Міністерство освіти і науки України

Харківський національний університет імені В.Н. Каразіна

Факультет комп'ютерних наук

Спеціальність 125 «Кібербезпека»

Освітня програма «Безпека інформаційних та комунікаційних систем»

«Допущено до захисту»

Зав.кафедрою БІСТ

Сергій РАССОМАХІН

_____

« » 2022 р.

Пояснювальна записка

до кваліфікаційної роботи магістра

на тему: «Математична модель та програмна реалізація стеганографічного приховування даних в аудіофайлах із розширенням спектру методом прямої послідовності»

оцінка « »

Голова ЕК

Доценко С.І. _____

Керівник д.т.н., проф. Кузнецов О. О.

Рецензент д.т.н.,с.н.с. Толстолузька О.Г.

Виконавець : студент(ка) групи КБ-61

Оникійчук О.О.

Харків – 2022

РЕФЕРАТ

Пояснювальна записка містить 78 сторінок, 74 рисунки, 1 таблицю, 2 додатки та 35 джерел посилань.

Об’єктом дослідження є методи приховування інформації в аудіоконтейнерах.

Предметом дослідження є техніки застосування прямого розширення спектру в комбінації з різними способами формування розширюючих послідовностей.

Метою роботи є дослідити такий спосіб приховування повідомлення в аудіоконтейнерах, як метод прямого розширення спектру, у комбінації з різними типами формування розширюючих послідовностей (чіп-кодів) та порівняти їх вплив на відсоток помилок у вбудованому повідомленні, кількість викривлень аудіоконтейнера, а також оцінити якість сприйняття стислих аудіофайлів за допомогою методу перцептивної оцінки якості звуку.

В роботі виконуються експериментальні дослідження над аудіофайлами, з використанням технології прямого розширення спектру, та проводиться оцінка результатів за такими критеріями, як інтенсивність бітових помилок в декодованих повідомленнях – BER (Bit Error Rate), спотворення аудіоконтейнерів за середньоквадратичною помилкою – MSE (Mean squared error), відношення сигнал / шум – SNR(Signal-to-noise ratio) і пікове відношення сигнал / шум – PSNR(Peak signal-to-noise ratio). Також проводяться дослідження аудіосигналів методом PEAQ.

Всі ці дослідження дозволяють оцінити залежність викривлень аудіоконтейнера та помилок в декодованому повідомленні від використовуваних чіп-кодів, а також обраної потужності сигналів і кількості вбудованих біт.

Ключові слова: ПРИХОВУВАННЯ ДАНИХ, АУДІОФАЙЛИ, ТЕХНОЛОГІЯ ПРЯМОГО РОЗШИРЕННЯ СПЕКТРА, СТЕГАНОГРАФІЯ, МЕТОДИ ГЕНЕРАЦІЇ СИГНАЛІВ, БЕЗПЕКА.

ABSTRACT

The explanatory note contains 78 pages, 74 figures, 1 table, 2 appendix and 35 sources of references.

The object of research is methods of hiding information in audio containers.

The subject of research is the technique of applying direct spread spectrum in combination with various ways of forming spread sequences.

The purpose of the work is to investigate such a method of hiding a message in audio containers, as a method of direct spread spectrum, in combination with different types of formation of spread sequences (chip-codes) and to compare their effect on the percentage of errors in the embedded message, the number of distortions of the audio container, and also to evaluate the quality of perception using method of perceptual assessment of sound quality.

In the work, experimental research is carried out on audio files, using the technology of direct spectrum expansion, and the results are evaluated according to such criteria as the intensity of bit errors in decoded messages - BER (Bit Error Rate), distortion of audio containers according to the mean squared error - MSE, signal/noise ratio – SNR and peak signal/noise ratio – PSNR. Audio signal studies are also conducted using the PEAQ method.

All these studies make it possible to evaluate the dependence of distortions of the audio container and errors in the decoded message on the chip codes used, as well as on the selected signal power and the number of embedded bits.

Keywords: DATA HIDING, AUDIO FILES, DIRECT SPREAD SPECTRUM TECHNOLOGY, STEGANOGRAPHY, SIGNAL GENERATION METHODS, SECURITY.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ СКОРОЧЕНЬ І ТЕРМІНІВ 6
ВСТУП..... 7
1 АНАЛІЗ ТА ДОСЛІДЖЕННЯ СУЧАСНИХ ТЕХНОЛОГІЙ РОЗШИРЕННЯ СПЕКТРУ, ФОРМАТІВ АУДІОФАЙЛІВ ТА АУДІОКОДЕКІВ 10
1.1 Технологія frequency-hopping spread spectrum, FHSS 10
1.2 Технологія Chirp Spread Spectrum, CSS 12
1.3 Технологія direct sequence spread spectrum, DSSS 15
1.4 Аналіз та дослідження форматів аудіофайлів та аудіокодеків..... 17
2 МАТЕМАТИЧНА МОДЕЛЬ СТЕГАНОГРАФІЧНОГО ПРИХОВУВАННЯ ДАНИХ В АУДІОФАЙЛАХ ІЗ РОЗШИРЕННЯМ СПЕКТРУ МЕТОДОМ ПРЯМОЇ ПОСЛІДОВНОСТІ 21
2.1 Математична модель розширення спектру методом прямої послідовності в телекомунікаційних системах 21
2.2 Математична модель розширення спектру методом прямої послідовності в стеганографічних системах 26
2.3 Показники та критерії ефективності аудіо-стеганосистем 30
2.4 Математична модель та алгоритми приховуванні та вилучення даних в аудіофайлах із розширенням спектру методом прямої послідовності 31
2.4.1 Спосіб вбудовування повідомлення методом прямої послідовності на основі робот Лізи Марвел 32
2.4.2 Спосіб вбудовування повідомлення методом адресації 33
2.4.3 Методи формування сигналів для приховування повідомлення 35
2.5 Показники ефективності приховування повідомлень 38
2.5.1. Частота бітових помилок, BER..... 39
2.5.2. Середньоквадратична помилка, MSE 39
2.5.3. Відношення сигнал/шум, SNR..... 39
2.5.4. Пікове відношення сигнал/шум, PSNR 40

2.5.5 Метод PEAQ 40
3 ПРОГРАМНА РЕАЛІЗАЦІЯ СТЕГАНОГРАФІЧНОЇ СИСТЕМИ ТА ЕКСПЕРЕМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ЇЇ ЕФЕКТИВНОСТІ 44
3.1 Обґрунтування вибору мови програмування та середовища розробки 44
3.2 Програмна реалізація стеганографічного приховування даних в аудіофайлах із розширенням спектру методом прямої послідовності 44
3.3 Експериментальні дослідження ефективності запропонованої стеганографічної системи 58
3.3.1 Спосіб вбудовування повідомлення методом прямої послідовності на основі робот Лізи Марвел 58
3.3.2 Спосіб вбудовування повідомлення методом адресації 68
3.3.3 Least Significant Bit - Найменш значущий біт 77
3.4 Обґрунтування практичних рекомендацій щодо впровадження розробленої стеганосистеми 77
ВИСНОВКИ 79
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ 81
ДОДАТОК А 85

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ СКОРОЧЕНЬ І ТЕРМІНІВ

WAV – Waveform Audio File Format
LSB – Least Significant Bit
BER – Bit Error Rate
MSE – Mean Squared Error
SNR – Signal-to-noise Ratio
PSNR – Peak Signal-to-noise Ratio
PEAQ – Perceptual Evaluation of Audio Quality
CHIRP – Compressed High-Intensity Radiated Pulse
FHSS – Frequency Hopping Spread Spectrum
DSSS – Direct Sequence Spread Spectrum
CSS – Chirp Spread Spectrum
CSMA – Carrier Sense Multiple Access
RTLS – Real-time Locating Systems
PN – Pseudo-noise Sequence
SS –Spread Spectrum
PCM – Pulse Code Modulation
FLAC – Free Lossless Audio Codec
AIFF – Audio Interchange File Format
ODG – Objective difference grade
ADB – Average Distorted Block
NMR – Noise-to-Masked-Ratio

ВСТУП

Для приховування інформації використовують різні прийоми [1-5]. Найбільш поширеними є методи криптографії [6-8]. І тут осмислені повідомлення перетворюються на безглузді, шумоподібні форми, тобто стають марними даними для неавторизованого користувача (що не знає секретний криптографічний ключ). Таким чином, криптографія приховує зміст даних, тоді як саме існування повідомлень не приховується. Інший підхід полягає у використанні стеганографічних методів [6]. У цьому випадку інформаційні повідомлення приховані всередині інших даних (названих також контейнерами, обкладинками, носіями тощо). Носії можуть бути представлені, наприклад, у вигляді різних мультимедійних файлів (зображень, відео, аудіо, текстів тощо). Існує низка програм з мультимедійними даними, тому передача відео- або аудіофайлів в Інтернеті не викликає жодних підозр. При цьому тільки той, хто має секретний стеганографічний ключ, знає, що в цих оболонках приховано інформаційне повідомлення. Він схожий на валізу з подвійним дном; друге дно надійно замасковане, і виявити приховане повідомлення без секретного ключа складно. Отже, стеганографія приховує факт існування повідомлень.

У цій дипломній роботі обговорюються методи приховування стеганографічної інформації, в яких аудіофайли виступають як мультимедійні контейнери [3]. Приховування даних аудіо має великі перспективи. По-перше, такі дані мають велику популярність в Інтернеті. Наприклад, як обкладинки можна використовувати аудіоповідомлення в месенджерах та соціальних мережах. При цьому обсяг прихованих даних може бути досить великим. Крім того, існує проблема із захистом авторських прав на аудіоконтент в Інтернеті. Приховування цифрових водяних знаків (з позначкою авторського права) в аудіоконтейнерах може бути ефективним рішенням. Таким чином приховування даних в аудіо дійсно дуже актуальне і може мати комерційну реалізацію.

У експериментах, наведених далі, були використані найпростіші звукові сигнали, записані у цифровому форматі WAV[9-10]. Однак, всі результати та пропозиції можна інтерпретувати для більш загального випадку з іншими форматами аудіоконтейнерів.

Для приховання інформації у цій роботі використовується спеціальна технологія, що традиційно застосовується в системах радіозв'язку[11-14]. Це пряме розширення спектра, що реалізує широкосмугову модуляцію псевдовипадковими послідовностями (чіп-кодами) [15-16]. Кореляційні властивості чіп-кодів гарантують відсутність взаємних перешкод каналу зв'язку [17]. В результаті система радіозв'язку набуває численних корисних властивостей: перешкодозахищеність, високу пропускну здатність і абонентську ємність, відсутність взаємних перешкод, екологічність зв'язку та багато інших.

Технологія прямого розширення спектра реалізована в багатьох галузевих стандартах та додатках, наприклад:

- Системи зв'язку множинного доступу з кодовим поділом каналів (IS-95, CDMA2000, WCDMA, DS-CDMA, TD-CDMA, TD-SCDMA та ін.);
- Глобальні супутникові навігаційні системи (американська GPS, європейська Galileo);
- Сімейство протоколів бездротової мережі Wi-Fi (сімейство стандартів IEEE 802.11) та багато іншого.

Технологія прямого розширення спектра також може бути використана у стеганографії [17-19]. Зокрема, в одній із перших робіт у цій галузі було запропоновано інтерпретувати несучу як шум у каналі зв'язку. В цьому випадку завдання приховування інформації в мультимедійних файлах еквівалентна передачі корисних сигналів каналами зв'язку з природним адитивним шумом.

В даний час вже проведено велику кількість досліджень з стеганографії та прямого розширення спектру [20]. Різні дослідники вивчали це питання стосовно різних комп'ютерних додатків. Виходить, що основні характеристики надійності та безпеки залежать від властивостей послідовностей, що розповсюджують. Зокрема,

були досліджені різні способи генерації кодів чіпів щодо зображень обкладинок. У цій роботі було продовжено ці дослідження стосовно іншого типу мультимедійних даних – аудіоносіїв.

У роботі були вивчені основні характеристики аудіо стегосистеми для різних послідовностей, що розширюють. У експериментах, наведених далі, була оцінена частоту помилок за бітами (BER) прихованих інформаційних даних. Це найважливіший показник якості відновлених повідомлень. Крім того, оцінювалися спотворення звукового контейнера, що характеризують якість мультимедійних даних після приховування інформаційних повідомлень. Тому були використані середньоквадратична помилка (MSE), відношення сигнал/шум (SNR) та пікове відношення сигнал/шум (PSNR). У цій дипломній роботі було продемонстровано, що різні способи створення чіп-кодів дають різні результати для BER, MSE, SNR і PSNR. Це стає областю можливої оптимізації, оскільки вибір методу формування послідовностей, що розширюють, дуже важливий для різних комп'ютерних додатків.

Останнім етапом роботи було дослідження якості сприйняття стислих аудіофайлів за допомогою методу перцептивної оцінки якості звуку (PEAQ), який напряму модулює поведінку людини та її сприйняття спотворень в аудіофайлах.

Таким чином метою роботи є дослідити такий спосіб приховування повідомлення в аудіоконтейнерах, як метод прямого розширення спектру, у комбінації з різними типами формування розширюючих послідовностей (сигналів) та порівняти їх вплив на відсоток помилок у вбудованому повідомленні, кількість викривлень аудіоконтейнера, а також оцінити якість сприйняття стислих аудіофайлів за допомогою методу перцептивної оцінки якості звуку.

1 АНАЛІЗ ТА ДОСЛІДЖЕННЯ СУЧАСНИХ ТЕХНОЛОГІЙ РОЗШИРЕННЯ СПЕКТРУ, ФОРМАТІВ АУДІОФАЙЛІВ ТА АУДІОКОДЕКІВ

Існує багато способів для покращення ефективності передачі інформації. Одним із таких способів є технологія розширення спектра, яка використовує модульовані сигнали через канал з лінійними згасаннями. Цей процес призводить до збільшення бази сигналу.

Існує три основні методи використання технології розширення спектра:

- FHSS – Frequency Hopping Spread Spectrum, або псевдовипадкове перестроювання робочої частоти
- DSSS – Direct Sequence Spread Spectrum, або розширення спектра методом прямої послідовності
- CSS – Chirp Spread Spectrum, або розширення спектра методом лінійної частотної модуляції.

1.1 Технологія frequency-hopping spread spectrum, FHSS

Технологія Frequency Hopping Spread Spectrum (FHSS) - це багаторазове перемикання несучої частоти під час радіопередачі для зменшення перешкод та запобігання перехопленню[21].

FHSS корисний для протидії підслуховуванню, а також для запобігання глушенню частот телекомунікацій та для забезпечення зв'язку з множинним доступом з кодовим поділом каналів. Це також може звести до мінімуму наслідки ненавмисного втручання.

У FHSS передавач перемикається між доступними вузькосмуговими частотами у вказаному широкому каналі в псевдовипадковій послідовності, відомої як відправнику, так і одержувачу.

Короткий пакет даних передається поточним вузькосмуговим каналом, а потім передавач і приймач налаштовуються на наступну частоту в послідовності

для наступного пакета даних. У більшості систем передавач перемикатиметься на нову частоту більше двох разів на секунду.

Оскільки жоден канал не використовується протягом тривалого часу, а ймовірність того, що будь-який інший передавач одночасно знаходиться на тому ж каналі, невелика, FHSS часто використовується як метод, що дозволяє кільком парам передавача і приймача працювати в одному і тому ж просторі на одній і тій же широкій смузі частот каналів одночасно.

Головною перевагою методу FHSS, є те, що оскільки смуга частот може бути розділена на піддіапазони, пристрої можуть швидко змінювати свої несучі частоти або стрибкоподібно змінювати частоти з мінімальними перешкодами.

Ця широкосмугова стратегія дає кілька переваг у порівнянні з фіксованою аналоговою передачею:

- Сигнали FHSS мають високу стійкість до вузькосмугових перешкод, оскільки сигнал перемикається в іншу смугу частот.
- Сигнали важко перехопити, якщо схема стрибкоподібної перебудови частоти невідома.
- Заглушити також складно, якщо шаблон невідомий; сигнал може бути заглушений тільки на один період стрибкоподібної перебудови, якщо послідовність розширення невідома.
- Передачі FHSS можуть спільно використовувати смугу частот з багатьма типами звичайних передач із мінімальними взаємними перешкодами. Сигнали FHSS вносять мінімальні перешкоди у вузькосмуговий зв'язок і навпаки.

З цих причин військові радіостанції зазвичай використовують сигнали з розширеним спектром для поширення закодованої мови, тому що вони повинні бути непомітними та стійкими до методів перешкод. Щоб порушити сигнал FHSS, обладнання противника повинне мати уявлення про схему стрибкоподібної перебудови частоти.

Уряди регулюють спектр мовлення та часто диктують деякі аспекти зв'язку з розширеним спектром.

Наприклад, у Північній Америці промисловий, науковий та медичний діапазон хвиль розділений на 75 стрибкоподібних каналів, і пристрої, що їх використовують, не можуть передаватися з потужністю більше 1 Вт по одному каналу.

Ці обмеження гарантують, що один пристрій не споживає занадто багато смуги пропускання та не затримується надто довго на одній частоті.

У 2000-х роках Федеральна комісія зі зв'язку США дозволила системам FHSS працювати в нерегульованому діапазоні 2,4 ГГц, щоб підтримувати використання FHSS у розгортанні Wi-Fi 802.11b, 802.11g і 802.11n.

Звід федеральних правил Федеральної комісія зі зв'язку США 47, частина 15.247, також містить рекомендації щодо стрибкоподібної перебудови частоти для кількох частотних діапазонів мегагерц.

1.2 Технологія Chirp Spread Spectrum, CSS

Chirp Spread Spectrum (CSS) — це радіочастотна технологія дальньої дії для бездротового зв'язку, яку можна використовувати для виявлення та відстеження розташування людей, майна та пристроїв як усередині, так і зовні на великих об'єктах[22]. Його дальність дії у поєднанні з високою надійністю, високою стійкістю до радіоперешкод та низьким енергоспоживанням роблять чірп унікальним для застосування у великих шумних середовищах, таких як промислові об'єкти. Як і інші протоколи зв'язку, включаючи Bluetooth, Chirp може використовуватися для передачі даних між пристроями за допомогою радіохвиль. Це досягається за допомогою методу широкосмугової модуляції, що створює сигнали з лінійною частотною модуляцією, також відомі як Chirps (Compressed High-Intensity Radiated Pulse, або стислий високо-інтенсивний випромінюваний сигнал).

Chirp був розроблений для роботи в діапазоні ISM 2,45 ГГц і відноситься до тієї ж категорії, що інші технології розширення спектру. Спочатку призначені для військових додатків, щоб допомогти забезпечити безпечний і надійний зв'язок, більш стійкий до виявлення перешкод, методи розширеного спектру поширюють

радіосигнали в ширшому діапазоні частот, створюючи сигнали з ширшою смугою пропускання, зберігаючи вихідну потужність сигналу.

Технологія Chirp збільшує смугу пропускання сигналів до значення, кратного значенням, вказаним у теоремі Шеннона-Хартлі, що допомагає зробити зв'язок більш стійким до перешкод. Реалізовано два типи Chirp-імпульсів - upchirps та downchirps. Для бездротового зв'язку Chirp-імпульсів відправляються від приймача до приймача або між приймачами, які можуть як передавати, так і приймати повідомлення з одним або декількома пристроями одночасно. Приймальні пристрої аналізують шаблони вхідних імпульсів і перетворюють їх на дані.

Хоча це дозволяє пристроям надійно відправляти дані на великі відстані, Chirp також можна використовувати для точного визначення розташування пристроїв. Це дозволяє пристроям з підтримкою Chirp, таким як прив'язки RTLS, точно визначати передавальний пристрій, наприклад тег відстеження активів, знаходити його точне місцезнаходження, а в деяких програмах забезпечувати зв'язок та послуги з урахуванням розташування.

На додаток до того, що RTLS використовується для позиціонування пристроїв у реальному часі, таких як теги відстеження, технологія Chirp також може забезпечувати двосторонню дальність та моніторинг відстані, а також програми бездротового зв'язку. За допомогою цих типів додатків технологія Chirp допомагає створювати рішення з урахуванням розташування, які забезпечують безліч варіантів використання, включаючи відстеження активів, запобігання зіткненням, відстеження транспортних засобів, промислову автоматизацію, пошук та порятунок робітників та багато іншого в різних типах об'єктів та промислових підприємств-середовищ, такі як фабрики, підземні шахти, склади та багато іншого.

Chirp має безліч унікальних переваг, які роблять його гнучким варіантом, придатним для розгортань, де особливе значення мають вимоги до далекого позиціонування, високої надійності та низького енергоспоживання.

Завдяки високому коефіцієнту посилення системи, а також стійкості до перешкод та багатопроменевому завмиранню, CSS забезпечує виняткову дальність

позиціонування навіть на галасливих промислових об'єктах. Цей діапазон поширюється як усередині приміщень, так і зовні, пропонуючи рішення, які працюють в обох середовищах без необхідності дорогого ліцензування спектра. Його велика дальність зв'язку і висока пропускна спроможність каналу в поєднанні з точним визначенням розташування в межах 1-2 м і дуже низькою затримкою, забезпечують визначення місцезнаходження в режимі реального часу та додаток двосторонньої дальності або зв'язку з особливою продуктивністю, що поєднує багато інших поширених радіочастотних технологій.

Технологія з підтримкою Chirp також споживає дуже мало енергії і була розроблена, зокрема, для додатків з низьким енергоспоживанням, що дозволяє створювати рішення RTLS з доступними та ефективними апаратними опціями, такими як мітки стеження з вбудованими батареями, які можуть працювати протягом кількох років без необхідності перезаряджати чи заміняти.

Завдяки Chirp імпульсам з ширшою смугою пропускання та підтримці CSMA, системи на основі Chirp мають високу стійкість до радіочастотних перешкод, включаючи як вузькосмугові, так і широкосмугові перешкоди. Розповсюдження лінійно частотної модуляції за частотою також робить такі системи дуже стійкими до багатопроменевого завмирання. У більшості вузькосмугових радіочастотних технологій вихідний сигнал від передавача зазвичай досягає приймача з кількома відбиттями від будівель або інших навколишніх об'єктів. Це часто призводить до посилення або ослаблення частот, а також деструктивних перешкод, які можуть призвести до розриву лінії зв'язку вузькосмугової системи. CSS відрізняється тим, що енергія, що міститься в символах, що передаються, рівномірно розподіляється по смузі пропускання, а це означає, що канал зв'язку не буде розірваний, якщо об'єкт, що створює перешкоди, не блокує всю смугу пропускання каналу. На відміну від інших радіочастотних технологій, лінійно частотна модуляція також стійкий до ефекту Доплера, який викликає частотний зсув сигналів, що передаються. Його широка смуга пропускання також допомагає покращити якість прийому практично за будь-якого положення антени. Ці унікальні якості дозволяють використовувати chirp для

високопродуктивних додатків у шумному середовищі, наприклад, на промислових об'єктах, де потрібна надійність та стійкість до перешкод.

Сигнали CSS також можна використовувати для оцінки часу прибуття та розрахунку розташування різниці в часі прибуття, забезпечуючи точну продуктивність системи для масштабованих рішень, здатних підтримувати тисячі об'єктів, що одночасно відстежуються. Рішення Chirp також пропонують економічне обладнання та вимагають менше інфраструктури, ніж інші технології, забезпечуючи високу рентабельність інвестицій та ще більше посилюючи його унікальну здатність пропонувати масштабовані корпоративні розгортання.

1.3 Технологія direct sequence spread spectrum, DSSS

Розширений спектр із прямою послідовністю (DSSS) є одним із найбільш поширених методів розширеного спектру (SS) у захищеному зв'язку. DSSS [23] вузькосмуговий вхідний сигнал розширюється шляхом модуляції з послідовністю псевдошумових (PN) елементарних посилок. Така модуляція призводить до того, що спектр сигналу виявляється набагато ширшим по смузі пропускання, ніж вихідний сигнал. Спільний приймач зі знанням послідовності розширення може відновити вихідний вузькосмуговий сигнал шляхом де-розширення. З розвитком теорії компресійного зондування (CS) в деяких пізніших роботах вивчалися структури сигналів і був розроблений оптимальний компресійний приймач для спільного виявлення очікуваних сигналів DSSS (наприклад, у множині з кодовим поділом каналів (CDMA)). Однак виявлення присутності сигналу DSSS без знання послідовності розширення є складним завданням, оскільки після розширення сигнали DSSS можуть бути нижчими за рівень фонового шуму. У літературі було запропоновано безліч методів (наприклад, на основі автокореляції та нейронних мереж) для виявлення DSSS. Незважаючи на те, що були розроблені різні методи, методи виявлення на основі енергії простіше та дешевше реалізувати.

Розширення спектра прямою послідовністю - це метод, при якому сигнал даних передається в діапазоні частот, рівномірно розподіляючи його виділеним спектром. Розширення спектра прямою послідовністю використовується для забезпечення того, щоб конкретна смуга частот (і відповідний діапазон частот)

залишалася вільною від перешкод. Цей метод може бути пов'язаний з вирішенням проблеми внутрішньоканальних перешкод (наприклад, двох різних бездротових мереж, що передають в одній і тій самій смузі частот) та перехресних перешкод. Розширення спектра прямою послідовністю також може використовуватися як альтернативний підхід до мультиплексування з ортогональним частотним поділом, при якому сигнал основної смуги частот кодується і передається за кількістю фіксованих заздалегідь визначених каналів. У цій ситуації кожен канал може передавати різну інформацію, сигнали даних або часові інтервали для різних програм в одній мережі. Розширений спектр прямої послідовності також використовувався для передачі зашифрованих даних, а в деяких процесах він використовується для передачі сигналів, які не є даними, таких як сигналізація потужності або сигнали керування.

Передачі з розширеним спектром прямої послідовності множать передані дані на псевдовипадкову послідовність, що розширює, яка має набагато більш високу швидкість передачі даних, ніж вихідна швидкість передачі даних. Результуючий сигнал нагадує білий шум з обмеженою смугою пропускання, як аудіозапис «статики». Однак цей шумоподібний сигнал використовується для точного відновлення вихідних даних на стороні, що приймає, шляхом їх множення на ту ж розширювальну послідовність. Цей процес, відомий як стиск, математично є кореляцією переданої послідовності розширення з послідовністю розширення, яку приймач вже знає, що передавач використовує. Після стиснення, відношення сигнал/шум приблизно збільшується на коефіцієнт розширення, який є відношенням швидкості послідовності розширення до швидкості передачі даних.

DSSS - це метод модуляції з розширеним спектром, який використовується для передачі цифрового сигналу по радіохвилях. Спочатку він був розроблений для використання у військових цілях і використовував широкосмугові сигнали, що важко виявляються, щоб протистояти спробам глушення. Він також розробляється для комерційних цілей у локальних та бездротових мережах.

Потік інформації у DSSS ділиться на невеликі частини, кожна з яких пов'язана із частотним каналом у спектрі. Сигнали даних у точках передачі

комбінуються з послідовністю бітів з вищою швидкістю передачі даних, яка поділяє дані на основі коефіцієнта розширення. Код чіпа в DSSS являє собою надлишковий бітовий шаблон, пов'язаний з кожним бітом, що передається. Це допомагає підвищити стійкість сигналу до перешкод. Якщо біти пошкоджені під час передачі, вихідні дані можуть бути відновлені завдяки надмірності передачі.

