як зменшення ентропії при отриманні повідомлень:

$$I = H_1 - H_2 \tag{1.2}$$

В науці вже виявлені елементи хаосу в функціонуванні нейронів і їх мереж, хаотична фрактальна динаміка на електроенцефалограмі і електрокардіограмі людини. Ще більше вражає, що хаос в функціонуванні організму виявився нормою і ознакою здоров'я, а упорядкований режим - свідченням патології (West, Goldberger, 1987; Голдбергер і ін., 1990). Наприклад, патологічна періодичність у функціонуванні нервової системи проявляється при епілепсії, паркінсонізмі, маніакально-депресивному психозі. Скорочення серця здорової людини позбавлені суворої періодичності, їх траєкторії в фазовому просторі утворюють хаотичний або дивний атрактор [52].

А. Н. Колмогорову належить не тільки істотна роль у перетворенні теорії інформації, сформульованої К. Е. Шенноном у вигляді, скоріше, технічної дисципліни, в строгу математичну науку, а й побудова основ теорії інформації на принципово іншому, відмінному від шеннонівського, фундаменті. У процесі цих робіт, а також робіт в області теорії динамічних систем, Ф. Н. Колмогоров [15] узагальнює поняття ентропії на ергодичні

випадкові процеси через граничний розподіл ймовірності, що має щільність $f(x)$.

У 1960 році угорський математик Альфред Реньї пропонує своє узагальнення ентропії [16]. Він вводить ентропію як q - момент заходи ε-розбиття (покриття):

$$R(q) = \frac{1}{1-q}\ln(\sum_{i=1}^{N(\varepsilon)} p_i^q), \sum_{i=1}^{N(\varepsilon)} p_i = 1. \quad (1.3)$$

системи, що припадають на і-елемент ε-розбиття, $N(\varepsilon)$-повне число елементів заданого ε покриття. Константа q може приймати будь-які значення, проте сенс ентропії Реньї при цьому, відповідно, змінюється.

Клаузіус, був першим, хто був переконаний в універсальному характері ентропії і вважав, що в усіх процесах, що відбуваються у Всесвіті, вона відіграє важливу роль, визначаючи їх напрямок розвитку в часі [52].

Певною мірою, але вже з інших позицій, до подібного висновку прийшов І. Пригожин [17]. Пригожин вважає, що принцип ентропії відповідальний за незворотність часу у Всесвіті і, можливо, грає важливу роль в розумінні сенсу часу як фізичного феномена.

Широка популярність формул типу $S = k\sum p\log(p)$ в різних областях пов'язана, з особливою роллю ступеневих законів в природі [18]. Для таких випадків вираз $p\log(p)$ є зручною адитивною мірою. Зручною в плані нашого сприйняття кількості чого б то не було - інформації, хаосу, нелокальності, інтенсивності світла зірки і так далі. Адже наші органи чуття працюють за логарифмічним законом Вебера-Фехнера: інтенсивність відчуття пропорційна логарифму інтенсивності стимулу. З цієї точки зору, ентропія, що виражається формулою типу $S = p\sum\log(p)$ є всього лише зручною для нашого сприйняття абстрактною адитивною міра. Її фізичний

зміст залежить від того, що саме ми оцінюємо, тобто тим, на яких даних і як ми визначаємо аргумент p - щільність ймовірності. Підстава логарифмів несуттєво і вибирається з міркувань зручності для того чи іншого завдання [52].

1.4. Хаотичність ЕКГ

В науковому світі хаотичність – не новий критерій діагностики. Він використовується для оцінки впливу алкогольних напоїв на серцево-судинну систему в роботі [19], для визначення глибини та стадій анестезії в роботі [20]. Але в цих роботах дослідження хаотичності зводиться до пошуку ентропії R-R інтервалів, не беручи до уваги інші цінні показники ЕКГ та діагностичні параметри, які можна отримати з них.

На практиці підтверджена діагностична ефективність показника, який оцінює величини розкиду траєкторій одноканальної ЕКГ на фазовій площині z (t), z’ (t) по хаусдорфовим відстаням [52].

Запропонований метод переходу від z (t) до параметрів A_P, $b_P^{(1)}$, $b_P^{(2)}$, μ_P,…, A_T, $b_T^{(1)}$, $b_T^{(2)}$, μ_T, t_0 також дозволяє оцінювати хаотичність ЕКГ: достатньо накопичити масив значень зазначених параметрів за певний період спостереження і оцінити індекс Хьорста, апроксимаційні або переставну ентропії або будь-який інший відомий показник ступеня хаотичності часового ряду [52].

Важливу діагностичну інформацію несуть інтервали PQ, QT, тривалості Δ_Q, Δ_{QRS}, зубця Q і комплексу QRS, які легко можуть бути обчислені за формулами:

$$PQ = \mu_Q - \mu_P + 3(b_p^{(1)} - b_Q^{(1)}), \tag{1.4}$$

$$QT = \mu_T - \mu_Q + 3(b_T^{(2)} + b_Q^{(1)}). \tag{1.5}$$

$$\Delta_Q = 3(b_Q^{(1)} + b_Q^{(2)}), \tag{1.6}$$

$$\Delta_{QRS} = \mu_S - \mu_Q + 3(b_Q^{(1)} + b_S^{(2)}). \tag{1.7}$$

Інші традиційні ознаки ЕКГ такі як глибина зубця Q, зміщення сегмента ST і амплітуда зубця T безпосередньо визначають параметри A_Q, A_{ST}, A_T відповідно [52].

Додаткову діагностичну цінність несуть показник β_T, симетрія зубця T і кут α_{QRS} орієнтації фазового портрету ЕКГ на площині $z(t), \dot{z}(t)$, які з високим коефіцієнтом детермінації можуть бути обчислені за рівнянням регресії [52]

$$\beta_T = 1{,}0082\eta^{-0.4248} \tag{1.8}$$

$$\alpha_{QRS} = 200{,}85e^{-0{,}7928\lambda} \tag{1.9}$$

де $\eta = b_T^{(1)} / b_T^{(2)}$, а $\lambda = A_Q / A_S$.

Для більш повної оцінки хаотичності ЕКГ можна в редукованому фазовому просторі спостерігати динаміку зміни нормованих ознак.

$$\overrightarrow{X_m} = (A_p[m], A_Q[m], A_{ST}[m], A_T[m], ..., PQ[m], QT[m], \beta_T[m], t_0[m]), \quad m = 1, ..., M. \tag{1.10}$$

від циклу до циклу і обчислити кореляційний інтеграл

$$I(\varepsilon) = \frac{1}{M^2}\sum_{m,l}^{M} \Theta(\varepsilon - \rho(\overrightarrow{X_m}, \overrightarrow{X_l})), \tag{1.11}$$

Тим самим оцінюється середня ймовірність того, що нормовані діагностичні ознаки, які фігурують у, виявляться близькими на двох різних циклах ЕКГ. Знаючи кореляційний інтеграл, можна визначити кореляційну розмірність атрактора ЕКГ [52].

Сучасні системи медичної діагностики зазвичай засновані на комп’ютерній обробці фізіологічних сигналів, які породжує організм в процесі свого функціонування. Наприклад, електрокардіограма (ЕКГ), яка несе інформацію про зміни електричної активності серця, вже понад сто років залишається одним з найбільш доступних і поширених методів функціональної діагностики в кардіології.

Бурхливий розвиток засобів обчислювальної техніки та інформаційних технологій заклало підґрунтя нової галузі - комп'ютерної електрокардіографії. Безумовно цифрові електрокардіографи, які забезпечують підтримку прийняття рішення лікаря-кардіолога, полегшують працю медичного персоналу та скорочують отримання результатів діагностики.

У той же час, як відзначають фахівці, комп'ютерна реалізації *традиційних підходів* до обробки ЕКГ у часовій області, не привела до досягнення більш важливої мети - підвищення достовірності результатів діагностики. До того ж, досвідчені клініцисти часто як і раніше надають перевагу візуальній інтерпретації ЕКГ, не в повному обсязі довіряючи комп'ютерним алгоритмам, які часом призводять до помилок ще на стадії вимірювання діагностичних ознак [1].

При постановці діагнозу лікарі орієнтуються не тільки на значення діагностичних ознак, але і враховують загальну клінічну картину і приймають «неформальні» рішення, спираючись на свій попередній досвід і інтуїцію. Тому в медичній практиці добре відомі приклади, коли кілька досвідчених кардіологів по-різному інтерпретують одну і ту ж ЕКГ.

Комп'ютерна електрокардіографія так чи інакше заснована на формальних алгоритмах аналізу відхилень показників ЕКГ від

популяційних норм. При цьому діагностичну цінність несуть як грубі, так і тонкі відхилення форми ЕКГ [52].

Грубими відхиленнями будемо називати патологічний (широкий і глибокий) зубець Q , Істотно розширений QRS - комплекс і ряд інших діагностичних ознак ЕКГ, аналіз яких не становить особливих труднощів як при візуальному, так і при комп'ютерному аналізі ЕКГ [52]. Значно більші проблеми викликає комп'ютерний аналіз тонких змін сигналу таких як альтернація або симетризація зубця T , які практично непомітні при візуальному аналізі ЕКГ, але несуть важливу діагностичну інформацію.

Вчені постійно шукають *нові* підходи до аналізу тонких змін ЕКГ сигналу. Один з таких інноваційних підходів - фазограф, заснована на переході від скалярного сигналу $z(t)$ в будь-якому з відведень до його обробки на фазовій площині $z(t), \dot{z}(t)$, де $\dot{z}(t)$ - оцінка швидкості зміни електричної активності серця [2].

Слід підкреслити, що такий підхід *принципово* відрізняє фазографії від аналогічних методів [3], заснованих на відображенні сигналу на площині з координатами $z(t), z(t-\tau)$, де τ - затримка в часі. Саме така відмінність дозволила розширити систему діагностичних ознак ЕКГ, заснованих на оцінці швидкісних характеристик процесу, і тим самим підвищити специфічність ЕКГ-діагностики.

Інтелектуальні можливості фазографії постійно розширюються. Останнім часом для вивчення динаміки поведінки складних медико-біологічних систем отримали популярність методи синергетики і теорії динамічного хаосу [7]. З позицій синергетики вдалося зрозуміти, як у відкритих системах з хаосу в результаті нелінійних процесів спонтанно виникають впорядковані структури.

Згідно [8] важливу інформацію про властивості системи несе не тільки сама ентропія, а й характер її зміни в часі. На основі аналізу форми зміни ентропії В.С. Аніщенко [9] виявив гендерні відмінності реакції

організму на стресові впливи навколишнього середовища. В роботі [10] отримано ряд цікавих результатів по використанню ентропійного методу при комплексній оцінці динаміки факторів ризику серцево-судинних захворювань.

До складу фазаграфії включений додатковий програмний модуль, забезпечуючий оцінку хаотичності показників ЕКГ [11]. Подальший розвиток ентропійного підходу для аналізу хаотичності ЕКГ та інших біомедичних сигналів неминуче дозволить отримувати додаткову інформацію при оцінці тонких змін сигналу, викликаних зовнішніми впливами на організм (фізичні навантаження, лікарська терапія, оперативне втручання та ін.), а значить є актуальною задачею як в науковому, так і в прикладному значеннях.

1.5. Проблеми обробки ЕКГ сигналу у часовому просторі

Електрокардіографія вже більше ста років залишається одним з найбільш доступних і поширених методів функціональної діагностики в кардіології. Довгий час цей метод розвивався на основі аналогових електрокардіографів, на стрічці яких відобразився графік зміни електричної активності серця - електрокардіограми (ЕКГ) [2]. У цей період склалися основні представлення про лікарські аспекти традиційної електрокардіографії, в тому числі, про точки накладення електродів на тіло пацієнта (відведень ЕКГ), генезисі інформативних фрагментів і діагностичних показників, що несуть інформацію про серцево-судинні патології.

Бурхливий розвиток засобів обчислювальної техніки та інформаційних технологій поклав основу нової галузі - комп'ютерної електрокардіографії, клінічне застосування якої пройшло декілька стадій свого розвитку.

Перші цифрові електрокардіографи мали лише одну, але дуже важливу функцію реєстрації і зберігання ЕКГ в цифровий формі. Друге покоління забезпечувало вже можливість автоматичного розпізнавання інформативних фрагментів ЕКГ (зубців, комплексів, сегментів) і вимір амплітудно-часових параметрів цих фрагментів [2]. І, нарешті, з'явилися цифрові електрокардіографи з вбудованими алгоритмами автоматичної інтерпретації ЕКГ, засновані на багаторічному досвіді традиційної електрокардіографії.

Безумовно, застосування таких електрокардіографів в амбулаторній і клінічній практиці, що забезпечує підтримку прийняття рішення лікаря-кардіолога, полегшує працю медичного персоналу і скорочує час отримання результату діагностики. В той же час комп'ютерна реалізація традиційних підходів до обробки ЕКГ у часовій області не призвела до досягнення більш важливої мети – підвищення достовірності результатів діагностики [2]. До того ж досвідчені клініцисти часто як і раніше вважають за краще візуальну інтерпретацію ЕКГ, в повному обсязі довіряючи комп'ютерним алгоритмам, які часом призводять до помилок на стадії вимірювання діагностичних ознак.

Справа в тому, що на реальних ЕКГ, як правило, немає чітких меж між інформативними фрагментами, що ускладнює їх автоматичне розпізнавання. Навіть таке, на перший погляд просте завдання, як автоматичне розділення тимчасового ЕКГ сигналу z(t) на окремі серцеві цикли потребують залучення досить складних обчислювальних процедур виділення QRS комплексів [2].

Ще більші проблеми викликає завдання визначення амплітудно-часових параметрів, зосереджених на інформативних фрагментах ЕКГ. Труднощі цього завдання обумовлені необхідністю визначення моментів початку і закінчення фрагментів реального сигналу, форма яких через дії різних збурень спотворена і відрізняється від ідеальної [53].

При цьому слід мати на увазі, що в реальних умовах заміщення далеко не завжди носять лише характер адитивної перешкоди, що вимагає застосування нових підходів до вирішення проблеми відновлення корисного сигналу по спотвореній реалізації.

Але навіть для придушення адитивної перешкоди, наприклад, мережевих перешкод 50 Гц, потрібно внести ряд удосконалень в традиційні режекторні фільтри, щоб не допускати існування спотворення форми сигналу, що несе інформацію про електричну активність серця [3]. Саме тому виробники цифрових електрокардіографів рекомендують всіляко боротися з джерелами таких перешкод, щоб з'явилася можливість проводити реєстрацію ЕКГ з відключенням частотно-вибіркової фільтрації.

Не менші проблеми виникають і при використанні іншого популярного підходу до підвищення співвідношення сигнал-шум, котрий застосовується в сучасних цифрових електрокардіографах і заснований на усередненні накопиченої послідовності циклів ЕКГ. Справа в тому, що навіть у здорових людей в стані спокою частота серцевих скорочень (ЧСС) не є постійною величиною, причому відбуваються нерівномірні зміни тривалостей окремих фрагментів ЕКГ. Наприклад, тривалість комплексу QRS в меншій мірі пов'язана зі зміною ЧСС, ніж тривалості зубців P і T [6].

Тому, використання тривіальних алгоритмів усереднення серцевих циклів в тимчасовій області неминуче призводить до «розмиття» інформативних фрагментів і, як наслідок, до помилок у вимірі значень діагностичних ознак, зосереджених на цих фрагментах.

Традиційна електрокардіографія передбачає реєстрацію ЕКГ в декількох відведеннях, кожне з яких несе інформацію про різниці потенціалів між двома певними точками електричного поля серця. Зазвичай використовують 12 відведень – три стандартних I, II, III і три посилених aVR, aVL, aVF відведень від кінцівок, а також шість грудних відведень V1, ..., V6.

Органічні ураження і функціональні порушення серцево-судинної системи породжують відповідні зміни полярності, амплітуди, тривалості і форми характерних сегментів і зубців ЕКГ. Зрозуміло, що при постановці діагнозу лікарі орієнтуються не тільки на такі зміни, але і враховують загальну клінічну картину і приймають «неформальні» рішення, спираючись на свій попередній досвід і інтуїцію. Тому, в медичній практиці добре відомі приклади, коли кілька досвідчених кардіологів по-різному інтерпретують одну і ту ж ЕКГ.

Комп'ютерна електрокардіографія, так чи інакше, заснована на формальних алгоритмах аналізу відхилень параметрів ЕКГ від популяційних норм [7]. При цьому діагностичну цінність несуть як «грубі», так і «тонкі» відхилення форми ЕКГ.

«Грубими» відхиленнями будемо називати патологічний (широкий і глибокий) зубець Q, істотно розширений QRS-комплекс [8] і ряд інших діагностичних ознак ЕКГ, виявлення яких не становить особливих труднощів як при візуальному, так і при комп'ютерному аналізі ЕКГ. Значно більші проблеми викликає комп'ютерний аналіз «тонких» змін, які практично непомітні при візуальному аналізі ЕКГ.

Прикладами таких «тонких» змін можна вважати так звані альтернацію (рис. 1.3) і симетризацію (рис. 1.4) зубця Т, знаходження яких дозволяє отримати важливу діагностичну інформацію.

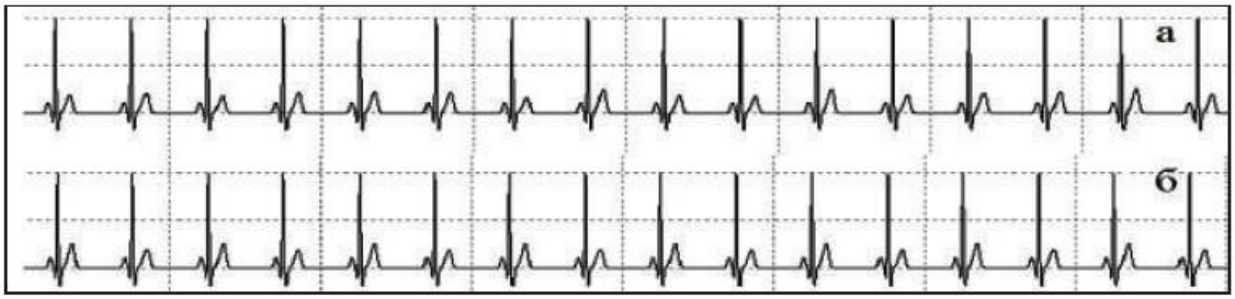


Рисунок. 1.3. ЕКГ з випадковим спотворенням (а)
і альтернацією (б) зубця Т.

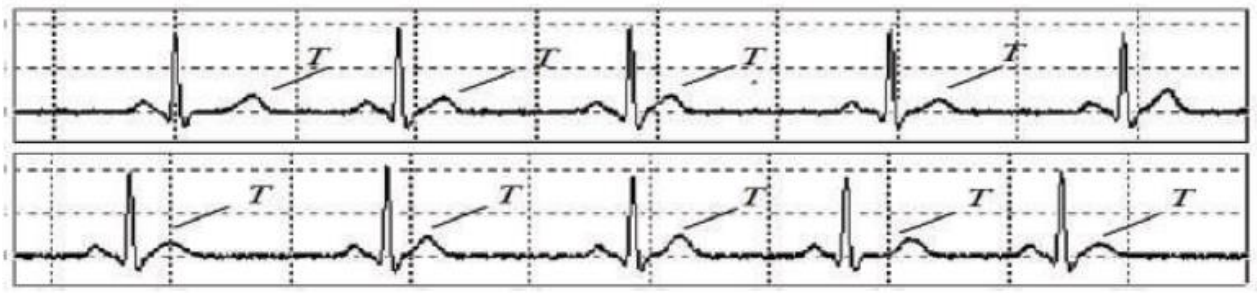


Рисунок. 1.4. ЕКГ з діагностично значущими відмінностями симетрії зубця Т.

Надійне розпізнавання таких дуже схожих сигналів - одна з функцій фазаграфії.

Інтелектуальні інформаційні технології (ІТ) отримують все більше поширення при вирішенні різних прикладних задач. Загальна концепція побудови таких технологій була розроблена в Україні ще в дев'яностих роках минулого століття.

На відміну від традиційних ІТ, заснованих на процедурах обробки числових даних, інтелектуальні ІТ оперують узагальненими поняттями (образами), які дають більш повну інформацію про зовнішнє середовище, а аналіз таких образів породжує цілісну картину досліджуваних явищ.

Важливий інструмент образного уявлення завдання - когнітивна комп'ютерна графіка, яка дозволяє або відразу побачити рішення задачі, або отримати підказку для його знаходження [2]. Такі можливості когнітивної графіки насамперед обумовлені тим, що людський мозок (на відміну від комп'ютера) набагато легше сприймає і інтерпретує графічний образ, ніж числові дані, які його породили.

Досвідчений лікар-кардіолог навряд чи зможе поставити діагноз, аналізуючи лише послідовність чисел (дискретних відліків), що відображають процес зміни в часі електричної активності серця. Але якщо ці ж відліки надати лікарю в вигляді графіка звичної ЕКГ, то при інтерпретації такого графіка включається механізм образного сприйняття

інформації, який в значній мірі спирається на аналогії, попередній досвід та інтуїцію лікаря [53].

Таким чином, якщо вдало представити дані будь-якої задачі у вигляді когнітивного графічного образу, то при аналізі такого образу рішення може бути знайдено без складних обчислень.

Інтелектуальні ІТ, що використовують когнітивну комп'ютерну графіку, набули поширення в самих різних сферах застосування. Фазаграфія - одна з таких перспективних технологій, яка дозволила розширити функціональні можливості одноканальної ЕКГ.

Термін «технологія» походить від грецького слова «téchnë» (мистецтво, майстерність, вміння), а задача технології як науки стоїть у виявленні загальних закономірностей з метою визначення та використання на практиці найбільш ефективних і економних виробничих процесів.

Слідуючи цим тлумаченням, введемо таке визначення. Фазаграфія - це наукомістка ІТ обробки сигналів різної фізичної природи, що мають складну форму, яка на основі ланцюжка інтелектуальних обчислювальних процедур забезпечує перехід від спостережуваного сигналу з локально-зосередженими діагностичними ознаками (сировина технології) до інформації, орієнтовної на конкретного користувача (продукт технології).

При цьому головне наукове завдання методу фазаграфії спрямоване на виявлення загальних закономірностей зазначених сигналів з метою визначення та використання на практиці найбільш ефективних обчислювальних процедур, що забезпечують такий перехід.

Для того, щоб продемонструвати інтелектуальні можливості фазаграфії, проаналізуємо думки про природний інтелект.

На основі цього аналізу і власних уявлень можна заключити, що інтелектуальна ІТ повинна володіти, принаймні, такими властивостями:

- адаптація - здатність пристосовуватися до ситуацій зовнішнього середовища, які постійно змінюються;

- здатність до навчання - здатність покращувати свої споживчі властивості в міру експлуатації;
- узагальнення - здатність розпізнавати класи ситуацій зовнішнього середовища;
- інваріантнісь - нечутливість до збурень зовнішнього середовища;
- прогнозування - можливість відновлювати загальну картину і передбачати майбутні ситуації по спостереженню тільки частин (фрагментів) зовнішнього середовища;
- розуміння - здатність осмислювати дійсність на основі порівняння поточних характеристик зовнішнього середовища з їх минулим значеннями;
- гнучкість - стійкість до можливих невдач і можливість корекції прийнятих рішень;
- взаємозамінність - використання альтернативних методів аналізу зовнішнього середовища;
- доступність - здатність надавати інформацію у формі, зрозумілій для сприйняття конкретним користувачем з урахуванням його кваліфікації.

1.4. Діагностичні ознаки ЕКГ у фазовому просторі

Перехід від скалярного представлення ЕКГ в тимчасовій області z (t) до векторного представлення в фазових координатах дозволяє ефективно відновити корисний сигнал, спотворений збуреннями, розділити z(t) на окремі серцеві цикли (R-R - інтервали) і провести селекцію нетипових циклів.

Виявилося, що цим не обмежуються переваги фазових портретів ЕКГ. Модельні експерименти показали, що традиційні діагностичні ознаки ЕКГ більш виразно проявляються при відображенні ЕКГ в фазових

координатах z(t), z’(t), ніж при її поданні в тимчасовій області z(t) (рис. 1.5) [53].

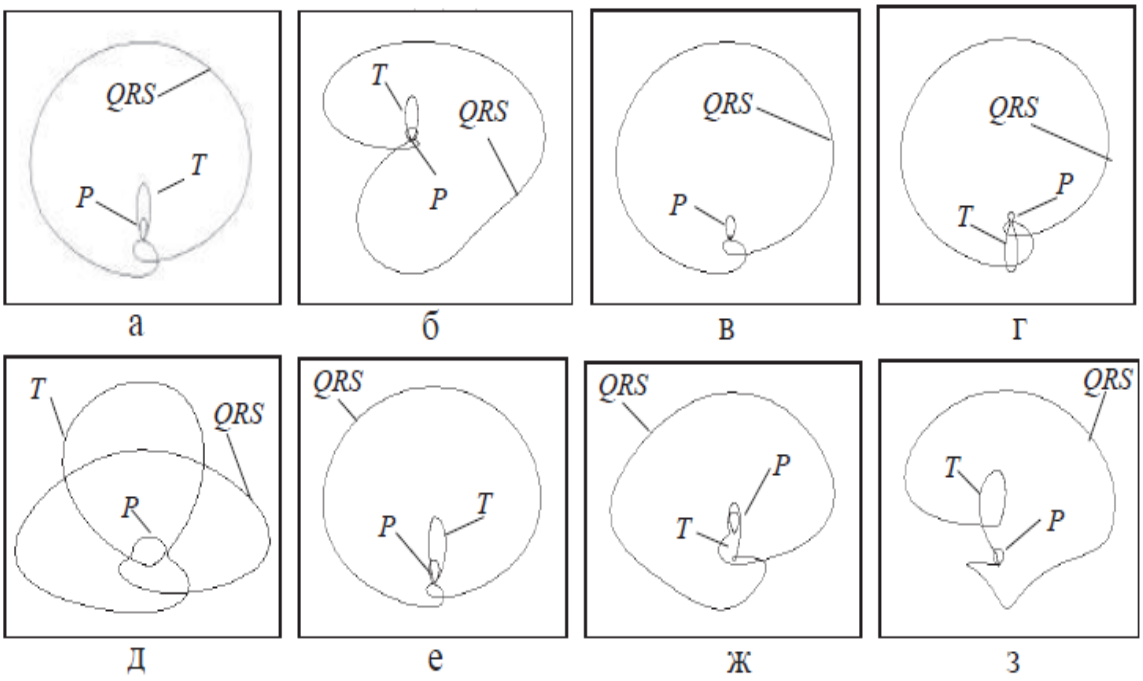


Рисунок. 1.5. Характерні зміни форми фазових портретів ЕКГ при зміні діагностичних ознак.

При позитивних зубцях Р, Т і незмінному стані QRS - комплексу фазовий портрет ЕКГ має вигляд характерного графічного способу, на якому відображаються три петлі, які відповідають зубцям Р, Т і комплексу QRS (рис. 1.5, а).

При патологічному (широкому і глибокому) зубці Q графічний образ розгортається за годинниковою стрілкою (рис. 1.5, б).

Плоский (рис. 1.5, в), негативний (рис. 1.5, г), надмірно високий (рис. 1.5, д) або асиметричний (рис. 1.5, е) зубець Т викликає адекватні зміни розміру і орієнтації відповідної петлі на фазовому портреті [53].

Характерний зсув вниз (рис. 1.5, ж) при депресії або вгору (рис. 1.5, з) при елевації сегмента ST зазнає відповідний фрагмент фазового портрета [53].

Виявлено, що діагностично значущі відхилення реальних ЕКГ в тимчасовій області викликають подібні зміни форми фазових портретів. Це добре видно з прикладів фазових портретів одноканальних записів ЕКГ, що зберігаються в спеціалізованих базах даних інтернет порталу PhisioNet (рис. 1.6).

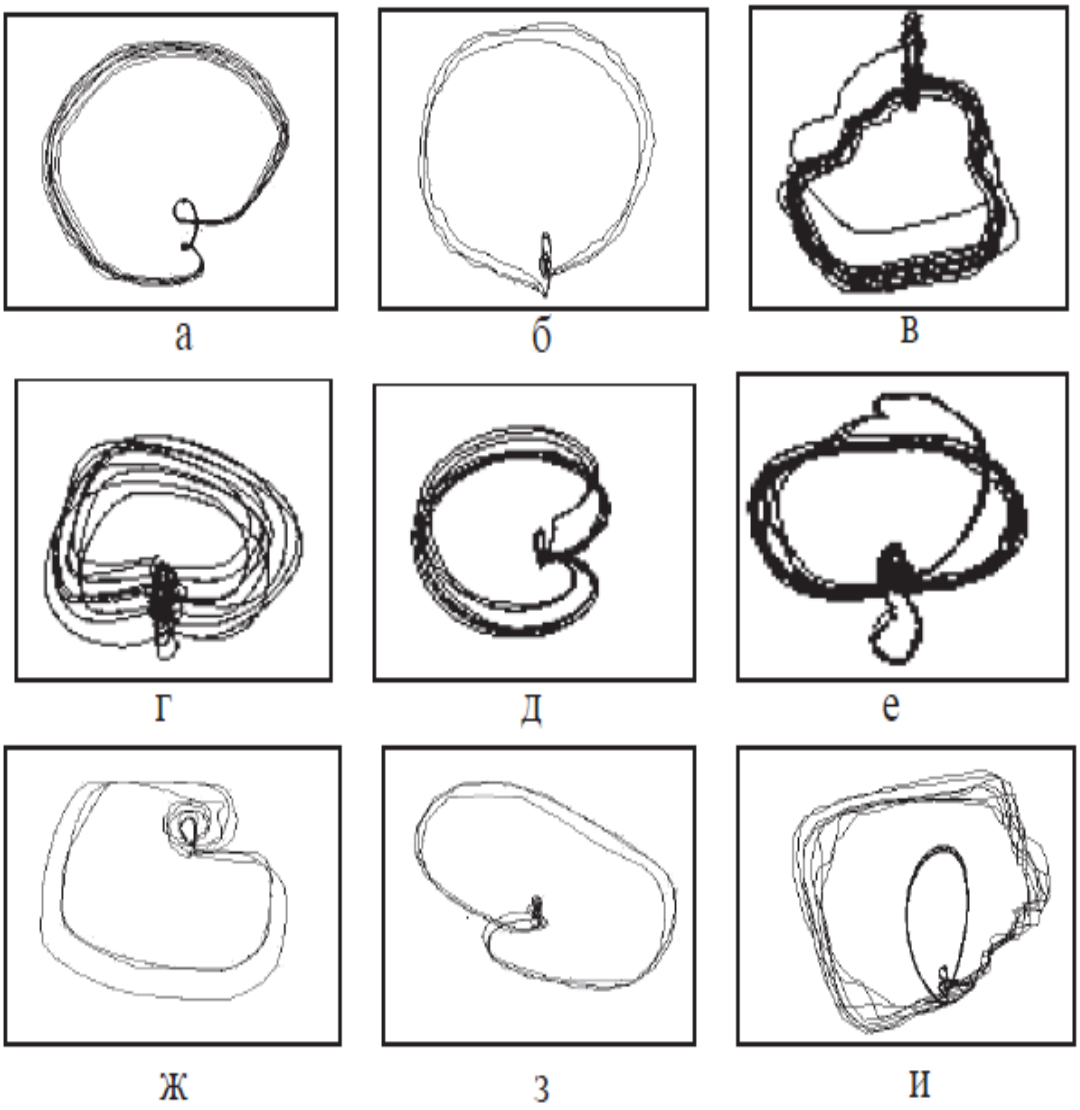


Рис. 1.6. Фазові портрети реальних ЕКГ різних пацієнтів.

Висновки до розділу 1

На підставі проведеного аналізу можна зробити висновок, що сучасна наука має у своєму розпорядженні ефективні методи і

обчислювальні алгоритми, які дозволяють з різних сторін оцінити ступінь хаотичність часових рядів, породжуваних медико-біологічними системами.

Було досліджено зміну ентропії показників ЕКГ і їх фазові портрети. На основі цього, було запропоновано новий метод оцінки зміни у часі ентропії показників електрокардіограми на основі її фазового портрету та аналізу його випуклої оболонки.

РОЗДІЛ 2
БАЗОВІ ПЕРЕДУМОВИ ТА ОСНОВИ ЗАПРОПОНОВАНИХ МЕТОДІВ

В основі багатьох математичних методів дослідження хаотичності динамічних рядів лежить відома формула ентропії Шеннона H [21].

$$H = -\sum_{i=1}^{n} p_i \log_2 p_i , \tag{2.1}$$

запропонована для оцінки невизначеності системи, яка знаходиться в одному із n станів з імовірностями p_i, $i = 1,...,n$. Чим більше величина H, тим далі система знаходиться від упорядкованого стану, причому максимальне значення ентропії Шеннона досягається, коли всі p_i рівні, тобто стани системи рівноймовірні [22].

Для аналізу хаотичності кінцевого часового ряду

$$A = a_1, a_2, \dots, a_N , \tag{2.2}$$

елементи якого являють собою скалярні величини $a_i \in [a_i^-, a_i^+]$, $i = 1,...,N$, що належать заданими інтервалам, в ряді робіт, зокрема, в [23], пропонується прямо скористатися формулою (2.1), оцінивши ймовірності (частоти) попадання значень ряду (2.2) в зазначені інтервали.

Однак величина (2.1) інваріантна відносно перестановок елементів ряду (2.2). Тому при безпосередньому використанні формули (2.1) оцінюється не хаотичність послідовності (2.2), а невизначеність випадкової величини, що породжує цю послідовність, що не одне і те ж [52].

Наприклад, дві послідовності - регулярна

$$A_1 = 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0$$

і хаотична

$$A_2 = 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1$$

матимуть однакові значення ентропії $H = 1$.

Оцінити хаотичність часового ряду при безпосередньому використанні формули (2.1) можна лише в тому випадку, якщо послідовність (2.2) обробляти по частинах (у вікнах) або оцінювати зміну ентропії по ходу накопичення даних [52]

$$H(k) = -\sum_{i=1}^{n} p_i(k) \log_2 p_i(k), \quad k \le N. \tag{2.3}$$

Існують й інші підходи до вирішення даного завдання, наприклад використання умовної ентропії [24]. Для цього послідовність (2.2) розбивається на підпослідовності (патерни)

$$x(i) = [a(i), a(i+1), ..., a(i+m-1)], \qquad i = 1, ..., N-m+1 \tag{2.4}$$

з розмірністю вкладення m, оцінюються ймовірності (частоти) появи конкретних патернів і обчислюється умовна ентропія $E(m \mid m-1)$ як приріст Шеннонівської ентропії при переході від патернів з розмірністю $m-1$ до m, тобто

$$E(m \mid m-1) = E(m) - E(m-1) = -\sum_{i=1}^{N-m+1} p_m \ln p_m + \sum_{i=1}^{N-m+2} p_{m-1} \ln p_{m-1}. \tag{2.5}$$

Згідно [25], перехід від послідовності скалярних величин (2.2) до векторів (2.4) можна інтерпретувати як перехід в фазовий (псевдофазовий) простір, в якому і проводиться подальший аналіз.