Весь процес виконується шляхом множення несучої радіочастоти на цифровий псевдошумовий (PN) сигнал. Код PN модулюється інформаційний сигнал з використанням декількох методів модуляції, таких як квадратурна фазова маніпуляція, двійкова фазова маніпуляція і т. д. Потім змішувач з подвійним балансуванням множує модульований PN інформаційний сигнал і повідомлення. Таким чином, сигнал TF замінюється сигналом смуги пропускання, що має спектральний еквівалент шумового сигналу. У процесі демодуляції PN-модульована несуча змішується або множиться на вхідний сигнал. Отриманий результат є сигналом з максимальним значенням, коли два сигнали корельовані. Потім сигнал відправляється на демодулятор двійкової фазової маніпуляції. Хоча ці сигнали здаються зашумленими в частотній області, смуга пропускання, що забезпечується PN-кодом, дозволяє потужності сигналу падати нижче за поріг шуму без втрати інформації.

1.4 Аналіз та дослідження форматів аудіофайлів та аудіокодеків

Усі аудіофайли можна згрупувати за трьома категоріями: несжаті, без втрат і з втратами. Що потрапляє в якусь категорію, залежить від того, наскільки стиснуті дані і, як наслідок, наскільки якість або втрати відчуває слухач.

Якщо для стиснення аудіо у файлі не використовувався алгоритм стиснення (або кодек), відбуваються дві речі: нульова втрата якості звуку і, швидке заповнення носію інформації. По суті, стисла доріжка є відтворенням вихідного аудіофайлу, в якому сигнали реального світу перетворюються на цифровий звук.

Аудіофайл без втрат стискається до меншого розміру файлу для більшої практичності та ефективності, але це не повинно впливати на якість звуку, в той час як треки з втратами є найменшими і, отже, їх найлегше зберігати та

завантажувати, але для того, щоб це було можливо, необхідно відкинути частину аудіо-інформації у процесі стиснення.

WAV та AIFF, можливо, є найпопулярнішими форматами несжатих аудіофайлів, обидва засновані на PCM (імпульсно-кодовій модуляції), який широко відомий, як найпростіший механізм зберігання аудіо в цифровій області. Файли WAV та AIFF використовують однакову технологію, але зберігають дані трохи по-різному. Вони можуть зберігати аудіофайли якості CD або високої роздільної здатності.

WAV був розроблений Microsoft та IBM, тому він використовується на платформах Windows і є стандартним форматом, у якому закодовано всі компакт-диски.

AIFF був розроблений Apple як альтернатива WAV, і, хоча файли AIFF не такі широко популярні, вони мають кращу підтримку метаданих.

Найбільшим недоліком цього типу аудіофайлів є те, що ці файли дуже великі. Файл CD-якості (16 біт, 44,1 кГц) займає близько 10 МБ на жорсткому диску за хвилину.

Файли без втрат, FLAC (безкоштовний аудіокодек без втрат) стискається майже до половини розміру стисненого WAV або AIFF з еквівалентною частотою дискретизації, але, в цьому випадку, не повинно бути втрат з погляду звучання. Файли FLAC також можуть забезпечувати роздільну здатність до 32 біт, 96 кГц, що краще, ніж якість CD.

Інші формати аудіофайлів без втрат включають Apple Lossless та Windows Media Audio Lossless. Перший є гарною альтернативою FLAC, сумісною з IOS та Apple Music, хоча файли трохи менш компактні, ніж FLAC.

Бітрейт, з яким записується MP3, також впливає на якість звуку. MP3, закодовані зі швидкістю 128 кбіт/с, матимуть більше втрат звуку, ніж закодовані зі швидкістю 320 кбіт/с.

Іншим форматом із втратами є Apple Advanced Audio Coding, який стискується так само, як MP3, але трохи ефективніший і звучить краще. Apple

Advanced Audio Coding використовується для потокової передачі Apple Music (зі швидкістю 256 кбіт/с) та потокової передачі YouTube.

Формат Vorbis, часто званий Ogg Vorbis через його контейнер Ogg, є альтернативою MP3 і AAC з відкритим вихідним кодом з втратами, необмежену. за патентами. Ogg Vorbis – це формат файлу, який використовується (зі швидкістю 320 кбіт/с) для потокової передачі Spotify.

Щоб використання стгенографічних методів для вбудовування інформації в аудіофайли було зручним, воно потребує застосування форматів аудіофайлів без стиснення. Саме тому для проведення експериментів в дипломній роботі в якості стеганоконтейнера використовувався аудіоконтейнер формату WAV.

Формат аудіофайлу Waveform (WAV) – це формат аудіофайлу [9]. Він вважається форматом першого покоління без стиснення, за винятком деяких маніпуляцій для збереження звуку в цифровому вигляді, що призводить до збільшення розміру в порівнянні з такими форматами, як MP3 і WMA. Цей стандарт був розроблений IBM та Microsoft для зберігання бітового аудіопотоку на персональних комп’ютерах.

WAV - це несжатий аудіоформат, що не потребує обробки; він зберігає необроблений звук, для якого не потрібні спеціальні кодери/декодери, що робить його дуже добрим стандартом для обміну з різними платформами або операційними системами, такими як Windows, Mac та Unix [10]. Хоча звуковий файл може містити стислий звук, несжатий звук із використанням формату лінійної імпульсно-кодової модуляції (LPCM) є найпоширенішим. Стандартне аудіокодування компакт-дисків також використовує формат LPCM, оскільки він стислий і зберігає всі семпли, записані з вихідного аудіо. З цієї причини багато аудіоекспертів та професіоналів використовують формат WAV з LPCM для збереження якості звуку. Формат є додатком формату файлу обміну ресурсами (RIFF), а також широко використовується у мовленні.

Головними параметрами для WAV формату файлів є наступні [24]:

- Кількість каналів (Number of Channels). Це значення представляє з себе кількість окремих аудіосигналів, які є закодованими в секції даних звуку. Якщо

параметр має значення 1, тобто одиниця, то це вказує на те що аудіофайл має моно сигнал, 2 - стерео і так далі.

- Частота вибірки (Sample Rate) - це кількість вібрацій звукового сигналу, яке випадає в секунду. На це значення не впливає кількість каналів.
- Середня кількість байтів в секунду (Average Bytes Per Second). Дане значення, зазначає яку кількість байтів даних в секунду має проходити через цифро-аналоговий перетворювач (DAC) під час програвання файлу. Даний параметр є корисним у використанні, коли потрібно визначити, з правильною чи ні швидкістю надходять дані з джерела, аби не відставати від зчитування. Для розрахунку даного параметру використовується формула 2.1:

$$AvgBytesPerSec = SampleRate * BlockAlign. \quad (1.1)$$

- Вирівнювання блоку(Block Align) визначається за допомогою кількості байтів на вибірку(Significant Bits Per Sample) та кількості каналів, це наводиться у формулі 2.2:

$$BlockAlign = SignificantBitsPerSample / 8 * NumChannels . \quad (1.2)$$

Кількість бітів на вибірку (Significant Bits Per Sample). Даний параметр визначає кількість бітів, які створюють кожну-вибірку сигналу. Частіше за все дане значення є 8, 16, 24 або 32. У випадках, коли даний параметр не ділиться без остачі на 8, то відбувається округлення параметру до мінімальної кількості байтів. Байти, які не використовуються приймають значення 0 та ігноруються. Формати даного типу не часто зустрічаються.

2 МАТЕМАТИЧНА МОДЕЛЬ СТЕГАНОГРАФІЧНОГО ПРИХОВУВАННЯ ДАНИХ В АУДІОФАЙЛАХ ІЗ РОЗШИРЕННЯМ СПЕКТРУ МЕТОДОМ ПРЯМОЇ ПОСЛІДОВНОСТІ

2.1 Математична модель розширення спектру методом прямої послідовності в телекомунікаційних системах

Фундаментальна теорема Шеннона-Хартлі встановлює обмеження на пропускну здатність каналу, тобто на верхню межу максимальної кількості інформації, яка може бути передана в системі зв'язку із заданою смугою частот та потужністю адитивного білого гаусівського шуму [25]:

$$C = \Delta F \log_2 1 + \frac{P_S}{P_N}, \tag{2.1}$$

де:

- C – пропускна спроможність каналу, біт/с;
- ΔF – смуга частот, Гц;
- P_S – потужність корисного сигналу;
- P_N – потужність адитивного білого шуму гауса;
- $\frac{P_S}{P_N}$ — співвідношення потужності сигналу до шуму (SNR).

З рівняння можна зробити висновок, що при низьких SNR, пропускну здатність можна збільшити лише за допомогою розширення смуги частот, і для цього добре підходить технологія прямого розширення спектра. Дійсно, використовуючи дуже довгі послідовності, що розширюють, навіть при дуже низькому значенні SNR можна реалізувати високошвидкісну передачу інформації. Ця деталь також може бути використана, наприклад, для побудови екологічності систем зв'язку, тобто за $P_S \approx P_N$. Тоді, використовуючи послідовності, що

розширюють, довжиною 10^8 біт і більше, можна реалізувати передачу інформації зі швидкостями в десятки і сотні Мбіт/с.

Тепер слід пояснити технологію розширення спектра прямою послідовністю в такий спосіб.

Для початку треба розглянути набір $s = \left(s_1, s_2, ..., s_k\right)$ псевдовипадкових бітових послідовностей (векторів):

$$\begin{aligned} s_1 &= \left(s_{1,0}, s_{1,1}, ..., s_{1,n-1}\right), \\ s_2 &= \left(s_{2,0}, s_{2,1}, ..., s_{2,n-1}\right), \\ &\ldots, \\ s_k &= \left(s_{k,0}, s_{k,1}, ..., s_{k,n-1}\right), \end{aligned} \tag{2.2}$$

представлені, наприклад, у полярній формі, тобто,

$$\forall i, j : s_{i,j} = \left\{ \begin{matrix} 1, \\ -1. \end{matrix} \right\} \tag{2.3}$$

Вектори призначені для розширення спектра вихідного повідомлення, тому їх називають чіп-коди або чіпові коди. Кожне значення $s_{i,j}$ називається чіпом або елементом послідовності.

Вектори $s_1, s_2, ..., s_k \in s$ формуються отже їх взаємна кореляція зневажливо мала, тобто,

$$\forall i \neq j : \rho\left(s_i, s_j\right) = \sum_{\upsilon=0}^{n-1} s_{i,\upsilon} s_{j,\upsilon} \approx 0 \tag{2.4}$$

Наприклад, якщо вектори s_1, s_2, ..., s_k будуть рівні рядкам (або стовпцям) матриці Адамара, то в результаті можливо отримати ортогональні чіп-коди, а саме коди наступного виду:

$$\forall i \neq j : \rho\left(s_i, s_j\right) = 0 \tag{2.5}$$

Існують інші способи генерації чіп-кодів, наприклад, коли елементи $s_{i,j}$ приймають випадкове значення на інтервалі $[-1,1]$. Однак умова, наведена в формулі (2.4) для генерації послідовностей розширення є вирішальною.

Тепер можна припустити, що повідомлення складається з k бітів, тобто $m_1, m_2, \ldots m_k$, наприклад, для цього можна записати інформаційні біти $m_i,\ i = 1, 2, \ldots, k$ також у полярній формі, тобто,

$$\forall i, j : m_i = \left\{ \begin{matrix} 1, \\ -1. \end{matrix} \right\}, \tag{2.6}$$

Модуляція інформаційних бітів $m_i,\ i = 1, 2, \ldots, k$ можна отримати різними способами, наприклад, за допомогою полярного запису за виразом:

$$\forall i : M_i = m_i s_i = \left(m_i s_{i,0}, m_i s_{i,1}, ..., m_i s_{i,n-1}\right) \tag{2.7}$$

Таким чином, замість одного бінарного елемента m_i буде вектор (послідовність) M_i бінарних елементів n, які передаються.

Після цього має сенс наступне припущення, що тривалість елемента m_i та його розширеної версії мікросхеми M_i збігаються. Тепер можна припустити,

наприклад, що вони дорівнюють T секундам. Тоді тривалість одного чіпа $s_{i,j}$ буде у n разів меншою: $T_{chip} = \frac{T}{n}$. Тому чіпи передаються на частоті (формула 2.8):

$$F_{chip} = \frac{1}{T_{chip}} = \frac{n}{T} Hertz \tag{2.8}$$

Ця частота в n разів більша за частоту інформаційних бітів $F = \frac{1}{T}$. Таким чином був отриманий розкид частоти вихідного повідомлення n разів.

Збільшення тривалості сигналу за його частоту називається основою B. Якщо $B > 1$, то сигнали називаються складними.

Для розглянутого вище випадку має істинність таке вираження:

$$B = T_{chip} F_{chip} = n > 1 \tag{2.9}$$

і сигнали прямого розширення є складними сигналами.

У канал зв'язку може одночасно передаватися кілька інформаційних бітів, тобто приймач отримує адитивну суміш:

$$Mix = N + \sum_{j=1}^{k} M_j \tag{2.10}$$

де символ N означає послідовність випадкових елементів, викликаних шумом (наприклад, адитивним білим Гауссівським шумом).

Приймач має синхронізовані копії кодів чипів $s_1, s_2, ..., s_k \in s$. Для відновлення інформаційних бітів необхідно обчислити кореляцію між прийнятим сигналом та відповідними кодами чіпа. Наприклад, щоб відновити значення m_i із

формули (2.7), приймач обчислює $\rho(s_i,\ Mix)$. Рівняння із формули (2.4) є лінійним, тобто можна записати:

$$\rho(s_i,\ Mix) = \rho(s_i, N) + \rho\left(s_i, \sum_{j=1}^{k} M_j\right) \tag{2.11}$$

Для випадкового шуму N є:

$$\rho(s_i, N) \approx 0 \tag{2.12}$$

У цьому $\forall i \neq j : \rho(s_i, s_j) \approx 0$; отже,

$$\rho(s_i,\ Mix) \approx \sum_j \rho(s_i, m_j s_j) \approx \rho(s_i, m_i s_i) \tag{2.13}$$

Таким чином, ми можемо написати правило відновлення інформаційного біта

$$m_i = \rho(s_i,\ Mix) = \begin{Bmatrix} -1, \rho(s_i,\ Mix) < 0; \\ +1, \rho(s_i,\ Mix) > 0. \end{Bmatrix} \tag{2.14}$$

Треба звернути увагу, що у наведених вище міркуваннях окремі біти $m_i,\ i = 1, 2, \ldots, k$ можуть належати різним відправникам. І тут реалізується множинний доступ із кодовим поділом каналів [17].

Слід зазначити, що є такі переваги використання технології прямого розширення спектра [26]:

- Стійкість до ненавмисного або навмисного глушіння;
- Спільне використання одного каналу кількома користувачами;

- Знижений рівень сигналу/фонового шуму ускладнює перехоплення;
- Визначення відносної синхронізації між передавачем та приймачем та багато іншого.

Також слід розглядати можливість використання цієї технології у стеганографії, тобто для приховування інформаційних повідомлень у мультимедійних файлах.

2.2 Математична модель розширення спектру методом прямої послідовності в стеганографічних системах

Технологія прямого розширення спектра вже давно використовується у стеганографії.

У перших роботах [11-13] за цією темою було запропоновано загальну схему стеганосистеми з прямим розширенням спектра. Автори використовували обкладинки, але їх міркування можна поширити і на інші мультимедійні дані.

Загальна ідея такої стегосистеми полягає в наступному: (Рис. 2.1).

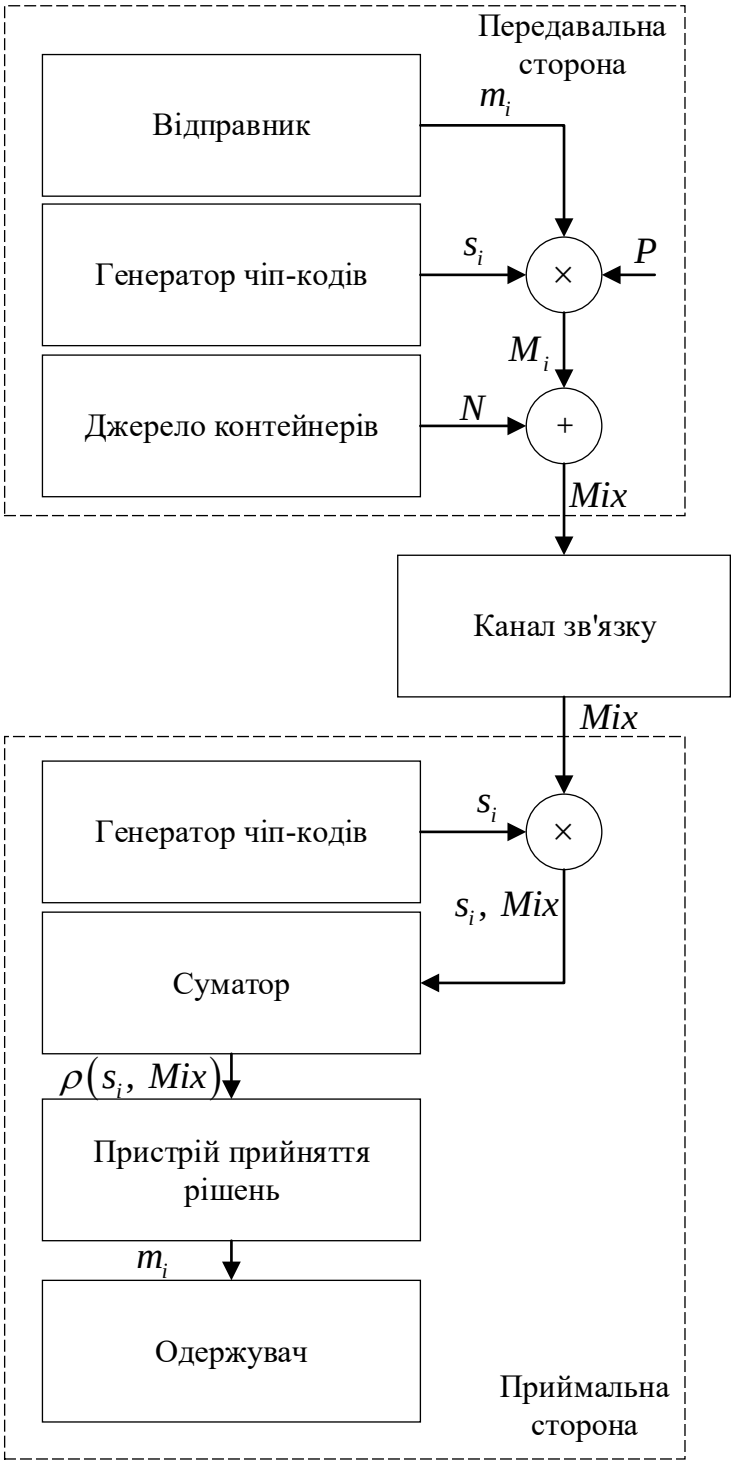

Рисунок 2.1 – Блок-схема приховування та відновлення.

Для того, щоб приховати дані, виконуються такі кроки:

1) Інформаційне повідомлення подається у полярній формі m_i;

2) m_i множиться біт за бітом на послідовність, що розширює, як в формулі (2.7);

3) Мультимедійні дані (зображення, звук, відео тощо) інтерпретуються як шум каналу зв'язку. У всіх вище позначених розрахунках це N , формула (2.10).

4) Далі ці дані ховаються в даних обкладинки за правилом, яке описане у формулі (2.10).

В розрахунках були використані такі позначки, як:

- N - мультимедійні дані (контейнер, обкладинка), усередині яких заховано повідомлення;
- Mix — змінені мультимедійні дані (заповнений контейнер) після приховування повідомлень.

Для відновлення даних виконуються такі кроки:

1) На першому кроці обчислюється кореляція.

- Продукт розрахований на змішування s_i, Mix;
- Обчислюється сума $\rho(s_i, Mix) = \sum_{\upsilon=0}^{n-1} s_{i,\upsilon} Mix_\upsilon$;

2) На другому кроці, якщо формули (2.12) і (2.13) вірні, значення відновленого біта обчислюється за рівнянням (2.14).

3) Після цього, відновлене повідомлення відображається в представленні користувача.

Слід зазначити, що у перших роботах автори зіштовхнулися зі значними труднощами. Наприклад, частота помилок у виділених повідомленнях була дуже висока. BER можна зменшити різними способами, наприклад, за допомогою фільтрації, коригувальних кодів і т. д. Найпростіший спосіб – збільшити потужність чіп-кодів. Для цього рівняння (2.10) можна переписати у вигляді:

$$Mix = N + P\sum_{j=1}^{k} M_j = N + \sum_{j=1}^{k} m_j \left(Ps_j\right) \qquad (2.15)$$

де значення P вказує коефіцієнт посилення потужності кодів мікросхем $s_j = \left(s_{j,0}, s_{j,1}, ..., s_{j,n-1}\right)$.

Збільшуючи потужність P, можна досягти зниження BER. Однак це неминуче призводить до спотворення контейнера N. Наприклад, при $k = 10$ і $P = 10$ кожне значення покриває файл N може бути спотворене на $\pm kP = 100$. Це може представляти реальну проблему, оскільки BER все ще високий. Наприклад, навіть при P = 100 частота помилок становить більше 10 %.

Після проведення досліджень за цією темою, був вивчений вплив різних способів формування послідовностей, що розширюють, на значення BER. Загальні висновки про високий BER підтвердилися, і пояснення цього явища, полягає в наступному. Перший член, формули (2.10) представляє деяку реалізацію випадкового адитивного білого шуму N. Це дійсно має місце для систем зв'язку з розширеним спектром. Припущення (2.12) виконується, що дає практично безпомилкове відновлення повідомлення за правилом (2.14). Однак у стеганосистемі (2.10) маються на увазі мультимедійні дані. Зазвичай це надмірні, сильно корельовані дані. Не випадкова реалізація Гаусовського шуму, і припущення (2.12) часто виконується. Очевидно, це призводить до помилок у відновлених бітах за правилом (2.14). У роботах [27, 28] було запропоновано ефективний спосіб боротьби з цим явищем, наприклад, формувати послідовності чіпів особливим, адаптивним чином. Дійсно, якщо врахувати статистичні властивості мультимедійних даних, можна сформувати коди чіпів $s_1, s_2, \ldots, s_k$ так, щоб умова (2.12) виконувалася гарантовано. У цьому випадку можна забезпечити практично безпомилкове вилучення інформаційних повідомлень за правилом (2.14).

В дипломній роботі були продовжені експерименти за даною темою. Були використані звукові обкладинки та проведені експерименти з різними способами генерації кодів чіпів. Різноманітність способів генерації розширюючих послідовностей, призводить до різних значень BER. Це область можливої оптимізації стеганосистеми.

2.3 Показники та критерії ефективності аудіо-стеганосистем

Безпека, непомітність і пропускна здатність - три атрибути для будь-якої стеганосистеми, в тому числі аудіо-стеганосистеми, вони для необхідні приховування секретних даних. Це найважливіші чинники, що визначають ефективність установки стеганографії. Відповідно до його використання існують певні специфічні вимоги [24].

Непомітність є головним пріоритетом у стеганографічних системах та спрямована на приховування секретних даних в інших засобах масової інформації. Людина не може зрозуміти це навіть при застосуванні статистичних методів. Статистичні методології є корисним засобом для зловмисників, щоб визначити, чи передаються секретні дані при обміні даними між двома сторонами. Отже, обкладинка має бути помітно змінена з погляду статистичних стандартів через вбудовування секретних даних; тобто, якщо схожі статистичні дані виявлені як у вихідному, так і в стеганографічному файлах, можна вважати, що безпека досить висока, щоб дозволити передачу даних. Якість обкладинки повинна підтримуватись при спільному використанні через незахищені мережі, незважаючи на шум від процесу вбудовування.

Термін "безпека" в стеганографії відноситься до "невиявлення" або "непомітності". Отже, стеганографічний підхід є безпечним, коли дані, які він приховує, не можуть бути виявлені за допомогою статистичних методів якоюсь третьою стороною. Безпека є основною вимогою для запобігання доступу незаконних осіб або комп'ютерів при спілкуванні незахищеним каналом, що забезпечує безпеку даних.

Ефективна стеганографічна система, як правило, призначена для надсилання максимальної кількості інформації з використанням найменш закритих носіїв. Це знижує ймовірність перехоплення при обміні даними незахищеним каналом і, як правило, вимагає високої ємності впровадження. Посилання описує швидкість вбудовування бітів залежно від розміру зображення обкладинки. Ключовим завданням стеганографії є підтримка високої ємності корисного навантаження при збереженні безпеки та непомітності [24].

Надійність означає здатність методу впровадження та вилучення протистояти будь-якому пошкодженню третьою особою у вигляді будь-яких методів обробки. У випадку стеганографії, якщо файли стеганографії не піддаються впливу або зміні під час відправки через Інтернет, це не вважається атакою, і одержувач отримує стегофайл, як передбачалося. В іншому випадку в процесі зв'язку можуть відбутися такі атаки, як стиснення, перетворення формату файлу та перетворення між цифровим та аналоговим форматами. Однак для систем відбитків пальців необхідна надійність у разі навмисної зміни або зміни файлів.

2.4 Математична модель та алгоритми приховуванні та вилучення даних в аудіофайлах із розширенням спектру методом прямої послідовності

Метод заснований на одночасній передачі вузькосмугового (контейнера) і широкосмугового (приховане повідомлення) сигналів [34]. Основними перевагами методів стеганографії з розширенням спектра сигналів є: стійкість до шуму, некорельовані з відліками повідомлення, мале збільшення енергії сигналів при впровадженні повідомлення, можливість роботи при низькому співвідношенні сигнал / шум, що зменшує ризик появи чутних (видимих) спотворень, підвищує стійкість до поширених спотворень сигналів. Ідея алгоритму полягає в тому, що приховуване повідомлення має стати рівномірно розподіленим за частотами несучого сигналу, при цьому має виконуватися умова – співвідношення сигнал/шум у каналі має бути дуже низьким, аби не виникла проблема зі спотворенням контейнеру і не з’явилися підозри наявності повідомлення в стеганоконтейнеру [24].

Використання в аудіоконтейнерах методу прямого розширення спектру має такий самий вигляд, як і використання в стеганосистемах загалом, тому розглянутої вище моделі достатньо для розуміння.

Далі наведено два основних метода вбудовування повідомлення, та види чіп-кодів, які комбінувалися з цими методами.

2.4.1 Спосіб вбудовування повідомлення методом прямої послідовності на основі робот Лізи Марвел

Загальна ідея для вбудовування повідомлення в контейнер полягає в інтерпретації мультимедійних даних (зображення, аудіо та ін.), як шуму в каналі зв'язку, більш детально цей процес описаний в роботах Лізи Марвел [12-13].