Для більш повного аналізу складності медико-біологічних систем здобули популярність інші ентропійні оцінки, зокрема, апроксимаційна ентропія (Approximation Entropy) [26], яку в російськомовній літературі називають ентропія подібності [27].

При її обчисленні вихідна послідовність також розбивається на патерни (2.4), близькість яких в фазовому просторі оцінюється відстанню

$$d\left[x(i), x(j)\right] = \max_{k=1,\ldots,m}\left\{\left|a(i+k-1) - a(j+k-1)\right|\right\} \tag{2.6}$$

між усіма парами $x(i)$ і $x(j)$, $i = 1,\ldots, N-m+1, j = i,\ldots, N-m+1$. Далі оцінюються ймовірності (частоти) появи в послідовності (2.2) таких пар патернів, відстань між якими не перевищує заданий поріг d_0 [28]. Для цього визначаються величини

$$C_r^{(m)}(i) = \frac{U^{(m)}(i)}{N-m+1}, \tag{2.7}$$

в яких $U^m(i)$ – кількість значень $d\left[x(i), x(j)\right]$, задовольняючих умову

$$d\left[x(i), x(j)\right] \le d_0, \quad j = 1,\ldots, N-m+1, \tag{2.8}$$

або, в еквівалентній формі запису,

$$C_r^{(m)}(i) = \frac{1}{N-m+1}\sum_{j=1}^{N-m+1}\Theta(d_0 - d[x(i), x(j)]), \tag{2.9}$$

де

$$\Theta(\eta) = \begin{cases} 1, \text{ если } \eta \geq 0, \\ 0, \text{ если } \eta < 0 \end{cases} \tag{2.10}$$

– функція Хевісайда.

Використовуючи вирази (2.7) або (2.9), можна обчислити величину

$$\theta^{(m)}(r) = \frac{1}{N - m + 1} \sum_{i=1}^{N-m+1} \ln C_r^{(m)}(i), \tag{2.11}$$

яку також прийнято називати безумовною ентропією [24].

Апроксимаційна ентропія визначається як приріст безумовної ентропії (2.11) при переході від послідовності паттернів довжиною m до послідовності довжиною $m+1$ за формулою

$$ApEn = \theta^{(m)}(r) - \theta^{(m+1)}(r). \tag{2.12}$$

Зазвичай [29], для розрахунку використовують наближену формулу

$$ApEn = \frac{1}{N - m} \sum_{i=1}^{N-m} \ln \frac{C_r^m}{C_r^{m+1}}. \tag{2.13}$$

Значення апроксимаційної ентропії для регулярної і хаотичної послідовностей A_1 и A_2 будуть відповідно дорівнювати $ApEn(A_1) = 0,08$ і $ApEn(A_2) = 0,4$.

Слід зауважити, що оцінка (2.13) є зміщеною оцінкою Апроксимаційні ентропії, значення якої асимптотично зростає зі збільшенням кількості елементів часового ряду [27]. Для усунення цього недоліку в роботі [30] запропонована невелика модифікація методу обчислення $ApEn$, яка передбачає використання величин

$$C_r^{(m)}(i) = \frac{\sigma + U^{(m)}(i)}{N - m + 1}, \tag{2.14}$$

чисельник яких, на відміну від (2.7), містить фіксовану добавку $\sigma > 0$, а в умову (2.8) вводиться додаткове обмеження $i \neq j$ [52].

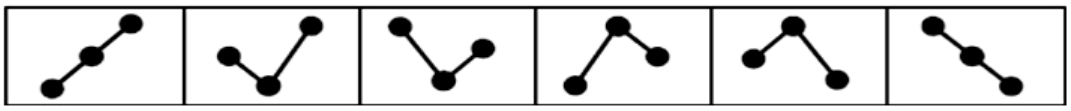
Подальша модифікація апроксимаційної ентропії дозволила запропонувати ентропію шаблонів (Sample entropy) [31], при обчисленні якої також використовується обмеження $i \neq j$, а саму ентропію визначає вираз

$$SampEn = -\ln \frac{U^{m+1}(r)}{U^m(r)} = \ln U^m(r) - \ln U^{m+1}(r) \tag{2.15}$$

Така оцінка, на відміну від (2.12), є незміщеною, а її значення практично не залежить від кількості елементів часового ряду [31].

Ще одним засобом оцінки хаотичності часових рядів є переставна ентропія (Permutation Entropy) [32], яка заснована на аналізі патернів (2.4) не з точки зору відстані між ними в фазовому просторі, а з точки зору їх форми (класу).

Для класифікації патернів кожному його елементу присвоюється мітка, що характеризує рівень елемента по відношенню до сусіднього, наприклад, мітки 0, 1, 2. Тоді при $m = 3$ можна розрізняти 3! (шість) класів патернів $\pi_1, \pi_2, \ldots, \pi_6$, утворених трійкою послідовних значень відповідного рівня (рис. 2.1) [52].



$\pi_1 = 012$ $\pi_2 = 102$ $\pi_3 = 201$ $\pi_4 = 021$ $\pi_5 = 120$ $\pi_6 = 210$

Рисунок 2.1. Класи патернів переставної ентропії при $m = 3$.

Переставна ентропія обчислюється за формулою

$$PE(m) = -\sum_{j=1}^{N-m+1} p(\pi_i)\log_2 p(\pi_i), \qquad i = 1,...m!, \tag{2.16}$$

в якій $p(\pi_i)$ – частота появи патерну i-го класу при обробці вихідної послідовності (2.2) ковзаючим вікном з m точок [52].

Значення переставної ентропії (2.16) для регулярної і хаотичної послідовностей A_1 і A_2 відповідно рівні $PE(A_1) = 1$ і $PE(A_2) = 1{,}96$.

У тому випадку, коли елементи часового ряду (2.2) утворюють не скалярні величини a_i, а R-вимірні вектори $\vec{a}_i = (a_{i1},...,a_{iR})$, для інтегральної оцінки хаотичності такого ряду використовують кореляційний інтеграл [33]. Для його обчислення компоненти векторів $\vec{a}_i = (a_{i1},...,a_{iR})$ попередньо нормуються, і подальша обробка здійснюється в фазовому просторі з нормованими координатами $a^*_{i1},...,a^*_{iR}$.

Оцінку кореляційного інтегралу $C(d_0, N)$ при кінцевому числі N спостережень дає вираз

$$C(d_0, N) = \frac{1}{N^2}\sum_{m,l=1}^{N} \Theta(d_0 - d(\vec{a}_l, \vec{a}_m)) \quad , \tag{2.17}$$

в якому $d(\vec{a}_l, \vec{a}_m)$ – евклідова відстань між l-м і m-м векторами в R- *вимірному* фазовому просторі, d_0 – заданий поріг, а $\Theta(\cdot)$ – функція Хевісайда, яка визначається виразом (2.10) [52].

Тим самим оцінюється середня ймовірність того, що нормовані вектори $\vec{a}_i = (a_{i1},...,a_{iR})$, $i = 1,...,N$, які утворюють оброблювану послідовність, виявляться на відстані, що не перевищує заданий поріг d_0.

Відомо [34, 35], що для фрактальних (самоподібних) множин характерна залежність:

$$\lim_{d_0 \to 0} C(d_0, N) = d_0^D \tag{2.18}$$

де D – розмірність атрактора (Fractal dimension), яка також оцінює хаотичність тимчасових сигналів в R- *вимірному* фазовому просторі [36].

Із (2.18) випливає, що за обчисленим кореляційним інтегралом (2.17), можна визначити розмірність атрактора за формулою

$$D = \lim_{d_0 \to 0} \frac{\log(C(d_0))}{\log d_0}. \tag{2.19}$$

На підставі (2.19) робимо висновок, що фрактальну розмірність D можна визначити за нахилом прямої лінії регресії, побудованої за експериментальними даними в координатах $\ln C(d_0)$, $\ln d_0$ [52].

Для зручності виразимо цю залежність в явному вигляді

$$D = \frac{N\sum_{i=1}^{N} \log d_0(i) \cdot \log(C(d_0(i))) - \sum_{i=1}^{N} \log d_0(i) \sum_{i=1}^{N} \log(C(d_0(i)))}{N\sum_{i=1}^{N} (\log d_0(i))^2 - (\sum_{i=1}^{N} \log d_0(i))^2}. \tag{2.20}$$

Відомий також інший підхід до обчислення фрактальної розмірності [36], заснований на розмірності Мінькowського D_{bc} (Box-counting dimension). Метод полягає у визначенні мінімального числа $M(d)$ «комірок» фазового простору діаметром $d \to 0$, якими можна «накрити» вихідну множину точок [52].

Після ряду перетворень отримаємо в явному вигляді остаточну формулу для обчислення розмірності Міньковського як кутовий коефіцієнт прямої лінії регресії, побудованої за експериментальними точкам в координатах $\ln M(d), \ln d_0(i)$:

$$D_{bc} = \frac{N\sum_{i=1}^{N} \ln d_0(i) \cdot \ln M(d(i)) - \sum_{i=1}^{N} \ln d_0(i) \sum_{i=1}^{N} \ln M(d(i))}{N\sum_{i=1}^{N} (\ln d_0(i))^2 - (\sum_{i=1}^{N} \ln d_0(i))^2}. \qquad (2.21)$$

Ще одним досить простим і ефективним методом аналізу нестаціонарних часових рядів є так званий R/S- аналіз, заснований на обчисленні індексу Хьорста H_0 (Hurst exponent) [22] за модифікованим часовим рядом

$$z_i = \sum_{i=1}^{k} (a_i - a_{cp}), \qquad i = 1,...,k, \qquad (2.22)$$

де $k \le N$, a_{cp} – середнє значення елементів у вихідній послідовності (2.2) із N елементів, або ж по іншому модифікованому ряду у вигляді логарифмів відносин

$$z_i = \log(\frac{a_{i+1}}{a_i}), \quad i = 1,2,...,N-1. \qquad (2.22)$$

Для обчислення індексу Хьорста досить визначити розмах

$$R^{(k)} = \max_{i=1,...,k}(z_i) - \min_{i=1,...,k}(z_i), \qquad (2.23)$$

і стандартне відхилення

$$S^{(k)} = \sqrt{\frac{1}{k-1}\sum_{i=1}^{N}(a_i - a_{cp})^2}\,, \qquad (2.24)$$

при деяких фіксованих значеннях $k \le N$.

Встановлено [37], що для більшості стохастичних рядів виконується співвідношення

$$R^{(k)} / S^{(k)} = Lk^{H_0}\,, \qquad (2.25)$$

тобто нормований розмах пропорційний ступеню k, де H_0 – показник Хьорста, а L – деяка константа.

Для визначення показника Хьорста в зручній формі прологарифмуємо рівняння (2.25):

$$\log(R^{(k)} / S^{(k)}) = \log L + H_0 \log k\,. \qquad (2.26)$$

і, також як в попередніх випадках, виразимо значення індексу Хьорста у вигляді кутового коефіцієнту прямої лінії регресії (2.26), яка найкращим чином апроксимує експериментальні точки $\log(R^{(k)} / S^{(k)})$ і $\log k$, знайдені при різних фіксованих значеннях $k \le N$ [52].

В результаті, після ряду очевидних перетворень, заснованих на обробці методом найменших квадратів значень ряду при всіх $k = 1,...,N$, отримаємо

$$H_0 = \frac{\log k \sum_{k=1}^{N} \log k \log(R^{(k)} / S^{(k)}) - \sum_{k=1}^{N} \log k \sum_{k=1}^{N} \log(R^{(k)} / S^{(k)})}{\log k \sum_{k=1}^{N} (\log k)^2 - (\sum_{k=1}^{N} \log k)^2}. \quad (2.27)$$

Зрозуміло, наведений аналіз існуючих методів не претендує на повноту, тим більше, що методи оцінки хаотичності часових рядів постійно розвиваються. Розглянуто лише найбільш популярні підходи і отримано остаточні формули, зручні для практичного застосування [52].

На підставі проведеного аналізу можна зробити висновок, що сучасна наука має у своєму розпорядженні ефективні методи і обчислювальні алгоритми, які дозволяють з різних сторін оцінити ступінь хаотичність часових рядів [52].

Нехай ЕКГ-сигнал $z(t)$, що спостерігається в дискретні моменти часу $t_k \equiv k\Delta$, $k = 1,..,K$, представлений кінцевою послідовністю окремих циклів $z_1(k)$, $z_2(k)$, …, $z_M(k)$, де Δ - крок квантування за часом, M - загальне число циклів.

Слідуючи [12] будемо апроксимувати кожен m-й цикл $z_m(k)$ деякою функцією $\varphi(k, \theta_1,..., \theta_G)$, заданою з точністю до кінцевого числа невідомих параметрів $\theta_1,...\theta_G$. Для визначення оптимальних значень цих параметрів скористаємося критерієм мінімума суми квадратів ухилень

$$Cr = \sum_{k=1}^{K_m} [\varphi(k, \theta_1,..., \theta_G) - z_m(k)]^2 \to \min, \quad (2.28)$$

де K_m - число дискретних відліків $z(t)$ на m -му циклі.

У цьому випадку кожен окремий цикл $z_m(k)$ представляє точка (вектор) $\vec{\theta}_m = (\theta_{1m},..., \theta_{Gm})$ в G-вимірному просторі параметрів, а

послідовність спостережених циклів $z_1(k)$, $z_2(k)$, …, $z_m(k)$ породжує в цьому просторі фазову траєкторію, яка однозначно відповідає спостереженому сигналу $z(t)$.

В якості функції, яка з прийнятною точністю описує цикли реальних ЕКГ, будемо використовувати суму несиметричних гаусових функцій

$$\varphi(k) = \sum_{\nu} A_\nu \exp\left[-\frac{(k-\mu_\nu)^2}{2[\sigma_\nu(k)]^2} \right], \quad \nu \in \{P, Q, R, S, ST, T\}, \tag{2.29}$$

в яких параметри A_ν і μ_ν визначають значення амплітуд і моментів часу, коли ν-й фрагмент приймає екстремальні значення, а функція $b_\nu(k)$, визначається виразом

$$b_\nu(t) = \begin{cases} b_\nu^{(1)} & \forall\ t \le \mu_\nu, \\ b_\nu^{(2)} & \forall\ t > \mu_\nu. \end{cases} \tag{2.30}$$

при $b_\nu^{(1)} \neq b_\nu^{(2)}$ функція (2.30) дає можливість окреслити несиметричні фрагменти, зокрема, несиметричний зубець T, якщо $b_T^{(1)} \neq b_T^{(2)}$.

Звідси випливає, що основні діагностичні показники ЕКГ, в тому числі, додатковий показник β_T симетрії ділянки реполяризації, можуть бути обчислені за допомогою співвідношень, наведених в таблиці 2.1.

Експериментальні дослідження показали [13], що навіть при високому рівні перешкод похибка оцінки зазначених показників за реальними ЕКГ різної форми не перевищує 1%.

Таблиця 2.1.

Формули для обчислення діагностичних показників

Показники	**Формули для обчислення**
Додатність зубця Q	$\Delta_Q \stackrel{\Delta}{=} t_Q^{(2)} - t_Q^{(1)} = 3(\sigma_Q^{(1)} + \sigma_Q^{(2)})$
Додатність інтервалу PQ	$\Delta_{PQ} \stackrel{\Delta}{=} t_Q^{(1)} - t_P^{(1)} = \mu_Q - \mu_P + 3(\sigma_p^{(1)} - \sigma_Q^{(1)})$
Додатність інтервалу QT	$\Delta_{QT} \stackrel{\Delta}{=} t_T^{(2)} - t_Q^{(1)} = \mu_T - \mu_Q + 3(\sigma_Q^{(1)} + \sigma_T^{(2)})$
Додатність комплексу QRS	$\Delta_{QRS} \stackrel{\Delta}{=} t_S^{(2)} - t_Q^{(1)} = \mu_S - \mu_Q + 3(\sigma_Q^{(1)} + \sigma_S^{(2)})$
Додатність RR - інтервалу	$\Delta_{RR} = \mu_R[m] - \mu_R[m-1], \quad m \geq 2$
Глубина зубця Q	A_Q
Зміщення сегмента ST	A_{ST}
Амплітуда зубца T	A_T
Симетрія зубца T	$\beta_T = \frac{\sigma_T^{(2)}}{\sigma_T^{(1)}}$

Теорія хаосу і синергетики дозволяють більш повно розкрити і проаналізувати механізми функціонування живої складно-організованої системи, що поєднує риси порядку і безладу, визначеності і невизначеності, організованості і дезорганізованості [7].

Для інтегральної оцінки динаміки хаотичності показників в процесі реєстрації ЕКГ пропонується наступний метод.

Потрібно оцінити хаотичність часового ряду

$$a_1, a_2, \ldots, a_M, \tag{2.31}$$

елементи якого являють собою послідовності значення будь-якого з показників, зазначених в таблиці 1, наприклад, послідовність значень *RR* - інтервалів або ж послідовність значення показника β_T від циклу до циклу.

Розділимо ряд (2.31) на L послідовних вікон, що містять W точок, в кожному такому l -м вікні оцінимо хаотичність H_l значень досліджуваного показника і обчислимо відношення H_l до хаотичності H_1 елементів в першому вікні:

$$h_l = \frac{H_l}{H_1} \cdot 100\,\%, \quad l = 1, ..., M, \tag{2.32}$$

вважаючи, що $H_1 \neq 0$.

Для оцінки H_l може бути використаний будь-який математичний метод аналізу хаотичності елементів динамічного ряду [28]. Зокрема таку оцінку можна проводити на основі обчислення шенонівскої ентропії

$$H_l = -\sum_{j=1}^{J} p_{jl} \log p_{jl}, \tag{2.33}$$

де p_{jl} - частота потрапляння j -й інтервал $\Delta_j = [a_j^-, a_j^+]$, $j = 1,..., J$ значень часового ряду, які спостерігаються в l -му вікні. Межі a_j^-, a_j^+ інтервалів Δ_j , В тому числі і при знакозмінних елементах a_i , визначають співвідношення

$$a_j^- = \min a_i + \delta(j-1), \quad a_j^+ = \min a_i + \delta j, \qquad j = 1,..., J, \tag{2.34}$$

де δ - заданий поріг нечутливості до змін показника.

Процедура (2.32) може бути реалізовані при зсуві $l+1$ - го вікна по відношенню l - му на ширину вікна W або ж при зсуві вікон на одну точку

(режим ковзного вікна). Зрозуміло, що в останньому випадку обсяг необхідних обчислень буде більше, але графік зміни ентропії буде більш плавним. При цьому вид такого графіка залежить від ширини вікна W і порога δ (рис. 2.2) [28].

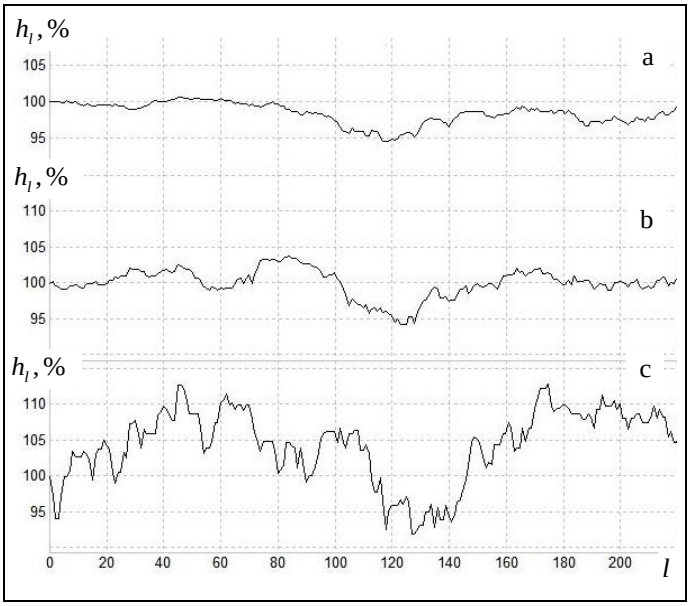


Рисунок. 2.2. Графіки ковзної ентропії $h(l)$ при оцінці хаотичності показника β_T :

a: $W = 100$ точ., $\delta = 0{,}1$ од.; b: $W = 60$ точ., $\delta = 0{,}04$ од.; c: $W = 30$ точ., $\delta = 0{,}02$ од.; .

Для інтегральної оцінки хаотичності показника в процесі спостереження ЕКГ пропонується перейти від ряду дискретних значень $h(l)$, обчислених методом ковзного вікна, до фазового портрету ентропії на площині $h(l), \dot{h}(l)$, де $\dot{h}(l)$ - оцінка першої виро з водної $h(l)$ в l-ій точці.

Незважаючи на те, що процедура чисельного диференціювання зашумленних даних відноситься до некоректно поставленого математичного завданням, застосування спеціальних процедур фільтрації і регуляризації [14] дозволило отримувати прийнятні оцінки похідної $\dot{h}(l)$.

В результаті вдається побудувати наочний графічний образ фазового портрета ентропії у вигляді точок на площині $h(l), \dot{h}(l)$ (рис. 2.3).

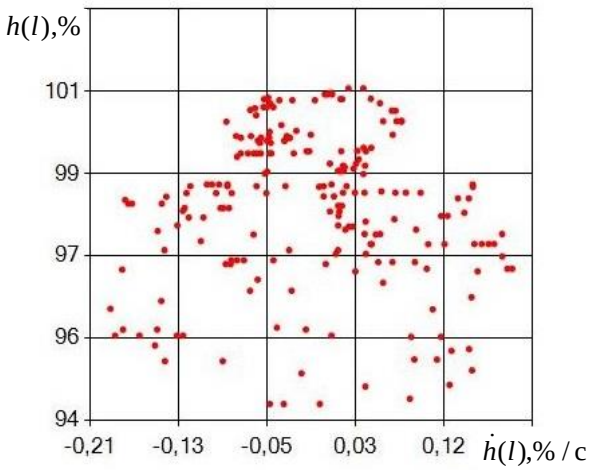


Рисунок. 2.3. Фазовий портрет ковзної ентропії показника β_T реальної ЕКГ.

Слід зауважити, що класична шенонівська ентропія (15) інваріантна відносно перестановок елементів у вікні. Наприклад, дві послідовності - регулярна

1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0

і хаотична

0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1

матимуть однакові значення ентропії $H = 1.$

Тому для більш глибокого вивчення динаміки хаотичності можна в кожному l -му вікні замість оцінки (2.33) обчислювати *переставну ентропію* (Permutation Entropy) [17], яка заснована на аналізі форми характерних патернів.

Для реалізації такої можливості ми дещо модернізували відому процедуру обчислення переставної ентропії і по трійці послідовних значень a_{m-1}, a_m, a_{m+1} часового ряду (2.31) оцінювали п'ять класів патернів (рис. 2.4).

Рисунок. 2.4. П'ять класів патернів модернізованої переставної ентропії.

Переставна ентропія обчислюється за формулою

$$PE_l = -\sum_{j=1}^{5} p(\pi_{jl}) \log_2 p(\pi_{jl}), \qquad (2.35)$$

в якій $p(\pi_j)$ - частота появи патерна j -го класу в l -му вікні.

Класи патернів однозначно визначаються послідовною перевіркою наступних умов:

клас π_1 , якщо $(a_m - a_{m-1}) > h \wedge (a_m - a_{m+1}) > h$,

клас π_2 , якщо $(a_{m-1} - a_m) > h \wedge (a_{m+1} - a_m) > h$,

клас π_3 , якщо $(a_m - a_{m-1}) > h \vee (a_{m+1} - a_m) > h \vee (a_{m+1} - a_{m-1}) > h$,

клас π_4 , якщо $(a_{m-1} - a_m) > h \vee (a_m - a_{m+1}) > h \vee (a_{m-1} - a_{m+1}) > h$,

клас π_5 , якщо не виконується жодне з наведених співвідношень,

в яких h - заданий поріг нечутливості до локальних змін сигналу.

Фазовий портрет можна будувати також на основі обчислення апроксимаційної ентропії *ApEn* (Approximation Entropy) [18] та інших відомих оцінок хаотичності, огляд яких представлений в статті [11].

При певних умовах між шенонівською ентропією і середньоквадратичним відхиленням (СКВ) існує однозначний зв'язок. Наприклад, при нормальному розподілі випадкової величини, що породжує ряд (2.31), цей зв'язок може бути описаний логарифмічною залежністю

$$H \cong 2{,}05 + \log_2 \text{СКО}. \qquad (2.36)$$

Звідси, здавалося б випливає, що результати аналізу мінливості динамічного ряду (2.33) будуть еквівалентними, якщо в ковзних вікнах замість ентропії (2.33) рахувати СКО спостережуваних значень.

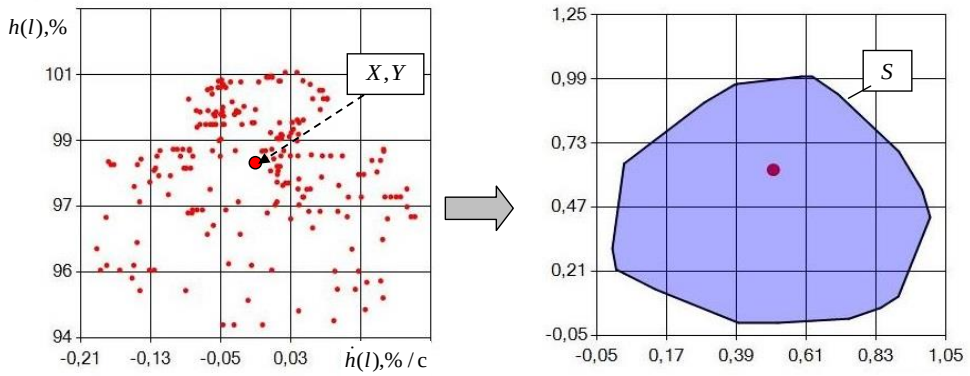


Рисунок 2.5. Фазовий портрет ковзної ентропії (зліва) і його опукла оболонка.

В той же час ентропія, на відміну від СКО, не залежить від значень величини, що спостерігається і тому характеризує не стільки розкид, скільки різноманітність значень цієї величини [21]. Отже, при обробці реальних даних результати будуть відрізнятися.

Для ілюстрації на рис. 2.6 представлені графіки зміни шенонівської ентропії H і СКО, обчислені по одній і тій же послідовності RR - інтервалів [28].

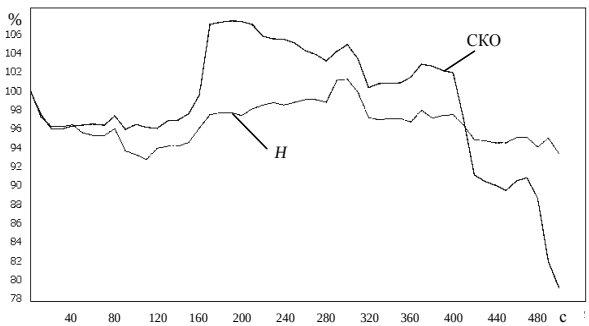


Рисунок 2.6. Графіки зміни ентропії H і середньоквадратичне відхилення СКО.

Вище було згадано про опуклу оболноку, на основі параметрів якої будуть проводитись дослідження. Для початку приведемо визначення опуклоъ оболонки, а потім розгялнемо існуючі алгоритми її пошуку. Важливо зауважити, що під опуклою оболонкою ми будемо вважати мінімальну опуклу оболонку.

Нехай на площині задана кінцева множина точок A. Оболонкою цієї множини називається будь-яка замкнута лінія H без самоперетинів така, що всі крапки з A лежать всередині цієї кривої (рис. 2.7). Якщо крива H є опуклою (наприклад, будь-яка дотична до цієї кривої не перетинає її більше ні в одній точці), то відповідна оболонка також називається опуклою (рис. 2.8). Нарешті, мінімальною опуклою оболонкою (далі коротко МОО) називається опукла оболонка мінімальної довжини (мінімального периметра) (рис. 2.9).

Рисунок 2.7. Оболонка множини точок.

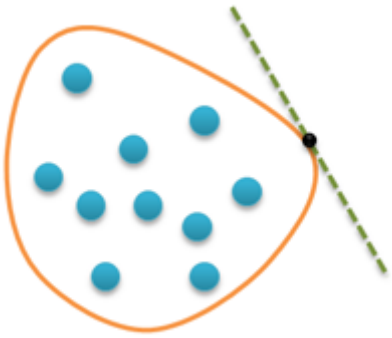


Рисунок 2.8. Опукла оболонка множини точок.

Рисунок 2.9. Мінімальна опукла оболонка множини точок.

Розглянемо алгоримти пошуку мінімальної опуклої оболонки множини точок, заданої на площині. Існує достатня кількість обчислювальних алгоритмів для вирішення даної задачі, проти при написанні магістерської дисертації було прийнято рішення зупинитись на двох з них:

- алгоритм Грехема;
- алгоритм Джарвіса.

Алгоритм Грехема є трикроковим. Будемо вважати розв’язком задачі зв’язний список точок P, які сформують собою МОО. На першому кроці шукається будь-яка точка в множині A, що гарантовано входить в МОО. Неважко здогадатися, що такою точкою буде, наприклад, точка з найменшою x-координатою (найлівіша точка в множині A). Цю точку (будемо називати її стартовою) переміщаємо в початок списку, вся подальша робота буде проводитися з рештою точок. Отже, перший крок алгоритму полягає в тому, щоб першою точкою в P виявилася точка з найменшою x-координатою. Другий крок у алгоритмі Грехема - сортування всіх точок, за ступенем їх лівизни щодо стартової точки. Будемо говорити, що B < C, якщо точка C знаходиться по ліву сторону від вектора RB (рис 2.10).

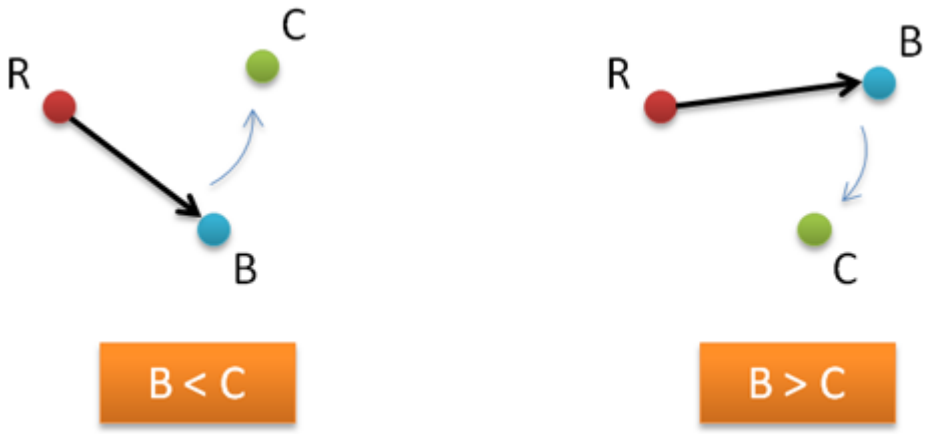


Рисунок 2.10. Другий крок алгоритму Грехема.

Результат сортування можна проілюструвати малюнком (рис. 2.11).

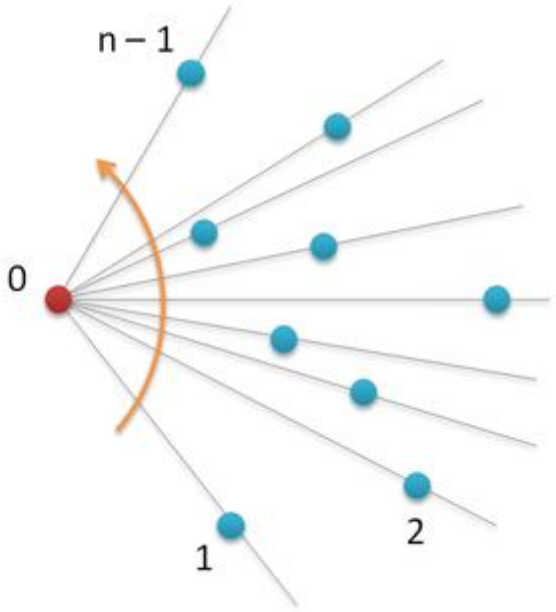


Рисунок 2.11. Результат сортування на другому кроці алгортма Грехема.

Якщо з'єднати точки в отриманому порядку, то отримаємо багатокутник, який, однак, не є опуклим (рис 2.12).

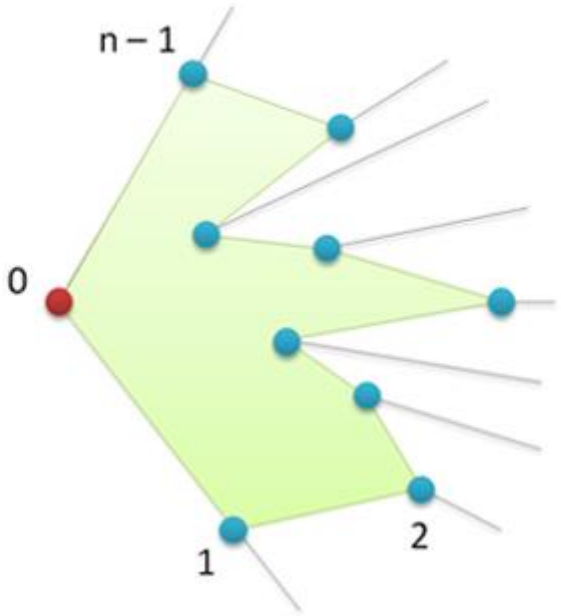


Рисунок 2.12. Багатокутник після об’єднання точок.

Крок 3 полягає у обрізанні кутів. Для цього потрібно пройтися по всіх вершин і видалити ті з них, в яких виконується правий поворот (кут в такій вершині виявляється більше розгорнутого).

Як можна побачити, у алгоритма Грехема є не дуже добра особливість - він не є адаптивним в тому сенсі, що не важливо, скільки вершин в результаті увійде в МВО (три, п'ять, десять або n), все одно час буде лінійно-логарифмічним. Такою адаптивністю володіє алгоритм Джарвіса, який і був використаний при написанні магістерської дисертації.

Висновки до розділу 2

Було розглянуто існуючі алгоритми ентропійних оцінок. На їх основі, було запропоновано новий підхід пошуку діагностичних ознак, оснований на аналізі зміні ентропії у часовому просторі, а саме: оцінці випуклої оболонки фазового портрету.

РОЗДІЛ 3
РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ

3.1. Програмні засоби, які були використані для реалізації

Для реалізації поставлених задач було використано наступні програмні засоби:

1. Microsoft Visual Studio 2017 (мова програмування – C#) система для розробки програмного засобу, що оцінює вплив зовнішніх ознак на організм людини базуючись на хаотичності параметрів ЕКГ.