Якщо описувати метод більш конкретно [24], то як і в стандартному методі розширення спектра, приховуване повідомлення інтерпретується в полярному вигляді і побітно множиться на псевдовипадкову послідовність (2.16):

$$\forall i : M_i = m_i s_i = (m_i s_{i,0}, m_i s_{i,1}, ..., m_i s_{i,n-1}) \tag{2.16}$$

Приховування даних в стеганосистемі виконується за формулою (2.17):

$$Mix = N + \sum_{j=1}^{k} M_j \tag{2.17}$$

Після цього інформація відновлюється за формулою (2.18):

$$\rho(s_i, Mix) = \rho(s_i, N) + \rho(s_i, \sum_{j=1}^{k} M_j) \tag{2.18}$$

Якщо виконуються наступні припущення, надані рівнянням 2.30 і рівнянням 2.31:

$$\rho(s_i, N) \approx 0 \tag{2.19}$$

$$\rho(s_i, Mix) \approx \sum_j \rho(s_i, m_j s_j) \approx \rho(s_i, m_i s_i) \tag{2.20}$$

тоді значення відновленого біта обчислюється за формулою (2.15).

Алгоритм цього методу показаний на рисунку. 2.2

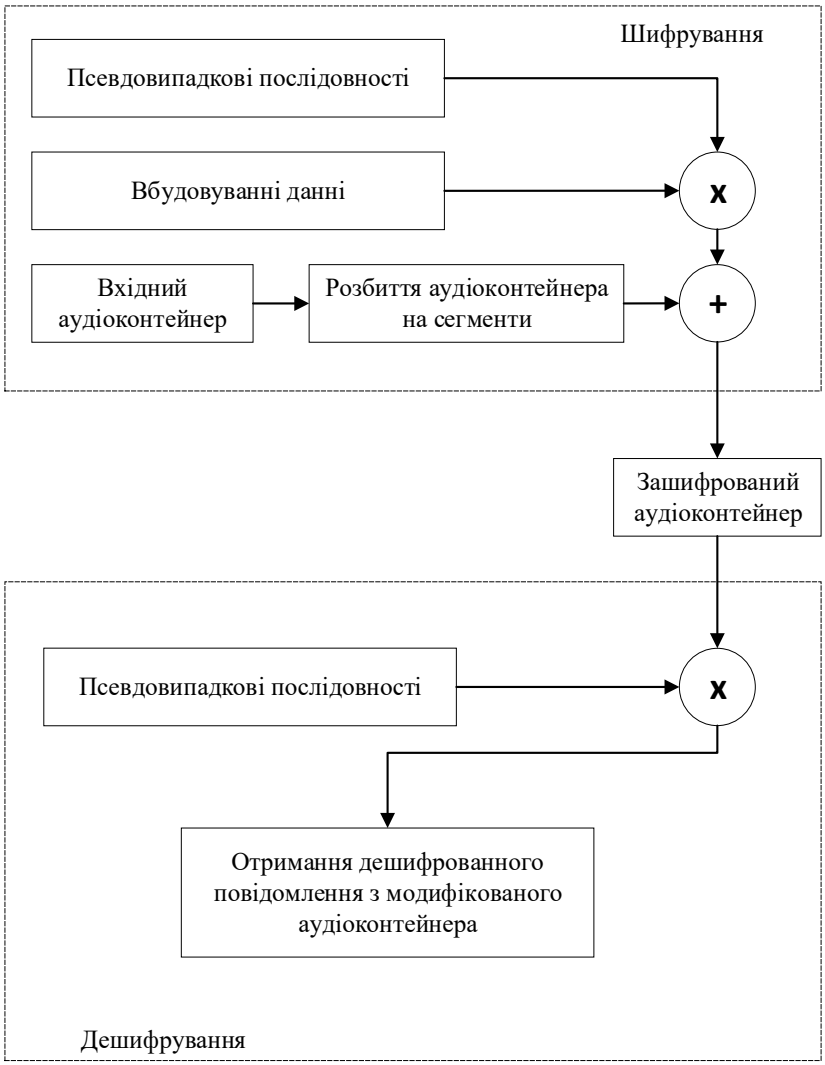

Рисунок 2.2 – Алгоритм вбудовування повідомлення методом Лізи Марвел

2.4.2 Спосіб вбудовування повідомлення методом адресації

Кодування методом адресації має суттєві відмінності від традиційного кодування з прямим розширенням спектру (запропонованого Лізою Марвел), а саме методами кодування та декодування бітів із контейнерів. Перше, що слід зазначити, це те, що режим адресації потребує функцію, яка призначена для ділить бітовий масив повідомлення на масив десяткових чисел, тобто формує так звані адреси. Кожне число десяткового масиву є адресом біту кодованого повідомлення. Декодування повідомлення з модифікованого аудіоконтейнера потребує функції, яка враховує, що в стеганоконтейнері закодоване не саме повідомлення, а адреси бітів повідомлення, тому після «витягування» адрес, вони конвертуються в біти повідомлення.

Найкращими показниками цього методу є низький MSE і високий PSNR. Саме тому, й було запропоновано використання цього методу на відміну від стандартного методу будови прямим розширенням спектру.

Основним недоліком методу є високий показник BER, що означає низьку ймовірність вилучити правильно повідомлення, і отримати дуже високі пошкодження закодованої інформації. На відміну від минулих досліджень, проблему зі швидкістю алгоритму вдалося частково вирішити, зміною мови програмування, але у порівнянні з іншими методами вбудовування метод все ще страждає від низької швидкодії.

Алгоритм кодування та декодування повідомлень за допомогою методу адресації показаний на Рисунку. 2.3.

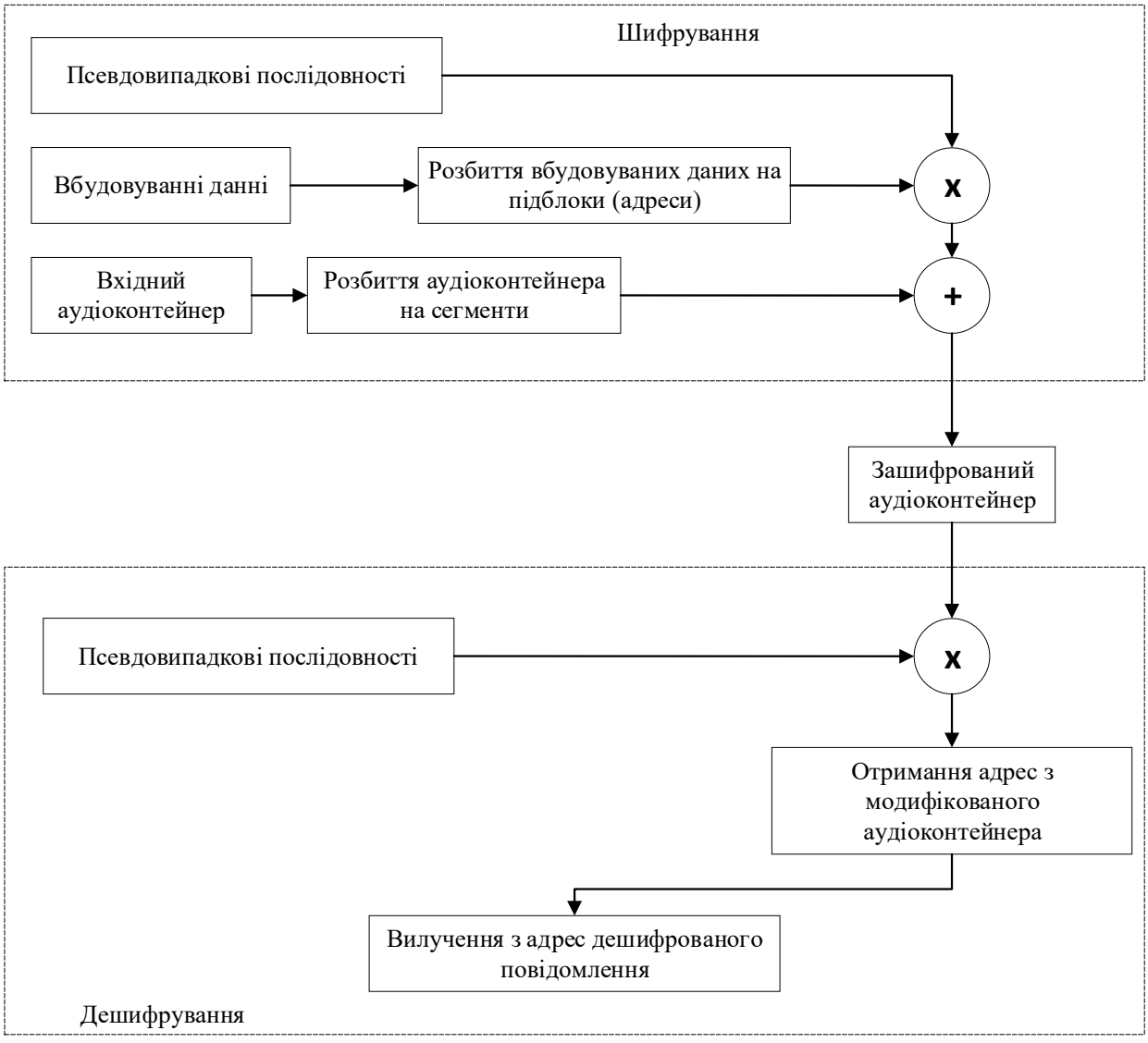

Рисунок 2.3 – Алгоритм вбудовування повідомлення методом адресації

2.4.3 Методи формування сигналів для приховування повідомлення

2.4.3.1 Метод №1

Метод №1 створений на основі використання ортогональних дискретних сигналів Уолша Адамара.

Це дискретні послідовності [34] довжини $n = 2^w$, $w = 1, 2, \ldots$, які можуть бути сформовані як рядки (або стовпці) матриці Адамара H_n порядку $n = 2^w$:

$$H_{2^w} = \begin{bmatrix} H_{2^{w-1}} & H_{2^{w-1}} \\ H_{2^{w-1}} & -H_{2^{w-1}} \end{bmatrix}, \; H_1 = [1] \tag{2.21}$$

Наприклад, для n = 4 буде чотири коди чіпа:

$$\begin{aligned} s_1 &= (1,1,1,1), \\ s_2 &= (1,-1,1,-1), \\ s_3 &= (1,1,-1,-1), \\ s_4 &= (1,-1,-1,1). \end{aligned} \tag{2.22}$$

В експериментах у роботі були використані розширюючі послідовності, що мають довжину $n = 2^{10} = 1024$. Тому були рекурсивно згенеровані H_{1024} за правилом (2.21) і сформовані $s = \left(s_1, s_2, \ldots, s_{k=1024}\right)$,як безліч струн H_{1024}.

Послідовності Уолша-Адамара утворюють ортогональну систему, тобто формула (2.23):

$$\forall i \neq j \; : \; \rho\left(s_i, s_j\right) = 0 \; , \tag{2.23}$$

що добре узгоджується з (2.10).

Проте, слід зазначити головний недолік таких чіп-кодів. Кількість різних ортогональних послідовностей довжини n не може перевищувати n. Тому для

приховування великої кількості бітів в одному контейнері цього методу генерації може виявитися недостатньо.

2.4.3.2 Метод №2

Метод на основі нелінійної модуляції. В основних роботах [12,13] з стеганографії з розширеним спектром використовувалося нелінійне правило формування чіп-кодів $s = (s_1, s_2, ..., s_k)$. Зокрема, кожен чіп $s_{i,j}$ формувався, як реалізація випадкової величини, розподіленої за стандартним законом.

Для цього було запропоновано правило:

$$s_{i,j} = \begin{cases} \Phi^{-1}(u_{i,j}), m_i = -1; \\ \Phi^{-1}(u'_{i,j}), m_i = 1, \end{cases} \quad (2.24)$$

де

$$u'_{i,j} = \begin{cases} u_{i,j} + 0.5, u_i < 0.5; \\ u_{i,j} - 0.5, u_i \geq 0.5, \end{cases} \quad (2.25)$$

- $u_{i,j}$ - Реалізація випадкової величини, рівномірно розподіленої на інтервалі $[0, 1]$;
- Φ^{-1} – зворотна кумулятивна функція розподілу для стандартної випадкової гауссівської величини.

У експериментах використовувалися чіп-коди довжиною $n = 1024$, згенеровані за правилом (2.24).

Розширювальні послідовності з Гауссівським розподілом чіпів (2.24) також вивчалися раніше стосовно зображень покриття [27, 28]. У дипломній роботі було розширено це дослідження на випадок аудіоконтейнерів.

2.4.3.3 Метод №3

Метод №3 створений на основі рівномірно розподілених випадкових чисел на інтервалі від -1 до 1.

Також реалізовано формування розширювальних послідовностей довжини $n = 1024$ випадкових чисел, рівномірно розподілених на інтервалі $[1, 1]$. Кожна чіп $s_{i,j}$ формується випадково з рівною ймовірністю і незалежно від інших значень.

2.4.3.4 Метод №4

Метод №4 створений на основі квазіортогональних дискретних сигналів

Для досліджень в дипломній роботі використовуються квазіортогональні дискретні сигнали. Ці сигнали можно вважати квазіортогональними, так як значення коефіцієнта взаємної кореляції сформованих сигналів значно не відрізняється від нуля, саме таку множину можна вважати ансамблем квазіортогональних дискретних сигналів.

Основними недоліками методу є високий MSE та низький PSNR. Головною перевагою є низька вірогідність помилок, тобто BER, але на жаль тільки в стандартному методі вбудовування даних.

2.4.3.5 Метод №5

Метод №5 створений на основі формування адаптивних квазіортогональних дискретних сигналів.

Адаптивна генерація послідовностей, для своєї реалізації, потребує введення деяких обмежень [24]. Перше обмеження вводиться на коефіцієнт кореляції, інше на модуль генерованого сигналу. Формула (2.26) демонструє більш конкретно ці обмеження:

$$\forall i : \left|\rho\left(S_i, C\right)\right| \le \rho_{\max} . \qquad (2.26)$$

Подібність контейнера C до згенерованого сигналу S_i, або його інверсії, визначається значенням – $\rho_{\max}$ Через достатньо довгий час пошуку потрібної

псевдовипадкової послідовності при малих значеннях $\rho_{\max}$, це значення не може бути занадто малим. Насправді кожен сигнал у цьому наборі генерується псевдовипадковим чином (за допомогою генератора псевдовипадкових чисел). У цьому випадку використовуються не всі сформовані послідовності, а лише ті, що задовольняють обмеження за формулою (2.26). Оскільки використовується обмеження $\rho_{\max}$, то відбувається так зване відхилення «непридатних» сигналів, тобто тих, які не підходять правилу $\left|\rho\left(\Phi_i, C\right)\right| > \rho_{\max}$.

Слід зазначити, безпомилкове відновлення інформаційних повідомлень буде відбуватися тільки у випадку, коли $\rho_{\max} < G \cdot n$ і разом з тим $\forall i \neq j : \rho\left(S_i, S_j\right) = 0$. Фактично в цьому випадку всі сигнали в групі взаємно ортогональні, а отже виникає відсутність перешкод.

Процес приховування та відновлення інформаційних повідомлень реалізується так само, як і відомі методи, тобто з використанням співвідношень (2.11), (2.14). Однак послідовності, які зараз використовуються в колекціях, справді не будуть пов’язані з контейнерами, тобто припущення буде справедливим (для малих $\rho_{\max}$, звичайно). Як показано нижче, адаптивна генерація сигналу з ансамблів дійсно може значно зменшити інтенсивність помилок у відновлених повідомленнях.

Основним недоліком методу є високий показник MSE і низький PSNR, але у комбінації з методом адресації ця проблема нівелюється. Також до недоліків слід віднести низьку швидкість алгоритму, яка також покращилася після зміни мови програмування, але все одно швидкодія алгоритму, залишає бажати кращого. Натомість перевага методу стосується низьким BER, який на відміну від методу Уолша Адамара залишається таким, навіть при комбінуванні з методом адресації, що не є дивовижним, оскільки сигнали майже взаємно ортогональні.

2.5 Показники ефективності приховування повідомлень

Щоб оцінити ефективність стеганосистеми, необхідно оцінити спотворення у відновлених повідомленнях, а також спотворення оригінального контейнера.

Тому була експериментально оцінена частота бітових помилок (BER) у відновлених повідомленнях. Також в роботі були оцінені спотворення звукового покриття за середньоквадратичною помилкою (MSE), відносини сигнал/шум (SNR) і піковим відношенням сигнал/шум (PSNR). Також більш точних показників досліджень, був використаний метод дослідження якості звуку PEAQ.

2.5.1. Частота бітових помилок, BER

Частота бітових помилок розраховується за рівнянням (2.27):

$$BER = \frac{k_{error}}{k}, \tag{2.27}$$

де

k_{error} – кількість помилково відновлених бітів,

k – загальна кількість бітів в інформаційному повідомленні.

2.5.2. Середньоквадратична помилка, MSE

Для оцінки спотворення контейнера використовується середньоквадратична помилка (MSE), тобто усереднений квадрат різниці між вихідними мультимедійними даними та даними покриття, отриманими після того, як повідомлення було приховано. Для звукового сигналу, представленого набором дискретних n семплів, MSE обчислюється за рівнянням (2.28):

$$MSE = \frac{1}{n}\sum_{j=0}^{n-1}[Mix_i - N_i]^2 \tag{2.28}$$

2.5.3. Відношення сигнал/шум, SNR

SNR описує загальний шум, що присутній на вихідному звороті, виявленому в обкладинці, в порівнянні з шумом вихідного рівня сигналу. SNR є показником якості і є грубим розрахунком можливості помилкового перемикання; він є засобом порівняння відносної продуктивності різних реалізацій. SNR оцінюється за рівнянням. (2.29):

$$SNR = \frac{\sigma^2_{signal}}{\sigma^2_{noise}} \tag{2.29}$$

2.5.4. Пікове відношення сигнал/шум, PSNR

Більш наочною характеристикою спотворення покриття є відношення максимально можливої потужності сигналу до потужності шуму, що спотворює. Зазвичай цю характеристику виражають у логарифмічному масштабі та розраховують за рівнянням:

$$\begin{aligned} PSNR &= 10 \cdot \log_{10}\left(\frac{N^2_{\max}}{MSE}\right) = 20 \cdot \log_{10}\left(\frac{N_{\max}}{\sqrt{MSE}}\right) = \\ &= 20 \cdot \log_{10}\left(N_{\max}\right) - 10 \cdot \log_{10}\left(MSE\right). \end{aligned} \tag{2.30}$$

де N_{max} – максимальне значення сигналу N .

Під N мається на увазі мультимедійні дані, тобто звуковий сигнал представлений набором дискретних відліків. Якщо кожен дискретний відлік кодувати в бітах, то (в лінійно-імпульсно-кодовій модуляції, ІКМ) максимально можливе значення $N_{max} = 2^B - 1$. У наведених експериментах використовувалися найпростіші звукові сигнали з $B = 8$, тобто $N_{max} = 255$ (2.30) для цієї величини набуває вигляду формули (2.31):

$$PSNR = 20 \cdot \log_{10}\left(255\right) - 10 \cdot \log_{10}\left(MSE\right) \tag{2.31}$$

2.5.5 Метод PEAQ

Перцептивна оцінка якості звуку (PEAQ) [29] - це стандартизований алгоритм для об'єктивного вимірювання якості звуку, що сприймається. У перцептивному кодуванні важливо визначити рівень шуму, який може бути введений в сигнал до того, як він стане чутним. Оскільки слухова система людини нелінійна, рівні шуму залежать від часу і частотних характеристик звукового

сигналу. Психоакустичні дослідження можуть надати порогові критерії для різних акустичних подій і виникають в результаті звуків, що сприймаються. Ключем є маскування, яке описує ефект, який звук справляє в інший одночасний звук. Маскування залежить від спектрального складу як маскуючого, так і сигналу, що маскують, а також інших змін у часі. Вхідний сигнал розкладається на субдискретизовані спектральні компоненти. Для кожної вибірки проводиться оцінка фактичного порогу з використанням правил, відомих з психоакустики. Це перцептивна модель системи кодування. Спектральні компоненти квантуються і кодуються, утримуючи шум квантування нижче порогу, що маскується. Нарешті формується бітовий потік.

Аналіз результатів ґрунтується на оцінці суб'єктивних відмінностей. Він порівнює тестований сигнал із вихідним еталонним сигналом.

За основу для тестування була версія Python алгоритму Perceptual Evaluation of Audio Quality (Додаток А), для кожного з вбудованих сигналів були проведені відповідні тести на основі робот [29].

Вихідною змінною об'єктивного методу вимірювання PEAQ є ступінь об'єктивної різниці (ODG) та індекс спотворення (DI). ODG відповідає ступеню суб'єктивної відмінності (SDG) у суб'єктивній області. Дозвіл ODG обмежений одним десятковим знаком. Однак слід бути обережним і взагалі не очікувати, що різниця між будь-якою парою ODG в одну десяту міру буде значною. DI має те саме значення, що й ODG. Однак DI та ODG можна порівнювати лише кількісно, але не якісно. Як правило, ODG слід використовувати як міру якості для значень ODG, що перевищують приблизно –3,6. ODG дуже добре корелює із суб'єктивною оцінкою в цьому діапазоні. Коли значення ODG менше –3,6, слід використовувати DI [29].

Інформація, отримана цим процесом, призводить до кількох значень, так званих MOV («Вихідні змінні моделі»), і може бути корисним для детального аналізу сигналу. Кінцева мета замість цього полягає в тому, щоб керувати мірою якості, що складається з одного числа, яке вказує на чутність спотворень, що присутні в сигналі, що тестується. Для цього потрібна додаткова обробка MOV, яка

імітує когнітивну частину слухової системи людини. Тому алгоритм PEAQ використовує штучну нейронну мережу.

Серед значень MOV були обрані найбільш важливі та побудовані графіки.

- Average Distorted Block (ADB)

Кількість достовірних кадрів [30] з ймовірністю виявлення бінаурального каналу $P_{bin}[n]$ вище 0,5 підраховується ($n_{distorted}$).

Для всіх допустимих кадрів обчислюється загальна кількість кроків вище за поріг бінаурального каналу $Q_{bin}[n]$:

$$Q_{sum} = \sum_{\forall n} Q_{bin}[n] \tag{2.32}$$

Розраховується спотворення середнього спотвореного блоку ADB:

– якщо $n_{distorted}$ дорівнює нулю, то $ADB = 0$ (спотворення не чути);

– якщо $n_{distorted} > 0$ і $Q_{sum} > 0$, то $ADB = log_{10}\left((Q_{sum}) \ / \ n_{distorted}\right)$;

– якщо $n_{distorted} > 0$ і $Q_{sum} = 0$, то $ADB = -0.5$.

- Relative Disturbed Frames

MOV Relative Disturbed Frames, вони ж відносно спотворені кадри MOV представляє кількість кадрів з:

$$\max_{\forall k} \left(10 \ log\left(\frac{P_{noise}[k,n]}{M[k,n]}\right)\right) \geq 1.5 dB \qquad k \in [0, Z-1] \tag{2.33}$$

відноситься до загальної кількості кадрів елемента.

Фрейми з низькою енергією на початку та в кінці елементів ігноруються.

- Noise-to-Masked-Ratio (NMR)

Схема вимірювання NMR (Noise-to-Masked-Ratio) оцінює [30] різницю рівнів між масованим порогом і шумовим сигналом. DFT із вікном Ханна близько

20 мс використовується для аналізу частотного вмісту сигналу. Коефіцієнти перетворення об'єднані у смуги за шкалою Барка. Маскований поріг оцінюється кожної смуги. Нахил замаскованого порога виводиться з використанням підходу для найгіршого випадку з урахуванням того факту, що нахили крутіші для слабких сигналів, але впираються в абсолютний поріг на вищих рівнях. Абсолютний поріг адаптується до роздільної здатності вхідного сигналу (зазвичай 16 біт), але не до психоакустичних вимог. Завдяки цим фактам NMR є стійким до змін рівня відтворення. Роздільна здатність шкали основного тону становить близько 1 Bark. Оскільки необхідна обчислювальна потужність невелика, NMR можна було реалізувати систему реального часу на ранній стадії її розробки.

Найбільш важливими вихідними значеннями NMR є швидкість прапора маскування, що дає відсоток кадрів з чутними спотвореннями, а також загальне та середнє значення NMR, які є різними способами усереднення відстані між енергією помилки і порогом маскування.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ СТЕГАНОГРАФІЧНОЇ СИСТЕМИ ТА ЕКСПЕРЕМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ЇЇ ЕФЕКТИВНОСТІ

3.1 Обґрунтування вибору мови програмування та середовища розробки

Для виконання поставлених завдань була створена реалізація на мові програмування Python.

В якості мови програмування був обраний Python, тому що він багато переваг на відміну від інших мов програмування. До основних переваг Python належить:

- Швидкість розробки. Для написання програм на Пітоні потрібно набагато менший обсяг коду, ніж у випадку з іншими популярними мовами (Java, C). Це помітно прискорює розробку, дозволяючи створювати складне програмне забезпечення швидше, ніж з іншими мовами програмування.
- Логічний синтаксис. Логічний синтаксис цієї мови спрощує читання та розуміння його коду
- Різноманітність бібліотек. Крім стандартної бібліотеки, для Пітона є великий вибір додаткових бібліотек. Серед найважливіших їх слід відзначити NumPy (для роботи з величезним обсягом даних), Pandas тощо.
- Масштабованість. Написані на Пітоні програми та програми легко розширюються і масштабуються завдяки можливості адаптації їх високорівневої логіки.
- Універсальність. Python – це інтерпретована мова, яка використовується для кодингу практично на всіх сучасних платформах. Він не потребує компіляції та його код можна писати у звичайному текстовому документі.

3.2 Програмна реалізація стеганографічного приховування даних в аудіофайлах із розширенням спектру методом прямої послідовності

Повідомленням для вбудовування в аудіоконтейнер був обраний текстовий файл формату txt, розміром 1 КБ (Рис. 3.1). Для вбудовування файл був

перероблений в двійковий полярний вигляд. Стеганоконтейнером був аудіофайл 8 бітного формату, розміром 345 КБ, з розширенням WAV та частотою дискретизації 48kHz (Рис. 3.2). Аудіоконтейнер представляє з себе звук запуску операційної системи Windows.

```
Interest in steganography has emerged in the last decade and is caused by the widespread use of multimedia technologies.
Steganography methods make it possible not only to transfer data covertly, but also to solve the problems of noise-resistant
authentication, protection of information from unauthorized copying, tracking the distribution of information over communication
networks, and searching for information in multimedia databases.
International symposiums on data hiding have been held since 1996, the first symposium on steganography was held in July 2002.
```

Рисунок 3.1 – Текст кодованого повідомлення

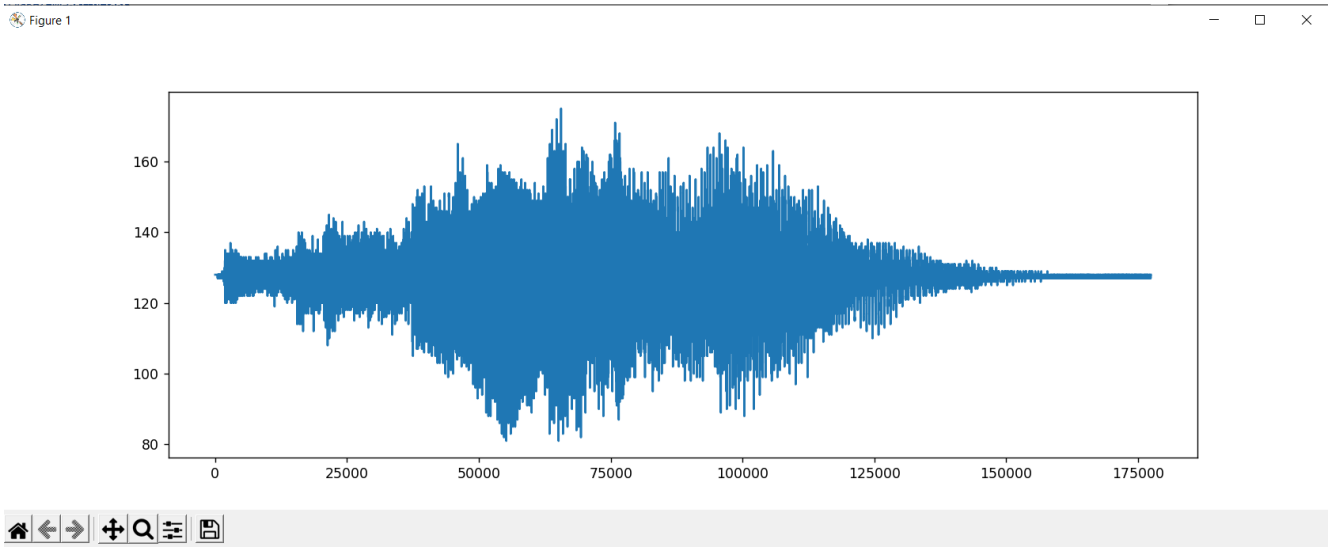

Рисунок 3.2 – Текст кодованого повідомлення

Спочатку для роботи були імпортовані модулі та бібліотеки(Рис. 3.3):

- Бібліотека scipy.io з модулем wavfile дозволяє Легко читати/записувати хвилеві аудіофайлів у списку власних типів Python.