Microsoft Visual Studio надає розвинений редактор коду, вбудований відладчик та інші засоби, що спрощують розробку додатків на мові C#. Також має наступні переваги:

1. CodeLens - являє із себе підказки, які з'являються над вашим кодом, що надають інформацію про те, які залежно є у цього коду, результати тестів цього методу, хто міняв цей код, пов'язані робочі елементи;

2. IntelliTrace - Можливості IntelliTrace значно підвищують продуктивність налагодження. IntelliTrace автоматично веде журнали виконання коду, запам'ятовує і відзначає події в таймлайнах, які далі можна переглядати, переміщатися і перевіряти стан [53];

3. CodeMap - Ця можливість стане в нагоді при роботі з великими кодовими базами. Підтримується також переміщення по створеній карті з паралельно відкритим кодом. Це допомагає відслідковувати ваше місцезнаходження в коді під час роботи;

4. Доступні інструменти для побудови процесів управління проектами та командною роботою: Team Foundation Server або Visual Studio Online Advanced;

5. Можливість підключення бібліотек з відкритим вихідним кодом.

C#:

1. Стандартизований. Перша версія C # стандартизована в ECMA (Standard ECMA-334 C # Language Specification, 3rd edition (June 2005)) та ISO (ISO / IEC 23270: 2003, Information technology - C # Language Specification);
2. JIT-компіляція;
3. Можливість використання стандартних бібліотек програмування включених в .NET Framework, що включає широкий спектр API-інтерфейсів для роботи як з самим обладнанням так і з операційною системою;
4. Компіляція в CIL (common intermediate language) код, який може виконуватися на будь-якій машині, де підтримується CLR (common language runtime);
5. CodeLens доступний тільки для C#;
6. Garbage collector.

3.2. Проектування програмного продукту

3.2.1. Модель життєвого циклу

Модель життєвого циклу – це структура, яку можна розбити на задачі, процеси та роботи, які включають в себе всі етапи «життя» ПП, а саме: розробку, експлуатацію і супровід програмного продукту. Модель життєвого циклу починається від від визначення вимог і закінчується припиненням її використання.

Послідовна модель. Послідовна модель – це модель, яка є строго послідовною в часі і передбачає однократне виконання всіх стадій проекту з точним плануванням і конкретними вимогами.

Поміж існуючих моделей ЖЦ (каскадна, гнучка, спіральна та еволюційна) було обрано каскадну модель. Оскільки це поетапне виконання визначених дій. Також в цій моделі значна увага приділяється

інженерії вимог та власне проектуванню, що застраховує від вагомих помилок [53].

3.2.2. Побудова ієрархічної структури та розрахунок нев'язки

Побудовану ієрархічну структуру зображено на рис. 3.2. Розрахуємо нев'язку за допомогою MATLAB (рис. 3.1) [53]

```
>> n = 8; %количество вершин полученного графа
E = 8; %количество ребер полученного графа
Et = n-1; %количество ребер дерева
Ec = n*(n-1)/2; %количество ребер полного графа
Nev = (E-Et)/(Ec-Et); %расчет невязки
Nev %вывод значений невязки

Nev =

    0.0476
```

Рисунок 3.1. Розрахунок нев'язки.

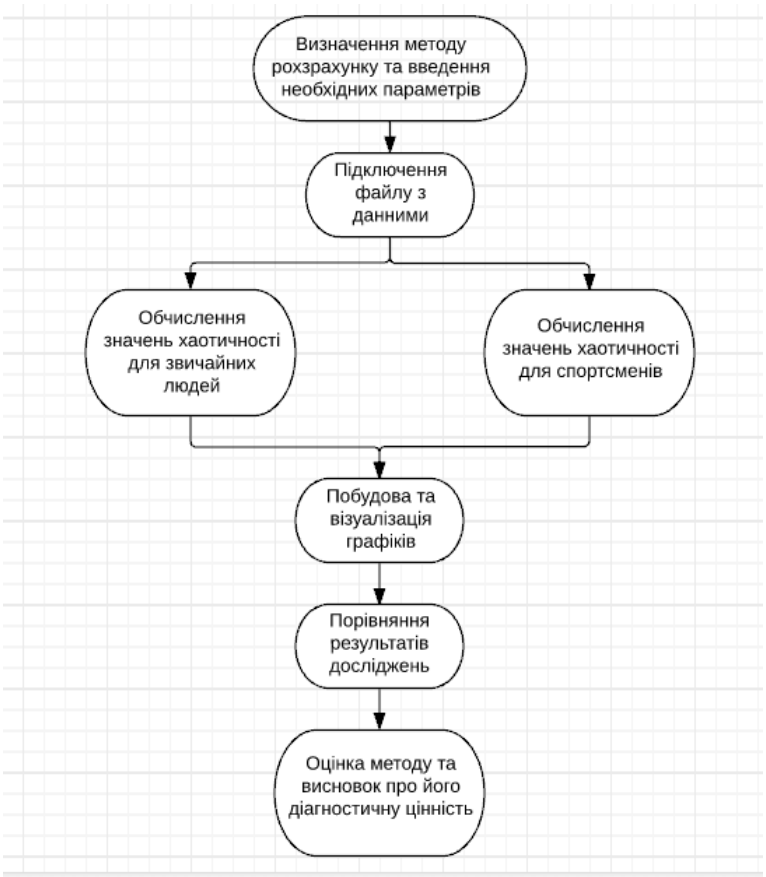


Рисунок 3.2. Ієрархічна структура ПП.

Як можна побачити, ми отримали дуже малу нев’язку, яка прагне о нуля, що прагне до нуля. Це свідчить про добре сплановану систему, яку доцільно розробляти [53].

3.2.3. Діаграма сутність-зв’язок (ERD)

ERD діаграма згідно до завдань, що були сформовані на етапі системного аналізу та аналізу вимог, показана на рис. 3.3.

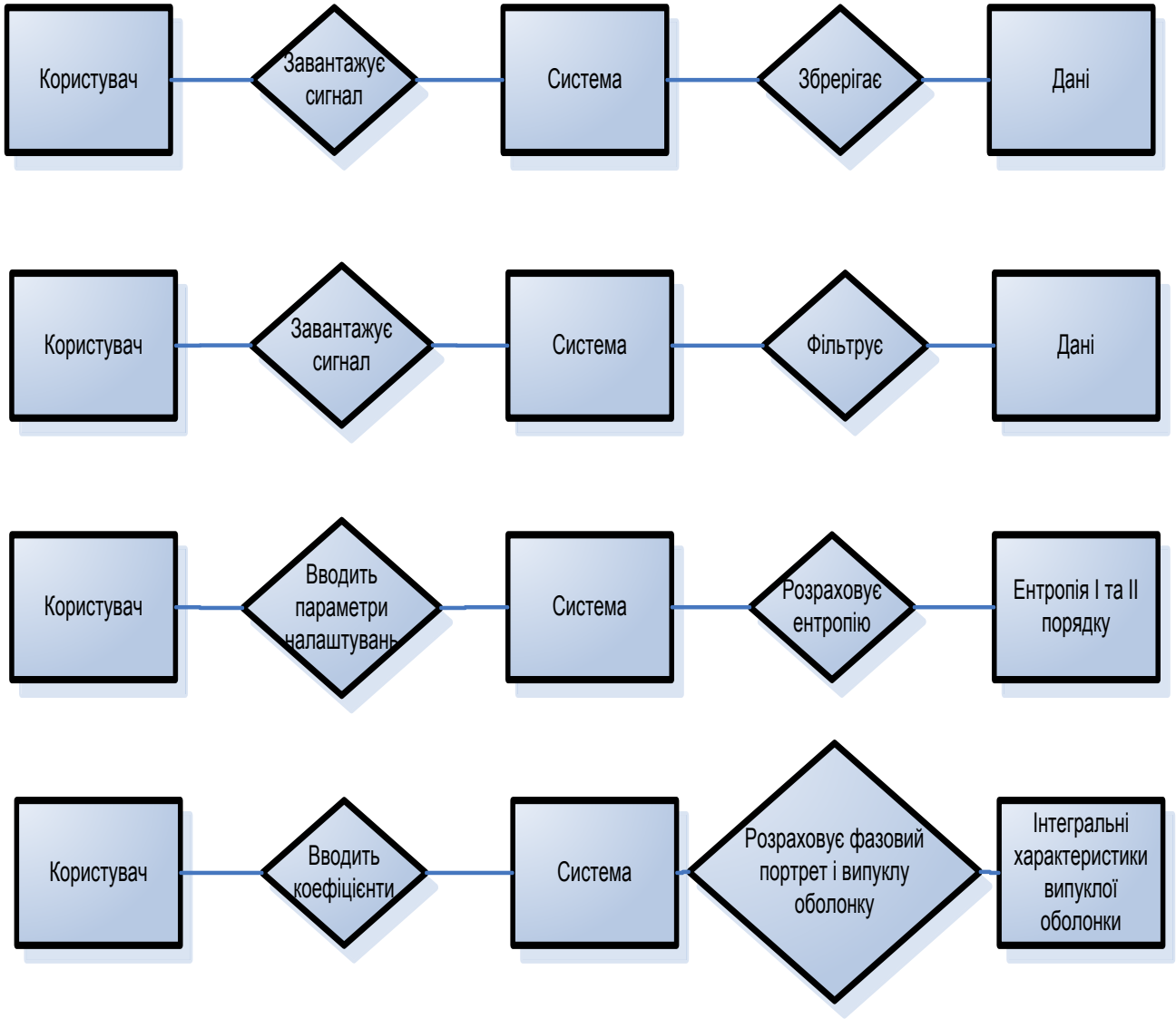


Рисунок 3.3. Діаграма сутність-зв’язок.

3.2.4. Контекстна діаграма IDEF0

Нотація IDEF0 створена для відображення ресурної інформації на кожному з етапів проектування систем шляхом опису документування процесів виробництва.

Таблиця 3.1

Використані ГОСТи

Позначення	Найменування
Стандарти ISO/IEC (ІСО/МЕК) в області розробки і документування програмних засобів	
ГОСТ Р ІСО/МЕК 12207-02	Інформаційна технологія. Процеси життєвого циклу програмних засобів
ГОСТ Р ІСО/МЕК 9126-93	Інформаційна технологія. Оцінка програмної продукції. Характеристики якості і керівництва щодо їх застосування
ГОСТ Р ІСО/МЕК 12119-94	Інформаційна технологія. Пакети програм. Вимоги до якості і тестування
Комплекс нормативних документів на автоматизовані системи	
ГОСТ 34.601-90	Автоматизовані системи. Стадії створення
ГОСТ 34.602-89	Технічне завдання на створення автоматизованої системи
Комплекс стандартів Єдиної системи програмної документації (ЄСПД)	
ГОСТ 19.201-78	Технічне завдання. Вимоги до змісту та оформлення
ГОСТ 19.701-90 (ІСО/МЕК 5807-85)	Схеми алгоритмів програм, даних і систем. Умовні позначення і правила виконання

На рис. 3.4 зображена контекстна діаграма IDEF0, що відображає використання всіх ресурсів в проекті. Діяльністю виступає власне

створення програмного продукту (ПП). Управлінням виступають перечислені вище ГОСТи [53].

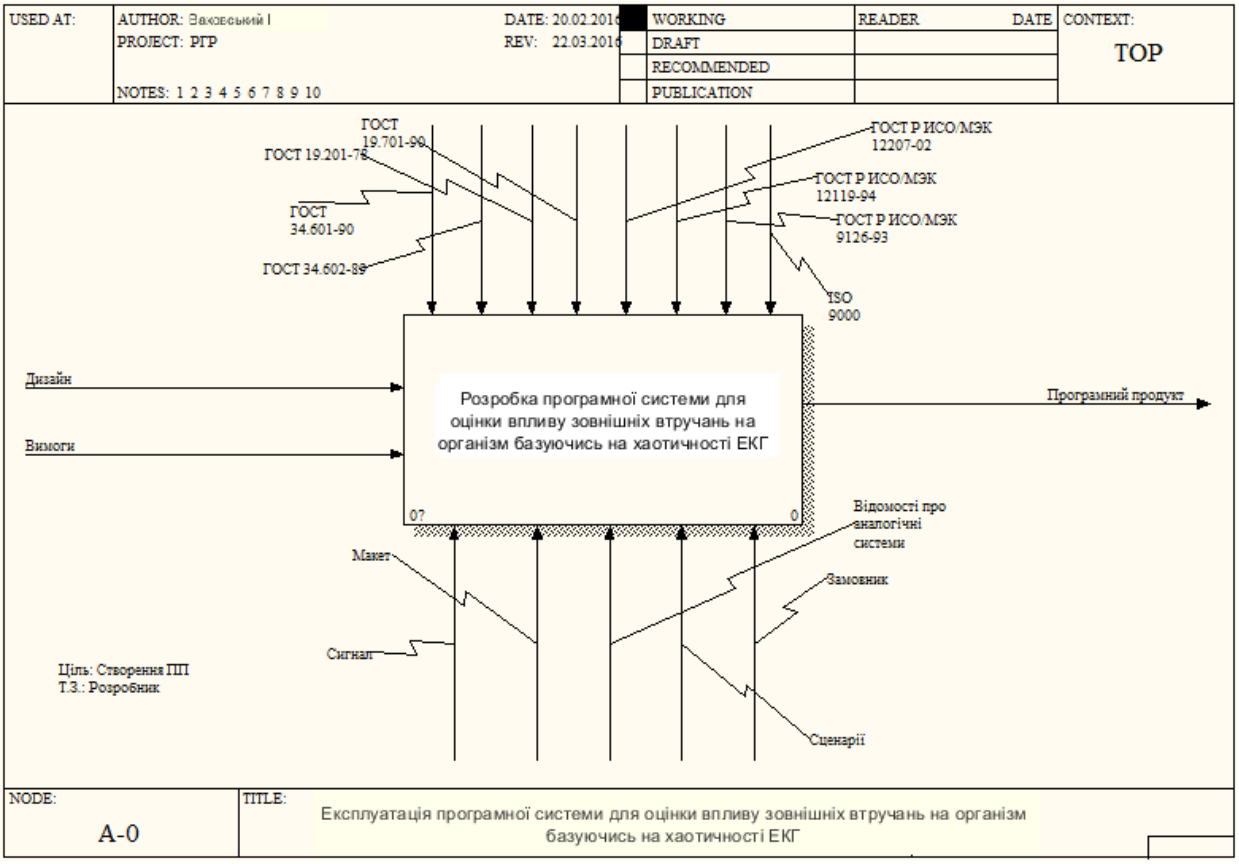


Рисунок 3.4. Контекстна діаграма.

На вхід подаються вимоги до ПП та його дизайн. Ресурсами виступають макет, сценарії, відомості про аналогічні системи, замовник. Виходом є власне ПП [53].

3.2.5. Діаграма декомпозиції IDEF0

Діаграма першого рівня декомпозиції A0, а також всі наступні діаграми декомпозиції, надають інтерфейсі обмеження (контекст) для дочірніх діаграм [53].

Було вирішено розбити створення програмного продукту на наступні складові: аналіз, проектування, реалізація та тестування, що зображено на рис.3.5 [53].

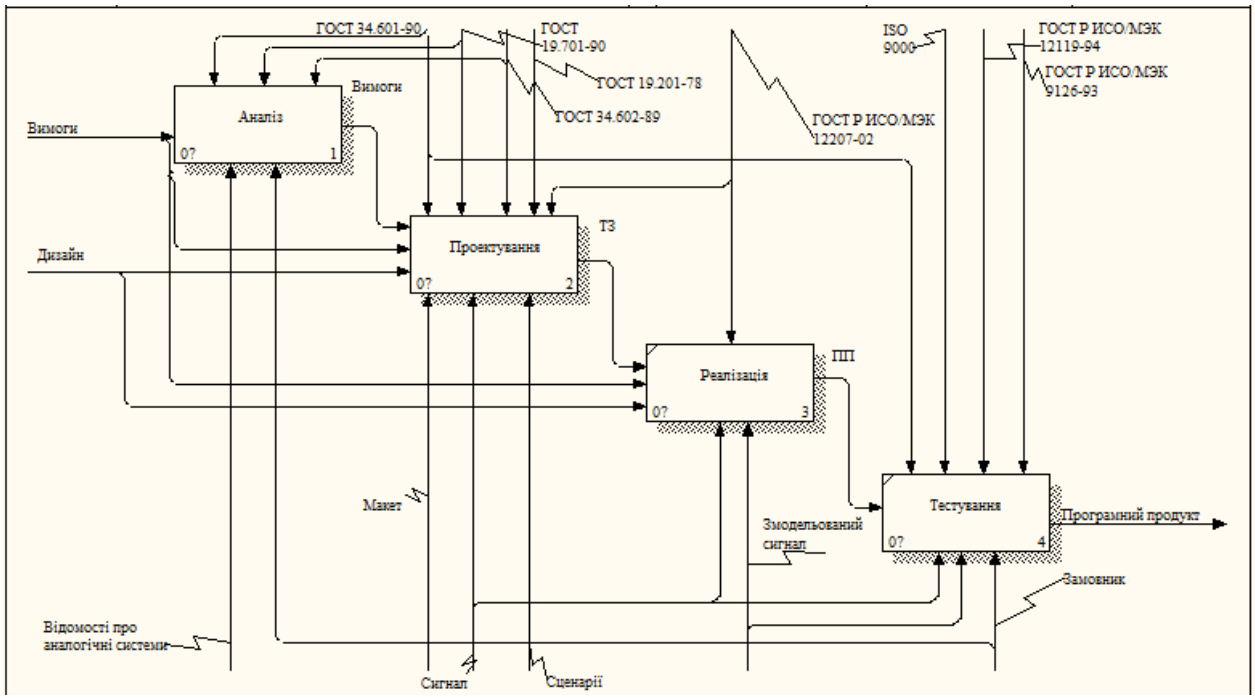


Рисунок 3.5. Діаграма декомпозиції IDEF0.

На рис. 3.6 зображена декомпозиція пункту Аналіз, що дозволяє більш детально розглянути його структуру [53].

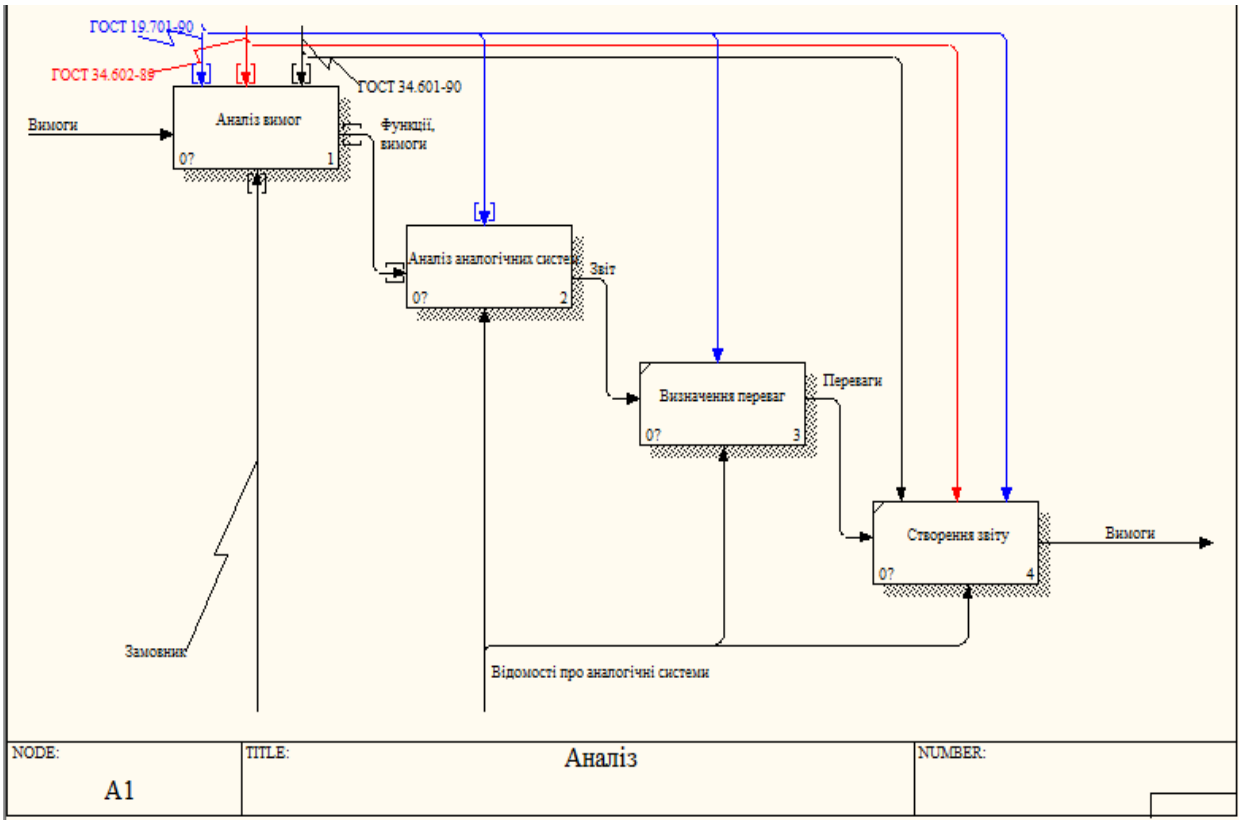


Рисунок 3.6. Діаграма декомпозиції IDEF0 (аналіз).

Не важко помітити, що аналізу вимог та аналізу використаних джерел відводиться важливе місце. Після них потрібно визначити переваги системи в порівнянні з вже існуючими [53].

Експлуатація програмної системи також може бути представлена у вигляді діаграм, подібно до розробки (рис. 3.7).

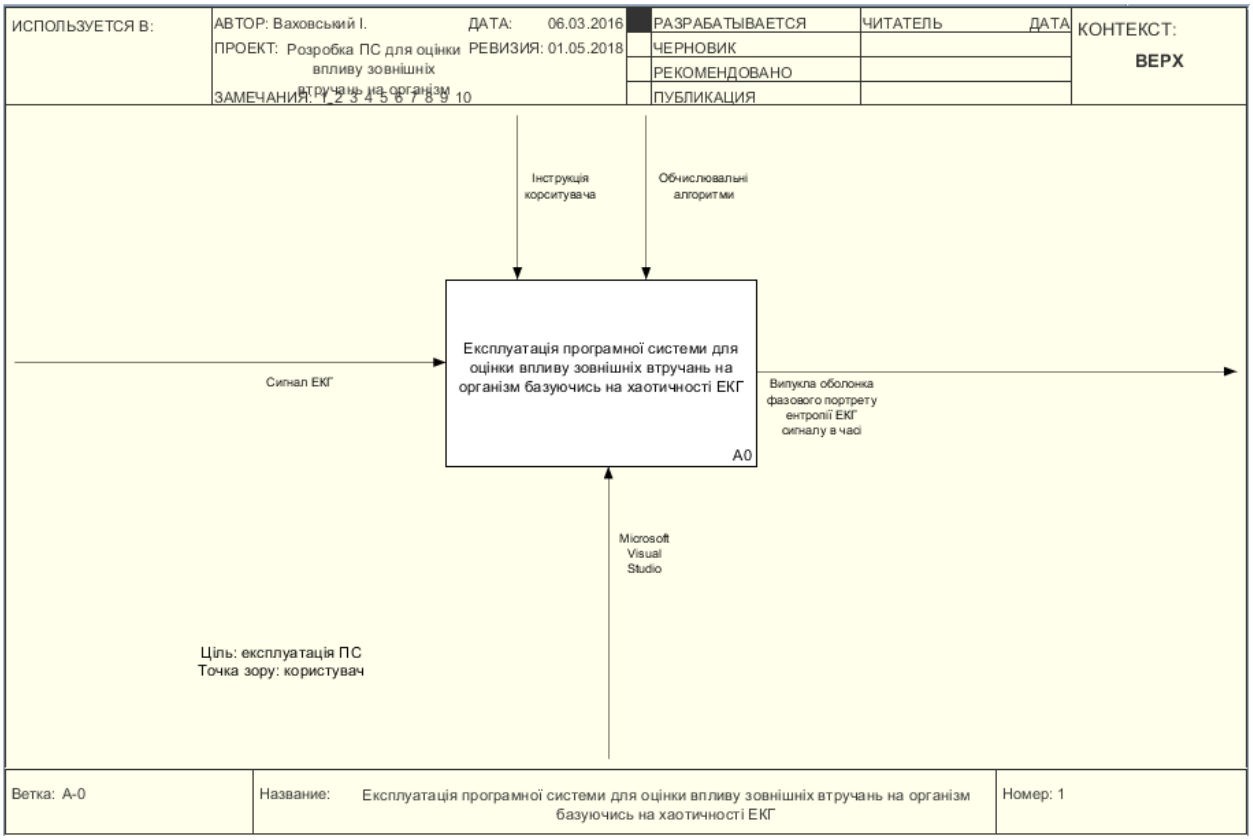


Рисунок 3.7. Контекстна діаграма.

На вхід подається сигнал ЕКГ. Ресурсами виступають інструкція користувача та обчислювальні алгоритми. Виходом є власне ПС.

Було вирішено розбити експлуатацію програмного продукту на наступні складові: зчитування вхідних даних з файлу, фільтрація даних, вибір алгоритму оцінки ентропії та його параметрів налаштування з подільшим її розрахунком, розрахунок фазового портрету та його випуклої оболонки, що зображено на рис. 3.8.

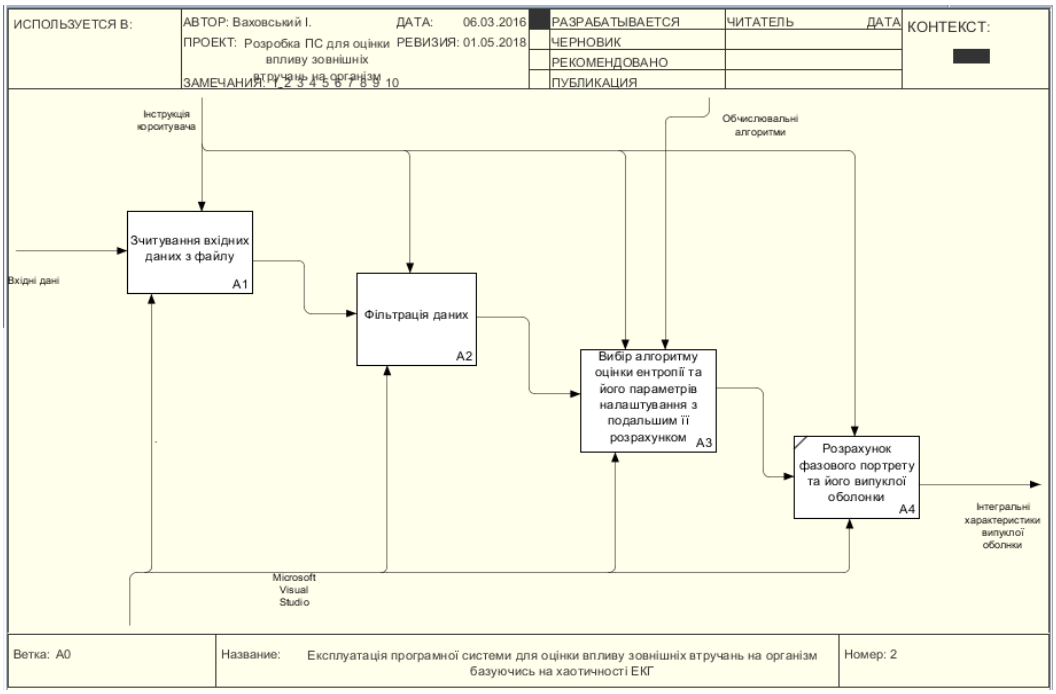


Рисунок 3.8. Діаграма декомпозиції IDEF0.

На рис. 3.9 зображена декомпозиція пункту «Вибір алгоритму оцінки ентропії та його параметрів налаштування з подальшим її розрахунком», що дозволяє більш детально розглянути його структуру.

На рис. 3.10 зображена декомпозиція пункту «Розрахунок фазового портрету та його випуклої оболонки», що дозволяє більш детально розглянути його структуру.

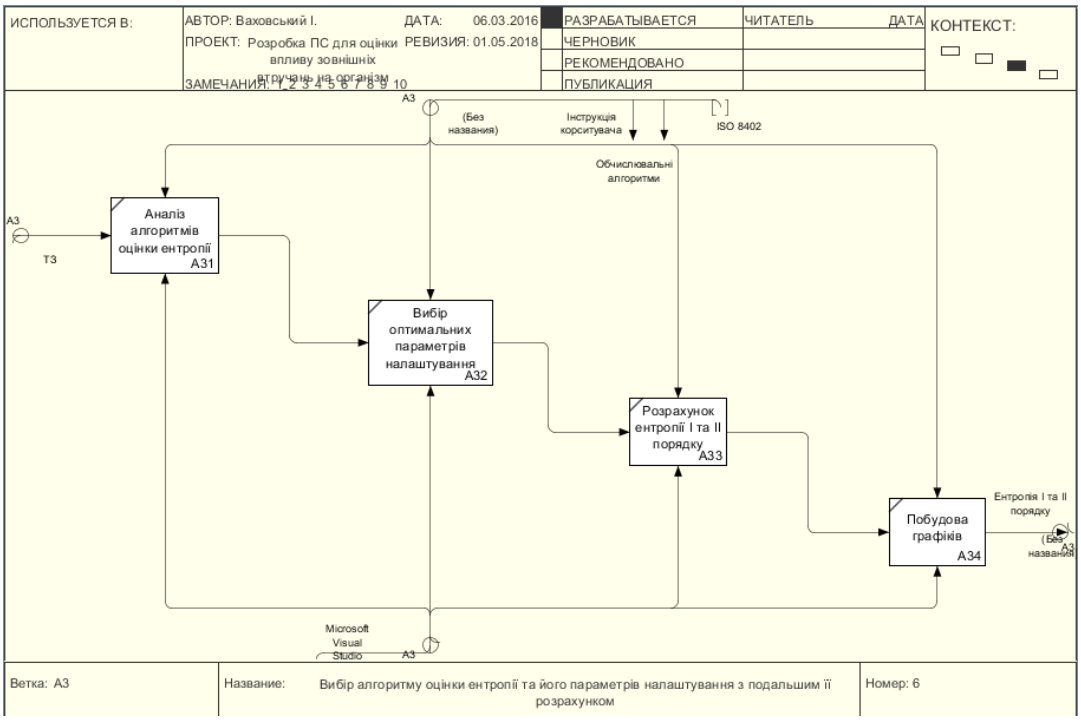


Рисунок 3.9. Діаграма декомпозиції IDEF0 (підпункт).

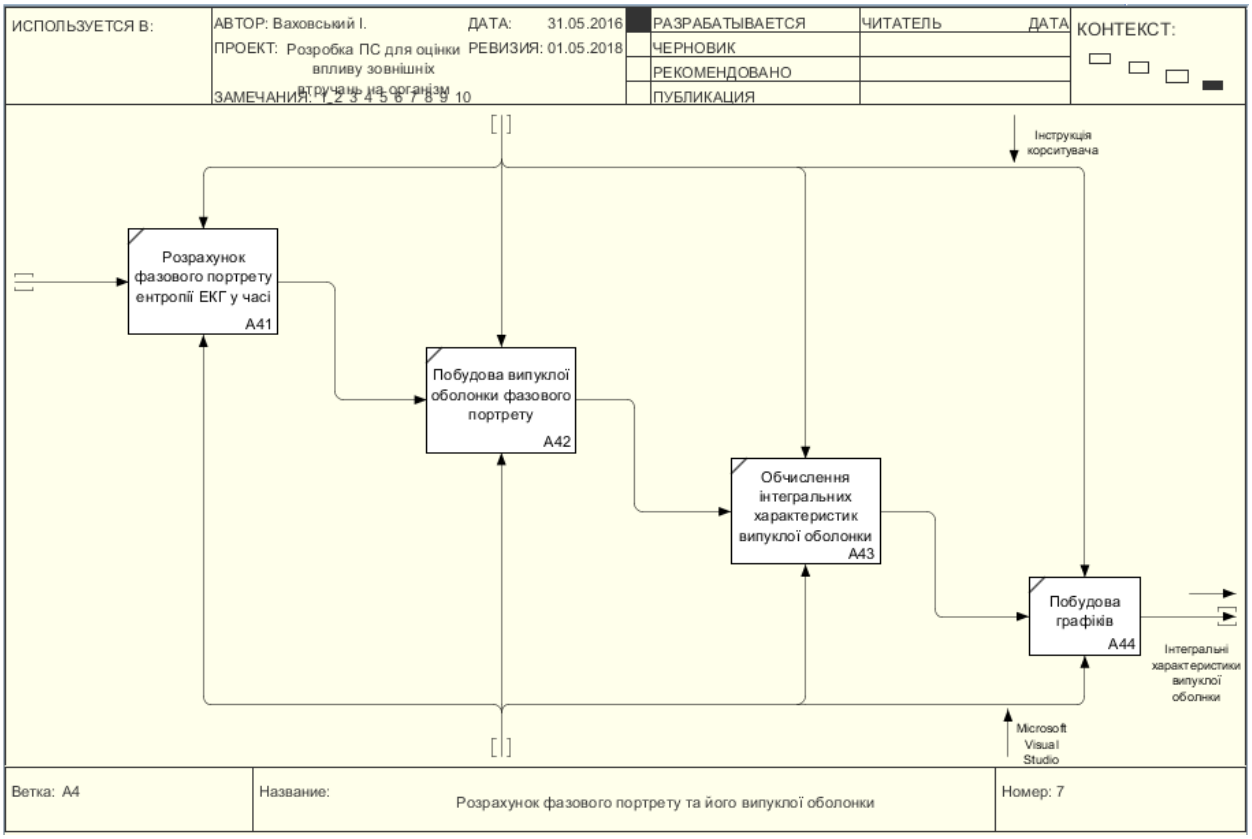


Рисунок 3.10. Діаграма декомпозиції IDEF0 (підпункт).

3.2.6. Методологія IDEF3

Для опису логіки взаємодії інформаційних потоків більше підходить IDEF3, звана також workflow diagramming, - методологія моделювання, яка в своїй основі має графічний опис взаємин між процесами обробки інформації, її потоків та об'єктів, які являються частиною цих процесів [53].

Доцільним є після створення технічного завдання одночасно виконувати проектування інтерфейсу адміністратора та користувача (рис 3.11).

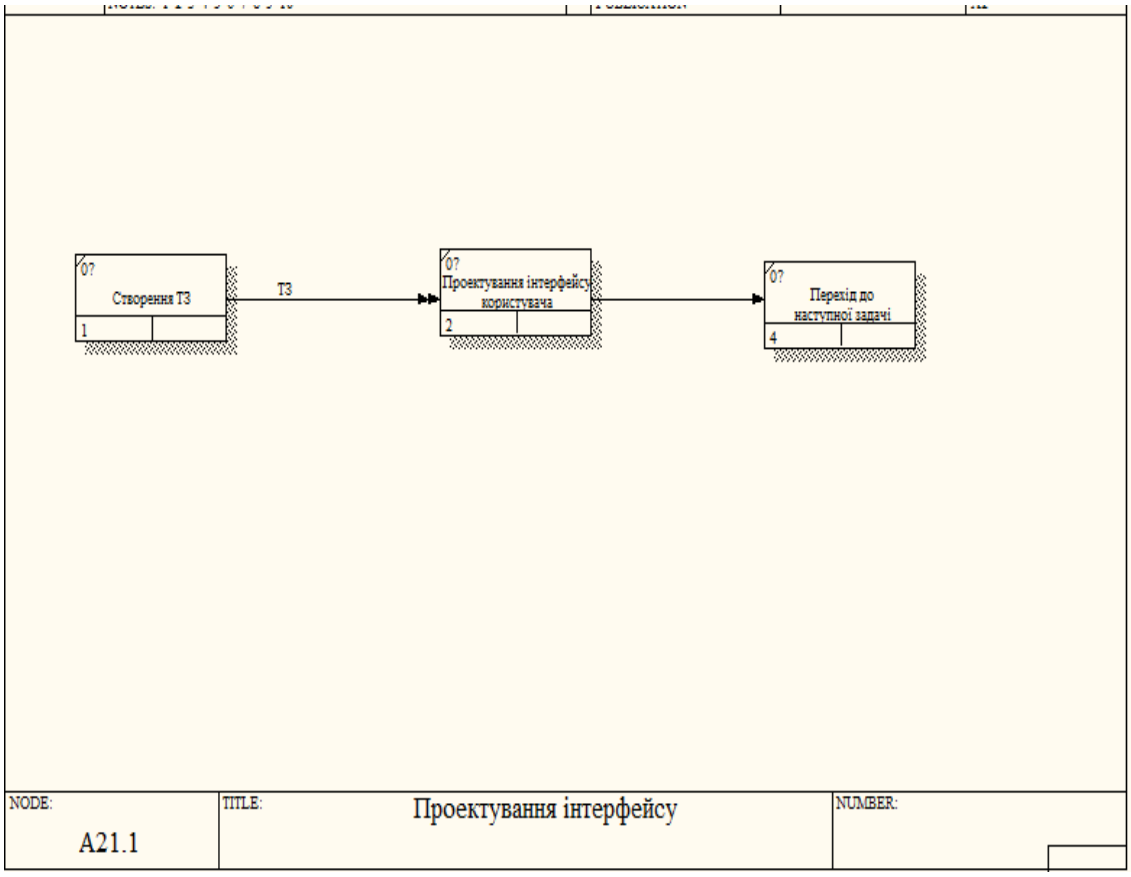


Рисунок 3.11. Діаграма IDEF3 (проектування інтерфейсу).

3.2.7. Методологія DFD

Найважливішим способом опису процесу є діаграми потоків даних (інформації) DFD (Data Flow Diagram). На рис. 3.12. зображено декомпозицію пункту «Аналіз вимог». Зовнішнім посиланням тут виступає замовник, котрий надає свої потреби. Сховищем даних є репозиторій ГОСТів [53].

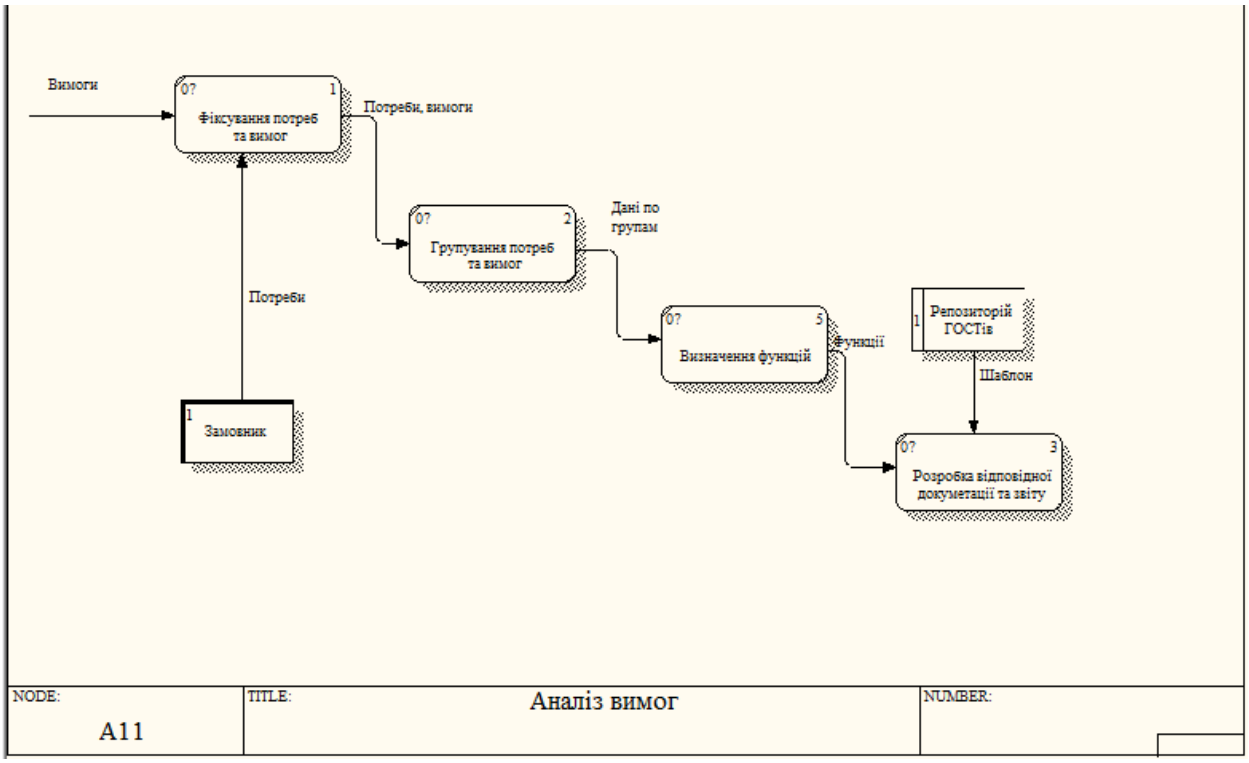


Рисунок 3.12. Діаграма DFD (аналіз вимог).

На рис. 3.13 зображено декомпозицію пункту «Аналіз аналогічних систем».

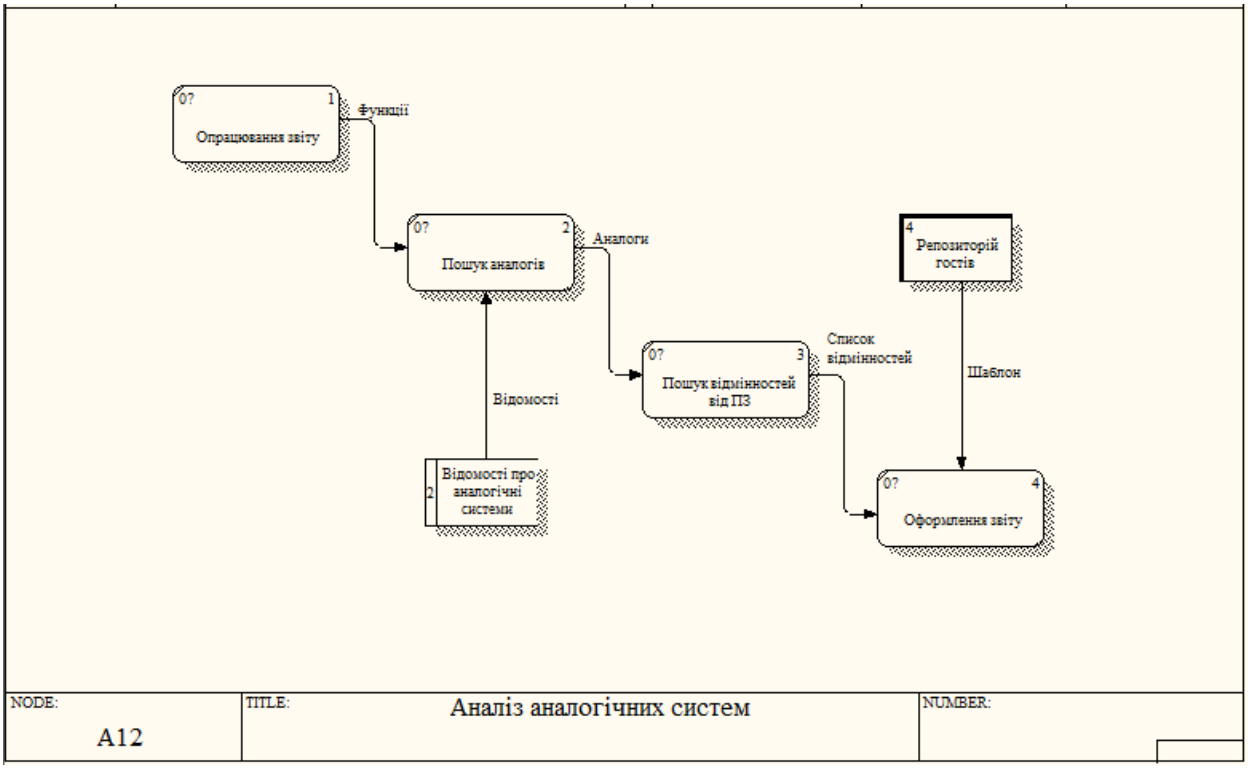


Рисунок 3.13. Діаграма DFD (аналіз аналогічних систем).

3.2.8. Діаграма дерева вузлів

Діаграма дерева вузлів показує ієрархічну залежність робіт, але не взаємозв'язки між роботами [53].

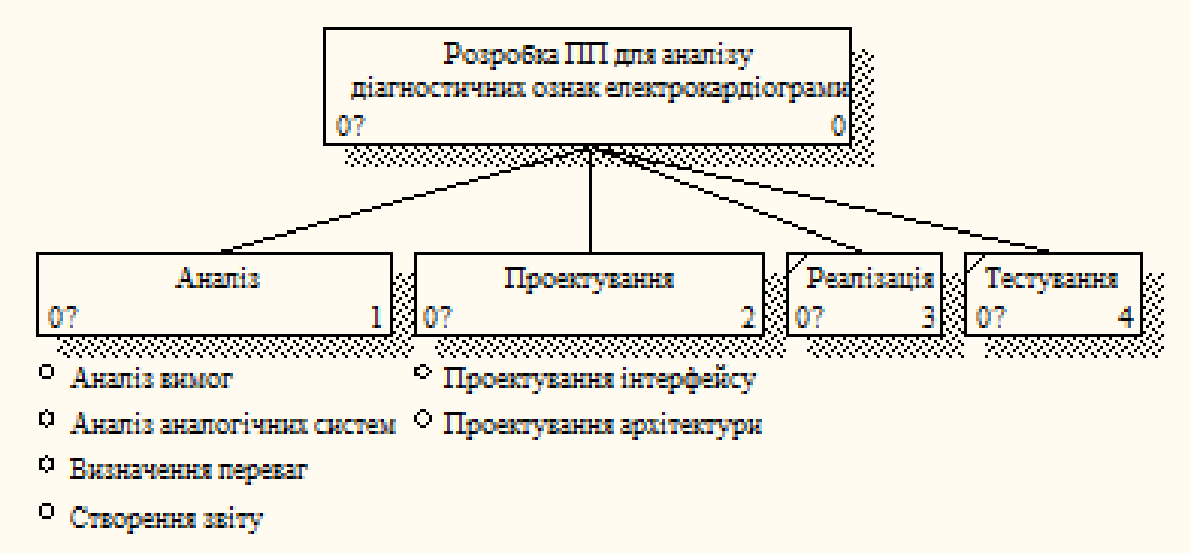


Рисунок 3.14. Діаграма дерева вузлів.

3.2.9. Діаграма варіантів

Сценарії використання системи, а також користувачів, які її використовують можуть описати «usecase» діаграми (діаграми варіантів використання).

Актором виступає користувач. Сценарії наступні: вибір файлу для дослідження; налаштування програмного середовища, що включає налаштування відображення графіків та налаштування парметрів алгоритмів оцінки ентропії; фільтрація сигналу, що розширюється можливістю скачати відфільтрований сигнал; розрахунок ентропії, що включає розрахунок ентропії I та II порядку, а також розширюється можливістю скачати дані числові ряди; розрахунок фазового портрету та випуклої оболонки, що включає побудову відповідних графіків.

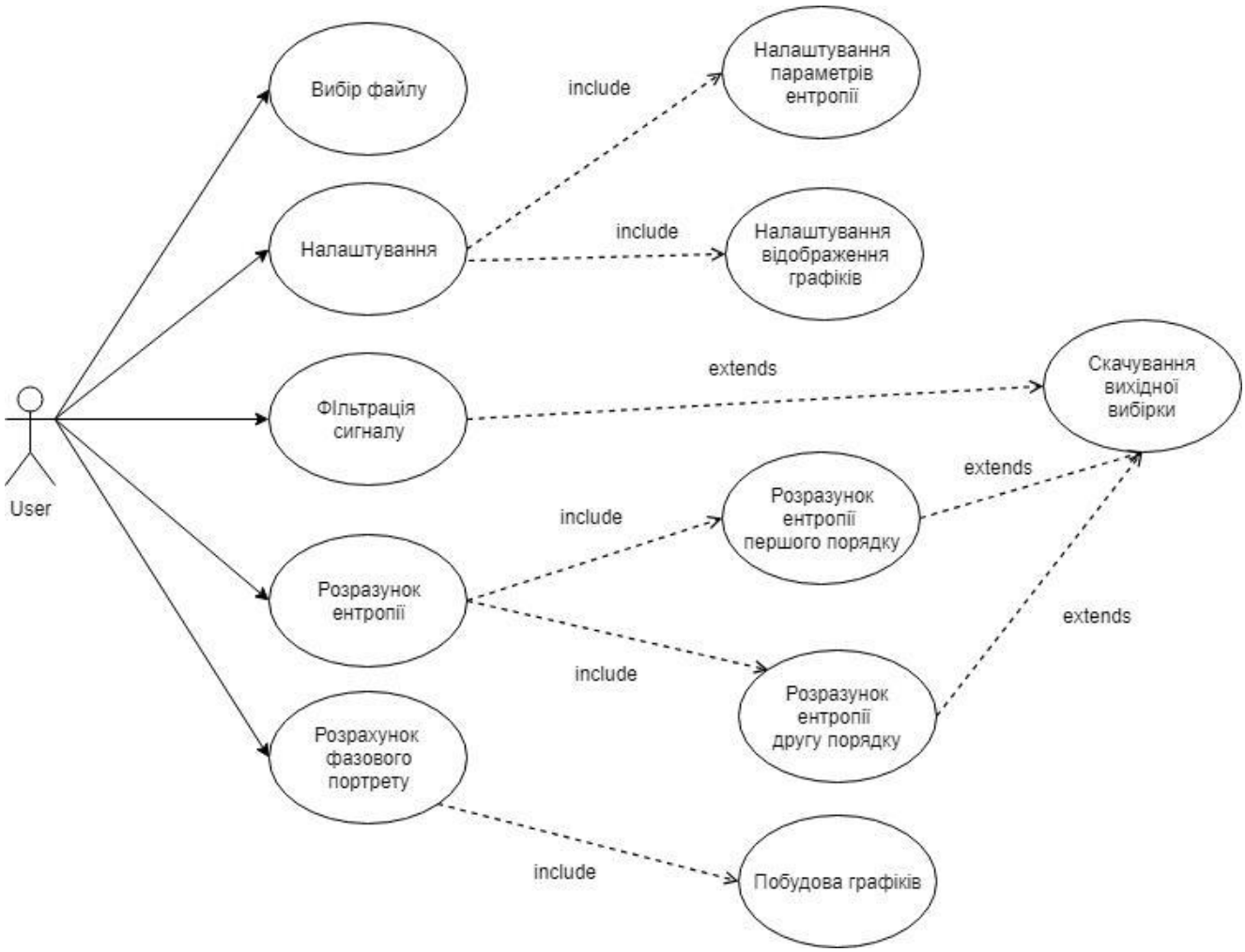


Рис.3.15. Use-case діаграма.

3.2.10. Діаграми послідовності

Опис головного успішного сценарію даного варіанта використання у короткій формі: Користувач має на екрані форму введення коефіцієнтів чи завантаження сигналу, після його дій алгоритм опрацьовує вхідний сигнал, розраховує коефіцієнти, виводить їх на екран та будує зображення графіків [53].

На рис. 3.16 наведено діаграму послідовності, створену в середовищі «draw.io» для даного прикладу. У верхній частині діаграми наведено перелік об’єктів (логічні сутності), які взаємодіють між собою у процесі виконання сценарію. Часова шкала на даній діаграмі направлена згори донизу. Нижче наведено короткий опис подій, що відбуваються при виконанні даного варіанта використання [53].

При запуску програми користувач вибирає файл для дослідження, програма взаємодіє з файловою системою і отримує вибраний файл, про що інформує користувача. Користувач вибирає параметри фільтрації сигналу, після чого відфільтровані дані використовуються для обчислення зміни ентропії та похідної від отриманого ряду. Ці дані далі передаються для обчислення фазового портрету та випуклої оболонки, розраховуються інтегральні характеристики випуклої оболонки фазового портрету і будуються відповідні графіки, після чого керування повертається до користувача.

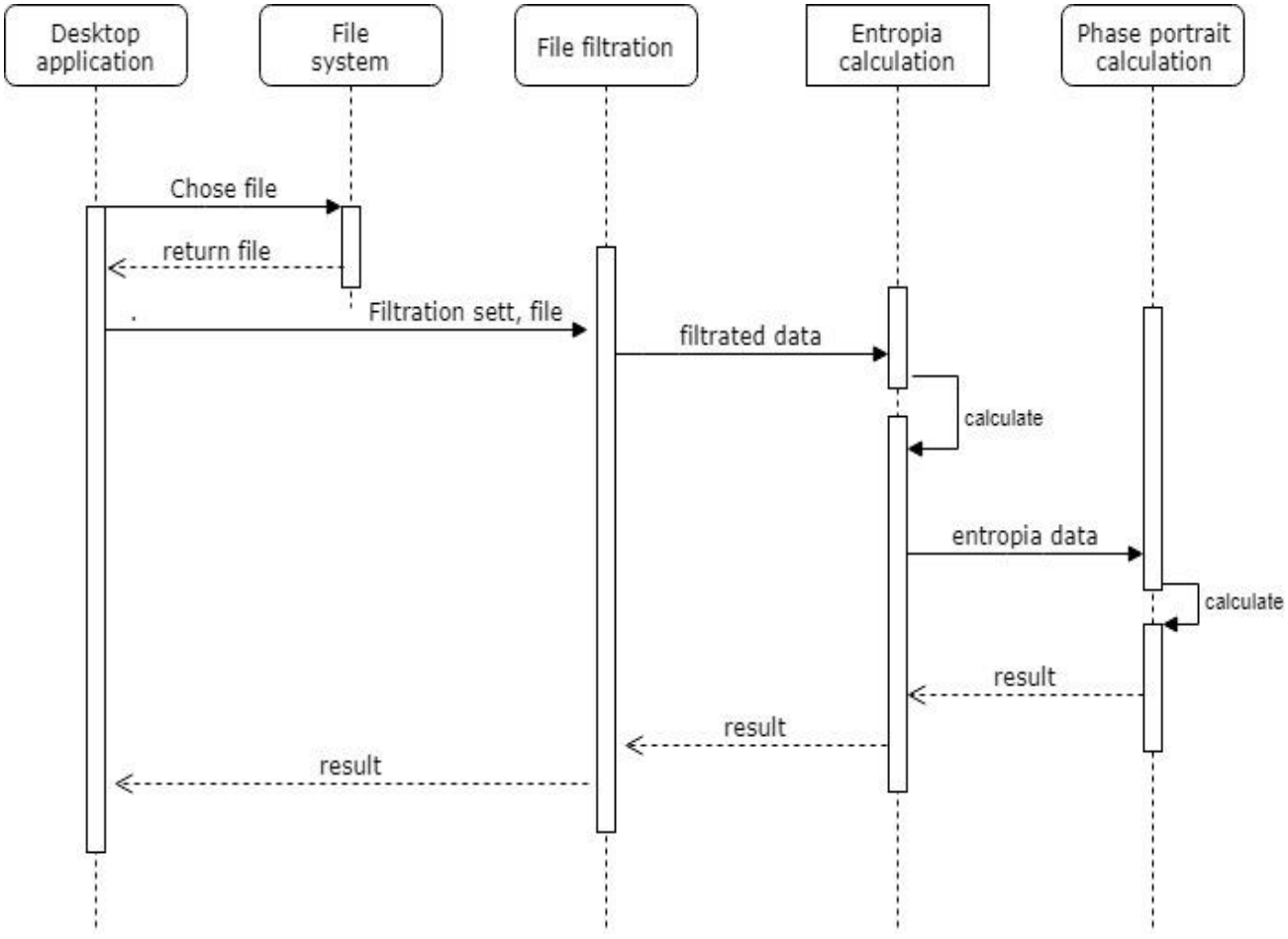


Рисунок 3.16. Діаграма послідовності ПП.

3.2.11. Діаграма класів

Діаграми класів (Class diagrams) – один з типів UML діаграм, які показують логічні складові програмної системи та певним чином впливають на розробку програмного продукту. Діаграма класів

складається з безпосередньо класів (classes) та відношень між цими класами (relations).

Було виділено наступні класи:

- інтерфейс «Calculation»
- класс Entropy
- класс PhasePortrait
- класс ConvexHull
- класс Graphics
- класс GUI

Їхні методи, атрибути та відношення між ними наведені на Рис. 3.17.

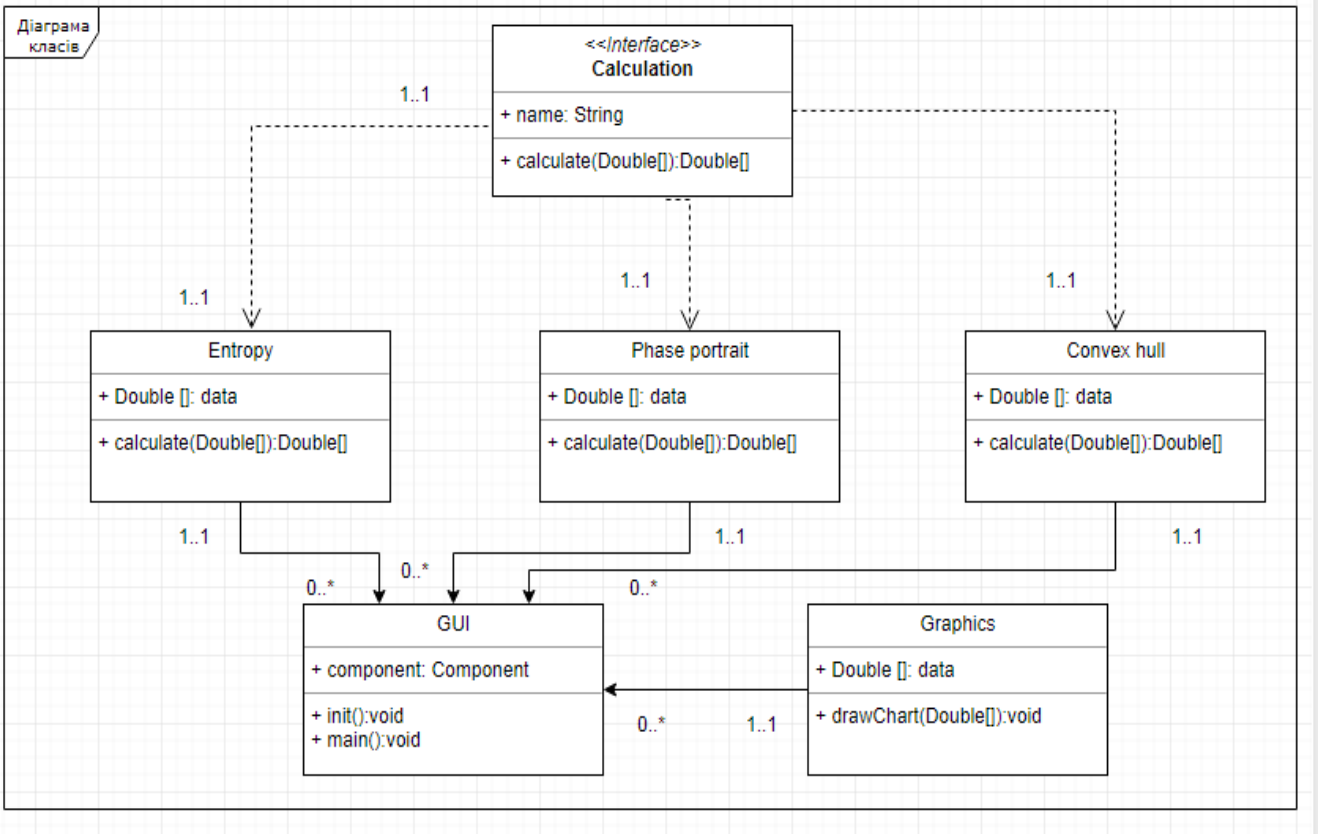


Рис. 3.17. Діаграма класів.

3.3. Реалізація програмного продукту

Розроблений програмний продукт написано мовою C# з використанням MS Visual Studio 2017 [39].

Метою розробки є виконання програмної реалізації оцінки впливу зовнішніх ознак на організм людини базуючись на аналізі зміні хаотичності параметрів ЕКГ.

Розроблений інтерфейс користувача є досить простим і інтуїтивно зрозумілим.

Вхідні дані при роботі з розробленою програмою:

- файл з даними;

Основні можливості системи, що повинні бути розроблені:

- можливість завантаження файлу;
- можливість фільтрації сигналу;
- можливість вибору методу дослідження;
- можливість обчислення зміни ентропії у часі;
- можливість побудови фазового портрету;
- можливість побудови опуклої оболонки фазового портрету;
- можливість зміни кольору графіків;
- можливість збереження проміжних результатів у файл;
- можливість налаштування відображення графіків.

Для реалізації програмного забезпечення було обрано MS Visual Studio 2017, тому що дане програмне забезпечення має необхідний інструментарій для виконання поставлених задач замовником.

Висковки до розділу 3

В ході проектування ПП була створена ієрархічна структура та розрахована нев’язка, яка свідчить про добре сплановану систему ПП, яку

буде доцільно розробляти. Здійснено проектування програмного продукту у вигляді діаграм, які описують розробку та функціонування програми.

РОЗДІЛ 4
РОБОТА З РОЗРОБЛЕНИМ ПРОГРАМНИМ ПРОДУКТОМ

Програма являє собою додаток для робочого стола, реалізований з використанням мови програмування C#. Головне вікно програми виглядає наступним чином:

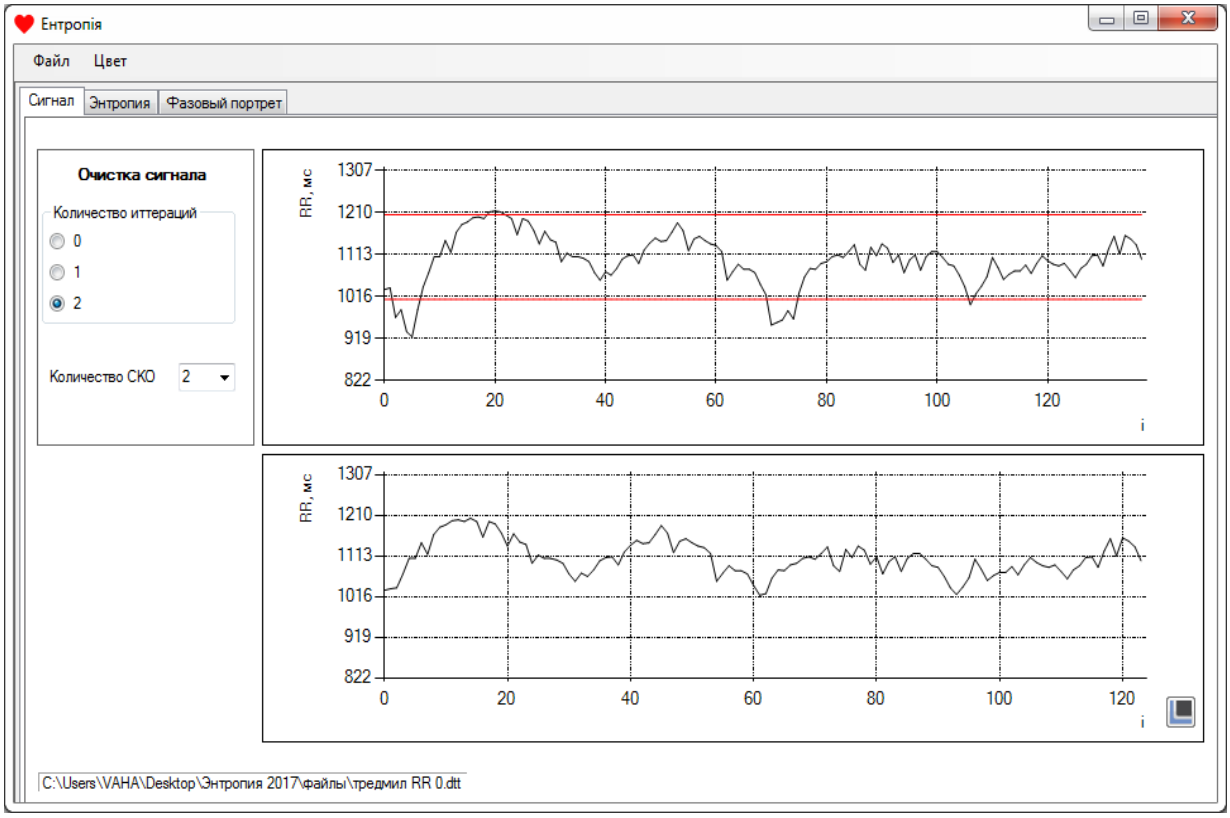


Рисунок 4.1. Стартове вікно програми.

Структурно, функціональну складову програми можна розділити на 3 вкладки:

1. «Сигнал» – вкладка, де проходить фільтрація сигналу. Ця вкладка являється також стартовою сторінкою програми (рис. 4.1). На даній вкладці користувач має можливість відфільтрувати вхідний сигнал. Алгоритм фільтрації обмежує вхідний сигнал в заданих рамках. Значення рамок визначається як певна кількість СКО вхідного сигналу. Для того, щоб встановити потрібний діапазон корисного сигналу, потрібно вибрати один із варіантів у випадаючому списку (від 0.25 до 3 з кроком в 0.25).

Також цю операцію можна зробити повторно, якщо вибрати відповідну кількість ітерацій.

2. «Ентропія» - вкладка, де відбувається розрахунок ентропії і похідної ентропії. На цій вкладці можна вибрати метод дослідження, встановити параметри для кожного методу, або вибрати оптимальні параметри (рис. 4.2).

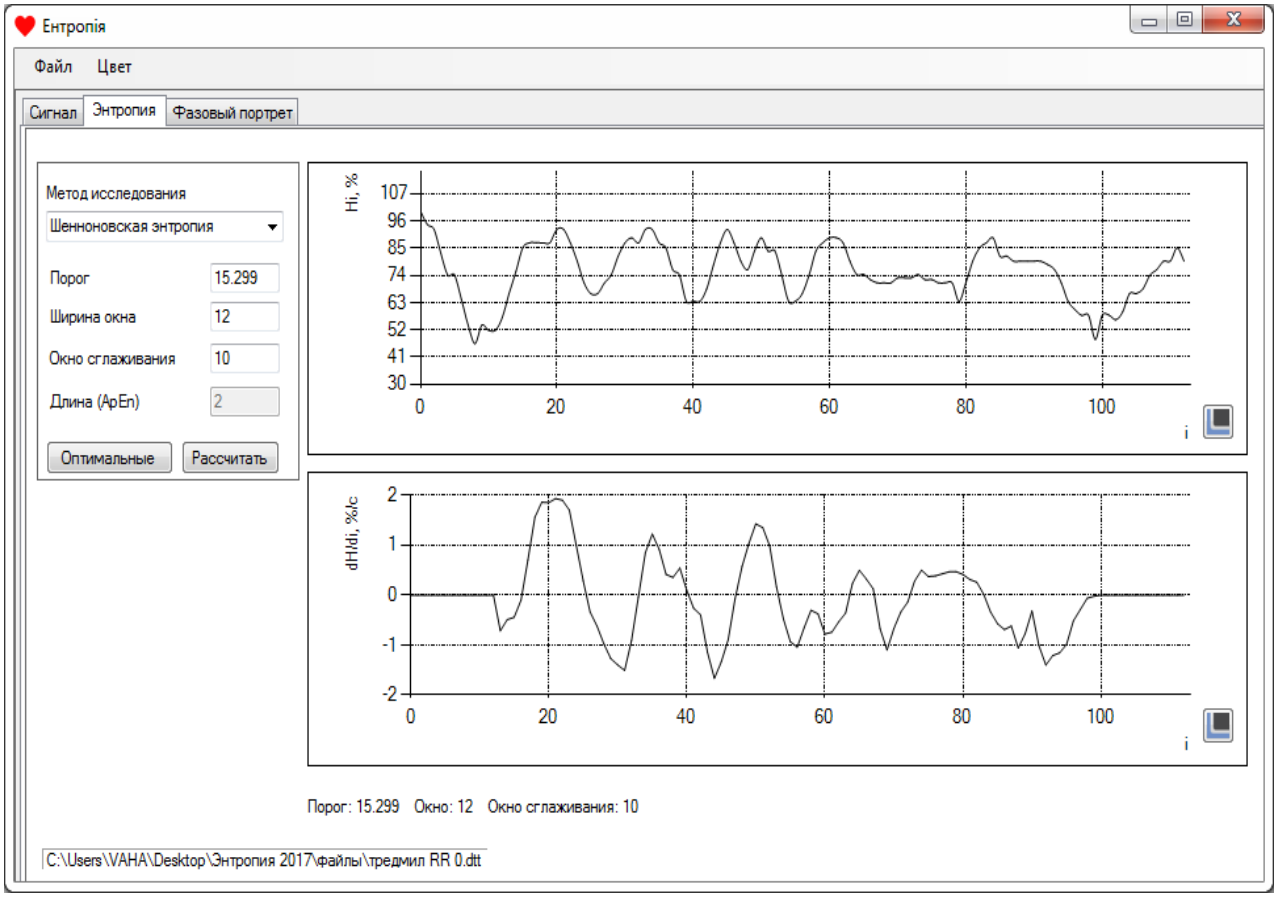


Рисунок 4.2. Вкладка «Ентропія».

На вкладці розташовуються два графіки. Перший графік відповідає за зміну ентропії у часі, а другий – за похідну отриманого вище ряду.

На графіках присутня кнопка «Зберегти», яка виконує збереження отриманого часового ряду в текстовий файл для подальших досліджень в інших програмних продуктах.

При виборі даної опції, програма запропонує вибрати розташування на файловій системі для збереження файлу (рис. 4.3). При успішному

збереженні файлу ми отримаємо повідомлення про успішно проведену операцію (рис. 4.4).