В даний час пакет підтримує формати з плаваючою комою PCM (ціле число) та IEEE, а також підтримує довільну цілочисельну точність, включаючи: 16-, 24-, 32- та 64-бітні вибірки.

- Бібліотека NumPy та модуль random дозволяють виробляти псевдовипадкові числа, використовуючи комбінації BitGenerator для створення послідовностей та Generator для використання цих послідовностей для вибірки з різних статистичних розподілів. Також крім модулю random була імпортована уся бібліотека NumPy, бо вона пропонує комплексні математичні функції, генератори випадкових чисел, процедури лінійної алгебри, перетворення Фур'є та багато

іншого. Швидкі та універсальні концепції векторизації, індексації та широкомовної розсилки NumPy сьогодні фактично є стандартами масивних обчислень.

- Модуль threading значно полегшує роботу з потоками і дозволяє програмувати запуск декількох операцій одночасно.
- Модуль multiprocessing дозволяє створювати процеси так само, як при створенні потоків за допомогою модуля threading. Суть у тому, що у зв'язку з тим, що тепер створюються процеси, можна обійти GIL (Global Interpreter Lock) і скористатися можливістю використання кількох процесорів на комп'ютері. Пакет multiprocessing також включає ряд API, яких взагалі немає в модулі threading.
- Pyplot – це набір функцій у популярному пакеті візуалізації Matplotlib. Його функції управляють елементами фігури, такими як створення фігури, створення області побудови, побудова ліній, додавання міток на графіку і т.д.
- pandas – популярний набір інструментів для аналізу даних на основі Python. Він є широкий спектр утиліт, починаючи від синтаксичного аналізу декількох форматів файлів і закінчуючи перетворенням всієї таблиці даних матричний масив NumPy. Це робить панд надійним союзником у науці про дані та машинне навчання. Подібно до NumPy, pandas в основному працює з даними в одновимірних і двовимірних масивах; але pandas обробляє їх по-різному.
- Модуль math – один з найважливіших у Python. Цей модуль надає великий функціонал для роботи з числами.

```
1 #IMPORT
2 from scipy.io import wavfile
3 from numpy import random
4 from threading import *
5 from multiprocessing import *
6 import matplotlib.pyplot as plt
7 import pandas as pd
8 import math
9 import numpy
```

Рисунок 3.3 – Імпортовані бібліотеки

Код запуску програми виглядає наступним чином, створюється декілька глобальних змінних, перша відповідає за біти аудіоконтейнера –

audioContainerFragmetedToSegments, друга – audioContainerFragmetedToSegments, відповідно за двійковий вид текстового повідомлення.(Рис 3.4) Змінна signalsKvazi – є сховищем, для зберігання сформованих псевдовипадкових послідовностей, змінна codedMessageInAudio – зберігає модифікований аудіоконтейнер, decodedMessage – біти декодованого повідмолення.

```
312 if __name__ == "__main__":
313     audioContainerFragmetedToSegments = None
314     BinaryTxtFileWithReversedNumbers = None
315     signalsKvazi = None
316     Summary = None
317     codedMessageInAudio = None
318     decodedMessage = None
```

Рисунок 3.4 – Головні змінні

Після оголошення глобальних змінних йде мультипоточність, в результаті чого запускаються одночасно 3 функції, оскільки функції не є дуже важкими, то мультипоточність чудово справляється з цим (Рис. 3.5).

```
thread1 = Thread(target = readAudio)
thread2 = Thread(target = readTxt)
thread3 = Thread(target = creatingKvaziSignals)

thread1.start()
thread2.start()
thread3.start()
thread1.join()
thread2.join()
thread3.join()
```

Рисунок 3.5 – Зчитування аудіофайлу, повідомлення та створення псевдовипадкових послідовностей

Мультипоточність запускає наступні функції readAudio (Рис. 3.6), readTxt (Рис. 3.7), creatingKvaziSignals (Рис. 3.8).

```
def readAudio():
    #Reading Audiofile into binaries
    samplerate, dataAudio = wavfile.read('S1.wav')
    #Fragmentation 2rows massive of AudioFile To 1row Massive
    row1dataAudio = fragmentationTo1rowAudioFileMassive(dataAudio)
    #Creating and showing the grafik of AudioFile
    '''
    plt.plot(row1dataAudio)
    plt.show()
    '''
    global audioContainerFragmetedToSegments
    audioContainerFragmetedToSegments = countOfSegments(row1dataAudio)

    el = []
    for i in range(len(audioContainerFragmetedToSegments)):
        for j in range(len(audioContainerFragmetedToSegments[i])):
            el.append((audioContainerFragmetedToSegments[i][j]))


    row1dataAudio1 = numpy.array(el)
    wavfile.write('main.wav',48000,row1dataAudio1)

```

Рисунок 3.6 – Функція зчитування аудіофайлу

```
def readTxt():
    global BinaryTxtFileWithReversedNumbers
    #Reading txt file into binaries
    bytetable = [("00000000"+bin(x)[2:])[-8:] for x in range(256)]
    binrep = "".join(bytetable[x] for x in open("4.txt", "rb").read())
    #Finding the Decimal Numbers Of TxtFile
    DemicalTxtFile = convertingTo8Numbers(binrep)
    #Finding the Binary Numbers Of TxtFile
    BinaryTxtFile = tuple(binrep)
    BinaryTxtFile = convertedToDemicalList(BinaryTxtFile)
    #Converting 0 To -1 and 1 To 1
    BinaryTxtFileWithReversedNumbers = convertingFrom0To_1From1To1(BinaryTxtFile)
```

Рисунок 3.7 – Функція зчитування повідомлення

```
#Method of forming signals by normal
def creatingKvaziSignals():
    n = 1024
    global signalsKvazi
    signalsKvazi = []
    for i in range(n):
        r = random.uniform(0, 2, size=n)
        for j in range(n):
            r[j] = math.ceil(r[j]) - 1
            r[j] = -1 if r[j] == 0 else 1
        signalsKvazi.append(r)
```

Рисунок 3.8 – Функція створення чіп-кодів методом №4

Функція readAudio відповідає за зчитування аудіофайлу за допомогою модулю wavfile. Після чого двійковий масив фрагментується в одинарний масив значень, таким чином роблячи з двоканального звуку в одноканальний. Також у функції реалізується побудова графіку(Рис 3.2), та створення аудіофайлів для подальших розрахунків PEAQ(Рис. 3.6).

Функція readTxt() відповідає за зчитування текстового файлу, та переведення його у бінарний вигляд, та заміну усіх значень 0, значеннями -1, для реалізації прямого розширення спектру(Рис. 3.7).

Функція creatingKvaziSignals(), це функція яка відповідає за створення псевдовипадкових послідовностей(Рис. 3.8). Для кожного методу створення псевдовипадкових послідовностей була створена своя функція, яка має назву залежно від методу.

Після закінчення роботи потоків, повідомлення комбінується з чіп-кодами та вбудовується у аудіоконтейнер, за допомогою функції codingAndDecodingMessage. Крім того, ця функція вилучає повідомлення з модифікованого контейнера зображення(Рис. 3.9).

```
339    codingAndDecodingMessage()
```

```
255 def codingAndDecodingMessage():
256     global Summary
257     global codedMessageInAudio
258     global decodedMessage
259 
260     Summary = []
261     codedMessageInAudio = []
262     decodedMessage = []
263 
264 
265     for g in range(1,41):
266         Summary1 = []
267         codedMessageInAudio1 = []
268         decodedMessage1 = []
269 
270         for k in range(1,11):
271             Summary1.append(findingSummary(g,k, signalsKvazi, audioContainerFragmetedToSegments, BinaryTxtFileWithReversedNumbers))
272 
273             codedMessageInAudio1.append(combinationOfSignalsAndMessage(audioContainerFragmetedToSegments,Summary1[k-1]))
274 
275             decodedMessage1.append(decodingOfMessage(k,audioContainerFragmetedToSegments,codedMessageInAudio1[k-1],signalsKvazi))
276 
277             fromSegmentsToFullSignal(codedMessageInAudio1[k-1],g,k)
278 
279         Summary.append(Summary1)
280         codedMessageInAudio.append(codedMessageInAudio1)
281         decodedMessage.append(decodedMessage1)
```

Рисунок 3.9 – Функція кодування та декодування повідомлення

Всередині функції відбувається заповнення глобальної змінної Summary, яка зберігає комбінацію повідомлення та псевдовипадкової послідовності, а також формується модифікований аудіоконтейнер, з якого декодується повідомлення (Рис. 3.10).

```
#Function for finding Summary of BinnTxtFile and Signals
def findingSummary(G,K,signals42Method1, audioContainerFragmetedToSegments1, BinaryTxtFileWithReversedNumbers1):
    Sum = []
    for i in range(len(audioContainerFragmetedToSegments1)):
        g1 = G/4
        x = 0
        for j in range(K):
            x += g1*(BinaryTxtFileWithReversedNumbers1[K*i+j]*signals42Method1[j+1])
        Sum.append(x)
    return Sum
#Combination Signals with Coded Message
def combinationOfSignalsAndMessage(audioContainerFragmetedToSegments1, Sum):
    combined2Massives = []
    for i in range(len(audioContainerFragmetedToSegments1)):
        combined2Massives.append(audioContainerFragmetedToSegments1[i]+Sum[i])
    return combined2Massives


#Decoding Message From AudioFile
def decodingOfMessage(K,audioContainerFragmetedToSegments1,codedMessageInAudio1,signals42Method1):
    m1 = []
    for i in range(len(audioContainerFragmetedToSegments1)):
        for j in range(K):
            ro = multForCoefficientOfCorrelation(codedMessageInAudio1[i], signals42Method1[j+1])
            if ro > 0:
                m1.insert(K*i+j, 1)
            else:
                m1.insert(K*i+j, -1)
    return m1
```

Рисунок 3.10 – Функції об’єднання повідомлення та чіп-кодів, а також кодування та декодування повідомлення

Обчислення головних показників для оцінки якості вбудовування відбувається за допомогою мультипоточності(Рис. 3.11). Кожен потік відповідає за виклик функції, яка розраховує свої показники, це зроблено для покращення швидкості роботи програми. Таким чином обчислюються показники BER(Рис. 3.12), MSE, PSNR(Рис. 3.13), SNR(Рис. 3.14).

```
    allBer = Array('f', range(40*10))
    allMSE = Array('f', range(40*10))
    allPSNR = Array('f', range(40*10))
    allSNR = Array('f', range(40*10))

    process1 = Process(target=compilingBER, args=(allBer,BinaryTxtFileWithReversedNumbers,decodedMessage))
    process2 = Process(target=compilingMSE_PSNR, args=(allMSE,allPSNR,audioContainerFragmetedToSegments, codedMessageInAudio))
    process3 = Process(target=compilingSNR, args=(allSNR,audioContainerFragmetedToSegments,Summary))

    process1.start()
    process2.start()
    process3.start()

    process1.join()
    process2.join()
    process3.join()
```

Рисунок 3.11 – Знаходження головних показників за допомогою мультипоточності

```
def findingBER(m1,BinaryTxtFileWithReversedNumbers1,BER1):
    A = 0
    for i in range(len(m1)):
        if m1[i] != BinaryTxtFileWithReversedNumbers1[i]:
            A += 1
    BER1.append(A/len(m1))
    return BER1
```

Рисунок 3.12 – Функція знаходження BER

```
#Finding MSE and PSNR
def findingMSE_PSNR(Seg1,Seg,MSE_PSNR1):
    A = 0
    for i in range(len(Seg)):
        for j in range(1024):
            A = A + (Seg[i][j]-Seg1[i][j])**2
    B = A/(len(Seg1)*1024)
    MSE_PSNR1[0].append(B)
    MSE_PSNR1[1].append(20*(math.log(255,10))-10*(math.log(B,10)))

    return MSE PSNR1
```

Рисунок 3.13 – Функція знаходження MSE та PSNR

```
def findingSNR(Sum1,Seg,SNR1):
    A = 0
    B = 0.0

    for i in range(len(Seg)):
        for j in range(1024):
            A = A + (Sum1[i][j])**2
            B = B + (Seg[i][j])**2

    SNR1.append(20*math.log(math.sqrt(A/B),10))

    return SNR1
```

Рисунок 3.14 – Функція знаходження SNR

Після виконання усіх головних функцій, формуються xlsx файли з результатами випробувань при різних показниках g – потужності сигналу та k – кількості вбудованих біт(Рис. 3.15).

```
    g = 40
    k = 10

    creatingFileOfBER(allBer,g,k)
    creatingFileOfMSE(allMSE,g,k)
    creatingFileOfPSNR(allPSNR,g,k)
    creatingFileOfSNR(allSNR,g,k)
```

Рисунок 3.15 – Створення файлів розрахунків основних показників

Кожна функція: creatingFileOfBER(Рис. 3.16), creatingFileOfMSE(Рис. 3.17), creatingFileOfPSNR(Рис. 3.18), creatingFileOfSNR(Рис. 3.19), створює файли xlsx, для різних показників якості вбудовування повідомлення.

```
def creatingFileOfBER(BER1,G,K):
    slovar = {}
    b = list(range(1,K+1))
    slovar[0] = b
    for i in range(G):
        a = []
        for j in range(K):
           a.append(BER1[K*i+j])
        slovar[(i+1)/4] = a

    df = pd.DataFrame(slovar)

    df.to_excel('BER.xlsx', index=False)
```

Рисунок 3.16 – Створення файлу показника BER

```
#Creating MSE.xlsx file
def creatingFileOfMSE(MSE1,G,K):
    slovar = {}
    b = list(range(1,K+1))
    slovar[0] = b
    for i in range(G):
        a = []
        for j in range(K):
           a.append(MSE1[K*i+j])
        slovar[(i+1)/4] = a

    df = pd.DataFrame(slovar)

    df.to_excel('MSE.xlsx', index=False)
```

Рисунок 3.17 – Створення файлу показника MSE

```
#Creating PSNR.xlsx file
def creatingFileOfPSNR(PSNR1,G,K):
    slovar = {}
    b = list(range(1,K+1))
    slovar[0] = b
    for i in range(G):
        a = []
        for j in range(K):
           a.append(PSNR1[K*i+j])
        slovar[(i+1)/4] = a

    df = pd.DataFrame(slovar)

    df.to_excel('PSNR.xlsx', index=False)
```

Рисунок 3.18 – Створення файлу показника PSNR

```
#Finding SNR and Creating it .xlsx file
def findingSNR(Sum1,Seg,SNR1):
    A = 0
    B = 0.0

    for i in range(len(Seg)):
        for j in range(1024):
            A = A + (Sum1[i][j])**2
            B = B + (Seg[i][j])**2

    SNR1.append(20*math.log(math.sqrt(A/B),10))

    return SNR1
```

Рисунок 3.19 – Створення файлу показника SNR

Крім того для наведених обчислень та виконання показаних головних функцій, було написано багато додаткових функцій. До таких функцій відноситься функція перекладу числа з десяткової системи числення у двійкову(Рис. 3.20)

```
12 #Function for Binary to Decimal
13 def binaryToDecimal(binary):
14     binary1 = binary
15     decimal, i, n = 0, 0, 0
16     while(binary != 0):
17         dec = binary % 10
18         decimal = decimal + dec * pow(2, i)
19         binary = binary//10
20         i += 1
21     return decimal
```

Рисунок 3.20 – Функція перекладу числа з десяткової системи числення у двійкову

Була написана додаткова функція для розбиття на 8 бітні форми повідомлень(Рис. 3.21) та розрахунку коефіцієнту кореляції (Рис. 3.22).

```
23 #Function to fragmentation Full Binary Code To Decimal Numbers
24 def convertingTo8Numbers(binrep1):
25     i = 0
26     a = []
27     while i<len(binrep1):
28         string = ''
29         for j in range(8):
30             string += binrep1[i+j]
31         i += 8
32         a.append(string)
33     return convertedToDemicalList(a)
```

Рисунок 3.21 – Функція повідомлень розбиття на 8 бітні форми

```
86 #Coefficient of correlation
87 def multForCoefficientOfCorrelation(a,b):
88     x = 0
89     x = x + numpy.dot(a,b)
90     return x
```

Рисунок 3.22 – Функція розрахунку коефіцієнту кореляції

Не слід забувати про дуже важливу функцію, яка реалізує розбиття аудіоконтейнеру на сегменти(Рис. 3.23)

```
#Function to fragmentate audiocontainer to segments
def countOfSegments(row1Audio):
    n0 = 0
    n = 1024
    segments = math.ceil(len(row1Audio)/n)-2
    Segments = []
    for i in range(segments):
        Segments.append([])
    for i in range(segments):
        for j in range(n0,n):
            Segments[i].append(row1Audio[j])
            a = 1
        n += 1024
        n0 += 1024
    return Segments
```

Рисунок 3.23 – Функція розрахунку коефіцієнту кореляції

Як вже було зазначено для кожного методу формування чіп-кодів, була створена своя функція. Так функція для створення квазіортогональних дискретних сигналів виглядає, як показано на Рисунку 3.8. Для реалізації чіп-коду на основі рівномірно розподілених випадкових чисел на інтервалі від -1 до 1, був застосований модуль random та метод uniform (Рис. 3.24).

```
#Method of forming signals by uniform
def creatingSignals42Method():
    n = 1024
    global signals42Method
    signals42Method = []
    for i in range(n):
        r = random.uniform(0, 2, size=n)
        signals42Method.append(r-1)
```

Рисунок 3.24 – Функція створення чіп-кодів методом №3

Для реалізації методу формування сигналів на основі матриці Уолша Адамара, достатньо було скористатися бібліотекою scipy там методом hadamard, де було достатньо вказати розмірність матриці (Рис. 3.25).

```
def creatingSignalsWA():
    n = 1024
    global signalsWA

    signalsWA = hadamard(n)
```

Рисунок 3.25 – Функція створення чіп-кодів методом №1

Метод, який використовує нелінійну модуляцію для формування сигналів потребував змін функцій кодування, а також декодування повідомлення. Це зображено на Рисунках 3.26 – 3.28.

```
def creatingSignals41Method():
    n = 1024
    global signals41Method
    signals41Method = [[],[]]
    for i in range(n):
        s = []
        s1 = []
        for j in range(n):
            r = random.uniform(0,1)
            q = norm.ppf(r, loc=0, scale=1)
            s.append(q)
            if r < 0.5:
                r1 = r + 0.5
            else:
                r1 = r - 0.5
            q1 = norm.ppf(r1, loc=0, scale=1)
            s1.append(q1)
        signals41Method[0].append(numpy.array(s))
        signals41Method[1].append(numpy.array(s1))
    signals41Method = numpy.array(signals41Method)
```

Рисунок 3.26 – Функція створення чіп-кодів методом №2

```
#Function for finding Summary of BinnTxtFile and Signals
def findingSummary(G,K,signals41Method1, audioContainerFragmetedToSegments1, BinaryTxtFileWithReversedNumbers1):
    Sum = []
    for i in range(len(audioContainerFragmetedToSegments1)):
        g1 = G
        x = 0
        for j in range(K):
            if BinaryTxtFileWithReversedNumbers1[K*i+j] == -1:
                x += g1*(signals41Method1[0][j])
            else:
                x += g1*(signals41Method1[1][j])
        Sum.append(x)
    return Sum
```

Рисунок 3.27 – Функція об’єднання повідомлення та чіп-кодів методу №2

```
#Decoding Message From AudioFile
def decodingOfMessage(K,audioContainerFragmetedToSegments1,codedMessageInAudio1,signals41Method1):
    m1 = []
    for i in range(len(audioContainerFragmetedToSegments1)):
        for j in range(K):
            a = multForCoefficientOfCorrelation(codedMessageInAudio1[i], signals41Method1[0][j+1])
            b = multForCoefficientOfCorrelation(codedMessageInAudio1[i], signals41Method1[1][j+1])
            if a > b:
                m1.insert(K*i+j, -1)
            else:
                m1.insert(K*i+j, 1)
    return m1
```

Рисунок 3.28 – Функція декодування повідомлення для чіп-кодів методу №2

На Рисунку 3.29 зображена функція створення адаптивних квазіортогональних дискретних сигналів, в якій сигнали формуються з урахування обмеження.

```
def creatingAdaptiveKvaziSignals(audioContainerFragmetedToSegments1):
    n = 1024
    global adaptiveKvaziSignals
    adaptiveKvaziSignals = []
    i = 0
    while i < n:
        r = random.uniform(0, 2, size=n)
        for j in range(n):
            r[j] = math.ceil(r[j]) - 1
            r[j] = -1 if r[j] == 0 else 1
        b = True
        for j in range(len(audioContainerFragmetedToSegments1)):
            a = multForCoefficientOfCorrelation(audioContainerFragmetedToSegments1[j], r)
            if abs(a)>1000:
                b = False
                break
        if b:
            i += 1
            adaptiveKvaziSignals.append(r)
```

Рисунок 3.29 – Функція створення чіп-кодів методом №5

Так як у роботі досліджувалися два методи вбудовування повідомлення в аудіо стеганоконтейнер, то для методу адресації, були написані додаткові функції створення, з бітів повідомлення, так званих адрес (Рис. 3.30). А також змін притерпіли функція вбудовування (Рис. 3.31) та вилучення повідомлення (Рис. 3.32).

```
def creatingAdress(K, audioContainerFragmetedToSegments1, BinaryTxtFileWithReversedNumbers1):
    adress = []
    for i in range(len(audioContainerFragmetedToSegments1)):
        a = 0
        for j in range(K):
            a += BinaryTxtFileWithReversedNumbers1[K*i+j]*(2**j)
        adress.append(a)
    return adress
```

Рисунок 3.30 – Функція створення адрес

```
def combinatingSignalsAndAdress(G,adaptiveKvaziSignals1, audioContainerFragmetedToSegments1,adress):
    Sum1 = []
    combined2Massives = []
    for i in range(len(audioContainerFragmetedToSegments1)):
        g1 = G/4
        Sum1.append(g1*adaptiveKvaziSignals1[adress[i]])
        combined2Massives.append(audioContainerFragmetedToSegments1[i]+Sum1[i])
    return Sum1, combined2Massives
```

Рисунок 3.31 – Функція комбінування адрес та чіп-кодів

```
def takingAdressesFromCodedMessage(codedMessageInAudio1, adaptiveKvaziSignals1, audioContainerFragmetedToSegments1):
    adress1 = []
    for i in range(len(audioContainerFragmetedToSegments1)):
        ad = 0
        a = 0
        for j in range(1024):
            b = multForCoefficientOfCorrelation(codedMessageInAudio1[i],adaptiveKvaziSignals1[j])
            if b > a:
                a = b
                ad = j
        adress1.append(ad)
    return adress1

def creatingMessageFromTAdresses(K, audioContainerFragmetedToSegments1, adress1):
    m1 = []
    for i in range(len(audioContainerFragmetedToSegments1)):
        x = adress1[i]
        for j in range(K):
            a = x%2
            m1.insert(K*i+j, a)
            x = math.floor(x/2)
    return m1
```

Рисунок 3.32 – Функції вилучення адрес та створення декодованого повідомлення

Для дослідження аудіоконтейнерів методом PEAQ використовувався код з Додатку А. Для більш точних порівнянь результати також були порівняні зі звичайним методом LSB для аудіоконтейнерів.

Реалізація методу LSB зображена на Рисунках 3.33 – 3.34.

```
def encodeLSB():
        print("\nEncoding Starts..")
        audio = wave.open("S_48000.wav",mode="rb")
        frame_bytes = bytearray(list(audio.readframes(audio.getnframes())))
        frame_bytes_copy = frame_bytes.copy()
        f = open('4_eng.txt')
        string = f.read()
        print(len(frame_bytes))
        string = string + int((len(frame_bytes)-(len(string)*8*8))/8) *'#'
        bits = list(map(int, ''.join([bin(ord(i)).lstrip('0b').rjust(8,'0') for i in string])))
        for i, bit in enumerate(bits):
            frame_bytes[i] = (frame_bytes[i] & 255) | bit
        frame_modified = bytes(frame_bytes)
        for i in range(0,10):
                print(frame_bytes[i])
        newAudio =  wave.open('S_LSB.wav', 'wb')
        newAudio.setparams(audio.getparams())
        newAudio.writeframes(frame_modified)

        newAudio.close()
        audio.close()
        return bits
```

Рисунок 3.33 – Функція кодування повідомлення методом LSB

```
def decodeLSB():
        print("\nDecoding Starts..")
        audio = wave.open("S_LSB.wav", mode='rb')
        frame_bytes = bytearray(list(audio.readframes(audio.getnframes())))
        extracted = [frame_bytes[i] & 1 for i in range(len(frame_bytes))]
        string = "".join(chr(int("".join(map(str,extracted[i:i+8])),2)) for i in range(0,len(extracted),8))
        decoded = string.split("###")[0]
        print("Sucessfully decoded: "+decoded)
        audio.close()
        return extracted
```

Рисунок 3.34 – Функція декодування повідомлення методом LSB

3.3 Експериментальні дослідження ефективності запропонованої стеганографічної системи

Усі вище зазначені методи були порівняні за основними критеріями: BER, MSE, PSNR та SNR. Також для порівняння аудіоконтейнерів був використаний метод PEAQ, так як PEAQ використовує 13 критеріїв для своїх тестів, з них було виділено 4 головних.

Усі графіки були побудовані при різних показниках потужності сигналу, тобто показника g та кількості одночасно вписаних біт – k. Показник g змінювався від 0.25 до 10 з кроком 0.25 (в деяких випадках, аби показати взагалі спроможність чіп-коду працювати був взятий більший діапазон та крок), показник k змінювався від 0 до 10 з кроком 1.

3.3.1 Спосіб вбудовування повідомлення методом прямої послідовності на основі робот Лізи Марвел

Наведені далі графіки є результатами комбінування методу Лізи Марвел з різними методами створення псевдовипадкових послідовностей .

3.3.1.1 Метод формування сигналів №1

Метод № 1, заснований на сигналах Уолша Адамара, були отримані наступні результати: дуже низький показник BER(Рис. 3.35), дуже високі показники MSE(Рис. 3.36) та SNR(Рис. 3.38), та занадто низький показник PSNR(Рис. 3.37). Головною проблемою методу є високі викривлення контейнера, але також слід зазначити про найнижчі ушкоджень повідомлення при декодуванні.

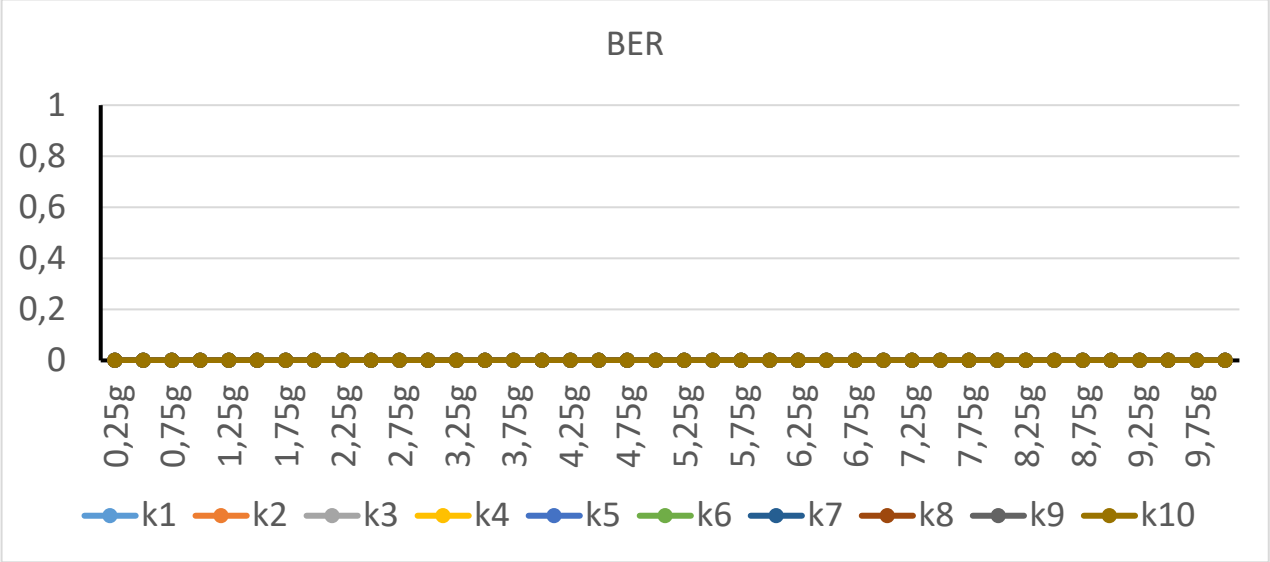

Рисунок 3.35 –Показник BER для методу формування сигналів №1 при різних параметрах k

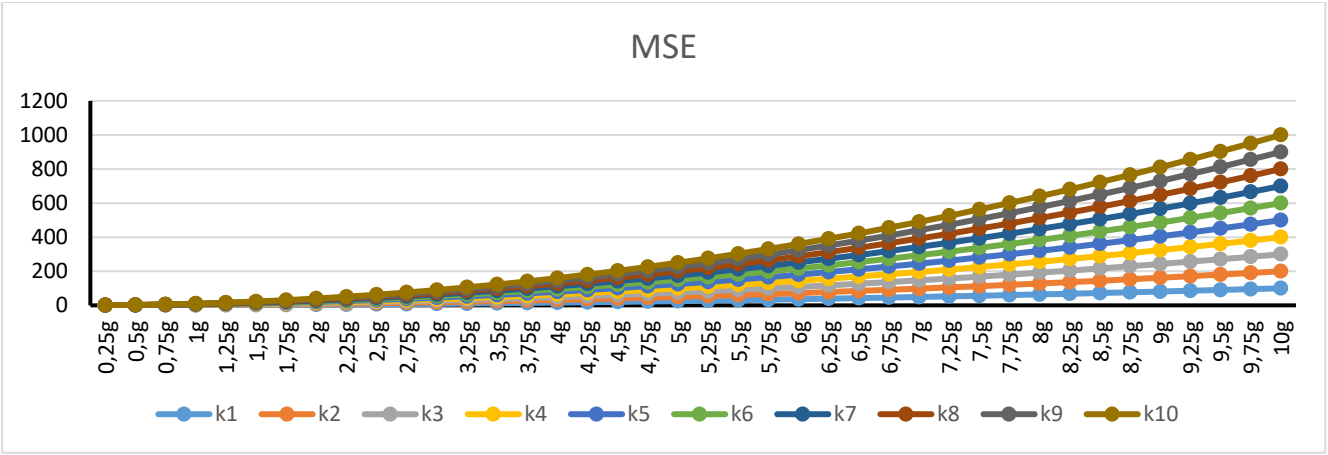

Рисунок 3.36 –Показник MSE для методу формування сигналів №1 при різних параметрах k

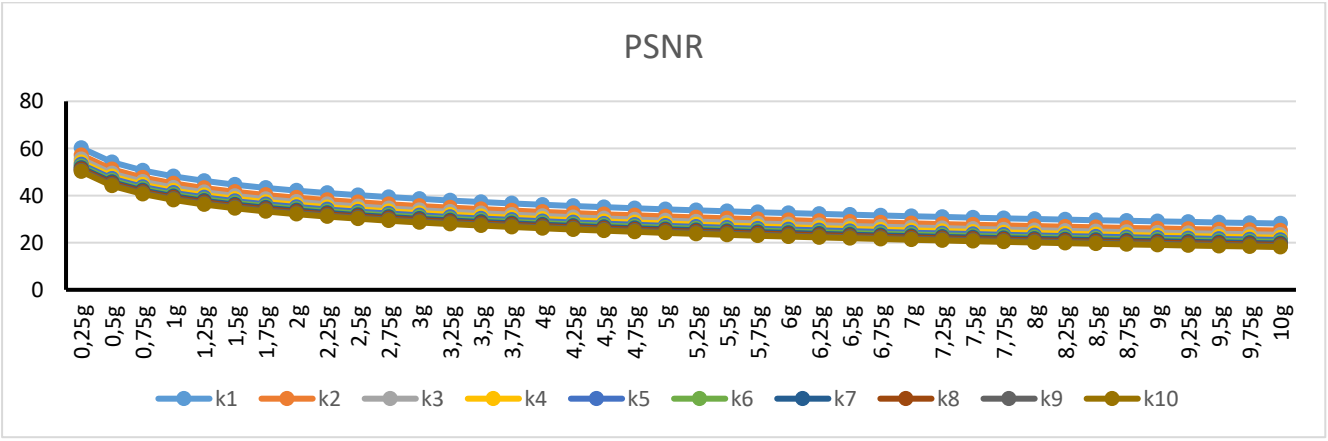

Рисунок 3.37 –Показник PSNR для методу формування сигналів №1 при різних параметрах k

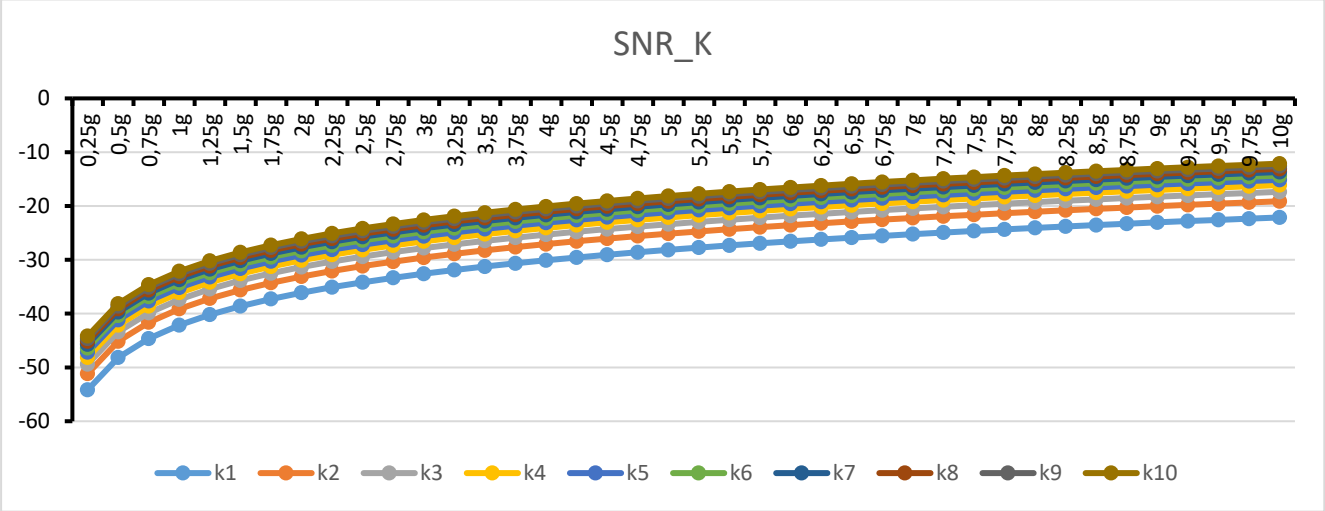

Рисунок 3.38 –Показник SNR для методу формування сигналів №1 при різних параметрах k

Що стосується дослідження методом PEAQ, слід зазначити, про гарні результати в усіх 4х показниках, тому різниця між початковим контейнером та вбудованим слабо помітна, це наведено у Додатку Б (див. Рис. Б.1 – Б.4).

3.3.1.2 Метод формування сигналів №2

Метод № 2, заснований на методі модуляції сигналів, були отримані наступні результати: дуже високий показник BER(Рис. 3.39), дуже низькі показники MSE(Рис. 3.40) та SNR(Рис. 3.42), та високий показник PSNR(Рис. 3.41). Але через занадто високі помилки при вилученні повідомлення, для отримання декодованого тексту потрібна занадто висока потужність сигналів, що негативно позначається на якості контейнера та погіршою якість звуку.

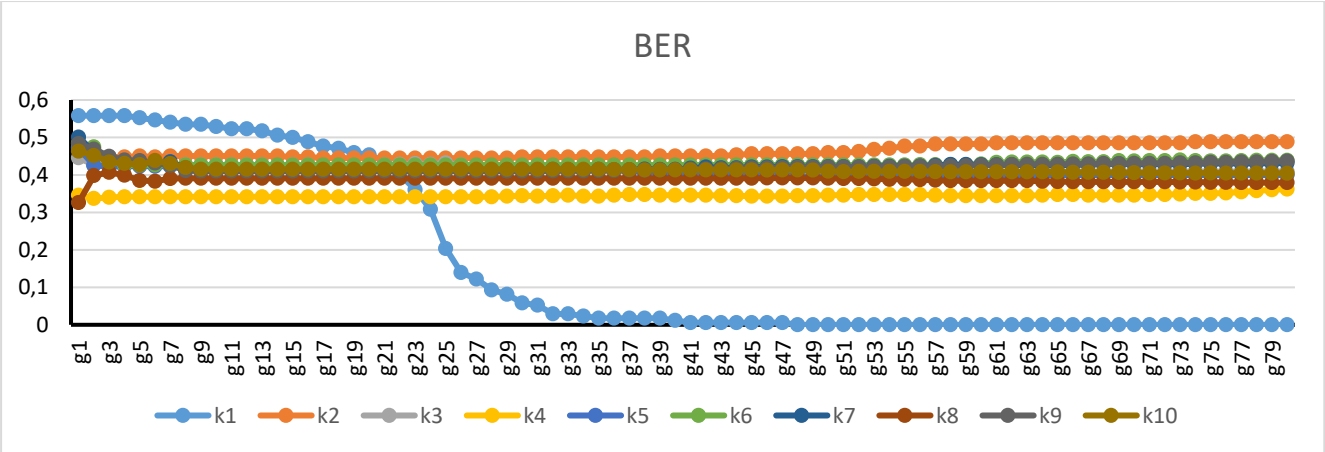

Рисунок 3.39 –Показник BER для методу формування сигналів №2 при різних параметрах k

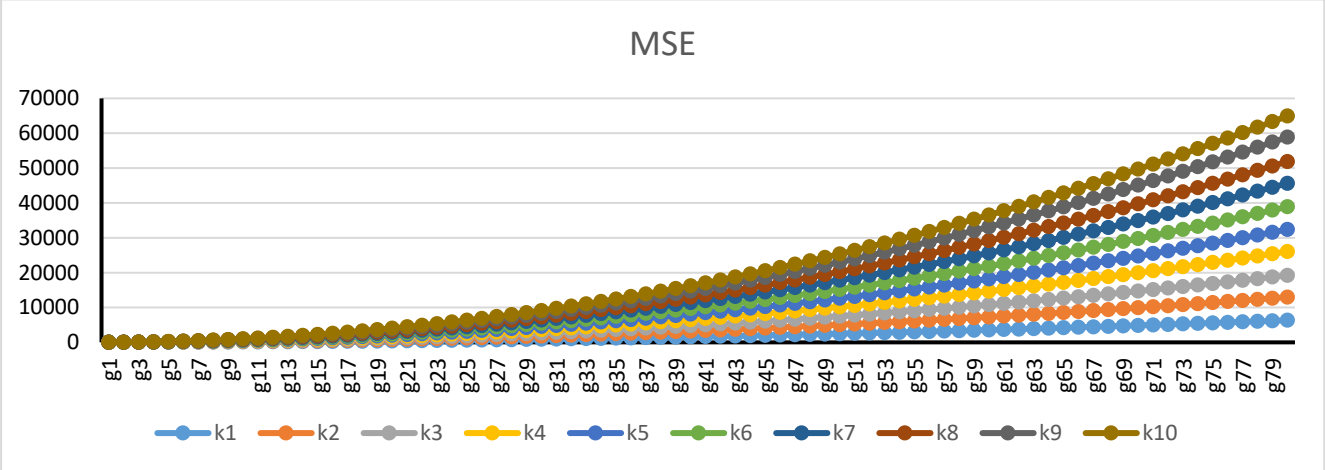

Рисунок 3.40 –Показник MSE для методу формування сигналів №2 при різних параметрах k

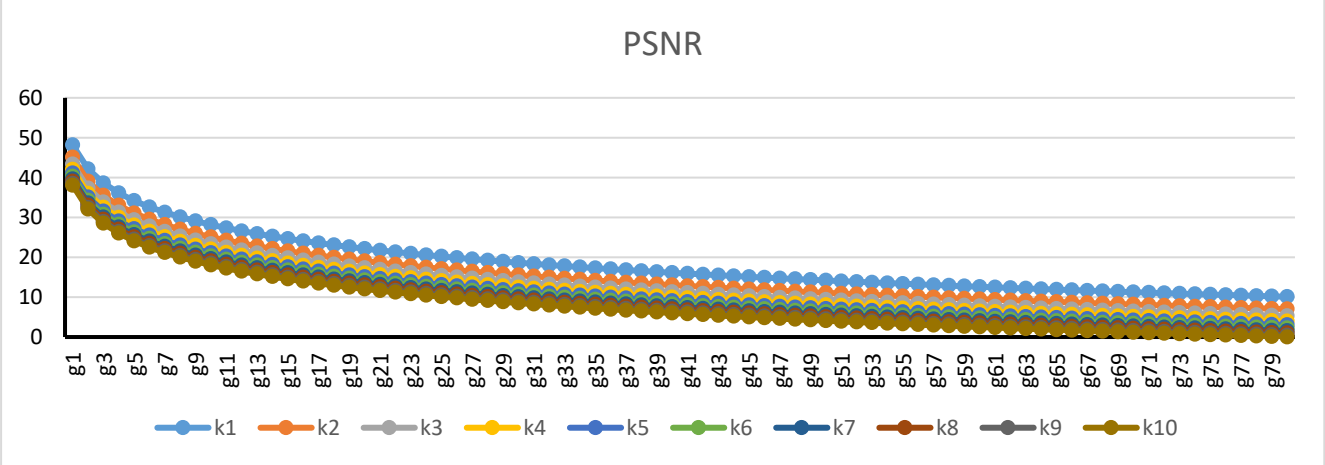

Рисунок 3.41 –Показник PSNR для методу формування сигналів №2 при різних параметрах k

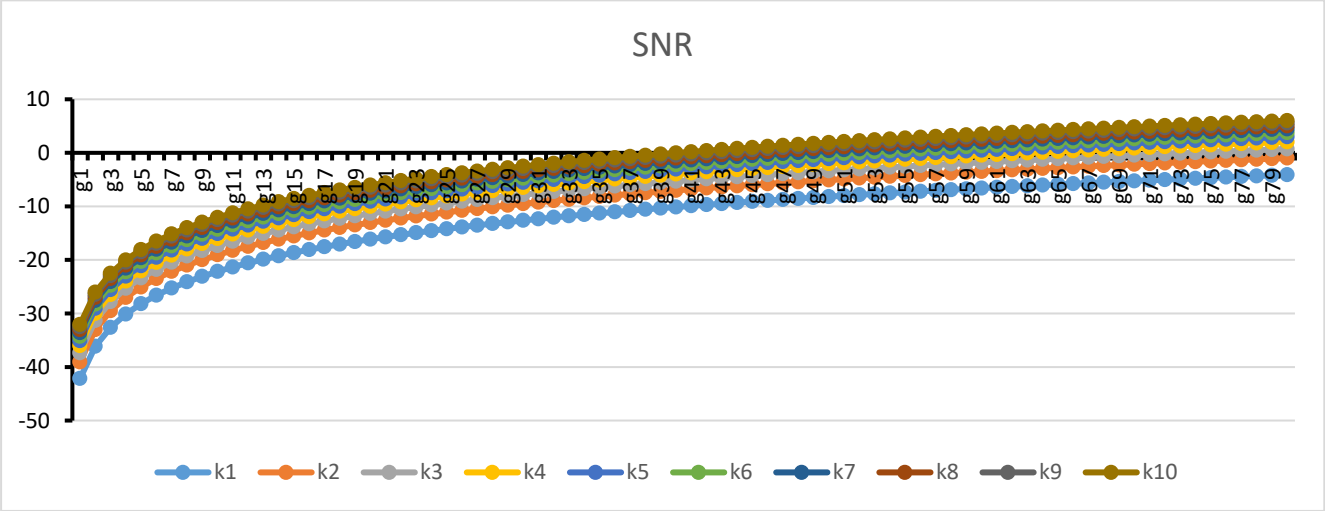

Рисунок 3.42 –Показник SNR для методу формування сигналів №2 при різних параметрах k

Що стосується дослідження методом PEAQ, слід зазначити, про дуже низький ODG та високі показники ADB, Relative Disturbed Frames, та NMR, тому різниця сильно відчувається між початковим контейнером та модифікованим контейнером, це наведено у Додатку Б (див. Рис. Б.5 – Б.8).

3.3.1.3 Метод формування сигналів №3

Метод № 3, на основі випадкових чисел на інтервалі від -1 до 1, були отримані наступні результати: досить низький показник BER(Рис. 3.43), середній показник MSE(Рис. 3.44), низький SNR(Рис. 3.46) та достатньо високий показник PSNR(Рис. 3.45). В порівнянні з минулими двома методами, метод виділяється середніми показниками по усім критеріям оцінки, тобто при вилученні повідомлення досить низька кількість помилок, при достатній потужності сигналу, а також викривлення контейнера залишаються досить низькими.

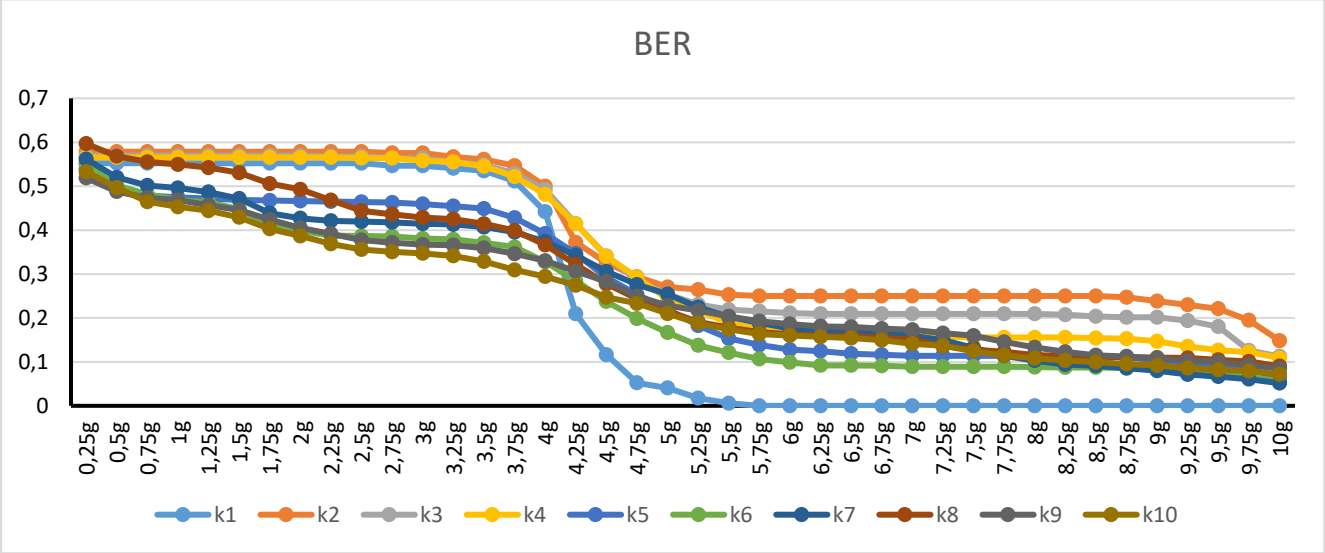

Рисунок 3.43 –Показник BER для методу формування сигналів №3 при різних параметрах k

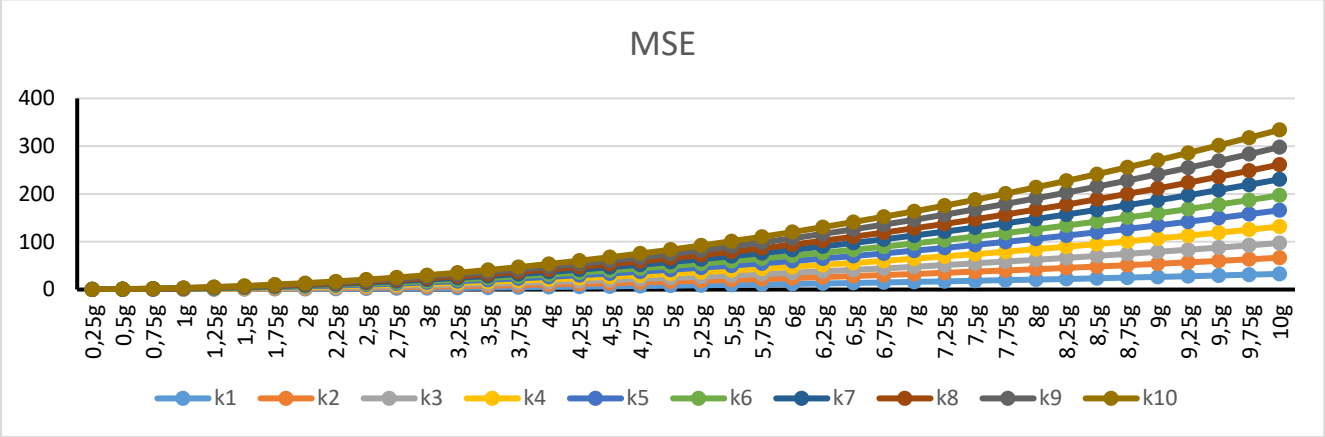

Рисунок 3.44 –Показник MSE для методу формування сигналів №3 при різних параметрах k

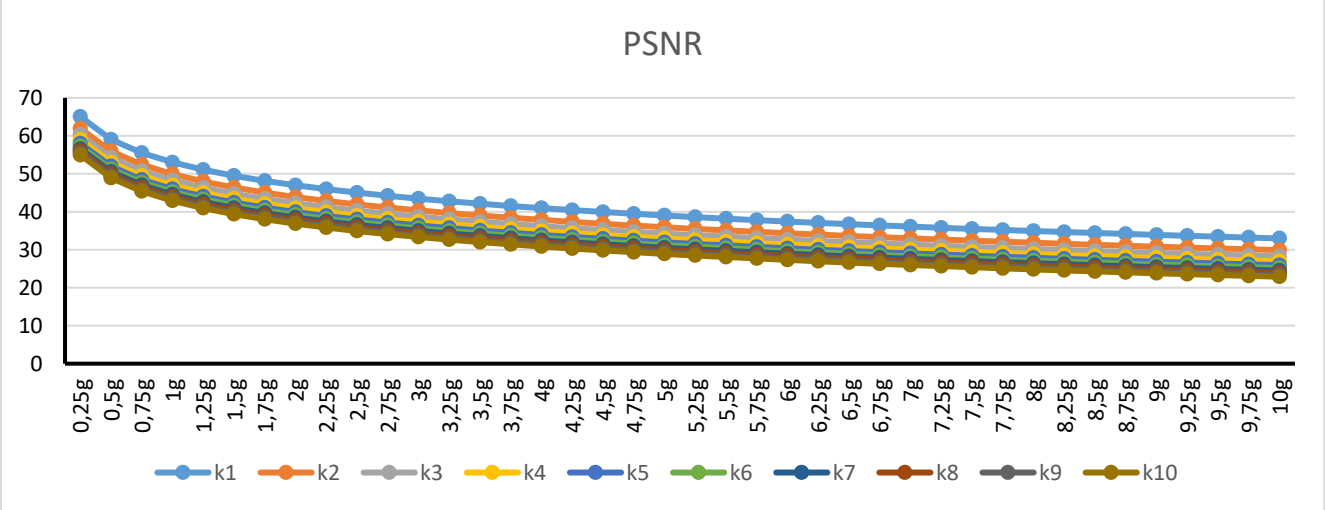

Рисунок 3.45 –Показник PSNR для методу формування сигналів №3 при різних параметрах k

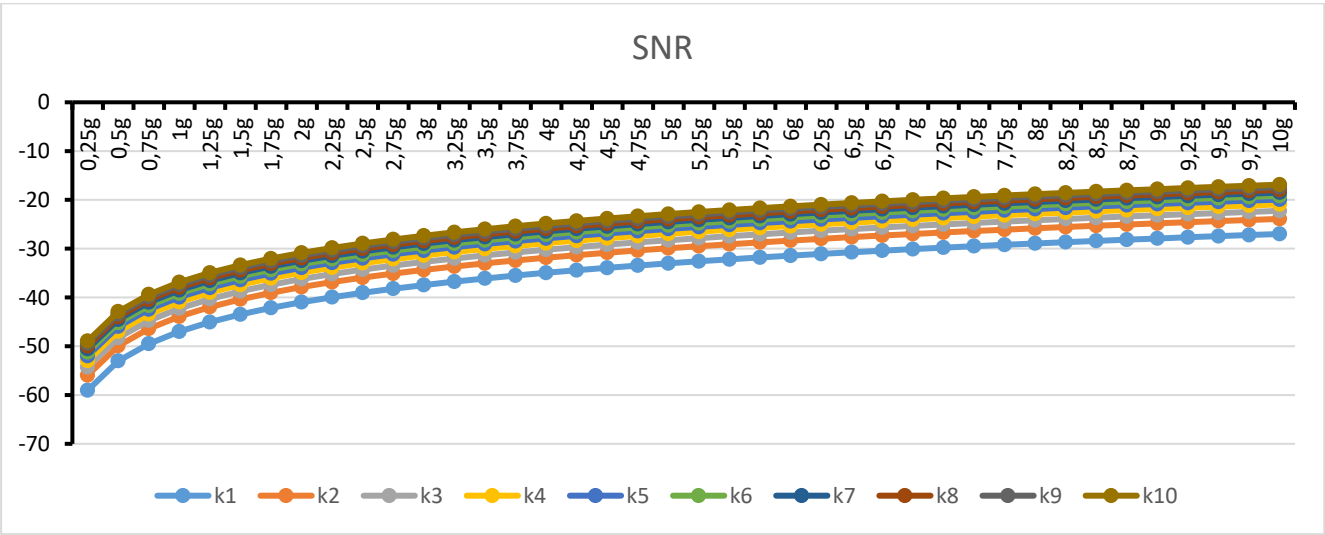

Рисунок 3.46 –Показник SNR для методу формування сигналів №3 при різних параметрах k

Що стосується дослідження методом PEAQ, слід зазначити, про дуже низький ODG та високі показники ADB, Relative Disturbed Frames, та NMR, тому різниця сильно відчувається між початковим контейнером та модифікованим контейнером , це наведено у Додатку Б (див. Рис. Б.9 – Б.12).