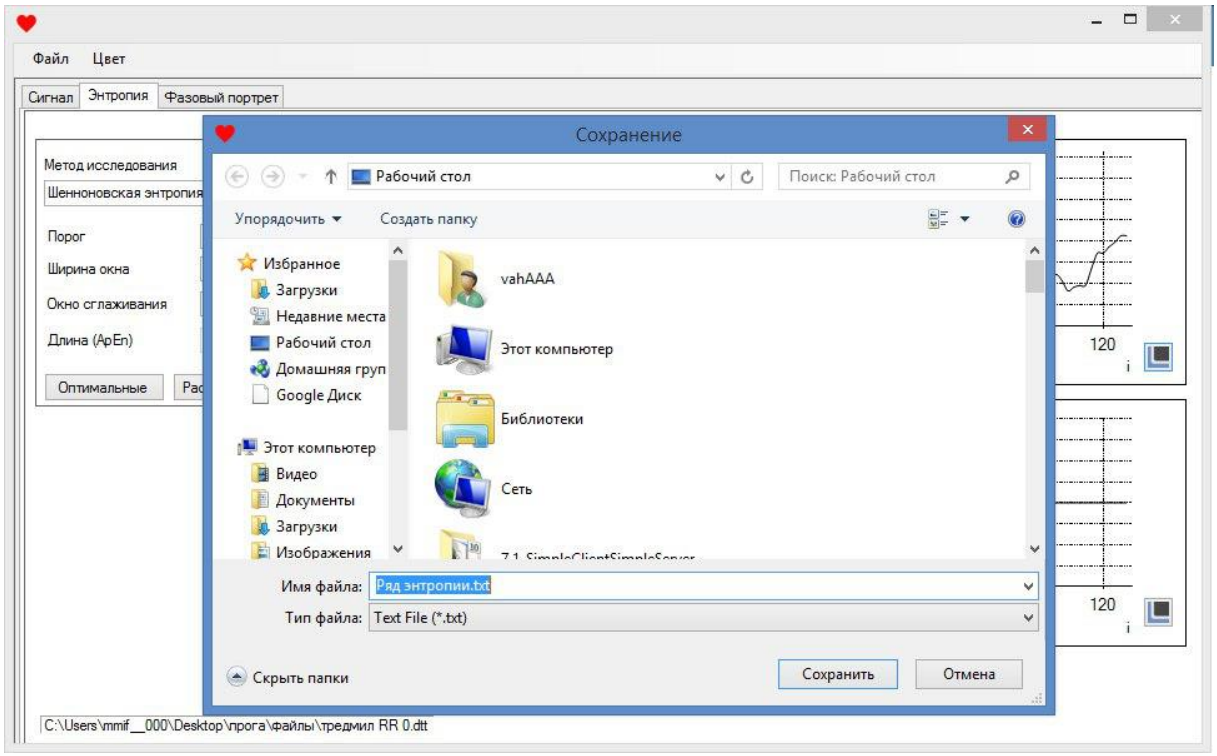


Рисунок 4.3. Модальне вікно «Збереження».

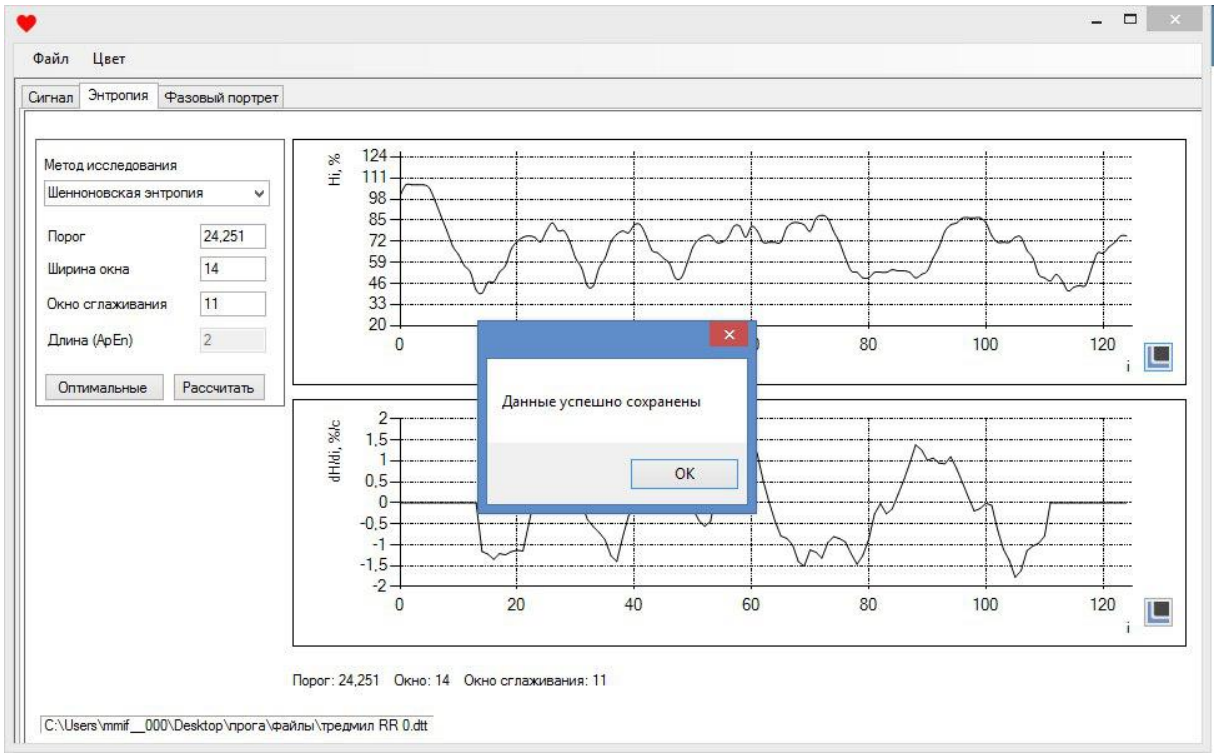


Рисунок 4.4. Повідомлення про успішне збереження файлу на файловій системі.

3. Фазовий портрет – вкладка, на якій показаний розрахований фазовий портрет (рис. 4.5).

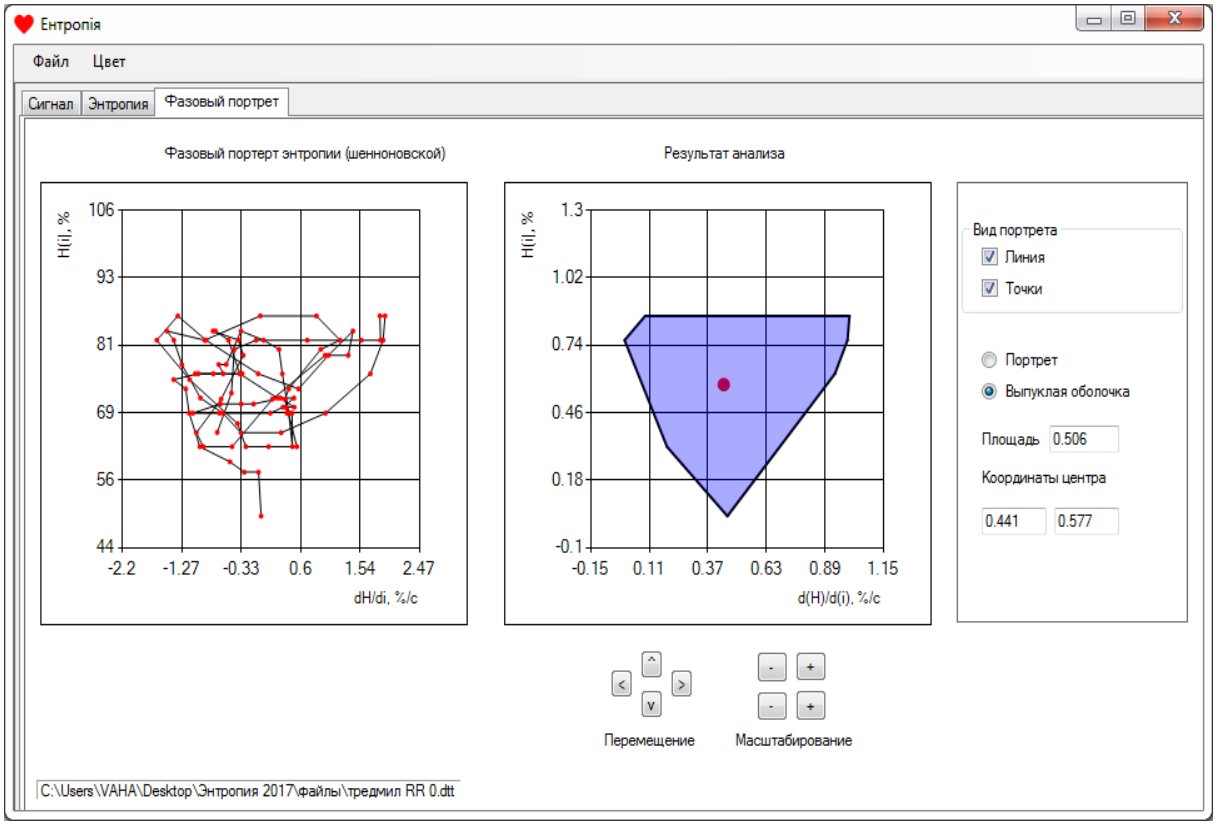


Рисунок 4.5. Вкладка «Фазовий портрет».

На цій вкладці показано два графіка:

- Фазовий портрет.
- Випукла оболонка фазового портрету.

Реалізована можливість налаштування відображення обох графіків. Для фазового портрету реалізована можливість відображення точками, лініями або і точками і лініями одночасно. Для випуклої оболонки реалізована можливість масштабування та переміщення по координатній площині.

Також на цій вкладці присутній блок з результатами досліджень, де відображаються інтегральні характеристики випуклої оболонки, за якими в

подальшому і проводяться дослідження. Цими характеристиками являються:

- Площа випуклої оболонки
- Координати центра мас випуклої оболонки.

Обрати файл для дослідження можна в меню «Файл - Обрати файл» (рис. 4.6, 4.7). В програмі використовуються файли з розширенням “.dtt”. Цей файл являє собою звичайний текстовий файл, але дане розширення дозволяє працювати з таким файлом і в уже існуючих програмних продуктах для дослідження ЕКГ.

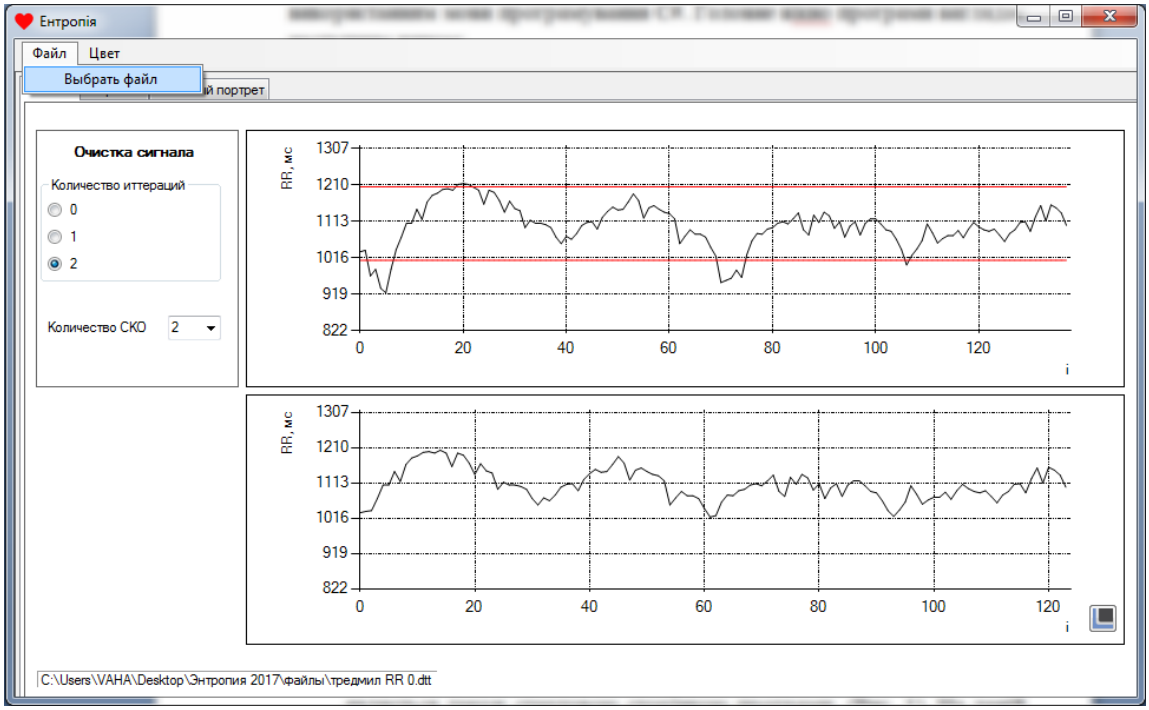


Рисунок 4.6. Вкладка меню «Обрати файл».

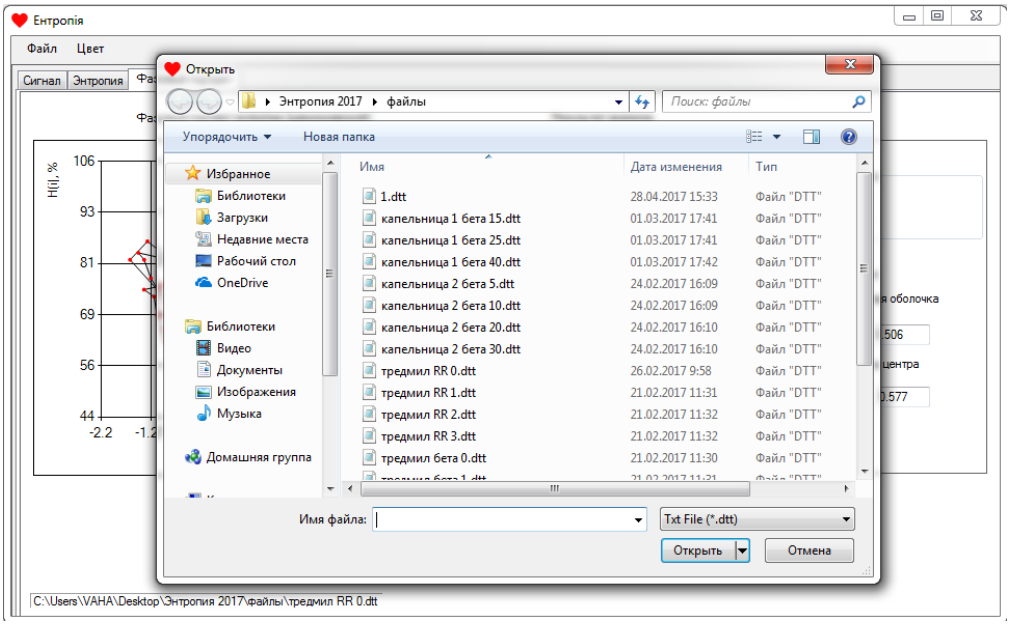


Рисунок 4.7. Вікно «Обрати файл».

Для покращення роботи користувача з програмою було реалізовано налаштування кольорів графіків (рис. 4.8, 4.9). Функція знаходиться на вкладці меню «Колір – Вибрати колір графіків».

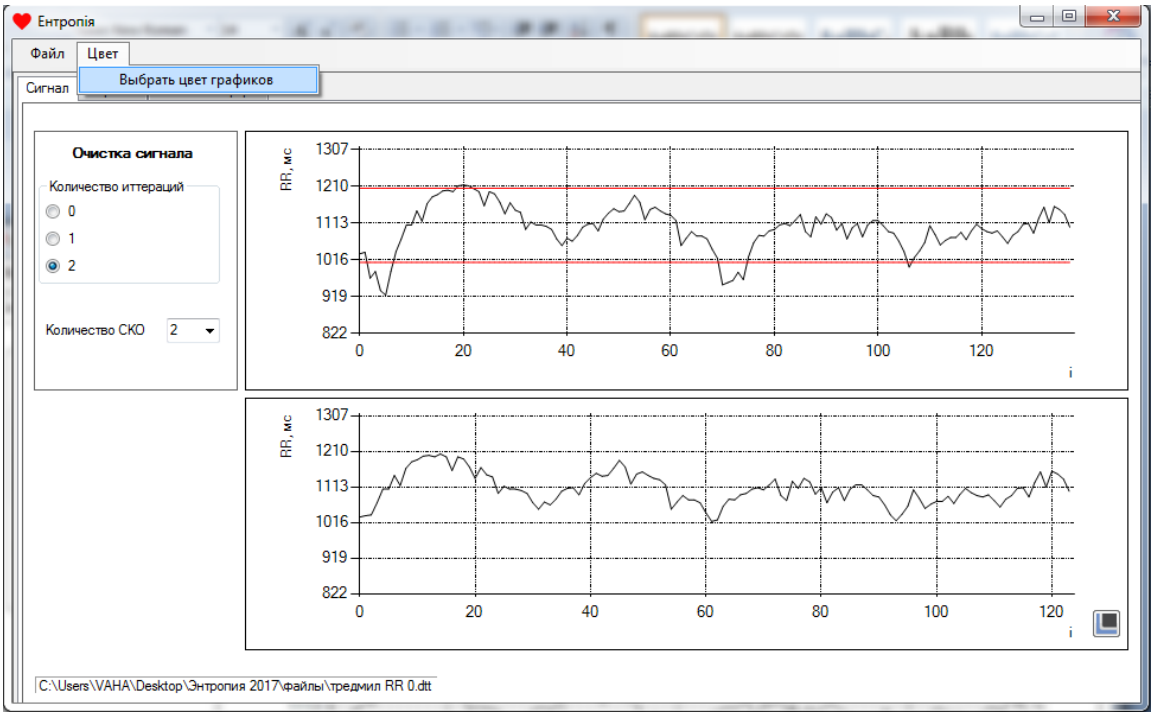


Рисунок 4.8. Вкладка меню «Колір».

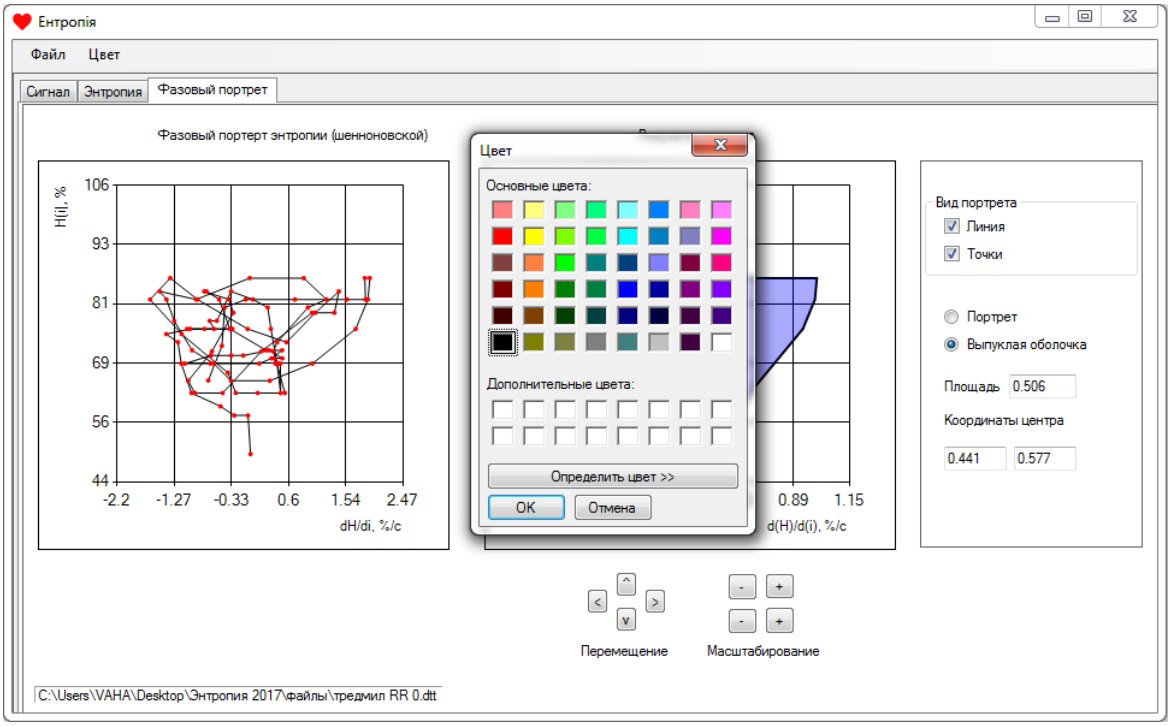


Рисунок 4.9. Вікно «Колір».

В програмі реалізована обробка непередбачуваних ситуацій та помилок. При неправильно вибраних даних чи неправильно підібраних параметрах для того чи іншого методу, програма видасть повідомлення про помилку (рис. 4.10).

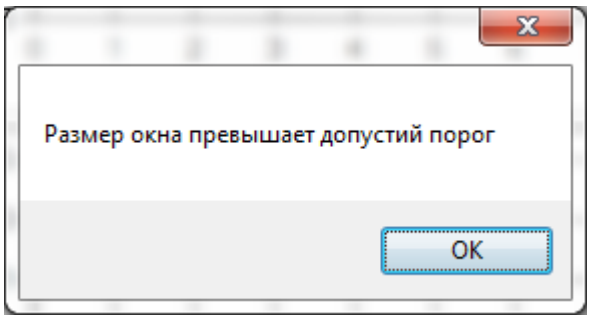


Рисунок 4.10. Обробка помилок.

Всі налаштування в програмі зберігаються в файлі «config.ini». При першому запуску програми цього файлу може не бути. Якщо даного файлу немає, програма сама його створить в тій же директорії, де знаходиться виконуваний файл. В створеному файлі будуть знаходитися налаштування за змовчуванням (рис. 4.11). Після того, як користувач проведе сесію

роботи з програмою і змінить будь-які налаштування: параметри фільтрації, обчислювальних алгоритмів, відображення графіків – програма зробить запит на збереження налаштувань (рис. 4.12). При виборі опції «Так», налаштування будуть збережені у конфігураційному файлі.

При переносі даного програмного продукту з однієї електронно-обчислювальної машини на іншу, можна скопіювати конфігураційний файл, щоб зберегти потрібні налаштування.

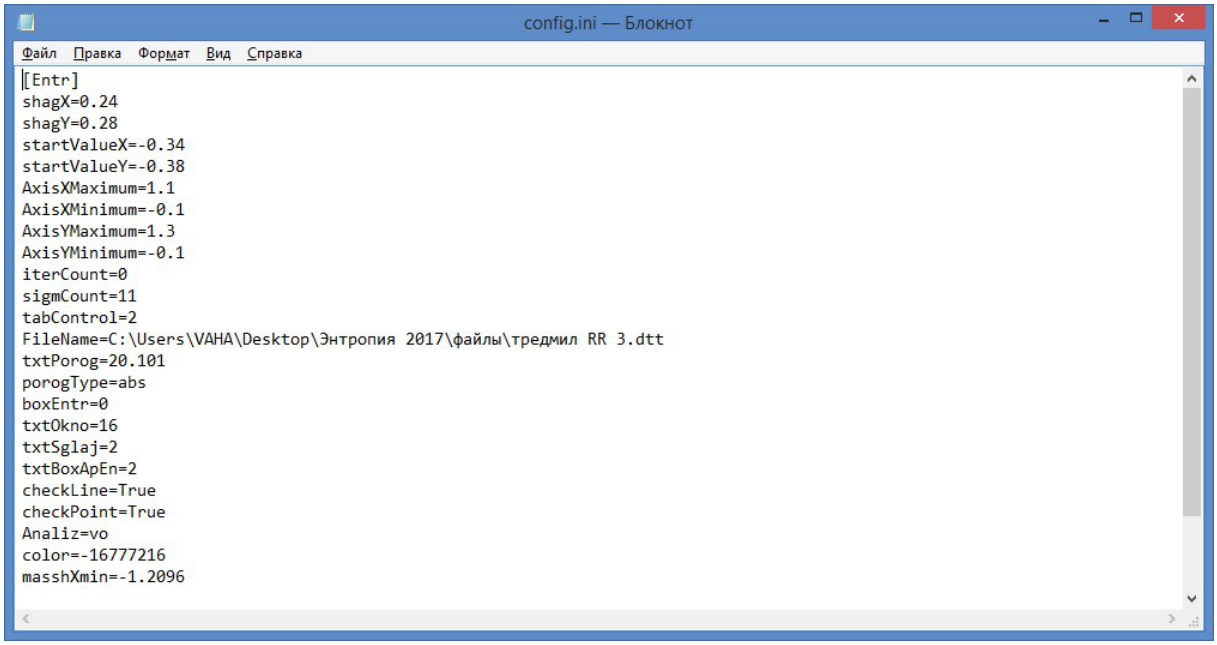


Рисунок 4.11. Конфігураційний файл.

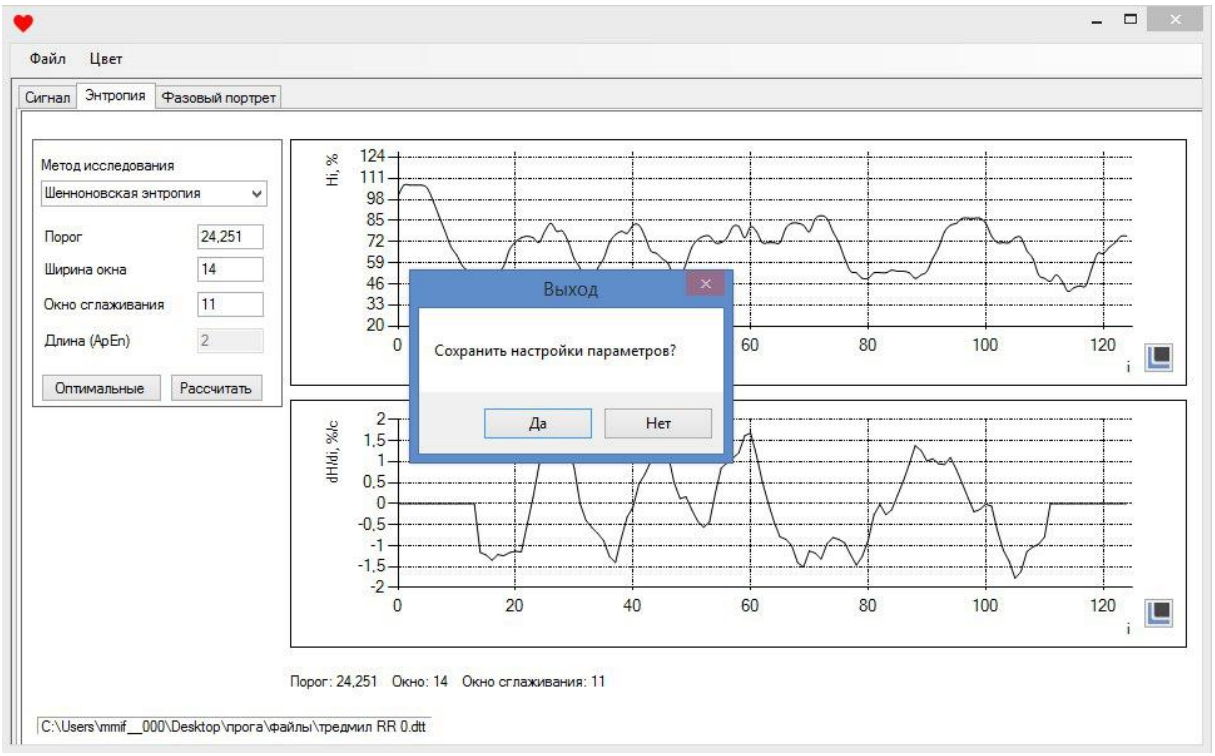


Рисунок 4.12. Запит на збереження налаштувань.

Висновки до розділу 4

В розділі детально описано всі можливі маніпуляції з програмою. Наведено знімки програми з поясненнями для кращого розуміння можливих дій. Розглянуто вибір даних для дослідження, налаштування обчислювальних алгоритмів, аналіз результатів.

РОЗДІЛ 5
РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ

При проведенні порівняльних експериментів між двома класами досліджуваних постало питання вибору оптимальних параметрів для обчислювальних алгоритмів ентропії. Враховуючи це, було запропоновано два варіанти:

1. Використовувати спільні параметри налаштування обчислювальних алгоритмів для всіх досліджуваних файлів. Спільні параметри знаходились, як середнє арифметичне між оптимальними параметрами для кожної окремої послідовності.

2. Використовувати індивідуальні параметри налаштування обчислювальних алгоритмів для всіх досліджуваних файлів окремо.

Для того, щоб вибрати один із варіантів, було проведено дослідження над контрольними пацієнтами, під час яких їм було запропоновано знаходитись під фізичними навантаженнями протягом 15 хв. Дані електрокардіограми, а саме:

– R-R інтервал;

– β параметр.

були зняті через такі інтервали:

– на початку експерименту;

– після 3 хвилин з початку експерименту;

– після 15 хвилин з початку експерименту;

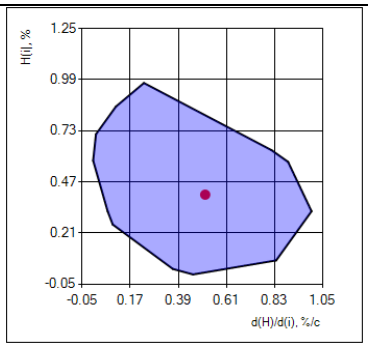
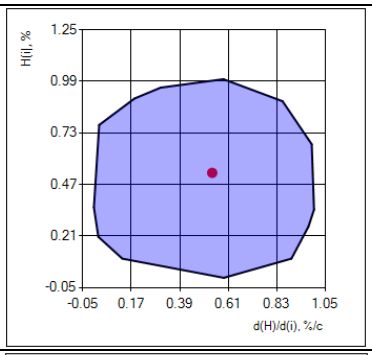
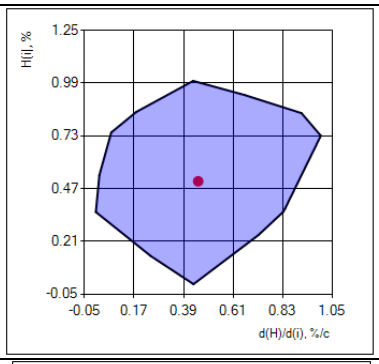
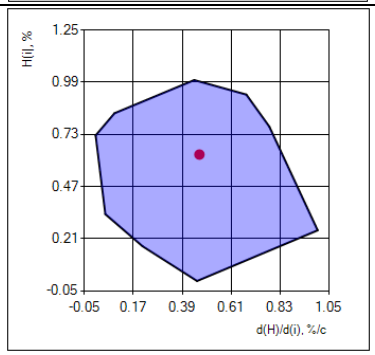
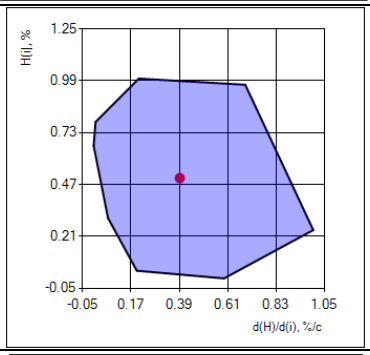
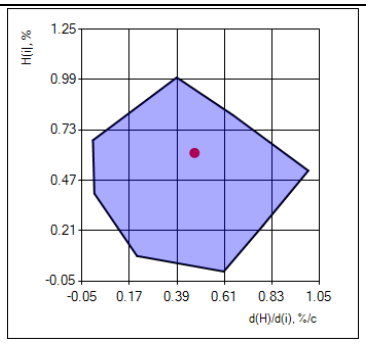
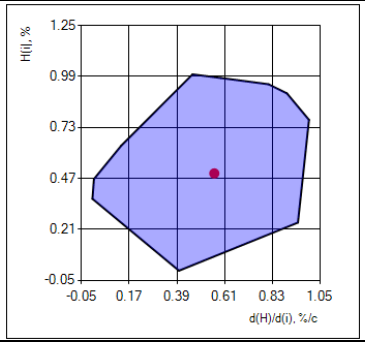
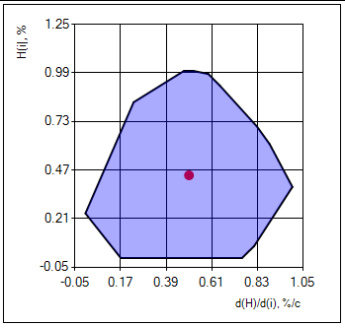
– після 3 хвилин відпочинку з моменту останнього навантаження.

В таблиці 5.1 та 5.2 наведені результати експерименту. З них можна побачити, що при індивідуально підібраних параметрах налаштувань обчислювальних алгоритмів ентропії (табл. 5.1) площа випуклої оболонки фазового портрету має лінійну залежність з інтервалами навантаження та

відпочинку досліджуваного. Якщо дивитись на табл. 5.2, де параметри знаходились, як середнє арифметичне між оптимальними для кожної окремої послідовності, то тут ми не можемо знайти таких закономірностей. Це зумовлено індивідуальними особливостями організму.