3.3.1.4 Метод формування сигналів №4

Метод № 4, на основі квазіортогональних дискретних сигналів, були отримані наступні результати: досить низький показник BER(Рис. 3.47), середній показник MSE(Рис. 3.48) та SNR(Рис. 3.50) та достатньо високий показник PSNR(Рис. 3.49). В порівнянні з минулими двома методами, метод виділяється середніми показниками по усім критеріям оцінки, тобто при вилученні повідомлення досить низька кількість помилок, при достатній потужності сигналу, а також викривлення контейнера залишаються досить низькими.

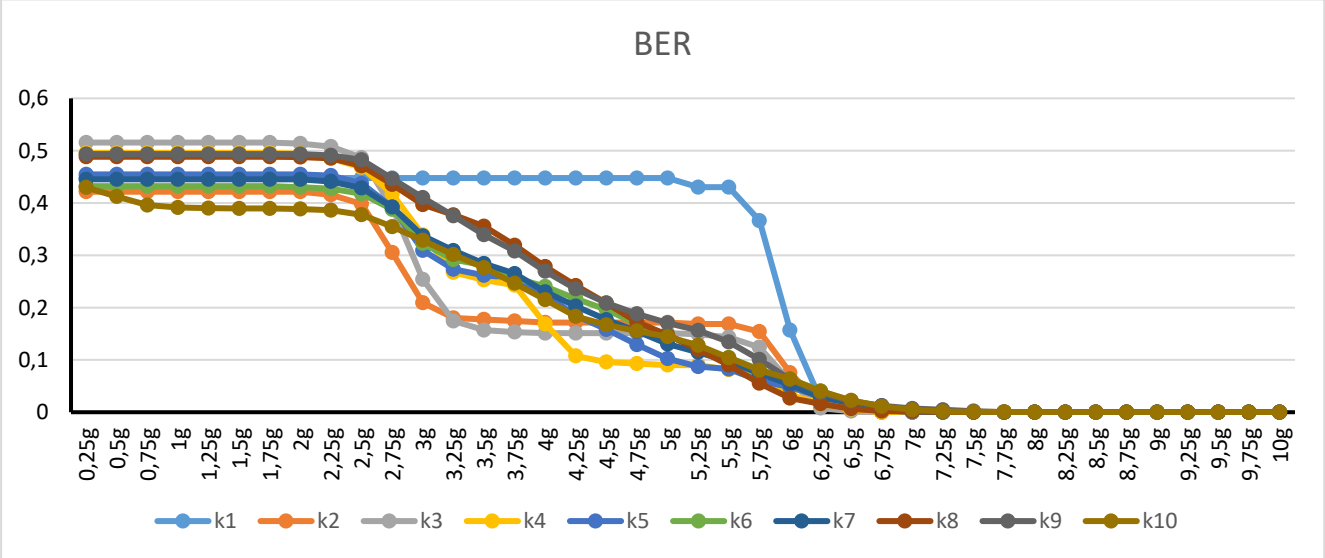

Рисунок 3.47 –Показник BER для методу формування сигналів №4 при різних параметрах k

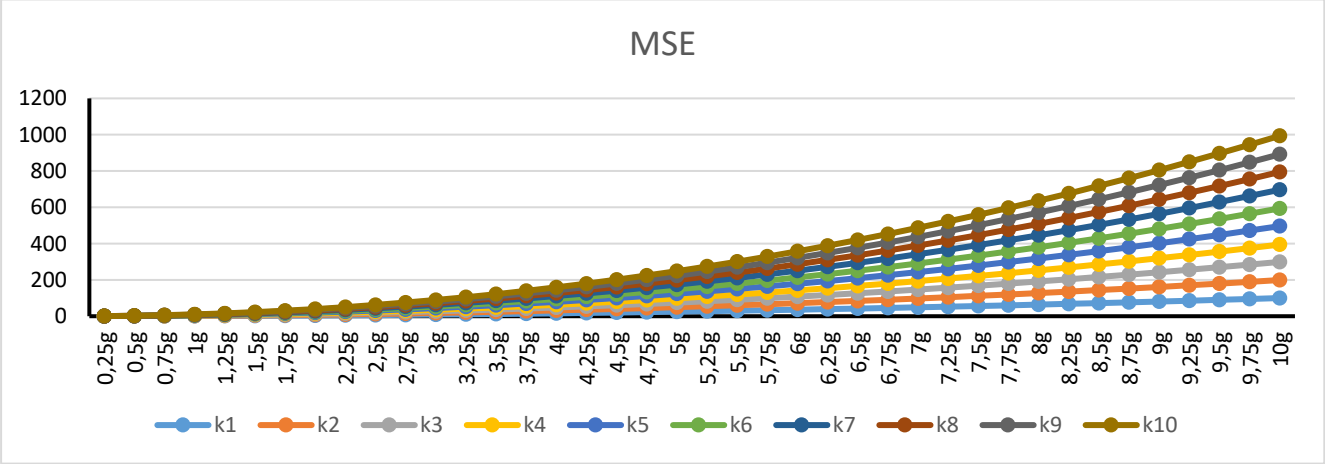

Рисунок 3.48 –Показник MSE для методу формування сигналів №4 при різних параметрах k

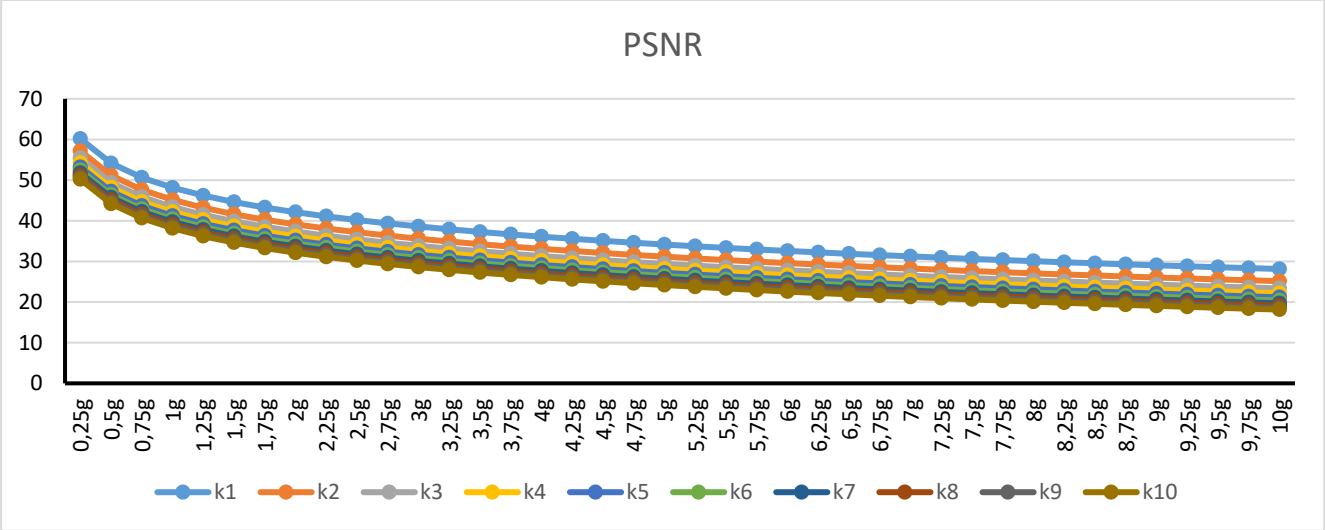

Рисунок 3.49 –Показник PSNR для методу формування сигналів №4 при різних параметрах k

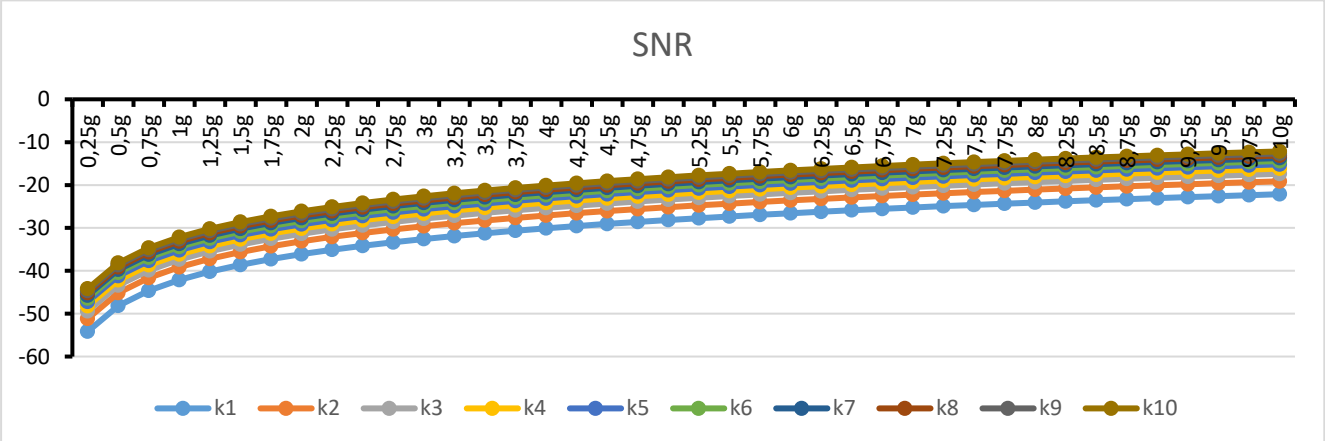

Рисунок 3.50 –Показник SNR для методу формування сигналів №4 при різних параметрах k

Що стосується дослідження методом PEAQ, слід зазначити, про дуже низький ODG та високі показники ADB, Relative Disturbed Frames, та NMR, тому різниця сильно відчувається між початковим контейнером та модифікованим контейнером , це наведено у Додатку Б (див. Рис. Б.13 – Б.16).

3.3.1.5 Метод формування сигналів №5

Метод № 5, заснований на адаптивних квазіортогональних дискретних сигналах, були отримані наступні результати: дуже низький показник BER(Рис. 3.51), показник MSE(Рис. 3.52) гірший за методи №2 та №3, але за рахунок низької потреби у підвищенні потужності сигналів MSE залишається на дуже низькому рівні, як і SNR(Рис. 3.54), та досить високий показник PSNR(Рис. 3.53).

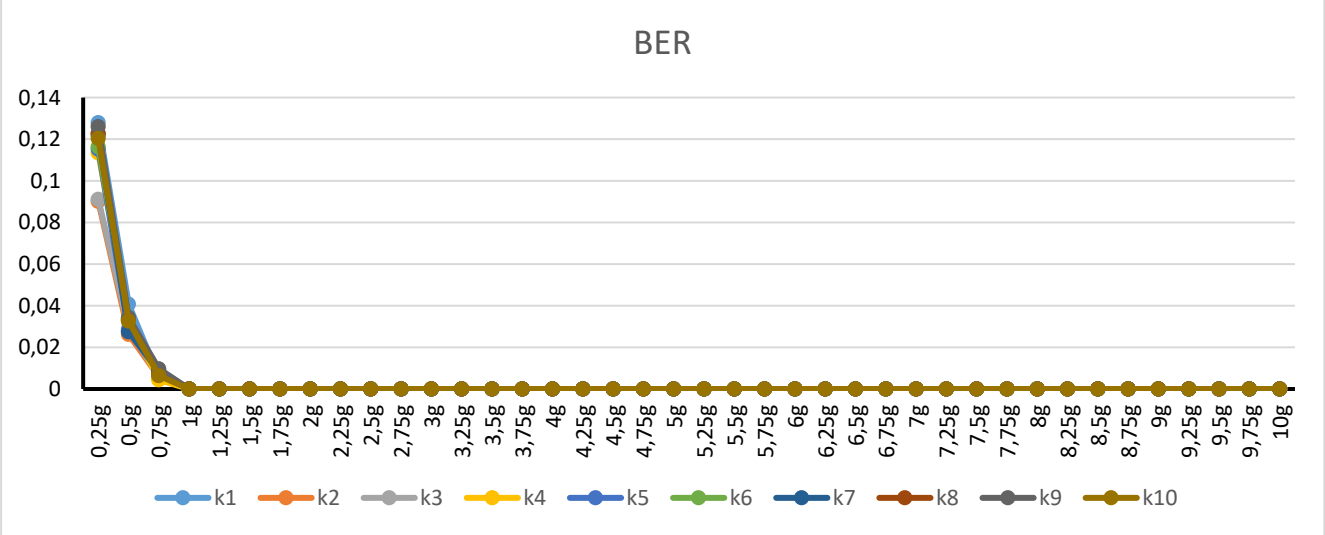

Рисунок 3.51 –Показник BER для методу формування сигналів №5 при різних параметрах k

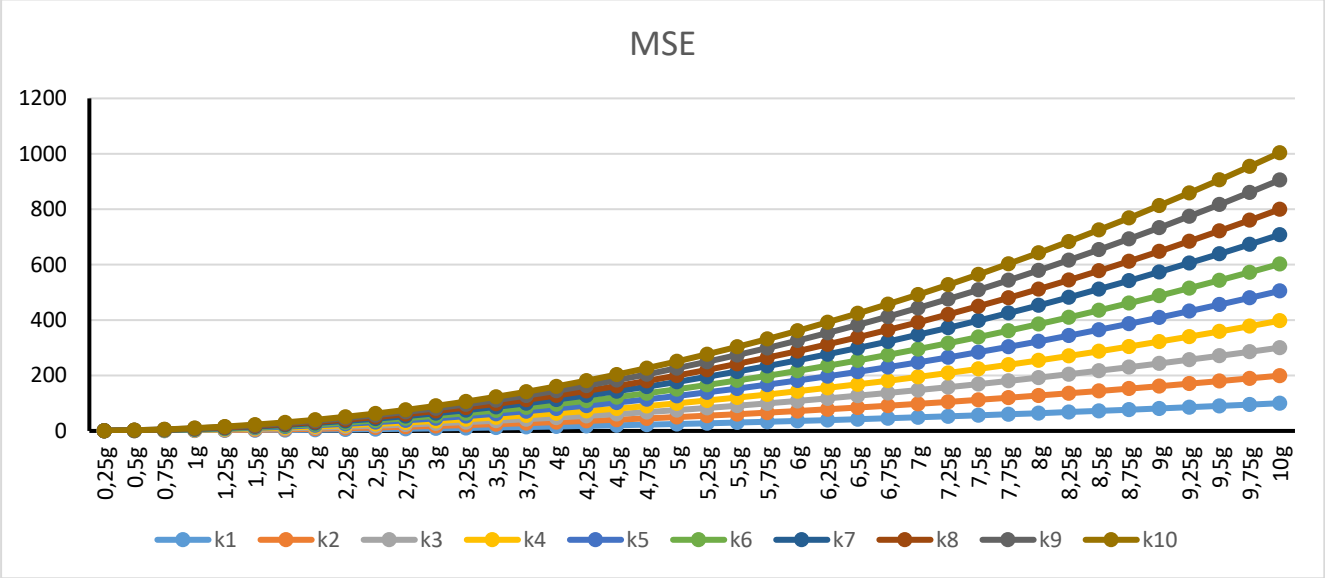

Рисунок 3.52 –Показник MSE для методу формування сигналів №5 при різних параметрах k

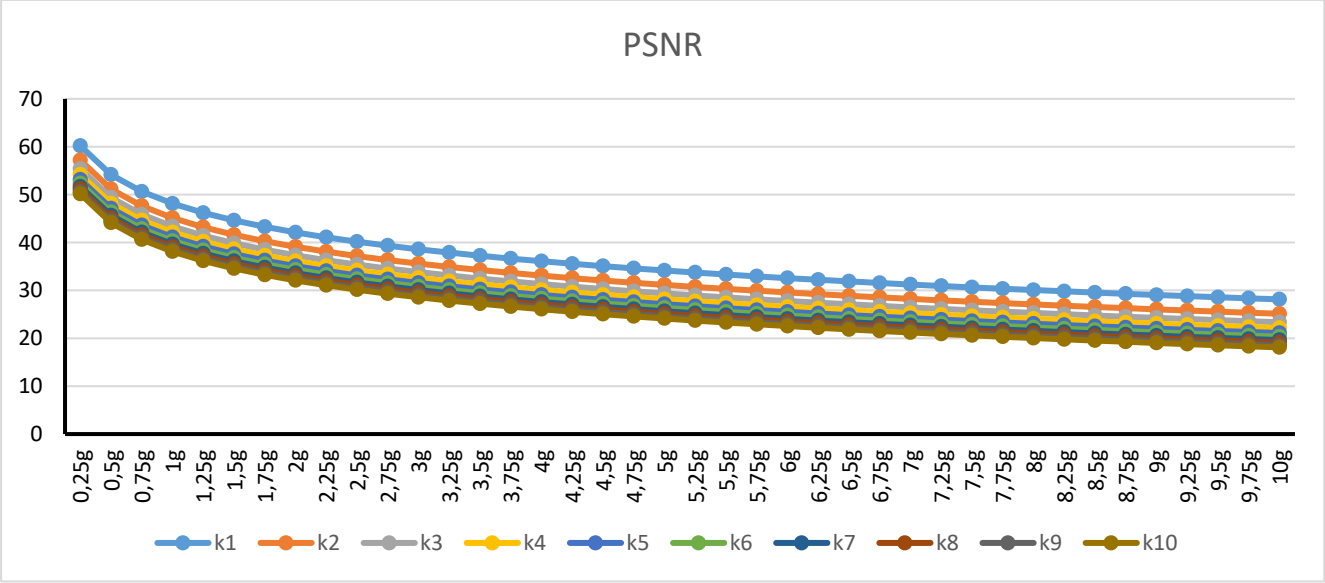

Рисунок 3.53 –Показник PSNR для методу формування сигналів №5 при різних параметрах k

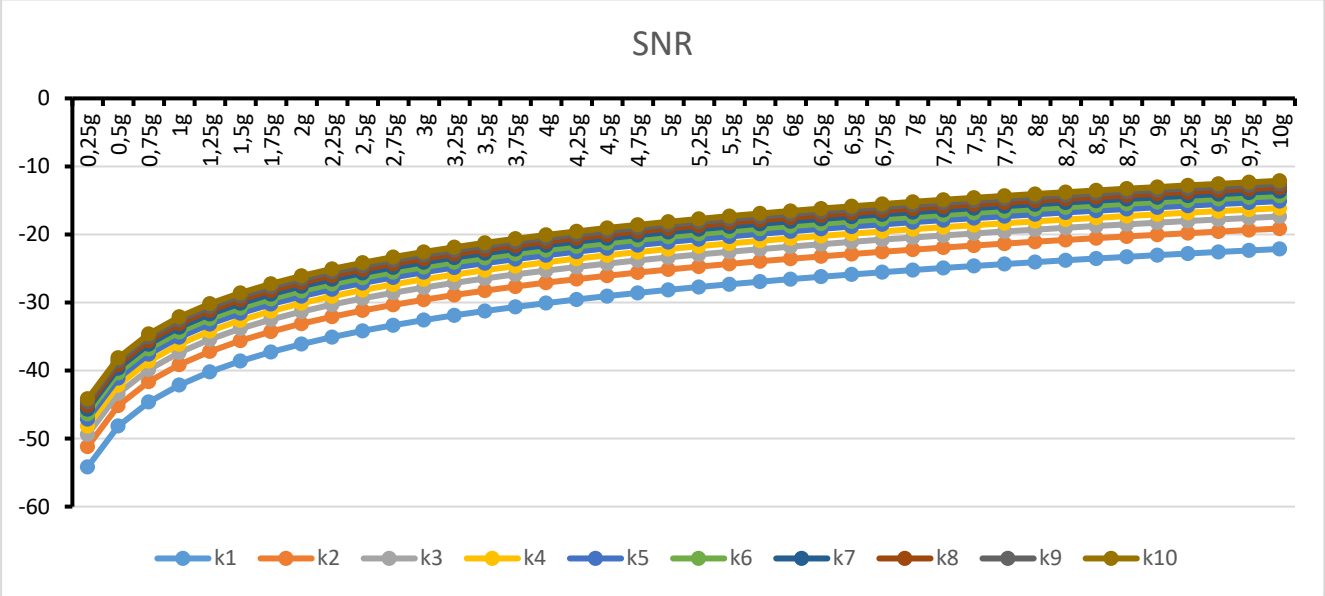

Рисунок 3.54 –Показник SNR для методу формування сигналів №5 при різних параметрах k

Що стосується дослідження методом PEAQ, слід зазначити, про дуже низький ODG та високі показники ADB, Relative Disturbed Frames, та NMR, тому відчувається різниця між початковим контейнером та модифікованим контейнером, це наведено у Додатку Б (див. Рис. Б.17 – Б.20).

3.3.2 Спосіб вбудовування повідомлення методом адресації

Наведені далі графіки є результатами комбінування методу адресації з різними методами створення псевдовипадкових послідовностей .

3.3.2.1 Метод формування сигналів №1

Метод № 1, в комбінуванні з методом адресації, надає наступні результати: дуже високий показник BER(Рис. 3.55), але натомість вдалося значно покращити показники MSE(Рис. 3.56), SNR(Рис. 3.58), та PSNR(Рис. 3.57). Головною проблемою методу є висока кількість помилок при вилученні повідомлення.

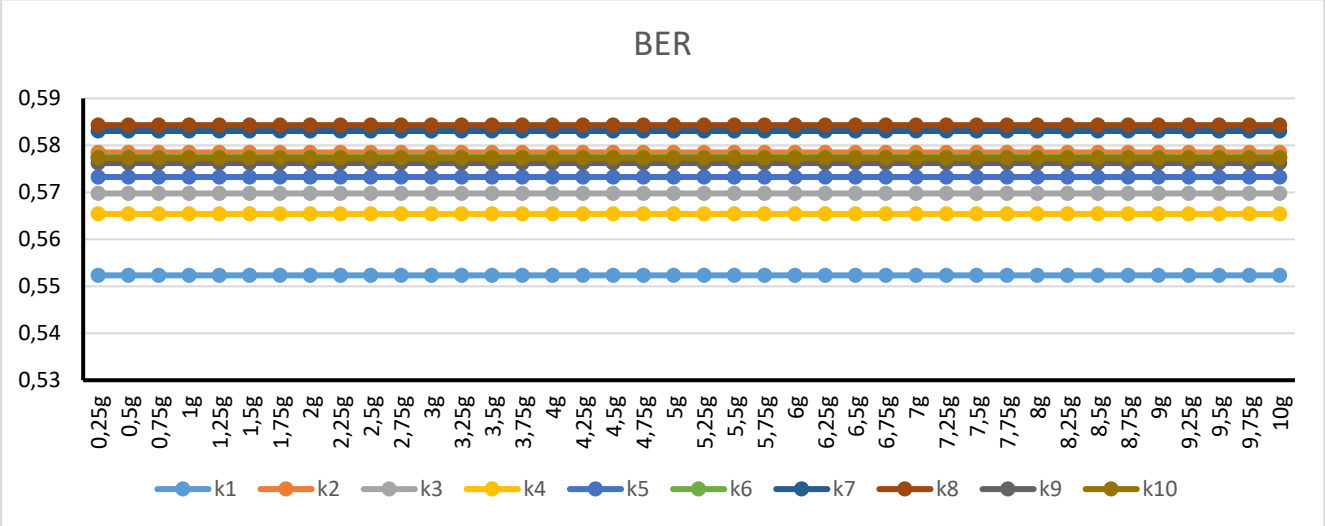

Рисунок 3.55 – Показник BER для методу формування сигналів №1 при різних параметрах k

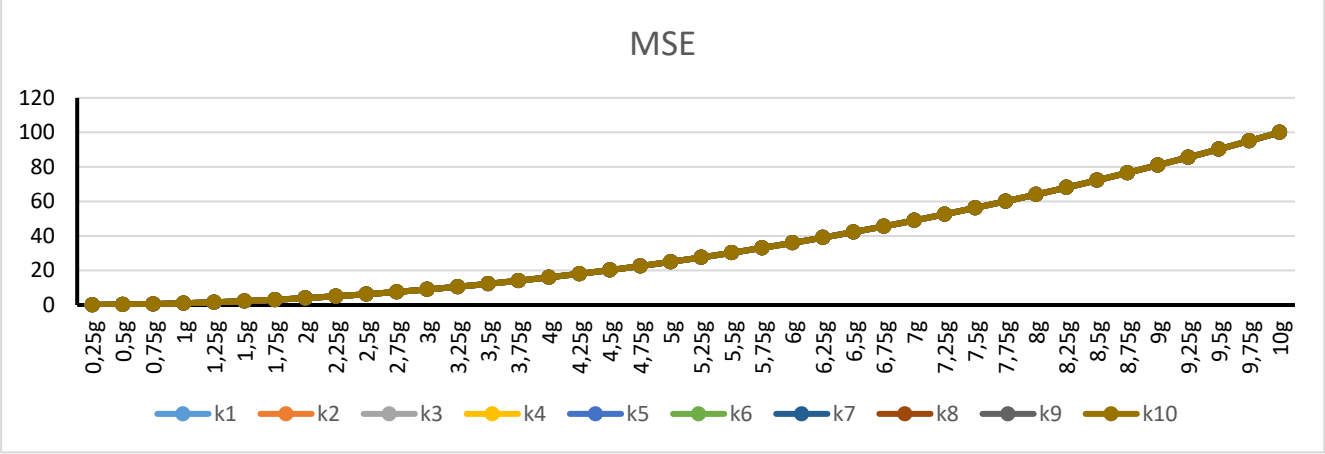

Рисунок 3.56 – Показник MSE для методу формування сигналів №1 при різних параметрах k

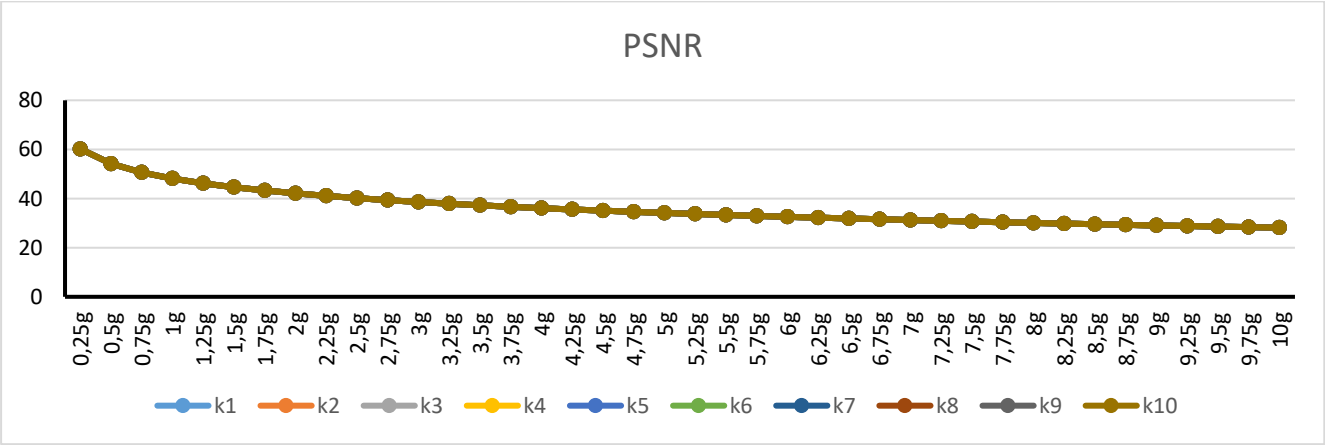

Рисунок 3.57 – Показник PSNR для методу формування сигналів №1 при різних параметрах k

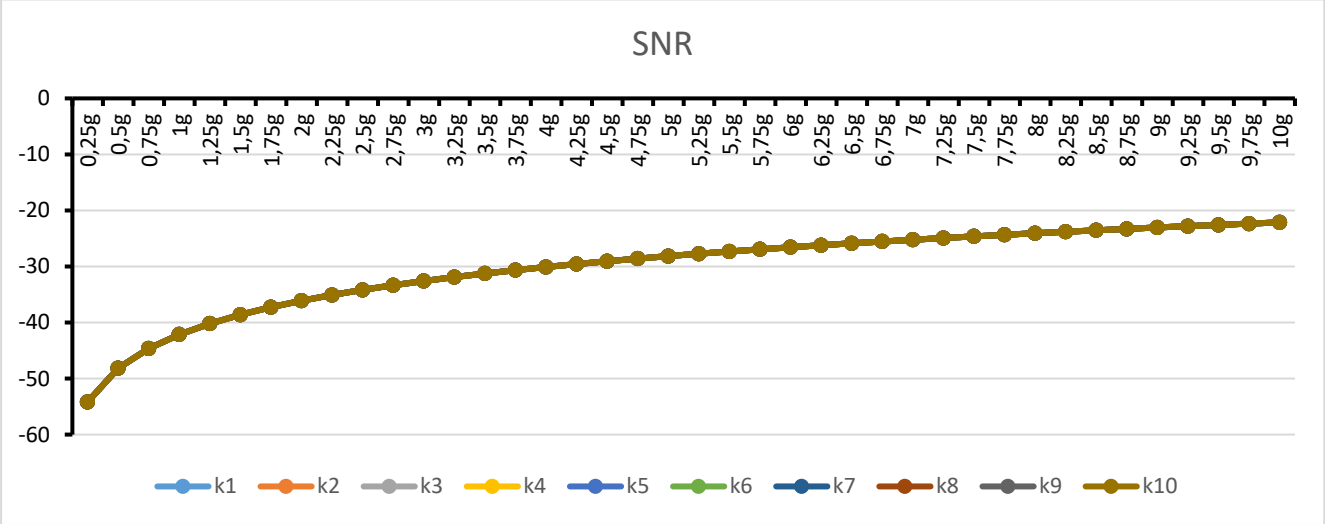

Рисунок 3.58 – Показник SNR для методу формування сигналів №1 при різних параметрах k

Що стосується дослідження методом PEAQ, слід зазначити, про дуже високі результати в усіх 4х показниках, тому різниця між початковим контейнером та модифікованим не відчувається, це наведено у Додатку Б (див. Рис. Б.21 – Б.24).

3.3.2.2 Метод формування сигналів №2

Метод № 2, в комбінуванні з методом адресації, надає наступні результати: дуже високий показник BER(Рис. 3.59), низькі показники MSE(Рис. 3.60) та SNR(Рис. 3.62), а також високий показник PSNR(Рис. 3.61). Головною проблемою методу є дуже висока кількість помилок при вилученні повідомлення.

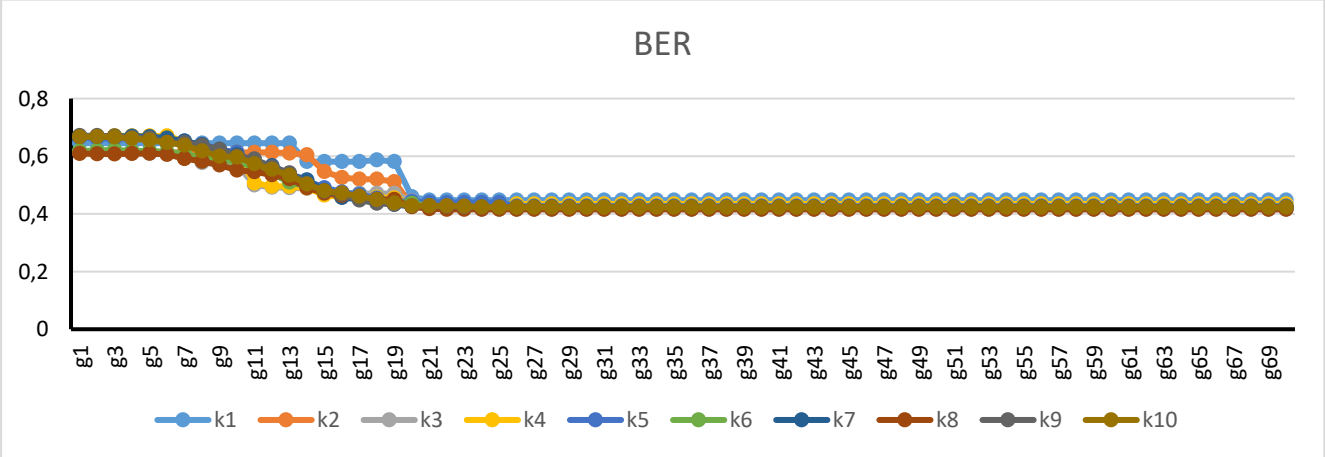

Рисунок 3.59 – Показник BER для методу формування сигналів №2 при різних параметрах k

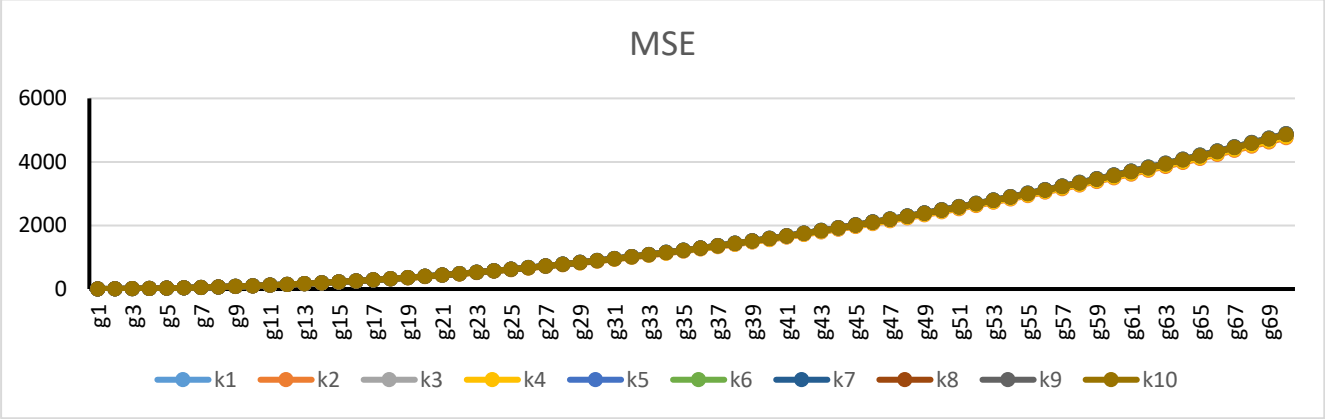

Рисунок 3.60 – Показник MSE для методу формування сигналів №2 при різних параметрах k

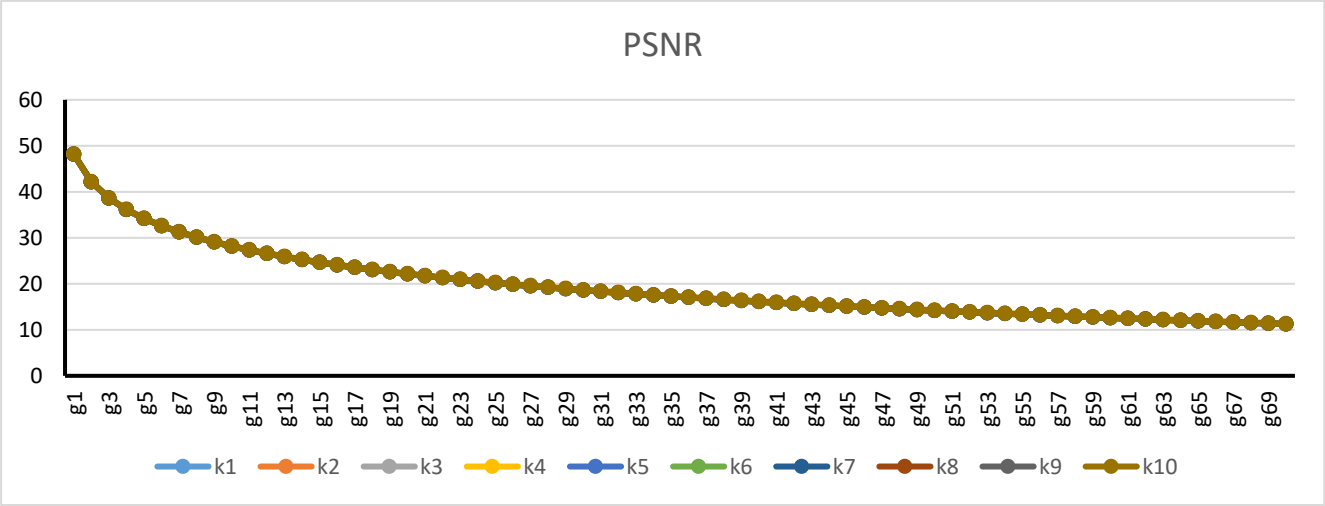

Рисунок 3.61 – Показник PSNR для методу формування сигналів №2 при різних параметрах k

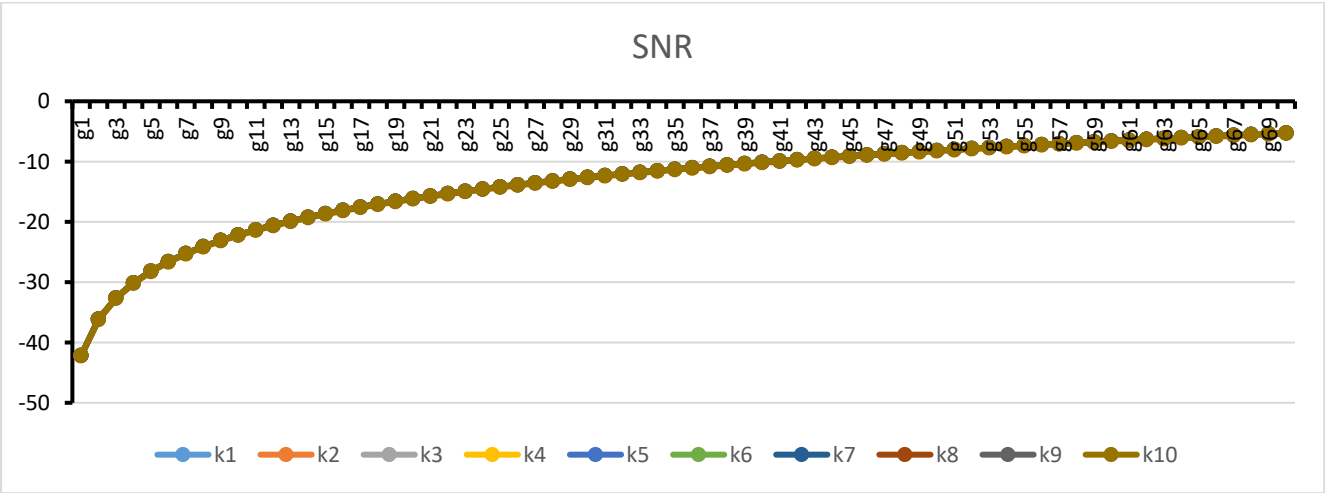

Рисунок 3.62 – Показник SNR для методу формування сигналів №2 при різних параметрах k

При дослідженні методом PEAQ, завдяки методу адресації вдалося дещо покращити результати, але нажаль вони залишаються на низькому рівні, це наведено у Додатку Б (див. Рис. Б.25 – Б.28).

3.3.2.3 Метод формування сигналів №3

Метод № 3, в комбінуванні з методом адресації, надає наступні результати: дуже високий показник BER(Рис. 3.63), але натомість вдалося значно покращити показники MSE(Рис. 3.64), SNR(Рис. 3.66), та PSNR(Рис. 3.65). Головною проблемою методу є висока кількість помилок при вилученні повідомлення.

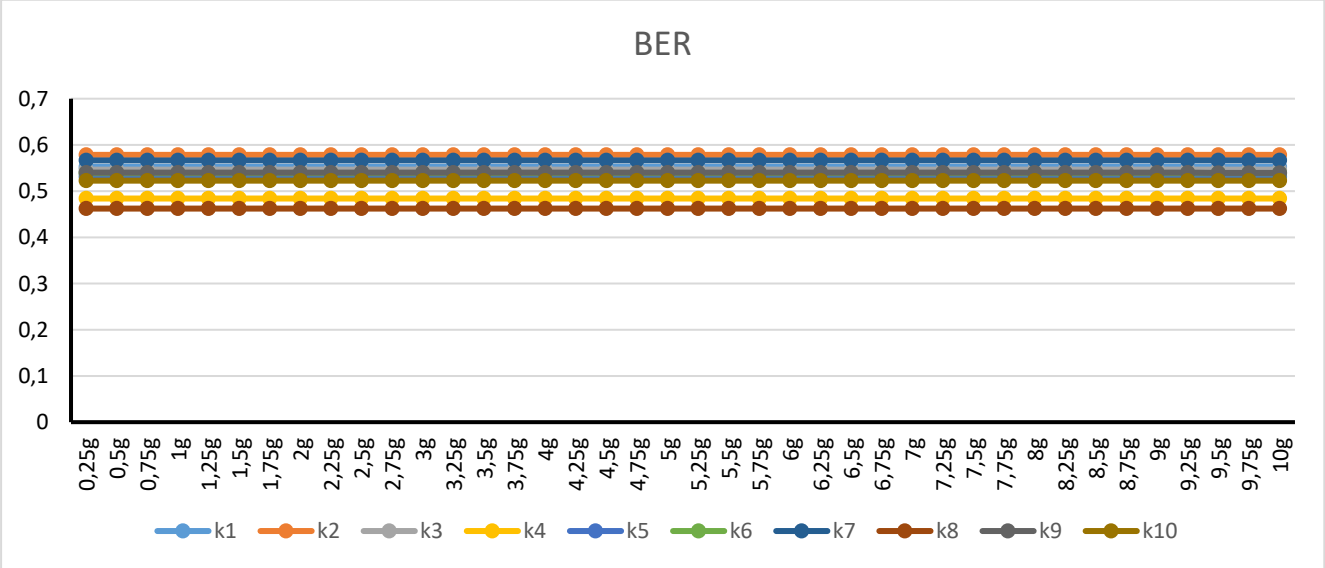

Рисунок 3.63 – Показник BER для методу формування сигналів №3 при різних параметрах k

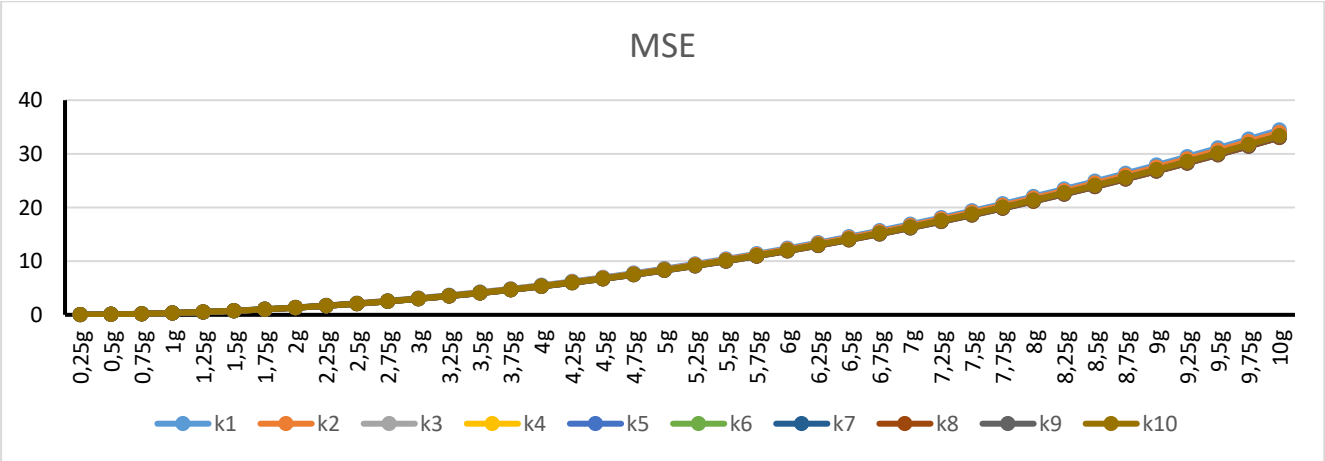

Рисунок 3.64 – Показник MSE для методу формування сигналів №3 при різних параметрах k

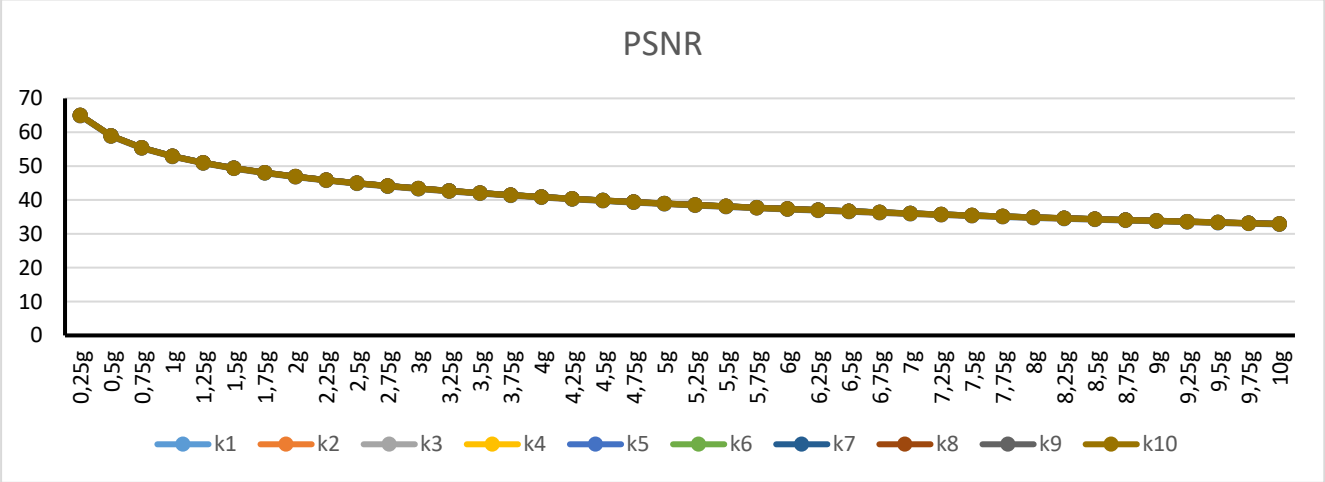

Рисунок 3.65 – Показник PSNR для методу формування сигналів №3 при різних параметрах k

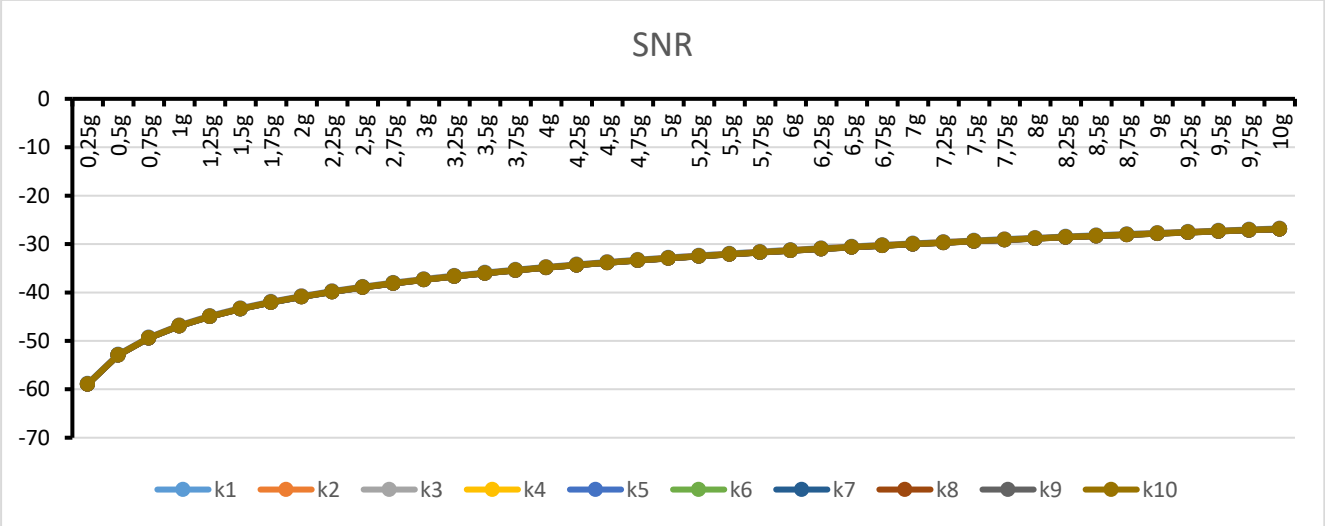

Рисунок 3.66 – Показник SNR для методу формування сигналів №3 при різних параметрах k

При дослідженні методом PEAQ, завдяки методу адресації вдалося дещо покращити результати, але нажаль вони залишаються на низькому рівні, це наведено у Додатку Б (див. Рис. Б.29 – Б.32).

3.3.2.4 Метод формування сигналів №4

Метод № 4, в комбінуванні з методом адресації, надає наступні результати: високий показник BER(Рис. 3.67), але натомість вдалося значно покращити показники MSE(Рис. 3.68), SNR(Рис. 3.70), та PSNR(Рис. 3.69). Головною проблемою методу є потреба в збільшенні потужності сигналу до показників вище 10 для отримання низької кількості помилок при вилученні повідомлення.