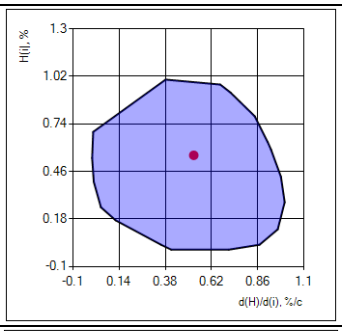
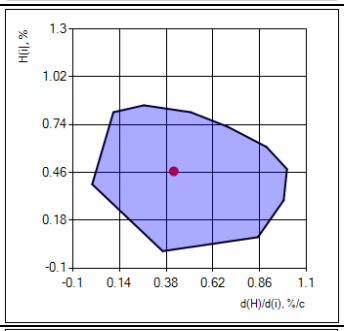
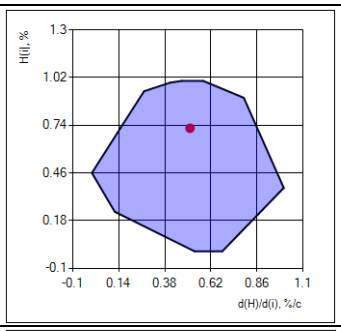
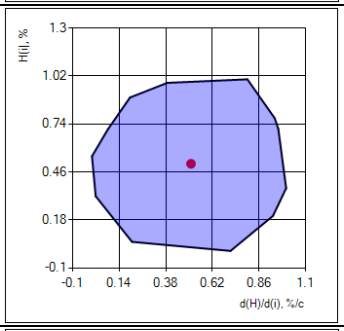
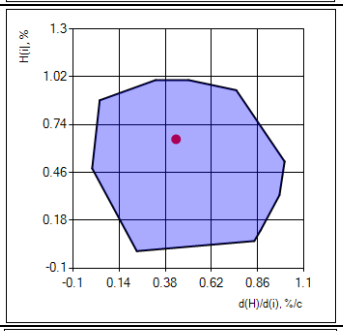
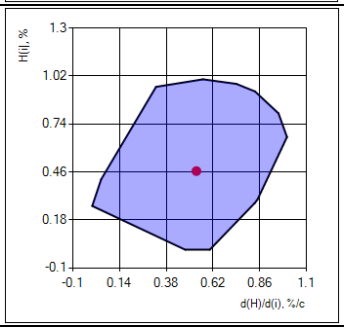
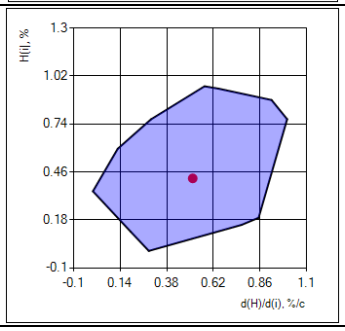
Таблиця 5.1

Дослідження з індивідуальними параметрами

Stage	CONVEX HULLS EPP		INTEGRAL PARAMETERS
	R-R interval	BETA parameter	
Baseline			$S_{RR} = 0.637$ $X_{RR} = 0.513$ $Y_{RR} = 0.405$ $S_{\beta R} = 0.819$ $X_{\beta R} = 0.537$ $Y_{\beta R} = 0.529$
Stress 3 min			$S_{RR} = 0.644$ $X_{RR} = 0.455$ $Y_{RR} = 0.506$ $S_{\beta R} = 0.666$ $X_{\beta R} = 0.468$ $Y_{\beta R} = 0.629$
Stress 15 min			$S_{RR} = 0.736$ $X_{RR} = 0.392$ $Y_{RR} = 0.502$ $S_{\beta R} = 0.625$ $X_{\beta R} = 0.472$ $Y_{\beta R} = 0.611$
Rest 3 min			$S_{RR} = 0.682$ $X_{RR} = 0.563$ $Y_{RR} = 0.496$ $S_{\beta R} = 0.695$ $X_{\beta R} = 0.5$ $Y_{\beta R} = 0.442$

Таблиця 5.2

Дослідження з груповими параметрами

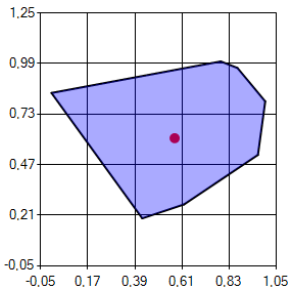
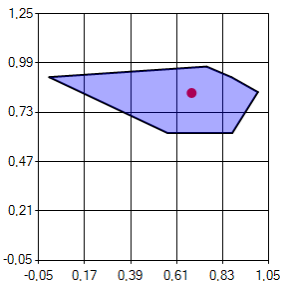
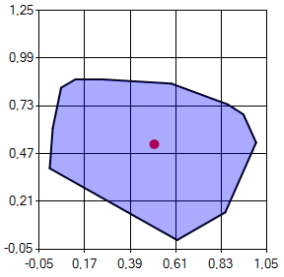
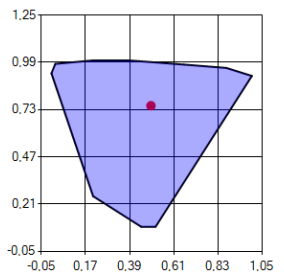
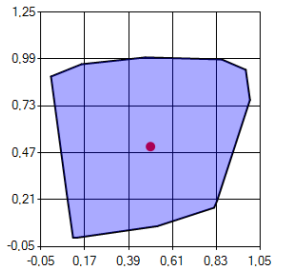
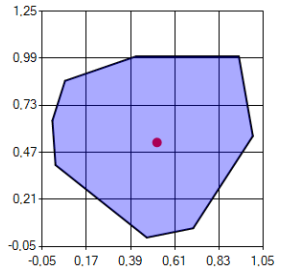
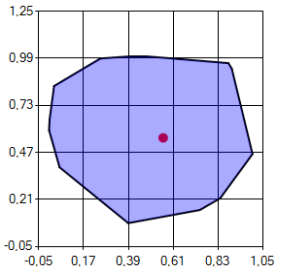
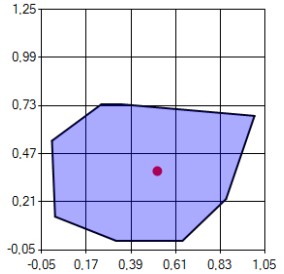
Stage	CONVEX HULLS EPP		INTEGRAL PARAMETERS
	R-R interval	BETA parameter	
Baseline		?	$S_{RR} = 0.777$ $X_{RR} = 0.528$ $Y_{RR} = 0.555$ $S_{\beta R} = 0.819$ $X_{\beta R} = 0.537$ $Y_{\beta R} = 0.529$
Stress 3 min			$S_{RR} = 0.610$ $X_{RR} = 0.419$ $Y_{RR} = 0.467$ $S_{\beta R} = 0.678$ $X_{\beta R} = 0.511$ $Y_{\beta R} = 0.722$
Stress 15 min			$S_{RR} = 0.793$ $X_{RR} = 0.510$ $Y_{RR} = 0.508$ $S_{\beta R} = 0.785$ $X_{\beta R} = 0.436$ $Y_{\beta R} = 0.654$
Rest 3 min			$S_{RR} = 0.666$ $X_{RR} = 0.535$ $Y_{RR} = 0.465$ $S_{\beta R} = 0.627$ $X_{\beta R} = 0.514$ $Y_{\beta R} = 0.423$

У табл. 5.3 представлені результати, отримані при тестуванні умовно здорового добровольця 44 років на тренажері тредміл. При тестуванні швидкість стрічки досягала 5,5 км/год, а кут нахилу поступово збільшувався до 14 %, що забезпечило на другій стадії метаболічний

еквівалент МЕТ = 10,2. Після цього тестований відпочивав протягом 3-х хвилин.

Таблиця 5.3

Динаміка зміни інтегральних показників при тредміл-тесті

№ п/п	СТАДІЯ	МЕТ	ОПУКЛІ ОБОЛОНКИ ФПЕ		ІНТЕГРАЛЬНІ ПОКАЗНИКИ
			***R-R* інтервалів**	**Показника β_T**	
0	Вихід	1			S_{RR} = 0,497 од. X_{RR} =0,576 од. Y_{RR} =0,606 од. $S_{\beta T}$ = 0,217 од. $X_{\beta T}$ =0,682 од. $Y_{\beta T}$ =0,831 од. *SDNN*=58 мс СКО β_T =0,02 од.
1	Навантаження 3 хв.	2,3			S_{RR} = 0,617 од. X_{RR} =0,507 од. Y_{RR} =0,521 од. $S_{\beta T}$ = 0,569 од. $X_{\beta T}$ =0,496 од. $Y_{\beta T}$ =0,751 од. *SDNN*=32 мс СКО β_T =0,04 од.
2	Навантаження 15 хв.	10,2			S_{RR} = 0,794 од. X_{RR} =0,5 од. Y_{RR} = 0,504 од. $S_{\beta T}$ = 0,749 од. $X_{\beta T}$ =0,522 од.; $Y_{\beta T}$ =0,524 од.; *SDNN*=20 мс СКО β_T =0,09 од.
3	Відпочинок 3 хв.	-			S_{RR} = 0,716, од. X_{RR} =0,561 од. Y_{RR} = 0,55 од. $S_{\beta T}$ = 0,58, од. $X_{\beta T}$ =0,52 од. $Y_{\beta T}$ = 0,376 од. *SDNN*=92 мс СКО β_T =0,08 од.

В процесі наростання навантаження показник $SDNN$ (СКО RR - інтервалів) зменшився на 66% (рис. 5.1), що узгоджується з відомими даними про збільшення впливу симпатичної частини вегетативної нервової системи при навантаженні [22]. Одночасно зі зменшенням варіабельності серцевого ритму інтегральний показник S_{RR} , який, на відміну від $SDNN$, характеризує не ступінь розкиду, а різноманітність RR - інтервалів, збільшився на 60%.

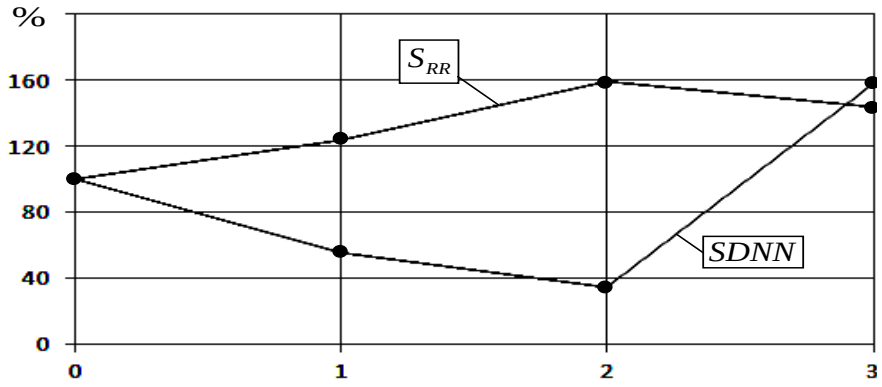


Рисунок 5.1. Динаміка зміни інтегральних показників серцевого ритму.

Можна припустити, що виявлений факт свідчить про пошук здоровим організмом найбільш економного способу регуляції серцевого *ритму.* Звісно така гіпотеза вимагає подальшого вивчення [28].

Два інших інтегральних показника, що характеризують мінливість (СКО β_T) І різноманітність ($S_{\beta T}$) значень симетрії зубця T при наростанні навантаження і відпочинку були співнапрямлені (рис. 5.2). У той же час зміни інтегрального показника $S_{\beta T}$ на першій стадії навантаження були більш виражені, ніж зміни показника СКО β_T , а на відпочинку більш вираженими виявилися зміни СКО β_T Виявлений ефект також вимагає додаткового вивчення [28].

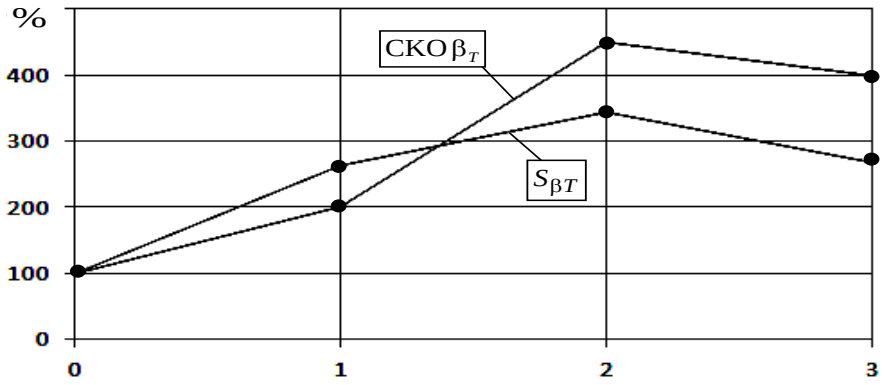


Рисунок 5.2. Динаміка зміни інтегральних показників симетрії зубця .

Цікаві результати отримані при дослідженні тонких змін ЕКГ у пацієнтів з ішемічною хворобою серця, яким проводилася операція аортокоронарного шунтування (АКШ). На рис. 5.3 представлена динаміка показника β_T до і після оперативного лікування хворій М. 60 років, якій у зв'язку з ураженням коронарних артерій встановлено три шунта [28].

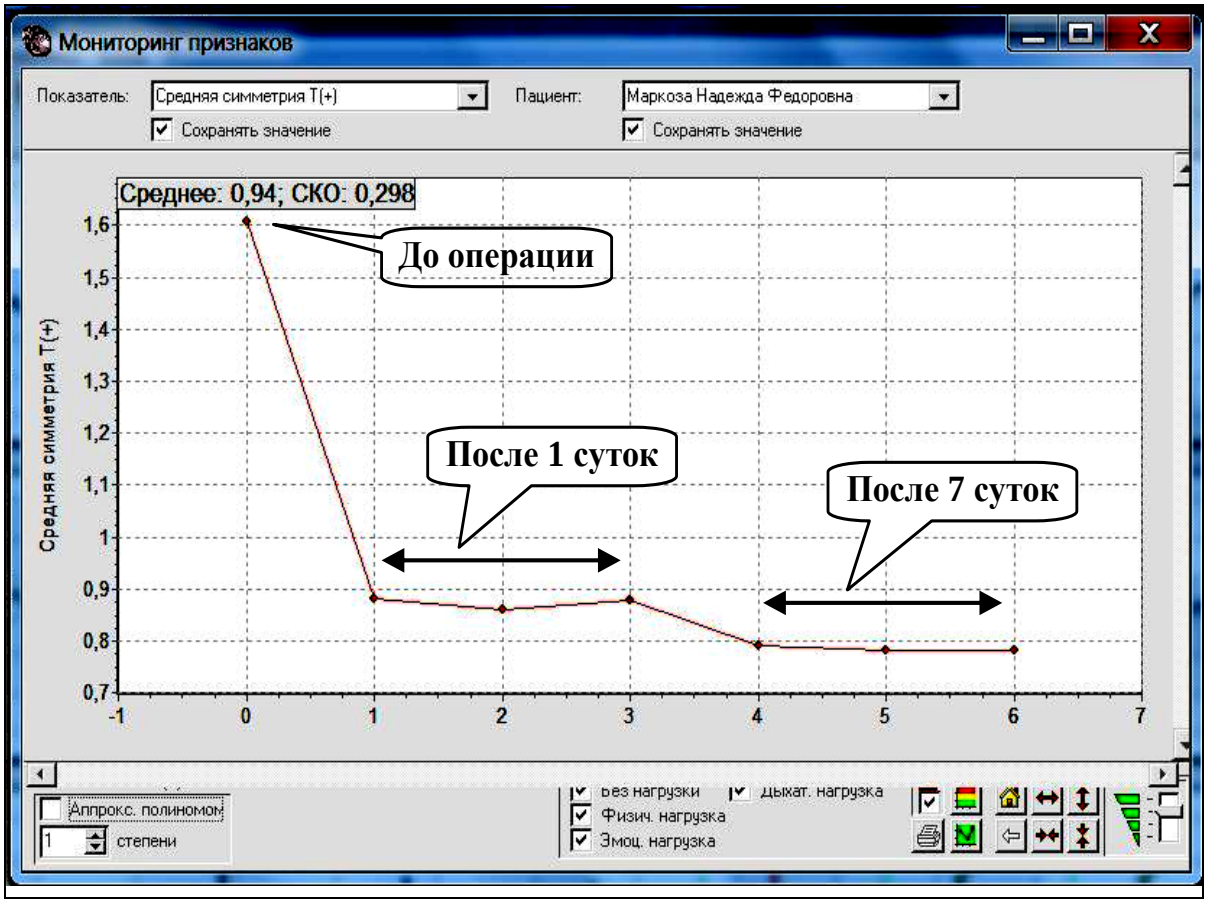


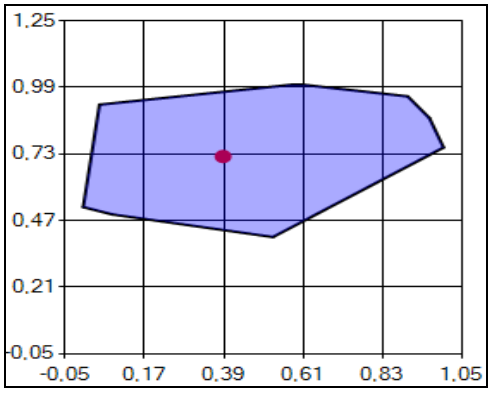
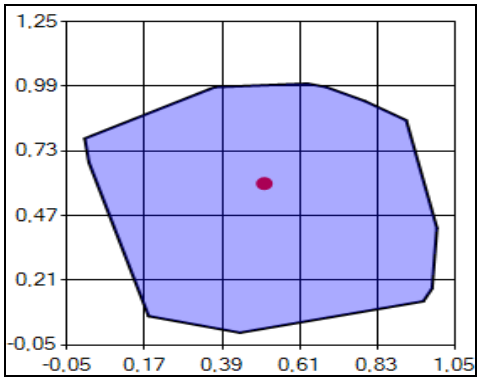
Рисунок 5.3. Динаміка показника β_T до і після операції АКШ

За день до операції показник симетрії зубця T дорівнював $\beta_T = 1{,}6$ од., що на 60% перевищувало нижню межу патологічних значень ($\beta_T = 1{,}05$ од). На першу добу після операції значення β_T нормалізувалися ($\beta_T \approx 0{,}9$ од.), що свідчить про відновлення кровотоку. На сьому добу після операції значення показника ще більш приблизилось до норми і досягло $\beta_T = 0{,}8$, що на 50% нижче патологічного значення до операції. Пацієнтка вдало пройшла реабілітаційний період і виписана через тиждень після операції [28].

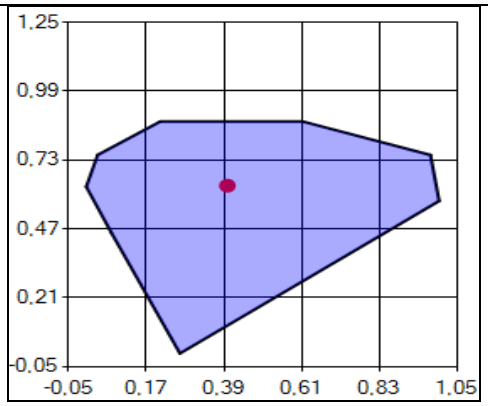
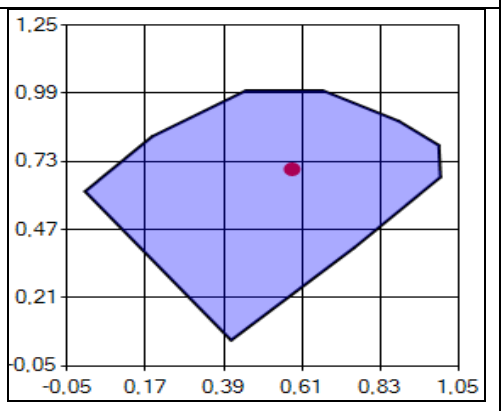
Цікаво, що на тлі нормалізації показника β_T спостерігалася динаміка фазових портретів ентропії (ФПЕ), побудованих для RR - інтервалів і показника β_T **(**табл. 5.4**).** При цьому вже на першу добу після операції площа S_{RR} опуклої оболонки ФПЕ RR - інтервалів збільшилася майже на 82%, а площа $S_{\beta T}$ опуклої оболонки ФПЕ показника β_T збільшилась на 2,4% [28].

Таблиця 5.4

Динаміка ФПЕ до і після аорто-коронарного шунтування (АКШ)

	ДО ОПЕРАЦІЇ	**1 ДОБА ПІСЛЯ ОПЕРАЦІЇ**
ФПЕ *RR* **- інтервалів**	 S_{RR} = 0,431 од.	 S_{RR} = 0,784 од.

Продовж. табл.5.4

	ДО ОПЕРАЦІЇ	1 ДОБА ПІСЛЯ ОПЕРАЦІЇ
ФПЕ показника β_T	 $S_{\beta T}$ = 0,535 од.	 $S_{\beta T}$ = 0,548 од.

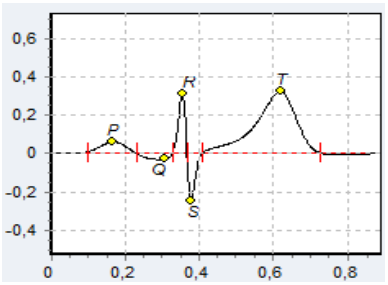
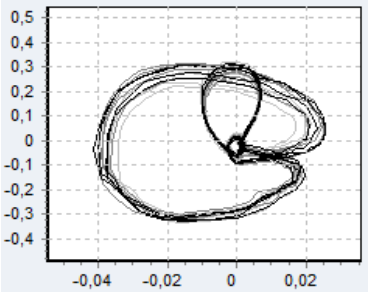
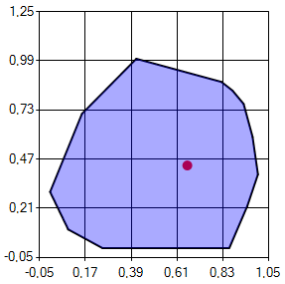
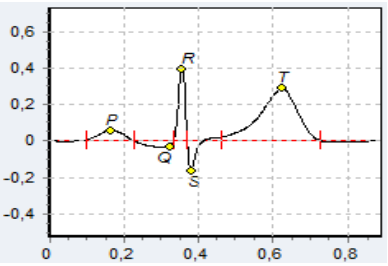
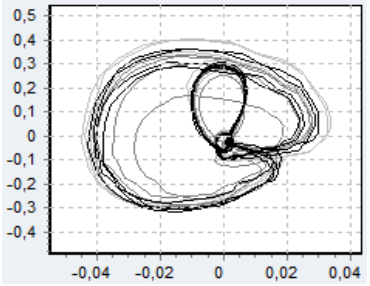
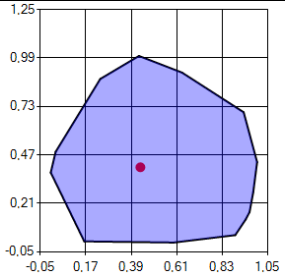
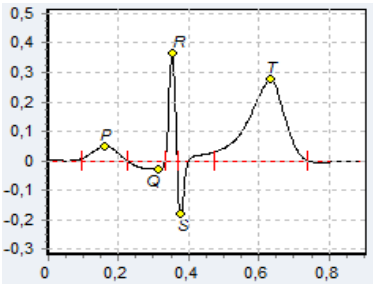
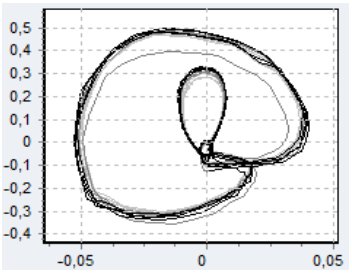
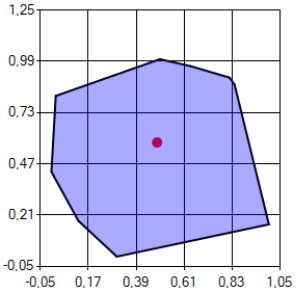
Виявлені властивості вказують на можливість використання методу фазаграфії для оцінки тонких змін ЕКГ до і після оперативного втручання і прогнозування результату лікування.

Багато лікарських препарати, в тому числі ті, що застосовуються в кардіологічній практиці, досить часто (від 30 до 70%) мають побічні дії [23]. Тому найактуальніше завдання оцінити можливості фазаграфії при аналізі тонких змін ЕКГ безпосередньо в процесі крапельного введення препаратів.

У таблиці 5.5 представлена динаміка зміни показників, які спостерігалися на ЕКГ пацієнтки І. 76 років в процесі крапельного внутрішньовенного вливання препаратів Панангін і Мексикор.

Таблиця 5.5

Динаміка показників ЕКГ під час першої крапельниці

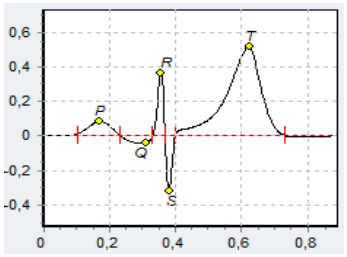
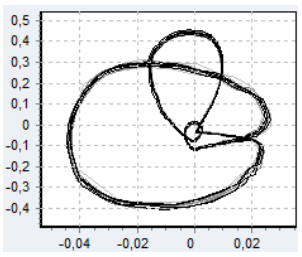
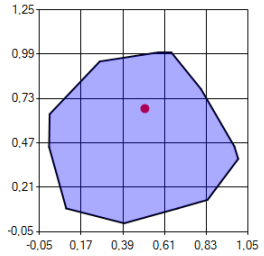
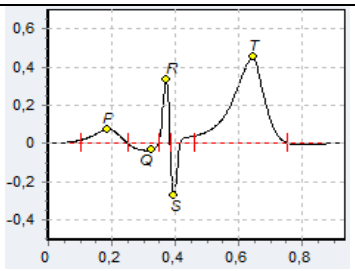
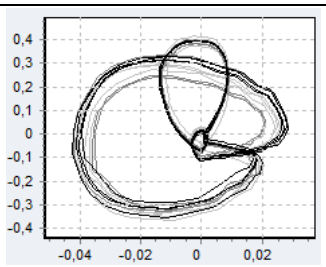
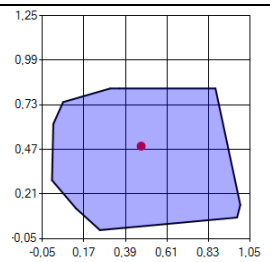
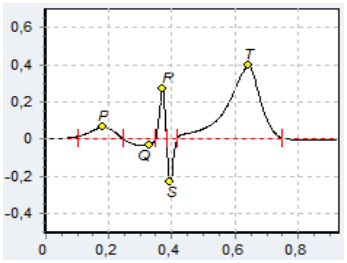
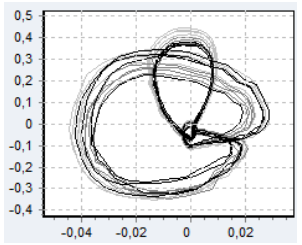
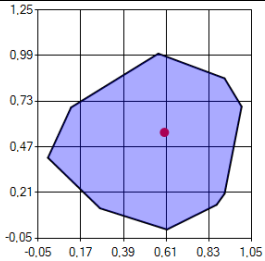
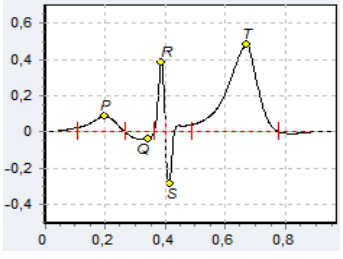
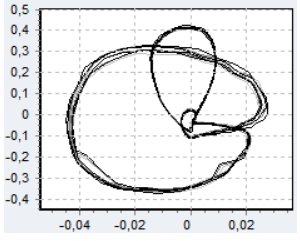
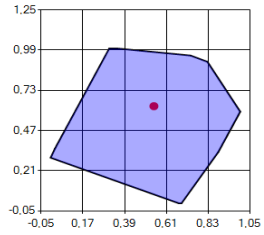
Час	СЕРЕДНІЙ ЦИКЛ	ФАЗОВИЙ ПОРТРЕТ ЕКГ	ОПУКЛА ОБОЛОНКА ФПЕ ПОКАЗНИКА β_T
15 хв.	 β_T = **0,65 од.**	 A_T = **0,44 мВ.**	 $S_{\beta T}$ = **0,764 од.**
25 хв.	 β_T = **0,71 од.**	 A_T = **0,29 мВ.**	 $S_{\beta T}$ = **0,754 од.**
40 хв.	 β_T = **0,64 од.**	 A_T = **0,28 мВ.**	 $S_{\beta T}$ = **0,744 од.**

Протягом всього періоду введення препаратів показник симетрії зубця T знаходилися в межах норми: $\beta_T = 0{,}653 \pm 0{,}014$ од. Стабільною була і площа ФПЕ показника β_T: $S_{\beta T} = 0{,}743 \pm 0{,}016$ од.

Через дві доби пацієнтці повторно була проведена процедура з цими ж препаратами (табл. 5.6) [28].

Таблиця 5.6

Динаміка показників ЕКГ під час другої крапельниці

Час	СЕРЕДНІЙ ЦИКЛ	ФАЗОВИЙ ПОРТРЕТ ЕКГ	ОПУКЛА ОБОЛОНКА ФПЕ ПОКАЗНИКА β_T
5 хв.	 β_T = 0,61 од.	 A_T = 0,52 мВ.	 $S_{\beta T}$ = 0,73 од.
10 хв.	 β_T = 0,65 од.	 A_T = 0,44 мВ.	 $S_{\beta T}$ = 0,697 од.
20 хв.	 β_T = 0,66 од.	 A_T = 0,38 мВ.	 $S_{\beta T}$ = 0,677 од.
30 хв.	 β_T = 0,63 од.	 A_T = 0,47 мВ.	 $S_{\beta T}$ = 0,65 од.

При введенні другої крапельниці показник симетрії зубця T також знаходився в межах норми: $\beta_T = 0{,}604 \pm 0{,}006$ од .. Однак спостерігалися

помітні зміни форми середнього циклу. Ці зміни викликані збільшенням на 36% амплітуди зубця T, яка перевищила амплітуду зубця R і привела до характерних змін форми фазового портрету ЕКГ.

Цікаво, що збільшення амплітуди зубця T, яке швидше за все пов'язане з гіперкаліміею від надмірного введення калієвих препаратів, супроводжувалися монотонним зниженням площі $S_{\beta T}$ ФПЕ показника β_T майже на 11%, тобто зменшенням різноманітності форми зубця T від циклу до циклу.

Серйозний прояв серцево - судинних захворювань – раптова серцева смерть, коли хворий гине практично миттєво (від декількох секунд до години) після початку серцевого нападу. Один із предикторів раптової серцевої смерті, який останнім часом став широко відомим в клінічних дослідженнях, заснований на виявленні ефекту електричної альтернації серця, який на ЕКГ проявляється в чередуванні елементів, наприклад, в чергуванні RR - інтервалів різної тривалості [28].

Комп'ютерний аналіз ефекту альтернації стає важливою характеристикою сучасних систем медичної діагностики. У той же час, на думку фахівців, існують комп'ютерні алгоритми не забезпечують необхідну надійність виявлення цього ефекту в реальних клінічних умовах.

Покажемо, що запропонований метод дозволяє виявити тонкі зміни сигналу, викликані ефектом альтернації серця, і відрізнити такі зміни від зовні схожих мікроколивань елементів ЕКГ, не пов'язаних з цим ефектом.

На рис. 5.4 показані реальні ритмограми двох пацієнтів. Одна з ритмограм належить умовно здоровому чоловіку 32 років, у якого варіабельність серцевого ритму а перебувала в межах фізіологічної норми ($SDNN$ = 50 мс). Друга ритмограма належить жінці 68 років, у якій ефект електричної альтернації серця привів до регулярного чергування тривалостей RR -інтервалів.

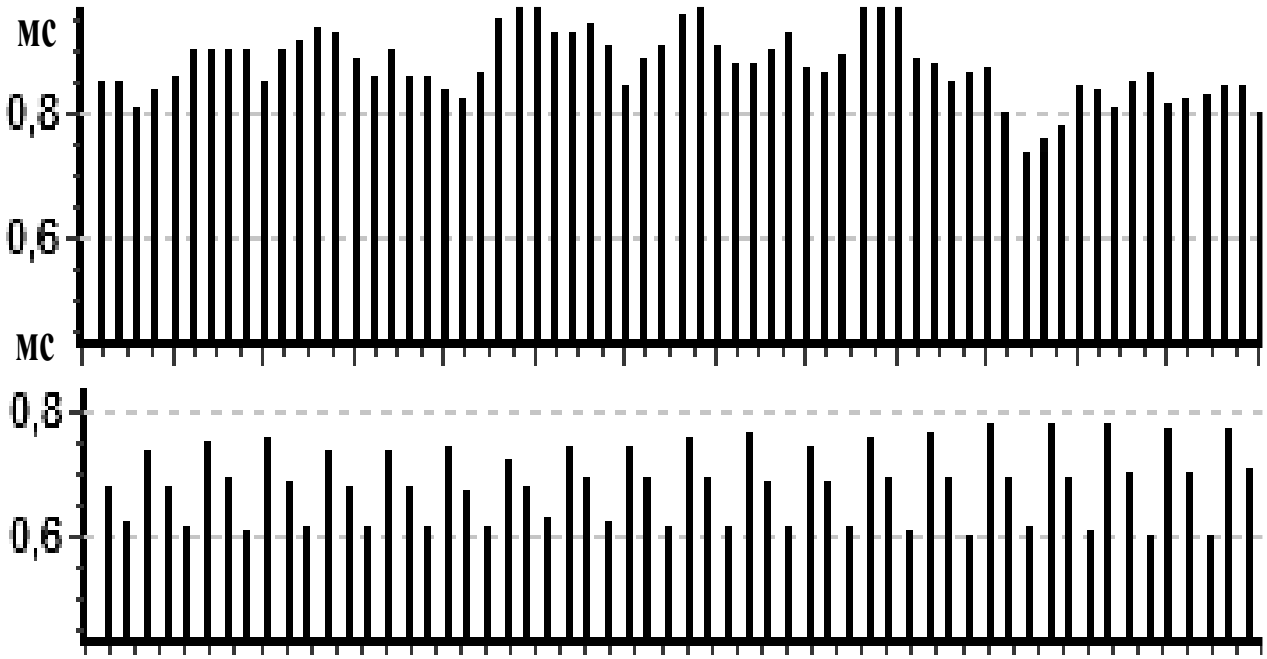


Рисунок. 5.4. Реальні ритмограми умовно здорового пацієнта (вгорі) і пацієнтки з альтернацією RR - інтервалів (внизу).

Зауважимо, що традиційний статистичний показник варіабельності серцевого ритму на ритмограмі з альтернацією RR - інтервалів також знаходився в межах фізіологічної норми ($SDNN$ = 62 мс). Саме за цим показником можна судити про валідність даного експерименту, адже пацієнти, над якими проводились дослідження належать до різних гендорних груп, що може викликати сумніви щодо результатів та реперехентативності експерименту в цілому [28].

Висновки до розділу 5

В розділі детально описано всі проведені експерименти, що дозволили показати діагностичну цінність досліджуваного методу.

Зміна площі випуклої оболонки фазового портрету несе важливу діагностичну інформацію про зміну хаотичності серцевого ритму.

Дослідження показали, що характеристики випуклої оболонки фазового портрету мають лінійну залежність з навантаженням, під яким знаходиться серцево-судинна система досліджуваного.

РОЗДІЛ 6
СТАРТАП-ПРОЕКТ

6.1. Призначення проекту

Програмна система дозволяє діагностувати захворювання ССС за допомогою нових алгоритмів.

6.2. Суть проекту

Складові:

– підсистема зчитування даних;

– підсистема обробки та фільтрації даних;

– підсистема розрахунку ентропії за даними;

– підсистема побудови випуклої оболонки фазового портрету та виведення результату.

6.3. Можливості

Прогнозування хвороб ССС за даними ЕКГ пацієнта. Бізнес-модель: орієнтація на клініки, медичні лабораторії, лікарні та дослідні інститути.

Цінний продукт:

1. Сукупність товарів-послуг

Комплекс послуг: діагностика захворювань ССС проводиться за рахунок роботи альтернативних алгоритмів обробки ЕКГ.

2. Вирішує такі проблеми клініки:

- встановлення діагнозу у пацієнтів із захворюваннями ССС

- прогнозування стану хворого;

3. Підтримка і сервіс програмної системи: додатковий сервіс надається шляхом підтримки системи розробником.

4. Продуктова лінійка (унікальність): для простих випадків - модуль «Діагностика захворювань ССС»

6.4. Унікальність пропозиції

На ринку не існує аналогів у даної програмної системи.

6.5. Ринок

Спочатку концентрація (фокусування) на одному сегменті (приватні клініки), потім розширення сегменту (державні заклади, міжнародний ринок).

Сегмент споживачів

1. Ринок. Середнє сегментування – надання послуг діагностичним клінікам. Спочатку послуга спрямована на приватні клініки. В майбутньому планується розширення списку сегментів (міжнародний ринок).

2. Що об’єднує клініки – мета надати якісне медичне обслуговування.

3. Звичний спосіб покупки - звичайний настільний додаток для комп’ютера.

Канали збуту

1. Прямі – прямий продаж клінікам через сайт.

Фріміум – частина послуг (наприклад, інформаційна) безкоштовна, частина – платна (або Shareware - 1 місяць безкоштовне надання послуг, далі – платно).

2. Функції

Продажні (велике рішення) – в системі присутні всі модулі. Надання рекомендацій в реальному часі – модуль вартістю 1000 грн.

Взаємодія із споживачами

1. Залучення клінік, їх утримання.

2. Підтримка (пошта, телефон, особисто, форум)

3. Залучення клієнтів завдяки науковим та медичним конференціям.

4. Повторний продаж системи завдяки реалізації додаткових модулів-послуг.

Дохід (монетизація):

1. Клієнт готовий платити за систему, яка буде працювати та підтримуватися розробником.

Ключові види діяльності

1. Процес створення цінності

Адміністрування, розробка системи (в тому числі і наукова діяльність), навчання користувачів системи, створення додаткових модулів (за потребою)

2. Виробництво товару-послуг, продаж.

3. Підтримка рішення

Ключові ресурси

1. Матеріальні (в т.ч. комп'ютерна техніка) - 3 персональні комп'ютери, принтер.

2. Інтелектуальні (в т.ч. патенти та ліцензії).

3. Людські:

- 2 інженери-програмісти відповідають за реалізацію (у тому числі, дизайнер - за інтерфейс та зовнішній вигляд),

- лікар – за стан здоров'я,

- маркетолог – за дослідження потреб, доцільності подальшого розширення системи,

- директор-керівник – за зміст проекту,

- бухгалтер – за фінанси.

4. Фінансові – фінансування всіх членів команди (в т.ч., програмістів-розробників)

5. Час реалізації проекту

Ключові партнери

1. Забезпечення ресурсами (комп’ютерною технікою та обладнанням)

2. Оптимізація та економія в продажах

3. Зниження ризиків і невизначеності

4. Подальший розвиток проекту

Витрати

1. Разові (придбання комп’ютерної техніки, обладнання, ліцензії, реклама, інше)

ПК, принтер; реклама продукції.

2. Постійні, не пов’язані з виробництвом (оренда, заробітна плата, зв'язок, інші)

Споживчі властивості товару

Базові (очікувані) властивості придбавши систему клініка очікує на підвищення відсотку вірно встановлених діагнозів. Основні (бажані) властивості зручність під час використання точність результату надійність у роботі Властивості, що захоплюють властивості зв'язок системи з лікарем

Дослідження ринку

Під час початкового періоду дослідження ринку потрібно відвідувати медичні конференції, на яких розглядаються дані проблеми.

Дослідження конкурентного оточення

Настільний додаток – це комплексна інформаційна система зчитування та обробки даних ЕКГ дослідження, яка забезпечує при одноразовому введенні ресурсу в систему можливість багатогранної обробки та використання інформації для задоволення різних інформаційних потреб, з оперативним розподіленим багатокористувацьким доступом через інтерфейс єдиного вигляду (характеру).

Повний доступ до всіх ресурсів можна здійснити одразу після встановлення додатку на комп’ютер (тобто процедура реєстрації не потрібна). Користувачам видається повний доступ до інформації зі змогою редагування даних (лікарям). Структура додатку включає в себе ресурси, які фізично розміщені у внутрішній базі даних, котра постачається разом із додатком.

Конкурентів у програмної системи немає.

Маркетингова стратегія просування

План розвитку товару

І. Обов'язкові властивості

Вирішення основної проблеми – діагностика захворювань ССС у пацієнтів.

Виконання основної функції – встановлення діагнозу у хворих.

ІІ. Користувацькі властивості продукту

Властивості комерційного призначення: затребуваність товару споживачами - одна група клієнтів - діагностичні клініки.

Додавання модулів до системи розширює споживчі властивості. Збільшення функцій (модулів) збільшує ціну товару. Додаток використовують в комплексі із комп’ютером.

Управління ціною

1. Формування ціни на продукт (витрати на розробку системи, організаційні заходи, рекламні заходи).

2. Формування системи лояльності: системи знижок, заохочень (для збільшення продажів, повторних продажів, партнерських продажів): робота із БД клінік для покращення алгоритмів.

Система збуту.

Спочатку давати можливість користуватися додатком на 50 пацієнтів безкоштовно. Потім, за бажанням купувати додаток. Процедура збуту через інтернет або через представника. Елементи фінансового плану

Вимоги, обмеження, рішення, його зовнішній вигляд, модель.

Опис бізнес-проекту:

Призначення бізнес-проекту - надання інформаційних послуг хворим на серцево-судинні захворювання.

Поточний етап реалізації бізнес-проекту – реалізований етап.

Можливості послуги щодо вирішення проблем споживачів:

– збір історичних даних (щоденник користувача);

– формування попереджень;

– Прогнозування стану хворого на серцево-судинні заховрювання за історичними даними

– Опис товару-послуги

Опис.

«Програмна система Entropia», метою якої є діагностика захворювань ССС. Вона проводиться за рахунок роботи альтернативних алгоритмів обробки та оцінки ЕКГ. Її основні функції:

- встановлення діагнозу у пацієнтів із захворюваннями ССС;

- прогнозування стану хворого.

Складові:

1) Безпосередньо програмний додаток

2) БД ЕКГ хворих та результатів діагностики:

Відмінні якості та інноваційність послуги.

Виконання діагностики за рахунок нових алгоритмів обробки ЕКГ.

Маркетинг та продаж

Ціна на товар схильна до частих коливань, і при потребі необхідно негайно внести зміни до цінової політики

Ціна продажу набагато перевищує витрати на виробництво.

Економічна модель

Створення додатку для «Android» буде проводитися у середовищі розробки «Android Studio», яке є безкоштовним.

Для можливості продажу програми у «Google Play» необхідно внести одноразовий платіж – реєстраційний внесок. Як вже було сказано, ми

створюємо комплементарний продукт, тож наш прибуток буде залежати від кількості збуту апаратних пристроїв. Ліцензійний ключ активації буде внесений у прошивку окремого пристрою, тож при його підключені до гаджету з ОС «Android» програмний додаток буде автоматично активізуватися і завантажуватися з «Google Play».

Фінансовий план

Витрати

Таблиця 5.1

Витрати на паливо й енергію на технологічні цілі

	Кількість	Ціна	Вартість (грн)	
Папір	2	88	175	Щомісячні витрати
Оренда	1	5000	5000	
Інтернет	1	130	130	
Комп'ютери	2	8000	16000	Перший місяць
Принтер	1	2000	2000	
Підписка Play Market	1	520	520	
Заробітна плата				
Програміст	2	22400	44800	Сума = 52600
Тестувальник	1	2800	2800	
Адмінастратор	1	5000	5000	

Докладний фінансовий план

Розрахунок точки беззбитковості.

Точка беззбитковості дозволить визначити, коли проект перестане бути збитковим. У нашому випадку якщо программу купить 3874 людей – ми вийдемо в 0, тому 3874 – наша точка беззбитковості (рис. 5.1.).

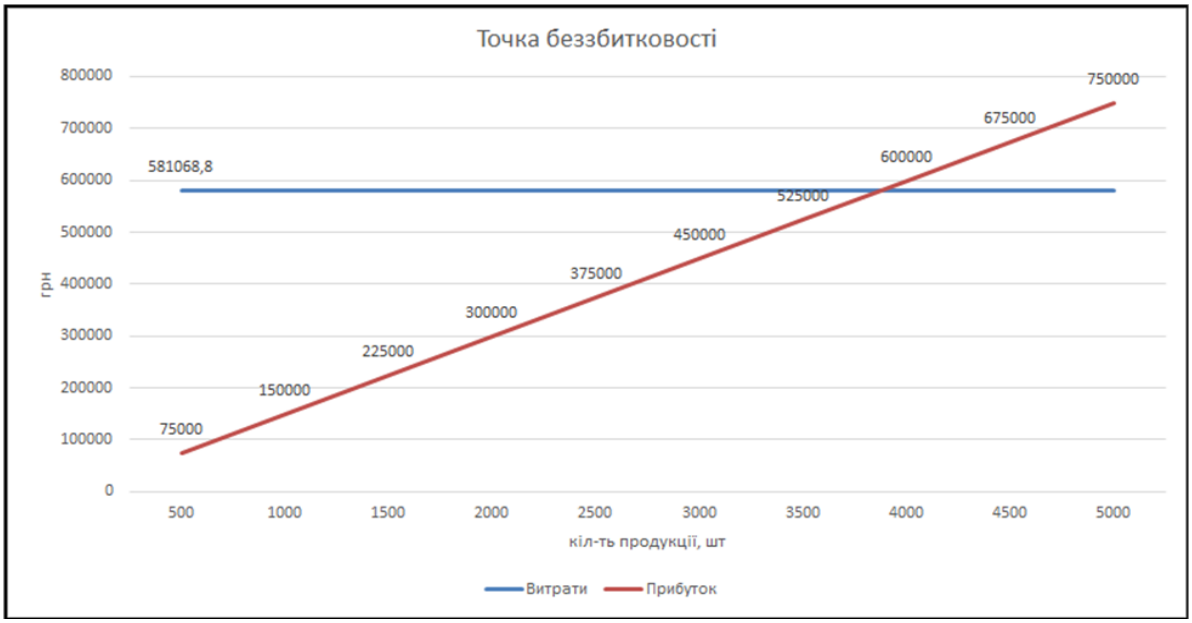


Рисунок 5.1. Точка беззбитковості

Окупність зображена на рис. 5.2.

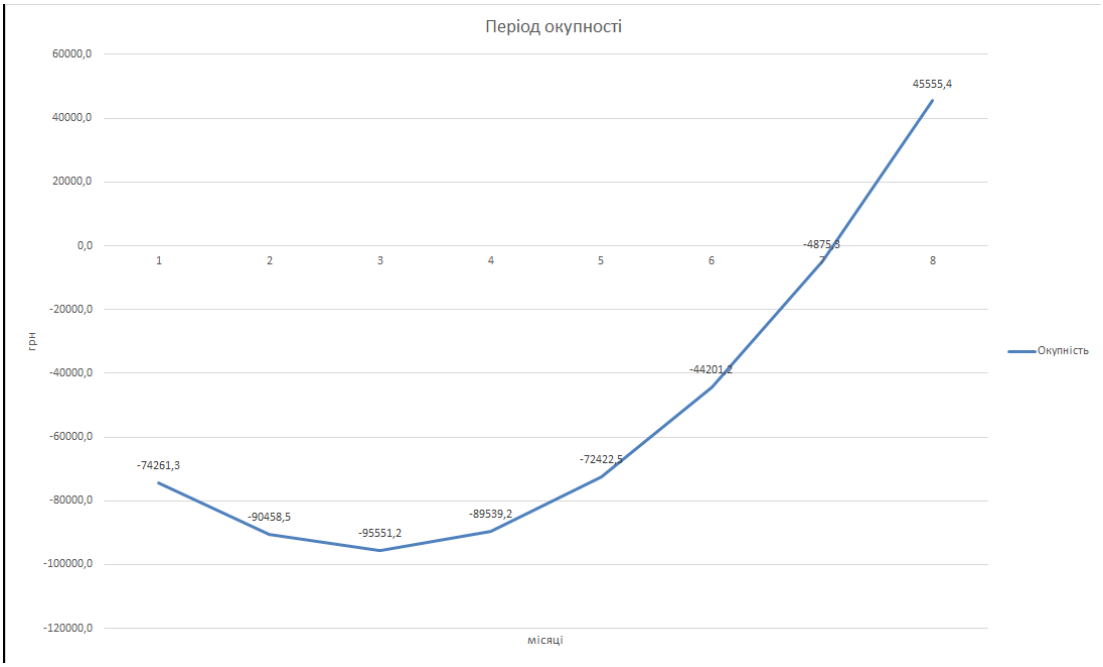


Рисунок 5.2. Період окупності

Період окупності становить 7,17 місяці.

Рентабельність

Рентабельність продажу (операційний прибуток ділимо на валовий прибуток):

(168,931 / 581,068) * 100% = 22,52%

Рентабельність інвестицій (операційний прибуток з вирахуваним податком на прибуток ділимо на початкові інвестиції):

((168,931 * (1 – 0.18)) / 95,551) * 100% = 144,97%

Висновки до розділу 6

Створено стартап-проект на основі магістерської дисертації. Розраховано його фінансовий план, точку беззбитковості. Створено план розвитку товару. Розраховано собівартість одиниці товару.

ВИСНОВКИ

1. В результаті виконання дипломної роботи було розроблено програмний продукт для оцінювання впливу зовнішніх втручань на організм людини на основі аналізу хаотичності фрагментів ЕКГ.

2. Був проведений детальний аналіз методів синергетики та їх застосування в медицині. Було досліджено наступні методи оцінки ентропії:

- шеннонівська ентропія
- переставна ентропія
- апроксимаціна ентропія

Всі перераховані алгоритми були реалізовані в програмному продукті.

3. На підставі проведеного аналізу можна зробити висновок, що сучасна наука має у своєму розпорядженні ефективні методи і обчислювальні алгоритми, які дозволяють з різних сторін оцінити ступінь хаотичність часових рядів, породжуваних медико-біологічними системами [28].

4. Було досліджено зміну ентропії показників ЕКГ і їх фазові портрети. На основі цього, було запропоновано новий метод оцінки зміни у часі ентропії показників елктрокардіограми на основі її фазового портрету та аналізу його випуклої оболонки.

5. Програмно реалізувано новий метод оцінювання зміни у часі ентропії показників електрокардіограми на основі її фазового портрету та алгоритму обчислення його опуклої оболонки.

6. Статистичні експерименти, проведені з використанням розробленої програмної системи, показали, що зміна площі випуклої оболонки фазового портрету несе важливу діагностичну інформацію про зміну хаотичності серцевого ритму.

7. Також, дослідження показали, що характеристики випуклої оболонки фазового портрету мають лінійну залежність з навантаженням, під яким знаходиться серцево-судинна система досліджуваного.

8. В ході проведення досліджень було встановлено, що оптимальні параметри налаштувань потрібно використовувати для кожного конкретного сигналу, а не для середнього по досліджуваній вибірці.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ослопов В. Н. Обмеження в автоматизованому комп'ютерному аналізі електрокардіорграми / В. Н. Ослопов, А. Р. Садикова, Т. С. Федосєєва. // 93. – 2012. – №4. – С. 687.

2. Файнзільберг Л. С. Основи фазаграфії / Л. С. Файнзільберг. – Київ, 2017. – (Освіта України). – 264 р.

3. Фрумин Л. Л., Шарк М. Б. О фазовом портрете электрокардиограммы // Автометрия – 1993 – № 2. – С. 51–54.

4. Файнзільберг Л. С. Компьютерная діагностика по фазовому портрету электрокардиограммы / Л. С. Файнзільберг. – Київ, 2013. – (Освіта України). – 191 р.

5. Файнзильберг Л. С. ФАЗАГРАФ® — эффективная информационная технология обработки ЭКГ в задаче скрининга ишемической болезни сердца // Клиническая информатика и телемедицина. – 2010. – Т. 6. – Вып. 7. – С. 22–30.

6. Скрининг ишемии миокарда путем оценки фазы реполяризации. / [Д. Д. Дячук, А. М. Кравченко, Л. С. Файнзильберг та ін.]. // Украинский кардио журнал. – 2016. – №6. – С. 82.

7. Майоров О. Ю. Повышение надежности исследований детерминированного хаоса (ЭЭГ, ЭКГ и вариабельность сердечного ритма) в методах нелинейного анализа биоэлектрической активности. / О. Ю. Майоров, В. Н. Фенченко. // 6. – 2009. – С. 10.

8. Шаповалов В. И. Об основных законах управления тенденциями / В. И. Шаповалов. // 2. – 2005. – №2. – С. 11.

9. Анищенко В. С. Степень хаотичности как критерий диагностики [Електронний ресурс] / В. С. Анищенко // Самоорганизация и неравновесные процессы в физике, химии и биологии. – 2006. – Режим доступу до ресурсу: http://sinsam.kirsoft.com.ru/KSNews_331.htm.

10. Яшин А. А. Жизнь имеет значение / А. А. Яшин. – Москва, 2010. – (Физика живых и эволюционных процессов). – 264 p.

11. Файнзильберг Л.С., Ориховская К.Б., Ваховский И.В. Оценка хаотичности формы фрагментов одноканальной электрокардиограммы // Кибернетика и вычислительная техника. – 2016. – №1. – С. 1-22.

12. Файнзільберг Л. С. Обобщенный метод обработки циклических сигналов сложной формы в многомерном пространстве параметров / Л. С. Файнзільберг. // Информатика и вычислительная техника. – 2015. – №3. – С. 24–39.

13. Файнзільберг Л. С. Ефективний метод аналізу діагностичних особливостей на основі зашумленої ЕКГ / Л. С. Файнзільберг, Н. А. Матушевич. // Конструкторські системи та машини. – 2016. – №2. – С. 84.

14. Файнзильберг Л. С. Информационные технологии обработки сигналов сложной формы. Теория и практика. – Киев: Наукова Думка, 2008. – С. 333.

15. Цирлин А. М. Оптимальные процессы в необратимой термодинамике и микроэкономике / А. М. Цирлин. – Москва: Физматлит, 2003. – 416 p.

16. Пригожин И. Р. Конец определённости: время, хаос и новые законы природы / И. Р. Пригожин. – Ижевск: Научно-издательский центр «Регулярная и хаотическая динамика», 2000. – 208 p.

17. Климонтович Ю. Л. Введение в физику открытых систем / Ю. Л. Климонтович. – Москва: «Янус», 2002. – 284 p.

18. Хадарцев А. А. Внутренние болезни с позиции теории хаоса и самоорганизации систем (научный обзор) / А. А. Хадарцев, В. М. Есков. // "Терапевт". – 2015. – №1. – С. 35–42.

19. Bandt C. Permutation entropy – a natural complexity measure for time series / C. Bandt, B. Pompe. // Phys Rev Lett. – 2002. – №17. – С. 17–41.

20. Pincus S. M. Approximate entropy as a measure of system complexity. / Pincus. // Proc Natl Acad Sci USA. – 1991. – №88. – С. 301.

21. Gorban I. Entropy of uncertainty / Gorban. // Math Machines and Systems. – 2013. – №2. – С. 105.

22. Novel predictor of prognosis from exercise stress testing: heart rate variability response to the exercise treadmill test / [F. E. Dewey, J. V. Freeman, G. Engel та ін.]. // American Heart Journal. – 2007. – №153. – С. 281.

23. Власова И. В. Все больше и больше побочных эффектов в лекарствах / И. В. Власова. // Коммерческие биотехнологии. – 2007. – №10. – С. 14.

24. Bian C., Qin C., Ma Q.D., Shen Q. Modified permutation-entropy analysis of heartbeat dynamics // Physical Review E. – 2012. – No. 85.

25. Parlitz U. Berg S. Luther, S. Schirdewan A. Kurths J. Wessel N. Classifying cardiac biosignals using ordinal pattern statistics and symbolic dynamics // Computers in Biology and Medicine. – 2012. – No. 42. – P. 319–327.

26. Frank B., Pompe B., Schneider U., Hoyer D. Permutation entropy improves fetal behavioural state classification based on heart rate analysis from biomagnetic recordings in near term fetuses // Medical & Biological Engineering & Computing. – 2006. – No. 44. – P. 179–187.

27. Дурнова Н.Ю., Довгалевский Я.П., Бурлака А.Н., Киселев А.Р., Фурман Н.В. Изучение зависимостей между показателями вариационной пульсометрии, энтропии ритма сердца, временного и спектрального анализов вариабельности ритма сердца в норме и при ишемической болезни сердца // Саратовский научно-медицинский журнал. – 2011. – том 7. – № 3. – С. 608-611.

28. Fainzilberg L., Orikhovska K., Vakhovskyi I. Analysis of subtle changes in biomedical signals based on entropy phase portrait // Биомедицинская инженерия и электроника. – 2017. – №3. – С. 44-66.

29. Файнзільберг Л.С., Оріховська К.Б., Ваховський І.В. Інструментальна система для оцінки хаотичності форми фрагментів

одноканальної електрокардіограми // Вітчизняні інженерні розробки для охорони здоров'я.-2016.-№2. – С. 1-2.

30. Ваховський І.В. Ступінь хаотичності біологічного сигналу як додаткова ознака стану серцево-судинної системи // VI Всеукраїнська школа-семінар молодих вчених і студентів «Сучасні комп'ютерні інформаційні технології». – 2016. - №6. – С. 1-2.

31. Мехтиев Т.Т., Кузнецова М.Н. Значение синергетики в медицине // Молодые ученные – здравоохранению. – 2016. – № 77. – С. 1-3.

32. Исаева В.В. Синергетика для биологов. / Рос. акад. наук. ДВО. Институт биологии моря ,Дальневосточный государственный униерситет . – М. : Наука, 2005 . – 158 с. - ISBN 5-02-033973-3.

33. Энтропии и фракталы в анализе данных / О. В. Чумак . – 2-е изд., испр. и доп . – М.; Ижевск : Регулярная и хаотическая динамика, 2012 . – 168 с. - ISBN 978-5-93972-940-6.

34. Берестин Д. К. Хаотическая динамика поведения параметров сердечно-сосудистой системы под воздействием крепких алкогольных напитков / Д. К. Берестин, А. Н. Булдин, Т. В. Гавриленко. // Вестник новых медицинских технологий. – 2013. – С. 11.

35. Энтропийные методы оценки уровня анестезии по ээг-сигналу / А. П.Немирко, Л. А. Манило, А. Н. Калиниченко, С. С. Волкова. // Информационно-управляющие системы. – 2010. – С. 6.

36. Вентцель Е.С. Теория вероятностей: учебник для вузов. – М.: Высшая школа, 1999. – 576 с.

37. Кузнецов А.А. Методы анализа и обработки электрокардиографических сигналов: новые подходы к выделению информации: монография. – В.: ВлГУ, 2008. – 140 с.

38. Сотников П.И. Выделение характерных признаков сигнала электроэнцефалограммы с помощью анализа энтропии // Наука и Образование. – 2014. – № 11. – С. 555–570.

39. Немирко А.П., Манило Л.А., Калиниченко А.Н., Волкова С.С. Сравнительный анализ применения различных оценок энтропии ЭЭГ-сигнала для распознавания стадий наркоза // Биотехносфера. – 2010. – № 3. – С. 3-10.

40. Takens F. Detecting strange attractors in turbulence // Dynamical systems and turbulence: lecture notes in mathematics. – 1981. – Vol. 898. – P. 366-381.

41. Pincus S.M. Approximate entropy as a measure of system complexity // Proceedings of the National Academy of Sciences. –1991. – Vol. 88. – P. 2297-2301.

42. Данильчук А.Б. Использование энтропийных показателей для моделирования динамики сложных социально-экономических систем // Economics. – 2014. – № 3. – С. 19-24.

43. Юшковская О.Г. Новый подход к оценке эффективности санаторно-курортной реабилитации больных ишемической болезнью сердца // Физкультура в профилактике, лечении и реабилитации. – 2004. – № 1. – С. 22-26.

44. Costa M., Ary L., Goldberger A.L., Peng C.-K. Multiscale entropy analysis of biological signals // Physical Review E. – 2005. – No. 71.

45. Pincus SM, Goldberger AL. Physiological time-series analysis: what does regularity quantify? // The American journal of physiology. – 1994. – Vol. 266. – P. 1643-1656.

46. Joshua S. Richman J., Moorman *R.* Physiological time-series analysis using approximate entropy and sample entropy // The American journal of physiology. – 2000. – Vol. 278. – No. 6. – P. 2039-2049.

47. Антипов И.Е., Захаров А. В., Повереннова О. И., Неганов В. А., Ерофеев А. Е. Возможности различных методов автоматического распознавания стадий сна // Саратовский научно-медицинский журнал. – 2012. – том 8. – № 2. – С. 374–379.

48. Мун Ф. Хаотические колебания: Вводный курс для научных работников и инженеров: Пер. с англ. – М.: Мир, 1990. — 312 с.

49. Майоров О.Ю., Фенченко В.Н. Повышение надежности исследований детерминированного хаоса в биоэлектрической активности (ЭЭГ, ЭКГ и вариабельности сердечного ритма) методами нелинейного анализа // Клиническая информатика и телемедицина. – 2009. – том 6. – № 6. – С. 10-17.

50. Киселева О.Г., Настенко Е.А., Герасимчук М.В. Метод оценки дизадаптационных состояний организма человека // Восточно-Европейский журнал передовых технологий. – 2011. – том 3. – № 2. – С. 57-64.

51. Зиненко А.В. R/S анализ на фондовом рынке // Бизнес-информатика. – 2012. – №3. – С. 24-30.

52. Ваховський І.В. Програмна система для оцінки хаотичності форми фрагментів одноканальної електрокардіограми [Текст]: дип. роб. бакалавра : 25.00.01 : захищена 30.06.16 : затв. 02.07.16 / Ваховський Іван Володимирович. — К., 2016. — 71 с.

53. Матушевич Н.А. Розробка методу аналізу діагностичних ознак за зашумленою електрокардіограмою та його програмна реалізація [Текст]: дип. роб. бакалавра : 25.00.01 : захищена 30.06.16 : затв. 02.07.16 / Матушевич Наталія Анатоліївна. — К., 2016. — 83 с.

Додаток А

Вихідний код програми.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using System.Collections;
using System.Dynamic;
using System.Windows.Forms.DataVisualization.Charting;
using System.Runtime.InteropServices;
using System.Drawing.Drawing2D;
using System.Globalization;
using System.Threading;


namespace Entropia
{


    public struct PointD
    {
        public double X;
        public double Y;

        public PointD(double x, double y)
        {
            X = x;
            Y = y;
        }

        public Point ToPoint()
        {
            return new Point((int)X, (int)Y);
        }

        public override bool Equals(object obj)
        {
            return obj is PointD && this == (PointD)obj;
        }
        public override int GetHashCode()
        {
            return X.GetHashCode() ^ Y.GetHashCode();
        }
        public static bool operator ==(PointD a, PointD b)
        {
            return a.X == b.X && a.Y == b.Y;
        }
        public static bool operator !=(PointD a, PointD b)
        {
            return !(a == b);
        }

        public bool Eq(PointD b)
        {
            if ((this.X - b.X) < 0.0000001 && (this.Y - b.Y) < 0.0000001)
                return true;
```

```
            else
                return false;
        }
    }



    public partial class Form1 : Form
    {
        IniFile INI = new IniFile("config.ini");

        public static String FileName = "";
        public static ArrayList Value = new ArrayList();
        public static ArrayList Value2 = new ArrayList();
        //   public static List<double> list = new List<double>();
        public static Boolean flag = false;
        public static double UpBound = 0, BotBound = 0;
        public static int iterNumber = 0;
        public static double sigmNumber = 0;
        public static bool checkValue = true;
        public static double[] sho;
        public static Color color;
        public static double[] pohidnaS;
        public static double[] pohidnaArray;
        public static double[] pohidnaSave;
        public static double[] clearSignal;
        public static bool ifcalc = false;
        public static bool ifDamaged = false;
        public static double masshXmin;
        public static double masshYmin;
        public static double masshXmax;
        public static double masshYmax;
        public static double shagX;
        public static double startValueX;
        public static double shagY;
        public static double startValueY;

        public static double AxisYMinimum;
        public static double AxisYMaximum;
        public static double AxisXMinimum;
        public static double AxisXMaximum;

        public static ArrayList val = null;

        public static double[] masYY;


        public static bool ifPerevod = true;
        public static bool ifClear = true;

        protected override void OnFormClosing(FormClosingEventArgs e)
        {
            DialogResult dialogResult = MessageBox.Show("Сохранить настройки
параметров?", "Выход", MessageBoxButtons.YesNo);
            if (dialogResult == DialogResult.Yes)
            {
                auto_write();
            }
            else if (dialogResult == DialogResult.No)
            {
                if (System.IO.File.Exists("configOld.ini"))
                    File.Copy("configOld.ini", "config.ini", true);
            }
```

```
        public Form1()
        {

            InitializeComponent();



            this.mainChart.ChartAreas[0].AxisX.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
            this.mainChart.ChartAreas[0].AxisY.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
            mainChart.BorderlineDashStyle = ChartDashStyle.Solid;
            chartClear.BorderlineDashStyle = ChartDashStyle.Solid;


            toolTip1.SetToolTip(btnSaveSE, "Сохранить сигнал скорости изменения
энтропии в Файл");
            toolTip1.SetToolTip(btnSaveE, "Сохранить сигнал энтропии в Файл");
            toolTip1.SetToolTip(btnSaveS, "Сохранить результат очистки в Файл");
            this.checkIter0.CheckedChanged += new
EventHandler(radioButtons_CheckedChanged);
            this.checkIter1.CheckedChanged += new
EventHandler(radioButtons_CheckedChanged);
            this.checkIter2.CheckedChanged += new
EventHandler(radioButtons_CheckedChanged);
            this.boxSigm.TextChanged += boxSigm_TextChanged;



            this.mainChart.MouseWheel += mainChart_MouseWheel;
            this.mainChart.MouseLeave += mainChart_MouseLeave;
            this.mainChart.MouseEnter += mainChart_MouseEnter;

            this.chartClear.MouseWheel += chartClear_MouseWheel;
            this.chartClear.MouseEnter += chartClear_MouseEnter;
            this.chartClear.MouseLeave += chartClear_MouseLeave;

            this.chartEntr1.MouseWheel += chartEntr1_MouseWheel;
            this.chartEntr1.MouseEnter += chartEntr1_MouseEnter;
            this.chartEntr1.MouseLeave += chartEntr1_MouseLeave;

            this.chartEntr2.MouseWheel += chartEntr2_MouseWheel;
            this.chartEntr2.MouseEnter += chartEntr2_MouseEnter;
            this.chartEntr2.MouseLeave += chartEntr2_MouseLeave;


            this.checkIter0.CheckedChanged += checkIter0_CheckedChanged;
            checkIter0.Checked = true;
            boxSigm.SelectedIndex = 0;

            if (checkIter0.Checked)
            {

                boxSigm.Enabled = false;

            }
            else
            {

                boxSigm.Enabled = true;
            }
```

```
            auto_read();

            if (System.IO.File.Exists("config.ini"))
                File.Copy("config.ini", "configOld.ini", true);

            ifcalc = true;

            if (FileName != "")
                calculate();
            else
            {

                double[] tm = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
                double[] tm1 = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
                mainChart.ChartAreas[0].AxisY.Interval = 100;
                mainChart.ChartAreas[0].AxisY.IntervalAutoMode =
IntervalAutoMode.FixedCount;
                this.mainChart.ChartAreas[0].AxisX.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
                this.mainChart.ChartAreas[0].AxisY.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
                chartClear.ChartAreas[0].AxisY.Interval = 100;
                chartClear.ChartAreas[0].AxisY.IntervalAutoMode =
IntervalAutoMode.FixedCount;
                this.chartClear.ChartAreas[0].AxisX.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
                this.chartClear.ChartAreas[0].AxisY.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
                // chartEntr1.ChartAreas[0].AxisY.Interval = 100;
                chartEntr1.ChartAreas[0].AxisY.IntervalAutoMode =
IntervalAutoMode.FixedCount;
                this.chartEntr1.ChartAreas[0].AxisX.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
                this.chartEntr1.ChartAreas[0].AxisY.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
                // chartEntr2.ChartAreas[0].AxisY.Interval = 100;
                chartEntr2.ChartAreas[0].AxisY.IntervalAutoMode =
IntervalAutoMode.FixedCount;
                this.chartEntr2.ChartAreas[0].AxisX.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
                this.chartEntr2.ChartAreas[0].AxisY.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;


                mainChart.Series["ser1"].Points.DataBindXY(tm1, tm);
                chartEntr1.Series["Series1"].Points.DataBindXY(tm1, tm);
                chartEntr2.Series["Series1"].Points.DataBindXY(tm1, tm);
                chartClear.Series["ser1"].Points.DataBindXY(tm1, tm);

            }

            labStatus.Text = FileName;
            labStatus1.Text = FileName;
            labStatus2.Text = FileName;

        }

        void chartEntr2_MouseLeave(object sender, EventArgs e)
        {
            if (chartEntr2.Focused)
                chartEntr2.Parent.Focus();

        }
```

```
        void chartEntr2_MouseEnter(object sender, EventArgs e)
        {
            if (!chartEntr2.Focused)
                chartEntr2.Focus();

        }

        void chartEntr2_MouseWheel(object sender, MouseEventArgs e)
        {
            try
            {
                if (e.Delta < 0)
                {
                    chartEntr2.ChartAreas[0].AxisX.ScaleView.ZoomReset();
                    chartEntr2.ChartAreas[0].AxisY.ScaleView.ZoomReset();
                }

                if (e.Delta > 0)
                {

                    double xMin =
Math.Round(chartEntr2.ChartAreas[0].AxisX.ScaleView.ViewMinimum, 2);
                    double xMax =
Math.Round(chartEntr2.ChartAreas[0].AxisX.ScaleView.ViewMaximum, 2);
                    // double yMin = Math.Round(
chartEntr1.ChartAreas[0].AxisY.ScaleView.ViewMinimum,2);
                    // double yMax =
Math.Round(chartEntr1.ChartAreas[0].AxisY.ScaleView.ViewMaximum, 2);

                    double posXStart =
Math.Round(chartEntr2.ChartAreas[0].AxisX.PixelPositionToValue(e.Location.X) - (xMax -
xMin) / 2, 2);
                    double posXFinish =
Math.Round(chartEntr2.ChartAreas[0].AxisX.PixelPositionToValue(e.Location.X) + (xMax -
xMin) / 2, 2);
                    // double posYStart =
Math.Round(chartEntr1.ChartAreas[0].AxisY.PixelPositionToValue(e.Location.Y) - (yMax -
yMin) / 2,2);
                    //double posYFinish =
Math.Round(chartEntr1.ChartAreas[0].AxisY.PixelPositionToValue(e.Location.Y) + (yMax -
yMin) / 2, 2);




                    chartEntr2.ChartAreas[0].AxisX.ScaleView.Zoom(posXStart,
posXFinish);
                    // chartEntr1.ChartAreas[0].AxisY.ScaleView.Zoom(posYStart,
posYFinish);

                    // chartEntr1.ChartAreas[0].AxisY.Interval = 0.05;

                    // chartEntr1.ChartAreas[0].AxisY.IntervalAutoMode =
IntervalAutoMode.VariableCount;

                }
            }
            catch { }
        }

        void chartEntr1_MouseLeave(object sender, EventArgs e)
        {
            if (chartEntr1.Focused)
                chartEntr1.Parent.Focus();
```

```
        }

        void chartEntr1_MouseEnter(object sender, EventArgs e)
        {
            if (!chartEntr1.Focused)
                chartEntr1.Focus();

        }

        void chartEntr1_MouseWheel(object sender, MouseEventArgs e)
        {
            try
            {
                if (e.Delta < 0)
                {
                    chartEntr1.ChartAreas[0].AxisX.ScaleView.ZoomReset();
                    chartEntr1.ChartAreas[0].AxisY.ScaleView.ZoomReset();
                }

                if (e.Delta > 0)
                {

                    double xMin =
Math.Round(chartEntr1.ChartAreas[0].AxisX.ScaleView.ViewMinimum, 2);
                    double xMax =
Math.Round(chartEntr1.ChartAreas[0].AxisX.ScaleView.ViewMaximum, 2);

                    double posXStart =
Math.Round(chartEntr1.ChartAreas[0].AxisX.PixelPositionToValue(e.Location.X) - (xMax -
xMin) / 2, 2);
                    double posXFinish =
Math.Round(chartEntr1.ChartAreas[0].AxisX.PixelPositionToValue(e.Location.X) + (xMax -
xMin) / 2, 2);
                    // double posYStart =
Math.Round(chartEntr1.ChartAreas[0].AxisY.PixelPositionToValue(e.Location.Y) - (yMax -
yMin) / 2,2);
                    //double posYFinish =
Math.Round(chartEntr1.ChartAreas[0].AxisY.PixelPositionToValue(e.Location.Y) + (yMax -
yMin) / 2, 2);




                    chartEntr1.ChartAreas[0].AxisX.ScaleView.Zoom(posXStart,
posXFinish);
                    // chartEntr1.ChartAreas[0].AxisY.ScaleView.Zoom(posYStart,
posYFinish);

                    // chartEntr1.ChartAreas[0].AxisY.Interval = 0.05;

                    // chartEntr1.ChartAreas[0].AxisY.IntervalAutoMode =
IntervalAutoMode.VariableCount;

                }
            }
            catch { }

        }

        void chartClear_MouseLeave(object sender, EventArgs e)
        {
            if (chartClear.Focused)
                chartClear.Parent.Focus();
```

```
        }

        void chartClear_MouseEnter(object sender, EventArgs e)
        {
            //throw new NotImplementedException();
            if (!chartClear.Focused)
                chartClear.Focus();

        }



        void mainChart_MouseEnter(object sender, EventArgs e)
        {
            if (!mainChart.Focused)
                mainChart.Focus();
        }

        void mainChart_MouseLeave(object sender, EventArgs e)
        {
            if (mainChart.Focused)
                mainChart.Parent.Focus();
        }

        private void mainChart_MouseWheel(object sender, MouseEventArgs e)
        {
            try
            {
                if (e.Delta < 0)
                {
                    mainChart.ChartAreas[0].AxisX.ScaleView.ZoomReset();
                    mainChart.ChartAreas[0].AxisY.ScaleView.ZoomReset();
                }

                if (e.Delta > 0)
                {
                    double xMin =
(int)mainChart.ChartAreas[0].AxisX.ScaleView.ViewMinimum;
                    double xMax =
(int)mainChart.ChartAreas[0].AxisX.ScaleView.ViewMaximum;
                    double yMin =
(int)mainChart.ChartAreas[0].AxisY.ScaleView.ViewMinimum;
                    double yMax =
(int)mainChart.ChartAreas[0].AxisY.ScaleView.ViewMaximum;

                    double posXStart =
(int)mainChart.ChartAreas[0].AxisX.PixelPositionToValue(e.Location.X) - (xMax - xMin)
/ 2;
                    double posXFinish =
(int)mainChart.ChartAreas[0].AxisX.PixelPositionToValue(e.Location.X) + (xMax - xMin)
/ 2;
                    double posYStart =
(int)mainChart.ChartAreas[0].AxisY.PixelPositionToValue(e.Location.Y) - (yMax - yMin)
/ 2;
                    double posYFinish =
(int)mainChart.ChartAreas[0].AxisY.PixelPositionToValue(e.Location.Y) + (yMax - yMin)
/ 2;

                    mainChart.ChartAreas[0].AxisX.ScaleView.Zoom(posXStart,
posXFinish);
                    mainChart.ChartAreas[0].AxisY.ScaleView.Zoom(posYStart,
posYFinish);
```

```
                    mainChart.ChartAreas[0].AxisY.Interval = Math.Round((maxi(Value,
true) - mini(Value, true)) / 7);

                    mainChart.ChartAreas[0].AxisY.IntervalAutoMode =
IntervalAutoMode.FixedCount;

                }
            }
            catch { }
        }

        public static double mini(ArrayList mas, bool check)
        {
            string[] array;
            double[] mas1;
            if (check)
            {
                array = mas.ToArray(typeof(string)) as string[];
                mas1 = new double[mas.Count];
                for (int i = 0; i < mas1.Length; i++)
                {
                    mas1[i] = double.Parse(array[i],
System.Globalization.CultureInfo.InvariantCulture);
                }
            }
            else
            {
                mas1 = mas.ToArray(typeof(double)) as double[];
            }


            double result = mas1[0];

            for (int i = 1; i < mas1.Length; i++)
            {
                if (mas1[i] < result)
                    result = mas1[i];
            }
            return result;
        }


        public static double maxi(ArrayList mas, bool check)
        {
            string[] array;
            double[] mas1;
            if (check)
            {
                array = mas.ToArray(typeof(string)) as string[];
                mas1 = new double[mas.Count];
                for (int i = 0; i < mas1.Length; i++)
                {
                    mas1[i] = double.Parse(array[i],
System.Globalization.CultureInfo.InvariantCulture);
                }
            }
            else
            {
                mas1 = mas.ToArray(typeof(double)) as double[];
            }

            double result = mas1[0];

            for (int i = 1; i < mas1.Length; i++)
```

```
    {
        if (mas1[i] > result)
            result = mas1[i];
    }
    return result;
}

private void button2_Click_1(object sender, EventArgs e)
{
    OpenFileDialog ofd = new OpenFileDialog();
    ofd.Filter = "Txt File|*.dtt|Text File|*.txt|All files (*.*)|*.*";
    if (ofd.ShowDialog() == System.Windows.Forms.DialogResult.OK)
    {
        // txtPath.Text += ofd.FileName;
        flag = true;
        FileName = ofd.FileName;
    }
}


private void выбратьФайлToolStripMenuItem_Click(object sender, EventArgs e)
{
    ifPerevod = true;
    fileChooser();
    //if (txtPorog.Text == "" || txtOkno.Text == "" || txtSglaj.Text == "")
    if (txtPorog.Text == "")
    {
        if (FileName.EndsWith("rr.dtt"))
            txtPorog.Text = "50";
        else
            txtPorog.Text = "0.5";
    }

    if (txtOkno.Text == "")
    {
        txtOkno.Text = "50";
    }

    if (txtSglaj.Text == "")
    {
        txtSglaj.Text = "11";
    }

    if (txtBoxApEn.Text == "")
    {
        txtBoxApEn.Text = "2";
    }

    calculate();
    btnSaveS.Visible = true;

}

public void fileChooser()
{
    OpenFileDialog ofd = new OpenFileDialog();
    ofd.Filter = "Txt File|*.dtt|Text File|*.txt|All files (*.*)|*.*";
    if (ofd.ShowDialog() == System.Windows.Forms.DialogResult.OK)
    {
        //txtPath.Text += ofd.FileName;
        flag = true;
        FileName = ofd.FileName;
        labStatus.Text = FileName;
        labStatus1.Text = FileName;
```

```
                labStatus2.Text = FileName;
            }
        }

        private void calculate()
        {

            if (txtOkno.Text == "" || txtPorog.Text == "" || txtSglaj.Text == "")
            {
                //  MessageBox.Show("Все поля должны быть заполнены");
                return;
            }

            Value.Clear();
            Value2.Clear();
            if (FileName == "")
            {
                // MessageBox.Show("Выберите файл");
                return;
            }
            string[] lines = null;

            try
            {
                lines = System.IO.File.ReadAllLines(@FileName);
            }
            catch (FileNotFoundException ex)
            {
                MessageBox.Show("Выбраный файл не найден. Выберите файл заново!");
                return;
            }


            List<String> str = new List<String>();
            String[] words;
            foreach (string line in lines)
            {

                words = line.Split(default(string[]),
StringSplitOptions.RemoveEmptyEntries);

                for (int i = 0; i < words.Length; i++)
                    Value.Add(words[i]);

            }


            string[] array = Value.ToArray(typeof(string)) as string[];
            double[] bufferArray = new double[Value.Count];
            for (int i = 0; i < Value.Count; i++)
            {
                bufferArray[i] = double.Parse(array[i],
System.Globalization.CultureInfo.InvariantCulture);
            }

            if (!checkIter0.Checked)
            {
                if (checkIter1.Checked)
                    iterNumber = 1;
                else
                    iterNumber = 2;

                switch (boxSigm.Text)
                {
```

```
            case "0.5":
                sigmNumber = 0.5;
                break;
            case "0.25":
                sigmNumber = 0.25;
                break;
            case "0.75":
                sigmNumber = 0.75;
                break;
            case "1.25":
                sigmNumber = 1.25;
                break;
            case "1.75":
                sigmNumber = 1.75;
                break;
            case "2.25":
                sigmNumber = 2.25;
                break;
            case "2.75":
                sigmNumber = 2.75;
                break;
            case "1":
                sigmNumber = 1;
                break;
            case "1.5":
                sigmNumber = 1.5;
                break;
            case "2":
                sigmNumber = 2;
                break;
            case "2.5":
                sigmNumber = 2.5;
                break;
            case "3":
                sigmNumber = 3;
                break;

        }

        UpBound = sred(bufferArray) + sko(bufferArray) * sigmNumber;
        BotBound = sred(bufferArray) - sko(bufferArray) * sigmNumber;

        List<double> buff = new List<double>();

        double[] promArray = medf(bufferArray, sigmNumber);
        double[] promArray2;


        for (int i = 0; i < promArray.Length; i++)
            Value2.Add(promArray[i]);

        if (iterNumber == 2) // 2 itteracii
        {
            promArray2 = medf(promArray, sigmNumber);
            UpBound = sred(promArray) + sko(promArray) * sigmNumber;
            BotBound = sred(promArray) - sko(promArray) * sigmNumber;
            Value2.Clear();
            for (int i = 0; i < promArray2.Length; i++)
                Value2.Add(promArray2[i]);
        }
        checkValue = false;
    }
    else
    {
```

```
                for (int i = 0; i < Value.Count; i++)
                    Value2.Add(Value[i]);

                checkValue = true;
            }

            int[] arrX = new int[Value.Count];
            double[] skoA = new double[Value.Count];
            double[] skoB = new double[Value.Count];

            for (int i = 0; i < arrX.Length; i++)
            {
                arrX[i] = i;
                skoA[i] = UpBound;
                skoB[i] = BotBound;

            }

            int[] arrX2 = new int[Value2.Count];
            double[] skoA2 = new double[Value2.Count];
            double[] skoB2 = new double[Value2.Count];

            for (int i = 0; i < arrX2.Length; i++)
            {
                arrX2[i] = i;
                skoA2[i] = UpBound;
                skoB2[i] = BotBound;

            }

            for (int i = 0; i < Value.Count; i++)
            {

                // MessageBox.Show("i: " + bufferArray[i].ToString());
            }

            this.mainChart.ChartAreas[0].AxisX.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
            this.mainChart.ChartAreas[0].AxisY.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
            mainChart.ChartAreas[0].AxisX.Minimum = 0.0;


            if (sko(bufferArray) < 1)
            {
                mainChart.ChartAreas[0].AxisY.Minimum = Math.Round(mini(Value, true))
- 0.5;
                mainChart.ChartAreas[0].AxisY.Maximum = Math.Round(maxi(Value, true))
+ 0.5;
            }
            else
            {
                // MessageBox.Show(maxi(Value, true).ToString());
                mainChart.ChartAreas[0].AxisY.Minimum = Math.Round(mini(Value, true))
- 100;
                mainChart.ChartAreas[0].AxisY.Maximum = Math.Round(maxi(Value, true))
+ 100;
            }

            mainChart.Series["ser1"].Points.DataBindXY(arrX, Value);
            mainChart.Series["ser2"].Points.DataBindXY(arrX, skoA);
            mainChart.Series["ser2"].Color = Color.Red;
            mainChart.Series["ser3"].Points.DataBindXY(arrX, skoB);
            mainChart.Series["ser3"].Color = Color.Red;
```

```
            mainChart.ChartAreas[0].AxisX.ScaleView.Zoomable = true;

            mainChart.ChartAreas[0].AxisY.Interval = Math.Round((maxi(Value, true) -
mini(Value, true)) / 3);
            chartClear.ChartAreas[0].AxisY.Interval = Math.Round((maxi(Value, true) -
mini(Value, true)) / 3);



            mainChart.ChartAreas[0].AxisY.IntervalAutoMode =
IntervalAutoMode.FixedCount;
            mainChart.ChartAreas[0].AxisY.Title = "RR, мс";
            mainChart.ChartAreas[0].AxisX.Title = "i";

            this.chartClear.ChartAreas[0].AxisX.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
            this.chartClear.ChartAreas[0].AxisY.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;

            chartClear.ChartAreas[0].AxisX.Minimum = 0.0;

            if (sko(bufferArray) < 1)
            {

                chartClear.ChartAreas[0].AxisY.Minimum = Math.Round(mini(Value, true))
- 0.5;
                chartClear.ChartAreas[0].AxisY.Maximum = Math.Round(maxi(Value, true))
+ 0.5;
            }
            else
            {
                chartClear.ChartAreas[0].AxisY.Minimum = Math.Round(mini(Value, true))
- 100;
                chartClear.ChartAreas[0].AxisY.Maximum = Math.Round(maxi(Value, true))
+ 100;
            }
            chartClear.Series["ser1"].Points.DataBindXY(arrX2, Value2);
            chartClear.ChartAreas[0].AxisX.ScaleView.Zoomable = true;
            chartClear.ChartAreas[0].AxisY.Title = "RR, мс";
            chartClear.ChartAreas[0].AxisX.Title = "i";
            chartClear.ChartAreas[0].AxisY.IntervalAutoMode =
IntervalAutoMode.FixedCount;

            mainChart.BorderlineDashStyle = ChartDashStyle.Solid;
            chartClear.BorderlineDashStyle = ChartDashStyle.Solid;

            mainChart.ChartAreas[0].AxisX.TitleAlignment = StringAlignment.Far;
            mainChart.ChartAreas[0].AxisY.TitleAlignment = StringAlignment.Far;
            chartClear.ChartAreas[0].AxisX.TitleAlignment = StringAlignment.Far;
            chartClear.ChartAreas[0].AxisY.TitleAlignment = StringAlignment.Far;



            double[] forsko;

            if (checkIter0.Checked == false)
                forsko = Value2.ToArray(typeof(double)) as double[];
            else
            {
                string[] arrayParse1;

                arrayParse1 = Value2.ToArray(typeof(string)) as string[];
                forsko = new double[Value2.Count];
                for (int i = 0; i < forsko.Length; i++)
```

```
                {
                    forsko[i] = double.Parse(arrayParse1[i],
System.Globalization.CultureInfo.InvariantCulture);
                }
            }


            toolTip1.SetToolTip(label3, "Предлагаемое значение порога: " +
Math.Round((maxArray(forsko) - minArray(forsko)) / 12, 3).ToString());
            toolTip1.SetToolTip(label2, "Предлагаемое значение ширины окна: " +
Math.Round(forsko.Length * 0.1).ToString());

            double porog = 0;

            string sep =
Thread.CurrentThread.CurrentCulture.NumberFormat.NumberDecimalSeparator;


            string porogSep = "";

            if (sep.Equals("."))
            {
                porogSep = txtPorog.Text;
                porogSep = porogSep.Replace(',', '.');
            }
            else
            {
                porogSep = txtPorog.Text;
                porogSep = porogSep.Replace('.', ',');
            }

            if (sko(forsko) * 0.05 < double.Parse(porogSep))
            {

                if (rbAbs.Checked == true)
                {
                    porog = double.Parse(porogSep);
                }
                else
                {
                    porog = double.Parse(porogSep) * sko(forsko) / 100;
                }
            }
            else
            {
                MessageBox.Show("Введен некорректный порог");
                return;
            }

            int okno = int.Parse(txtOkno.Text);

            if (okno > forsko.Length - 50)
            {
                MessageBox.Show("Размер окна превышает допустий порог");
                return;
            }



            if (okno < 4)
            {
                MessageBox.Show("Размер окна меньше допустимого порога");
                return;
            }
```

```
            int oknoSglaj = int.Parse(txtSglaj.Text);

            if (oknoSglaj > forsko.Length - 20)
            {
                MessageBox.Show("Размер окна сглаживания превышает допустий порог");
                return;
            }

                txtPorog.Text = Math.Round((maxArray(forsko) - minArray(forsko)) / 12,
3).ToString();
                txtOkno.Text = Math.Round(forsko.Length * 0.1).ToString();

                porog = Math.Round((maxArray(forsko) - minArray(forsko)) / 12, 3);
                okno = int.Parse(Math.Round(forsko.Length * 0.1).ToString());


                /* txtPorog.Text = 20.101.ToString();
                 txtOkno.Text = 16.ToString();

                 porog = 20.101;
                 okno = 16;*/


            switch (BoxEntr.Text)
            {
                case "Шенноновская энтропия" :
                     sho = shenEntr(Value2, okno, porog);// MASSIV ENTROPII
                    break;

                case "Перестановочная энтропия":
                    sho = funcPerest(Value2, okno, porog);
                    break;

                case "Аппроксимационная энтропия":
                    sho = funcAprox(Value2, okno, porog, int.Parse(txtBoxApEn.Text));
                    break;
            }


            if (sho == null)
            {
                this.chartEntr1.Series["Series1"].Color = Color.Transparent;
                this.chartEntr2.Series["Series1"].Color = Color.Transparent;
                return;
            }
            else
            {
                this.chartEntr1.Series["Series1"].Color = color;
                this.chartEntr2.Series["Series1"].Color = color;
            }


            mainChart.Series["ser1"].Color = color;
            chartClear.Series["ser1"].Color = color;

            int[] entrCounter = new int[sho.Length];
            int[] entrTest = new int[sho.Length];
            for (int i = 0; i < sho.Length; i++)
            {
                entrCounter[i] = i;
                entrTest[i] = 100;
                //  MessageBox.Show("# " + i + ": " + sho[i]);
```

```
}


pohidnaArray = pohidnaEntr(sho); // PROIZVODNAYA ENTROPII


int[] entrCounter2 = new int[pohidnaArray.Length];

for (int i = 0; i < pohidnaArray.Length; i++)
{
    entrCounter2[i] = i;

    // MessageBox.Show("# " + i + ": " + sho[i]);
}



if (pohidnaArray.Length < int.Parse(txtSglaj.Text) * 2 + 1)
{
    MessageBox.Show("Размер окна сглаживания выбран некорректно");
    return;
}

pohidnaS = skolzSglaj(pohidnaArray, int.Parse(txtSglaj.Text));



List<double> pohView = new List<double>();

int bufCount = int.Parse(txtSglaj.Text) + 3;

if (pohidnaS == null)
    return;

for (int i = 0; i < bufCount; i++)
    pohView.Add(0);

for (int i = 0; i < pohidnaS.Length; i++)
{
    pohView.Add(pohidnaS[i]);
    // MessageBox.Show(pohidnaS[i].ToString());
}

for (int i = 0; i < bufCount; i++)
    pohView.Add(0);

double[] pohidnaView = pohView.ToArray()

int[] counterS = new int[pohidnaView.Length];
for (int i = 0; i < counterS.Length; i++)
{
    counterS[i] = i;
}

chartEntr1.ChartAreas[0].AxisX.Minimum = 0.0;
chartEntr2.ChartAreas[0].AxisX.Minimum = 0.0;

chartEntr1.Series["Series1"].Points.DataBindXY(entrCounter, sho);
chartEntr2.Series["Series1"].Points.DataBindXY(counterS, pohidnaView);

chartEntr1.ChartAreas[0].AxisX.Title = "i";
chartEntr1.ChartAreas[0].AxisY.Title = "Hi, %";
chartEntr1.ChartAreas[0].AxisX.TitleAlignment = StringAlignment.Far;
```

```
            chartEntr1.ChartAreas[0].AxisY.TitleAlignment = StringAlignment.Far;

            chartEntr2.ChartAreas[0].AxisX.Title = "i";
            chartEntr2.ChartAreas[0].AxisY.Title = "dH/di, %/c";

            chartEntr2.ChartAreas[0].AxisX.TitleAlignment = StringAlignment.Far;
            chartEntr2.ChartAreas[0].AxisY.TitleAlignment = StringAlignment.Far;


            chartEntr1.BorderlineDashStyle = ChartDashStyle.Solid;
            chartEntr2.BorderlineDashStyle = ChartDashStyle.Solid;

            chartEntr1.ChartAreas[0].AxisX.TitleAlignment = StringAlignment.Far;
            chartEntr1.ChartAreas[0].AxisY.TitleAlignment = StringAlignment.Far;
            chartEntr2.ChartAreas[0].AxisX.TitleAlignment = StringAlignment.Far;
            chartEntr2.ChartAreas[0].AxisY.TitleAlignment = StringAlignment.Far;


            this.chartEntr1.ChartAreas[0].AxisX.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
            this.chartEntr1.ChartAreas[0].AxisY.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
            this.chartEntr2.ChartAreas[0].AxisX.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;
            this.chartEntr2.ChartAreas[0].AxisY.MajorGrid.LineDashStyle =
ChartDashStyle.DashDotDot;


            if (sko(sho) > 0.5) // для того случая, если график вообще регулярный и
все значения одиннаковые
            {
                chartEntr1.ChartAreas[0].AxisY.Minimum = Math.Round((minArray(sho) -
(maxArray(sho) - minArray(sho)) * 0.3));
                chartEntr1.ChartAreas[0].AxisY.Maximum = Math.Round((maxArray(sho) +
(maxArray(sho) - minArray(sho)) * 0.3));
                chartEntr1.ChartAreas[0].AxisY.IntervalAutoMode =
IntervalAutoMode.FixedCount;
                chartEntr1.ChartAreas[0].AxisY.Interval = Math.Round((maxArray(sho) -
(minArray(sho))) / 5);



            }
            else
            {
                chartEntr2.ChartAreas[0].AxisY.Minimum = -1;
                chartEntr2.ChartAreas[0].AxisY.Maximum = 1;
                chartEntr2.ChartAreas[0].AxisY.IntervalAutoMode =
IntervalAutoMode.FixedCount;
                chartEntr2.ChartAreas[0].AxisY.Interval = 0.5;
            }

            double[] normPohidna = norma(pohidnaS, minArray(pohidnaS),
maxArray(pohidnaS));
            double[] normNewsho = norma(sho, minArray(sho), maxArray(sho));



            double[] newsho = new double[normNewsho.Length - 6 - 2 *
int.Parse(txtSglaj.Text)];
            double[] newshoTest = new double[normNewsho.Length - 6 - 2 *
int.Parse(txtSglaj.Text)];
            masYY = new double[normNewsho.Length - 6 - 2 * int.Parse(txtSglaj.Text)];
```

```
            int jj = 0;
            for (int i = 3 + int.Parse(txtSglaj.Text); i < sho.Length - 3 -
int.Parse(txtSglaj.Text); i++)
            {
                // MessageBox.Show("i " + i + "  " + normNewsho[i].ToString());
                newsho[jj] = normNewsho[i];

                masYY[jj] = sho[i];
                jj++;

            }


            // int razm = (int)(maxArray(sho)-minArray(sho));
            double[,] pixeldata = new double[100, 800];

            double[] masX = new double[normPohidna.Length];
            double[] masY = new double[normPohidna.Length];

            for (int i = 0; i < normPohidna.Length; i++)
            {
                masX[i] = normPohidna[i];
                masY[i] = masYY[i];
            }


            if (checkLine.Checked == false)
            {
                this.chartPortret.Series["Series1"].Color = Color.Transparent;
            }
            else
            {
                this.chartPortret.Series["Series1"].Color = color;
            }

            if (checkPoint.Checked == false)
            {
                this.chartPortret.Series["Series2"].Color = Color.Transparent;
            }
            else
            {
                this.chartPortret.Series["Series2"].Color = Color.Red;
            }



            chartPortret.Series["Series1"].Points.DataBindXY(normPohidna, newsho);
            chartPortret.Series["Series2"].Points.DataBindXY(normPohidna, newsho);


            chartPortret.ChartAreas[0].AxisX.Title = "dH/di, %/c";
            chartPortret.ChartAreas[0].AxisY.Title = "H(i), %";
            chartPortret.ChartAreas[0].AxisX.TitleAlignment = StringAlignment.Far;
            chartPortret.ChartAreas[0].AxisY.TitleAlignment = StringAlignment.Far;


            chartPortret.ChartAreas[0].AxisY.Minimum = AxisYMinimum;
            chartPortret.ChartAreas[0].AxisX.Minimum = AxisXMinimum;
            chartPortret.ChartAreas[0].AxisX.Maximum = AxisXMaximum;
            chartPortret.ChartAreas[0].AxisY.Maximum = AxisYMaximum;

            if (chartPortret.ChartAreas[0].AxisX.CustomLabels.Count != 0)
            {
```

```
                chartPortret.ChartAreas[0].AxisX.CustomLabels.RemoveAt(0);
                chartPortret.ChartAreas[0].AxisX.CustomLabels.RemoveAt(0);
                chartPortret.ChartAreas[0].AxisX.CustomLabels.RemoveAt(0);
                chartPortret.ChartAreas[0].AxisX.CustomLabels.RemoveAt(0);
                chartPortret.ChartAreas[0].AxisX.CustomLabels.RemoveAt(0);
                chartPortret.ChartAreas[0].AxisX.CustomLabels.RemoveAt(0);
            }

            chartPortret.ChartAreas[0].AxisX.CustomLabels.Add(startValueX, startValueX
+ shagX * 2, Math.Round(antiNorm(startValueX + shagX, minArray(pohidnaS),
maxArray(pohidnaS)), 2).ToString());
            chartPortret.ChartAreas[0].AxisX.CustomLabels.Add(startValueX + shagX,
startValueX + shagX * 3, Math.Round(antiNorm(startValueX + shagX + shagX,
minArray(pohidnaS), maxArray(pohidnaS)), 2).ToString());
            chartPortret.ChartAreas[0].AxisX.CustomLabels.Add(startValueX + shagX * 2,
startValueX + shagX * 4, Math.Round(antiNorm(startValueX + shagX * 2 + shagX,
minArray(pohidnaS), maxArray(pohidnaS)), 2).ToString());
            chartPortret.ChartAreas[0].AxisX.CustomLabels.Add(startValueX + shagX * 3,
startValueX + shagX * 5, Math.Round(antiNorm(startValueX + shagX * 3 + shagX,
minArray(pohidnaS), maxArray(pohidnaS)), 2).ToString());
            chartPortret.ChartAreas[0].AxisX.CustomLabels.Add(startValueX + shagX * 4,
startValueX + shagX * 6, Math.Round(antiNorm(startValueX + shagX * 4 + shagX,
minArray(pohidnaS), maxArray(pohidnaS)), 2).ToString());
            chartPortret.ChartAreas[0].AxisX.CustomLabels.Add(startValueX + shagX * 5,
startValueX + shagX * 7, Math.Round(antiNorm(startValueX + shagX * 5 + shagX,
minArray(pohidnaS), maxArray(pohidnaS)), 2).ToString());


            //double shagY = 0.26;
            // double startValueY = -0.31;
            if (chartPortret.ChartAreas[0].AxisY.CustomLabels.Count != 0)
            {

                chartPortret.ChartAreas[0].AxisY.CustomLabels.RemoveAt(0);
                chartPortret.ChartAreas[0].AxisY.CustomLabels.RemoveAt(0);
                chartPortret.ChartAreas[0].AxisY.CustomLabels.RemoveAt(0);
                chartPortret.ChartAreas[0].AxisY.CustomLabels.RemoveAt(0);
                chartPortret.ChartAreas[0].AxisY.CustomLabels.RemoveAt(0);
                chartPortret.ChartAreas[0].AxisY.CustomLabels.RemoveAt(0);
            }

            if (ifPerevod)
            {
                chartPortret.ChartAreas[0].AxisY.CustomLabels.Add(startValueY,
startValueY + shagY * 2, Math.Round(antiNorm(startValueY + shagY, minArray(masYY),
maxArray(masYY))).ToString());
                chartPortret.ChartAreas[0].AxisY.CustomLabels.Add(startValueY + shagY,
startValueY + shagY * 3, Math.Round(antiNorm(startValueY + shagY + shagY,
minArray(masYY), maxArray(masYY))).ToString());
                chartPortret.ChartAreas[0].AxisY.CustomLabels.Add(startValueY + shagY
* 2, startValueY + shagY * 4, Math.Round(antiNorm(startValueY + shagY * 2 + shagY,
minArray(masYY), maxArray(masYY))).ToString());
                chartPortret.ChartAreas[0].AxisY.CustomLabels.Add(startValueY + shagY
* 3, startValueY + shagY * 5, Math.Round(antiNorm(startValueY + shagY * 3 + shagY,
minArray(masYY), maxArray(masYY))).ToString());
                chartPortret.ChartAreas[0].AxisY.CustomLabels.Add(startValueY + shagY
* 4, startValueY + shagY * 6, Math.Round(antiNorm(startValueY + shagY * 4 + shagY,
minArray(masYY), maxArray(masYY))).ToString());
                chartPortret.ChartAreas[0].AxisY.CustomLabels.Add(startValueY + shagY
* 5, startValueY + shagY * 7, Math.Round(antiNorm(startValueY + shagY * 5 + shagY,
minArray(masYY), maxArray(masYY))).ToString());
            }
```

```
chartPortret2.ChartAreas[0].AxisY.Minimum = AxisYMinimum;
chartPortret2.ChartAreas[0].AxisX.Minimum = AxisXMinimum;
chartPortret2.ChartAreas[0].AxisX.Maximum = AxisXMaximum;
chartPortret2.ChartAreas[0].AxisY.Maximum = AxisYMaximum;
chartPortret2.ChartAreas[0].AxisX.RoundAxisValues();


chartPortret2.ChartAreas[0].AxisX.Title = "d(H)/d(i), %/c";
chartPortret2.ChartAreas[0].AxisY.Title = "H(i), %";
chartPortret2.ChartAreas[0].AxisX.TitleAlignment = StringAlignment.Far;
chartPortret2.ChartAreas[0].AxisY.TitleAlignment = StringAlignment.Far;
chartPortret2.Series["Series1"].Color = color;
chartPortret2.Series["Series2"].Color = Color.Red;




double sredX = sred(normPohidna);
double sredY = sred(newsho);
boxX.Text = Math.Round(sredX, 3).ToString();
boxY.Text = Math.Round(sredY, 3).ToString();

double[] sredinaX = { sredX };
double[] sredinaY = { sredY };
chartPortret2.Series["Series4"].Points.DataBindXY(sredinaX, sredinaY);
chartPortret2.Series["Series4"].MarkerSize = 10;


//VO(normPohidna, newsho, chartPortret,0);



for (int i = 0; i < normPohidna.Length; i++)
    lb1.Items.Add(normPohidna[i].ToString());

PointD[] data = new PointD[normPohidna.Length];

for (int i = 0; i < normPohidna.Length; i++)
{
    data[i].X = normPohidna[i];
    data[i].Y = newsho[i];
}

if (data.Length < 3)
{
    MessageBox.Show("Введены некорректные параметры");
    return;
}


double[] VOx = new double[a.Count];
double[] VOy = new double[a.Count];




for (int i = 0; i < a.Count; i++)
{
    VOx[i] = data[a[i]].X;
    VOy[i] = data[a[i]].Y;
}
```

```
            chartPortret2.Series["Series3"].Points.DataBindXY(VOx, VOy);
            // chartPortret2.Series["Series3"].ChartType = SeriesChartType.Line;
            chartPortret2.Series["Series3"].Color = Color.Black;



            //MessageBox.Show(square(data).ToString());

            PointD[] dataTest = new PointD[a.Count - 1];
            for (int i = 0; i < a.Count - 1; i++)
            {
                dataTest[i].X = VOx[i];
                dataTest[i].Y = VOy[i];
            }


            boxSquare.Text = Math.Round(square(dataTest), 3).ToString();


            labParams.Text = "Порог: " + txtPorog.Text + "    Окно: " + txtOkno.Text +
"    Окно сглаживания: " + txtSglaj.Text;

            if (BoxEntr.SelectedIndex == 0)
                label5.Text = "Фазовый портерт энтропии (шенноновской)";
            else
                label5.Text = "Фазовый портерт энтропии (перестановочной)";


        }


        public double antiNorm(double xNorm, double min, double max)
        {
            return xNorm * (max - min) + min;
        }

        public double square(PointD[] data)
        {
            double s = 0;


            for (int i = 1; i < data.Length - 1; i++)
            {
                s = s + triangleSquare(data[0], data[i], data[i + 1]);
                // MessageBox.Show(triangleSquare(data[0], data[i], data[i +
1]).ToString());
            }

            return Math.Abs(s);
        }

        public double triangleSquare(PointD a, PointD b, PointD c)
        {
            double s = 0;
            // MessageBox.Show(a.X.ToString() + " " + a.Y.ToString() + " " +
b.X.ToString() + " " + a.Y.ToString());
            s = ((a.X - c.X) * (b.Y - c.Y) - (b.X - c.X) * (a.Y - c.Y)) / 2;

            return s;
        }

        public double geron(double a, double b, double c)
```

```
{
    double s = 0;
    double p = (a + b + c) / 2;
    s = Math.Sqrt(p * (p - a) * (p - b) * (p - c));
    return s;
}

public List<int> ConvexHullJarvis(PointD[] mas)
{
    List<int> convex = new List<int>();
    int _base = 0;
    for (int i = 1; i < mas.Length; i++)
    {
        if (mas[i].Y < mas[_base].Y)
            _base = i;
        else
            if (mas[i].Y.Equals(mas[_base].Y) && mas[i].X < mas[_base].X)
                _base = i;
    }
    // эта точка точно входит в выпуклую оболочку
    convex.Add(_base);



    PointD first = mas[_base];
    PointD cur = first;
    PointD prev = new PointD(first.X - 1, first.Y);
    do
    {
        double minCosAngle = 1e9; // чем больше угол, тем меньше его косинус
        double maxLen = 1e9;
        int next = -1;


        for (int i = 0; i < mas.Length; i++)
        {
            double curCosAngle = CosAngle(prev, cur, mas[i]);

            if (curCosAngle < minCosAngle)
            {
                next = i;
                minCosAngle = curCosAngle;
                maxLen = dist(cur, mas[i]);
            }
            else if (EqualsD(curCosAngle, minCosAngle))
            {
                double curLen = dist(cur, mas[i]);
                if (curLen > maxLen)
                {
                    next = i;
                    maxLen = curLen;
                }
            }
        }
        prev = cur;
        //  MessageBox.Show(mas.Length.ToString());
        if (next == -1)
        {
            next = 0;

            convex.Add(next);
            // convex.Add(next);
            //convex.Add(next);
```

```
            break;

        }

        cur = mas[next];
        convex.Add(next);

    }
    // while (cur!=first);
    while (!cur.Eq(first));

    return convex;
}
```