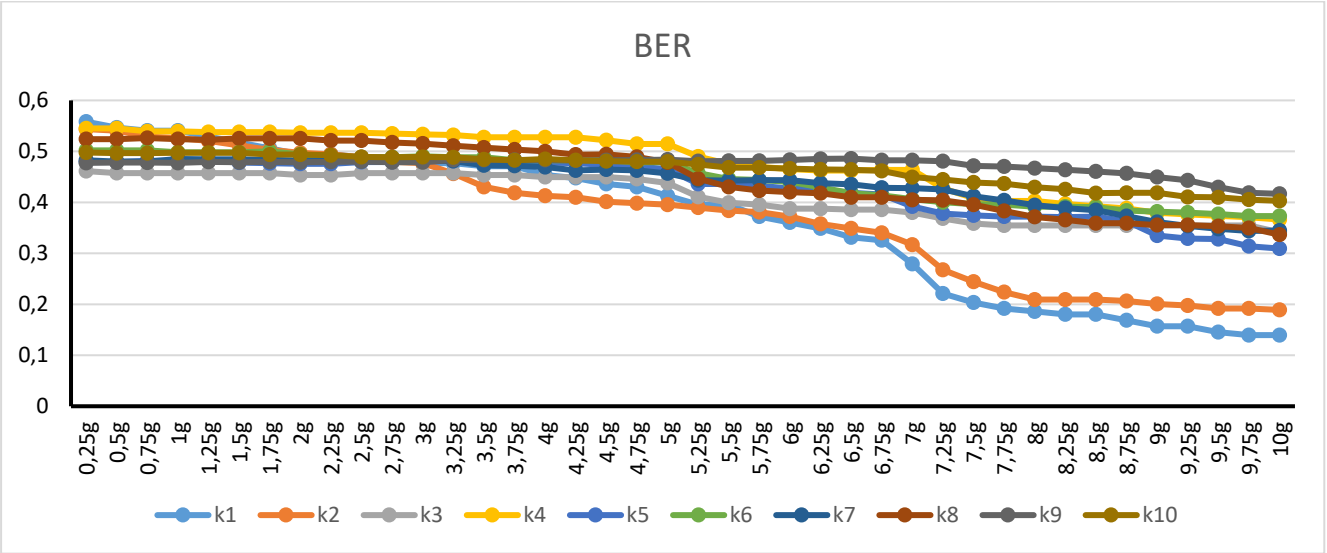

Рисунок 3.67 – Показник BER для методу формування сигналів №4 при різних параметрах k

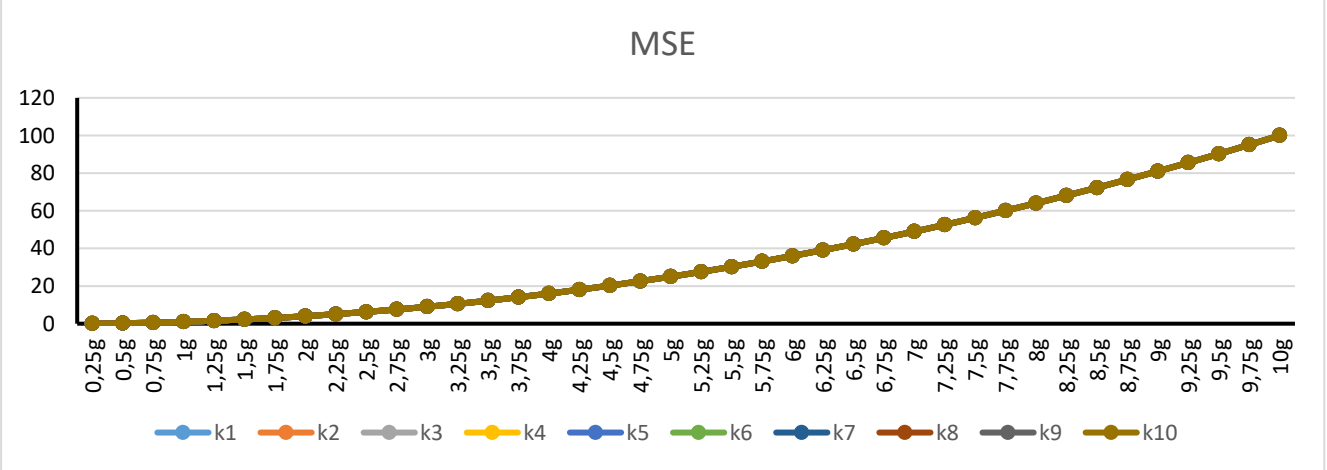

Рисунок 3.68– Показник MSE для методу формування сигналів №4 при різних параметрах k

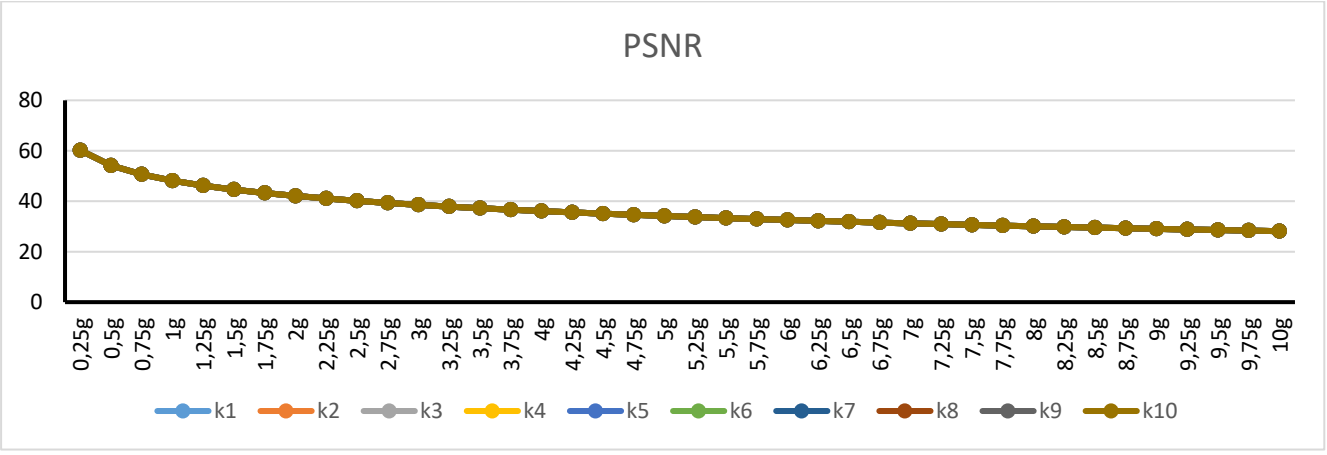

Рисунок 3.69 – Показник PSNR для методу формування сигналів №4 при різних параметрах k

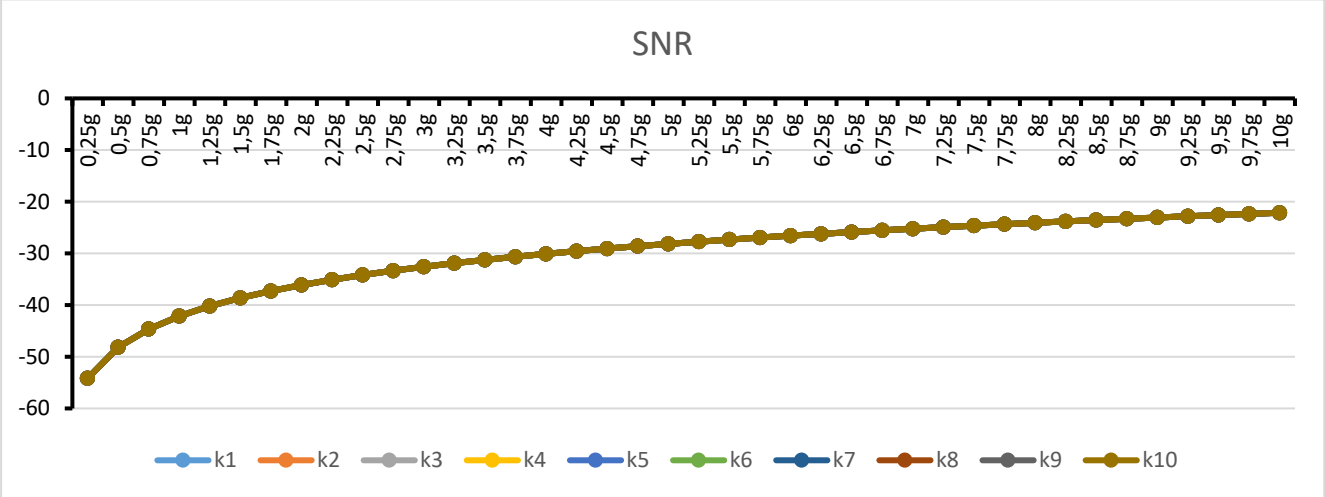

Рисунок 3.70 – Показник SNR для методу формування сигналів №4 при різних параметрах k

При дослідженні методом PEAQ, завдяки методу адресації вдалося дещо покращити результати, але нажаль вони залишаються на низькому рівні, це наведено у Додатку Б (див. Рис. Б.33 – Б.36).

3.3.2.5 Метод формування сигналів №5

Метод № 5, в комбінуванні з методом адресації, надає найкращі результати: дуже низький показник BER(Рис. 3.71), вдалося значно покращити показники MSE(Рис. 3.72), SNR(Рис. 3.74), та PSNR(Рис. 3.73). Головною проблемою методу є висока кількість помилок при вилученні повідомлення.

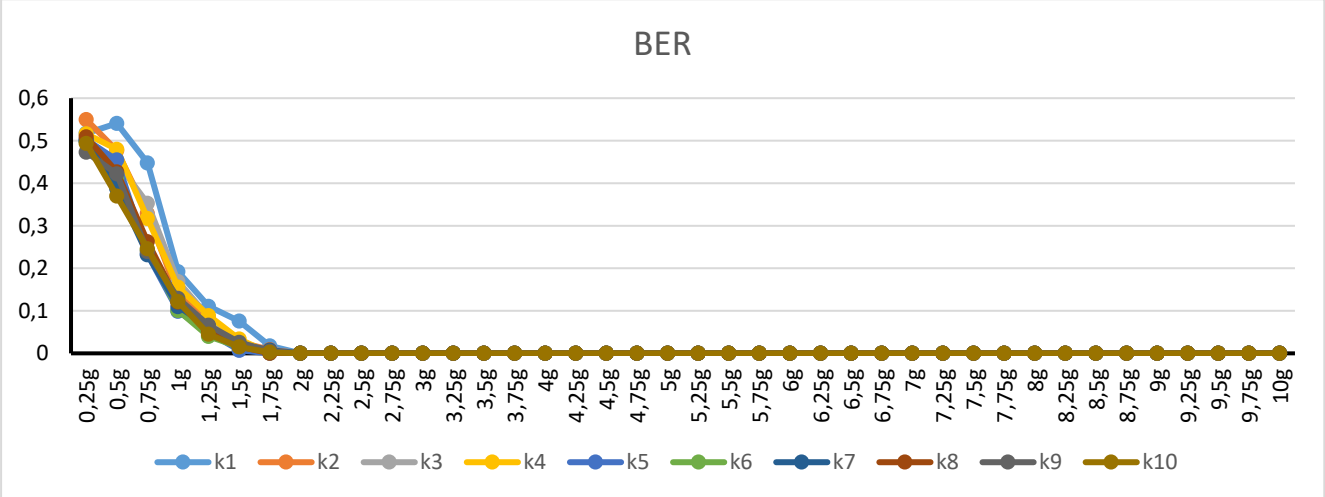

Рисунок 3.71 – Показник BER для методу формування сигналів №5 при різних параметрах k

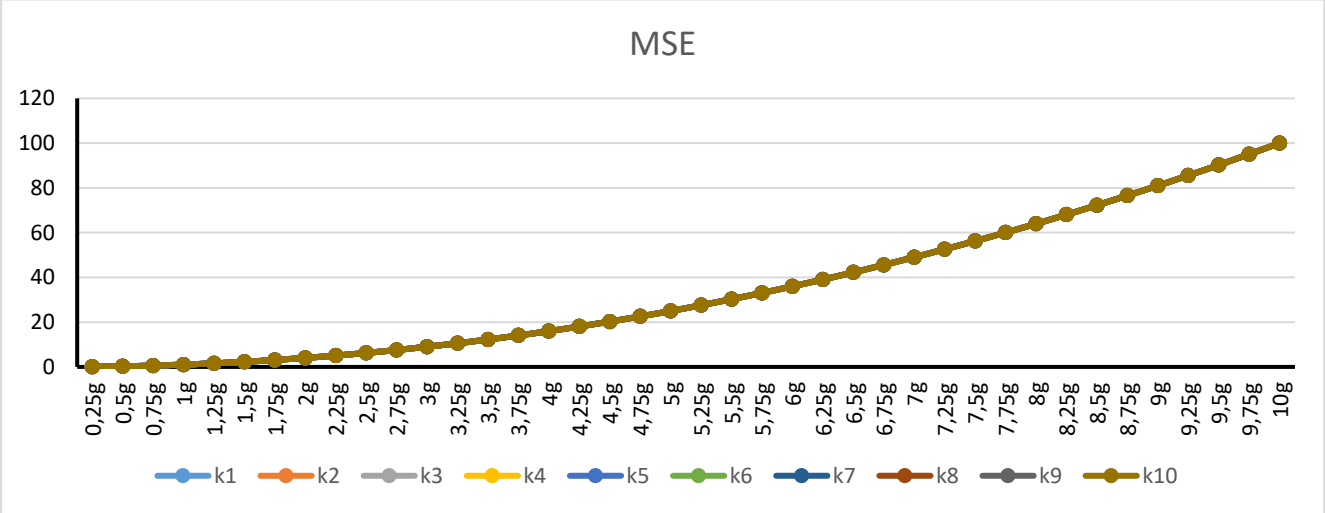

Рисунок 3.72 – Показник MSE для методу формування сигналів №5 при різних параметрах k

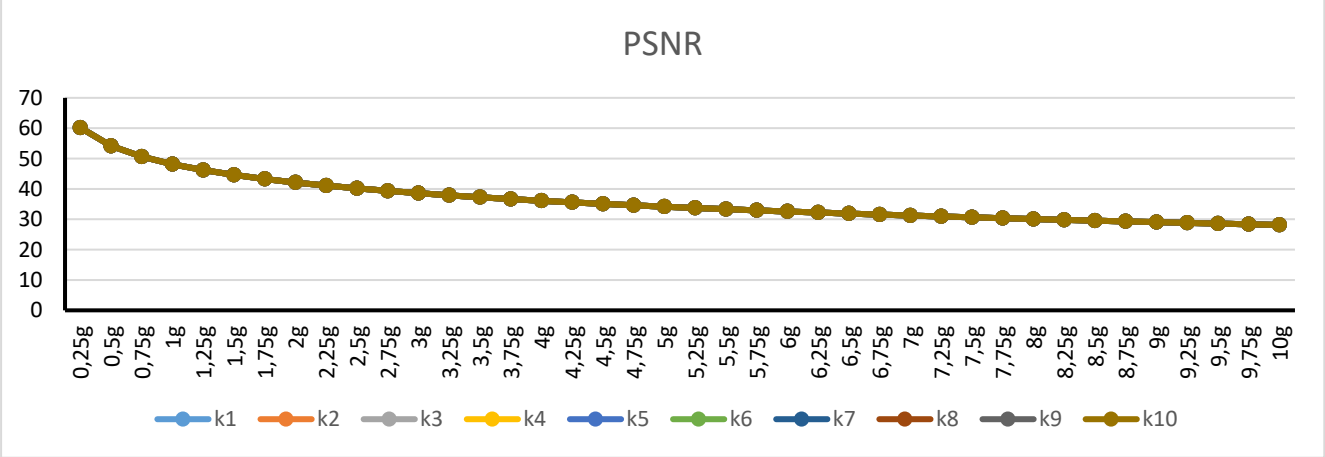

Рисунок 3.73 – Показник PSNR для методу формування сигналів №5 при різних параметрах k

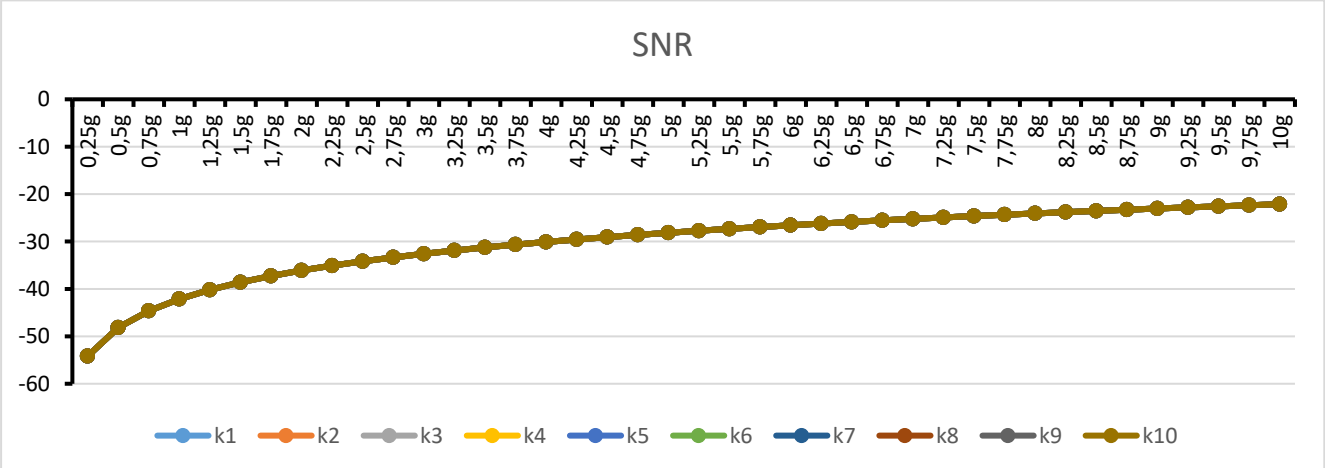

Рисунок 3.74 – Показник SNR для методу формування сигналів №5 при різних параметрах k

При дослідженні методом PEAQ, завдяки методу адресації вдалося покращити результати, вдалося підвищити показник ODG, це призвело до покращення результатів в інших показниках, наведено у Додатку Б (див. Рис. Б.37 – Б.40).

3.3.3 Least Significant Bit - Найменш значущий біт

Метод LSB має низький BER, MSE, SNR високий PSNR.

Результати тестування методу LSB для вбудовування повідомлення в аудіоконтейнер дозволяє зробити висновок, що метод PEAQ треба проводити з аудіофайлами кращої якості, оскільки 8-бітний одноканальний звук має занадто високі базові спотворення, тому при зміні якості звуку з моно сигналу у стерео сигнал показники PEAQ покращуються.

Таблиця 3.1 – Результати використання методу LSB з 8 бітними аудіофайлами з різною кількістю каналів

Кількість каналів	BER	MSE	PSNR	SNR	ODG	ADB	RelDistFrames	NMR
1	0.0	0.543	50.4623	-4.2013	-3.7266	1.0000	15.6603	42.6808
2	0.0	0.4578	51.5236	-4.4068	-2.6862	1.0000	14.0534	49.7189

3.4 Обґрунтування практичних рекомендацій щодо впровадження розробленої стеганосистеми

Аналіз представлених на рис. 3.35-3.74 результатів дозволяє зробити такі висновки.

По-перше, важко реалізувати приховування інформації без значного спотворення контейнера. Наведені вище залежності наочно демонструють це. Збільшуючи обсяг прихованого повідомлення k та/або коефіцієнт посилення потужності g, так чи інакше спотворюється контейнер, тобто зменшується PSNR.

Основну перевагу отримують такі чіп-коди для приховування інформації, як сигнали Уолша Адамара. Єдиний вид сигналів, який може конкурувати з зазначеними чіп-кодами – адаптивні квазіортогональні дискретні сигнали.

Другий висновок — це можливий компроміс між кількістю прихованих даних і спотворенням аудіоконтейнера, що вноситься. Наприклад, зменшення значення k призводить до збільшення PSNR приблизно. Це можна використовувати для налаштування бажаних параметрів стегосистеми у конкретній практичній програмі.

Третій і, мабуть, найважливіший висновок у тому, що досягти великих обсягів прихованої інформації відомими методами практично неможливо. Наприклад, навіть для найкращого випадку, коли $PSNR = 35\ дБ$, а $BER < 1\%$ можна приховати лише 16 бітів інформації. Поліпшити ці значення традиційними методами можна лише за рахунок збільшення довжини чіпових кодів, що неминуче призводить до збільшення обчислювальної складності перетворень. Однак, це може стати проблемою. Оскільки із збільшенням довжини чіп-кодів знижується продуктивність, що може суттєво ускладнити практичну реалізацію.

Четвертий висновок полягає у тому, що використання методу адресації, замість стандартного методу вбудовування повідомлення в контейнер, дійсно знижує кількість спотворень контейнера. На жаль, в цій ситуації дуже сильно збільшується кількість помилок при вилученні повідомлення. На щастя, завдяки використанню методу адаптивної генерації сигналів дозволяє суттєво знизити кількість помилок.

П’ятий висновок, стосується методу PEAQ. Після проведення тестування за допомогою методу PEAQ, модифікований аудіоконтейнер все одно відрізняється від свого базового аналогу. Для перевірки правильності роботи тестів PEAQ, був також реалізований метод LSB для вбудови повідомлення. Показник ODG мав так само погані результати, але змінивши якість аудіоконтейнеру, вдалося суттєво покращити результат. Тому для наступних тестів рекомендується обирати контейнери кращої якості, тобто з кращим Sample Rate та більшою кількістю каналів. Це також є перспективним напрямком подальших досліджень.

ВИСНОВКИ

У цій дипломній роботі були досліджені методи приховування інформації в аудіоконтейнерах із використанням прямого розширення спектру. Були розглянуті різні послідовності розширення (чіп-коди) та вивчені їх вплив на частоту появи помилок у відновлених повідомленнях. Також було досліджено спотворення аудіоконтейнера з точки зору MSE, SNR та PSNR.

Отримані результати наочно демонструють перспективність цього напряму. Саме в кожному сегменті контейнера удалося надійно приховати частину інформаційного повідомлення. У той же час значення BER, MSE, SNR та PSNR знаходяться у допустимих межах. Найкращі характеристики показали чіп-коди Уолша-Адамара, які показали найменшу частоту помилок (при порівнянних контейнерних спотвореннях).

У той самий час слід зазначити, що є об'єктивно суперечливі чинники, негативно впливають друг на друга. Наприклад, збільшуючи обсяг прихованих повідомлень, неминуче збільшується BER та спотворюється контейнер. Збільшуючи потужність чіп-кодів, можна зменшити BER. Однак це ще більше спотворює контейнер. Це узгоджується з відомими результатами стеганографії, що ґрунтуються на прямому розширенні спектру. Наприклад, за рахунок збільшення потужності чіп-кодів можна зменшити BER, але спотворення покриттів стають дуже значними. Вдалося уявити наявність компромісних рішень. Це особливо вірно для чіп-кодів Уолша-Адамара, для яких діапазон можливих рішень набагато ширший, ніж для інших розширюючих послідовностей. Проте, слід зазначити, що з використанням традиційних методів приховування даних обсяг прихованих повідомлень може бути великим.

В роботі було досліджено формування псевдовипадкових послідовностей за допомогою адаптивних квазіортогональних дискретних сигналів. В цьому методі відбувається відбраковка послідовностей, за рахунок враховування статистичних властивостей контейнера, що забезпечує виконання базового припущення про

відсутність кореляції аудіо стеганоконтейнеру та дискретних послідовностей. Як показали тести метод адаптивного формування сигналів посідає друге місце за результатами, маючи також дуже низькі показники BER і задовільні показники MSE, SNR та PSNR.

При цьому для покращення результатів, в був розглянутий інший метод вбудовування повідомлення в контейнер, метод адресації. В комбінації з різними методами формування псевдовипадкових послідовностей, метод адресації сильно знижує кількість викривлень контейнера при вбудовуванні повідомлення. На жаль використання методу також супроводжується сильними ушкодженнями повідомлення при його декодуванні. Але, цю проблему можна вирішити, використовуючи адаптивний метод формування сигналів. Це максимально знижує вірогідність помилок в декодованому повідомленні.

Були проведені тести методом PEAQ, який ґрунтується на загальноприйнятих психоакустичних принципах. На жаль спостерігаються високі показники шуму у контейнері-зображенні. За результатами тесту можна сказати, що вирішення цієї проблеми є перспективним напрямком для подальших досліджень, оскільки викривлення аудіоконтейнера залишаються незначними, лише для малих показників потужності сигналів та кількості вбудованих біт.

Також було помічено, що тест PEAQ показує кращі результати при використанні аудіоконтейнеру вищої якості, тому в подальшому слід дослідити цей феномен.

Експерименти, проведені в даній дипломній роботі, дозволяє підтвердити можливість використання псевдовипадких послідовностей, сформованих за допомогою квазіортогональних дискретних сигналів, у комбінації з методом адресації, який дозволяє зменшити викривлення стеганоконтейнеру.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Digital Watermarking and Steganography / Ingemar J. Cox et al. Burlington, 2008. 587 p. DOI: 10.1016/b978-0-12-372585-1.x5001-3.
2. F. Y. Shin. Digital Watermarking and Steganography. New Jersey, 2017. 293p. DOI: 10.1201/9781315219783.
3. Host Cancelation-Based Spread Spectrum Watermarking for Audio Anti-Piracy over Internet/ Li R.; Xu, S.; Rong, B.; Yang, H. Ottawa: Secur. Commun. Netw., 2016. Vol. 6. P. 4691–4702.
4. Security and PrIvacy foR the Internet of Things: An Overview of the Project / Aroua S. et al. In Proceedings of the 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC). Bari, 2019. P. 3993–3998.
5. Moulin P., O'Sullivan J.A. Information-Theoretic Analysis of Information Hiding. IEEE Trans. Inf. Theory, 2003. P. 563–593.
6. I. V. S. Manoj. Cryptography and Steganography. New York: International Journal of Computer Applications, 2010. Vol.1, №12. P. 63–68. DOI: 10.5120/257-414.
7. N. F. Johnson, S. Jajodia. Exploring steganography: Seeing the unseen, in Computer. Virginia, 1998. Vol. 31, №2. P. 26-34. DOI: 10.1109/MC.1998.4655281.
8. A. Z. Tirkel, C. F. Osborne, R. G. Van Schyndel. Image watermarking-a spread spectrum application. Mainz: Proceedings of ISSSTA'95 International Symposium on Spread Spectrum Techniques and Applications, 1996. Vol.2. P. 785-789. DOI: 10.1109/ISSSTA.1996.563231.
9. WAV File Extension-What Is It? How to Open a WAV File? All about WAV Files: веб-сайт. URL: https://filext.com/file-extension/WAV (дата звернення 25.09.2022).
10. Fleischman E. WAVE and AVI Codec Registries. RFC – Informational. 1998. URL: https://tools.ietf.org/html/rfc2361 (дата звернення 25.09.2022).
11. J. R. Smith, B. O. Comiskey. Modulation and information hiding in images. Cambridge: Lecture Notes in Computer Science, 1996. P. 207–226. DOI: 10.1007/3-

540-61996-8_42.

12. L. M. Marvel, C. G. Boncelet, R. Jr., Charles T. Methodology of Spread-Spectrum Image Steganography. Delaware, 1998. Vol.8, №8. 33p. DOI: 10.21236/ada349102.
13. L. M. Marvel, C. G. Boncelet, C. T. Retter. Spread spectrum image steganography, Delaware: IEEE Transactions on Image Processing, 1999. P. 1075-1083. DOI: 10.1109/83.777088.
14. F. S. Brundick, L. M. Marvel. Implementation of Spread Spectrum Image Steganography. Maryland, 2001. 23p. DOI:10.21236/ada392155.
15. Song T., Zhou K., Li T. CDMA System Design and Capacity Analysis Under Disguised Jamming. IEEE Trans. Inf. Forensics Secur. 2016, Vol. 11. P. 2487–2498.
16. Complementary Coded CDMA Systems With CP-Free OFDM \ Wang X., Liu X., Chen H.-H., Meng W. IEEE Trans. Veh. Technol. 2020, Vol. 69. P. 11515–11528.
17. Ipatov, V.P. Spread Spectrum and CDMA: Principles and Applications; John Wiley & Sons, Ltd.: Chichester, UK, 2005. 6-14 p. ISBN 978-0-470-09180-7.
18. Yu. V. Stasev, A.A. Kuznetsov, A.M. Nosik. Formation of pseudorandom sequences with improved autocorrelation properties. Kharkiv: Cybernetics and Systems Analysis, 2007. Vol. 43, №1. P. 1-11. DOI:10.1007/s10559-007-0021-2
19. Periodic Properties of Cryptographically Strong Pseudorandom Sequences/ A. Kuznetsov et al. Kharkiv: International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), 2018. P. 129-134. DOI: 10.1109/INFOCOMMST.2018.8632021
20. Formation of Pseudorandom Sequences with Special Correlation Properties/ A. Kuznetsov et al. Lviv: 3rd International Conference on Advanced Information and Communications Technologies (AICT), 2019. P. 395-399. DOI: 10.1109/AIACT.2019.8847861
21. Motlagh, N.H. Frequency Hopping Spread Spectrum: An Effective Way to Improve Wireless Communication Performance. In Advanced Trends in Wireless Communications; IntechOpen Limited: London, UK, 2011.

https://doi.org/10.5772/15482.

22. Reynders B., Pollin S. Chirp Spread Spectrum as a Modulation Technique for Long Range Communication. In Proceedings ofthe 2016 Symposium on Communications and Vehicular Technologies (SCVT), Mons, Belgium, 2016; P. 1–5.
23. Torrieri, D. Chapter 2 Direct-Sequence Systems. In Principles of Spread-Spectrum Communication Systems; Torrieri, D., Ed.; Springer International Publishing: Cham, Switzerland, 2015; P. 79–145, ISBN 978-3-319-14096-4.
24. Оникійчук О. О. Моделі та методи стеганографічного приховування інформації в аудіоконтейнери із застосуванням технології прямого розширення спектру : Пояснювальна записка до дипломної роботи бакалавра: напрям підготовки 125 – Кібербезпека / О. О. Оникійчук; Харківський національний університет імені В. Н. Каразіна. – Харків, 2021. 62 с.
25. Sklar B. Digital Communications: Fundamentals and Applications, 2nd ed.; Prentice Hall: Upper Saddle River, NJ, USA, 2017. P. 1-5. ISBN 978-0-13-472405-8.
26. Direct Spread Spectrum Technology for Data Hiding in Audio / Kuznetsov A. et al. Sensors 2022, Vol. 22. 3115 p. DOI:10.3390/s22093115
27. Pseudorandom Sequences for Spread Spectrum Image Steganography. In Proceedings of the International Workshop on Cyber Hygiene (CybHyg-2019) Co-Located with 1st International Conference on Cyber Hygiene and Conflict Management in Global Information Networks \ Kuznetsov A. et al. Kyiv, Ukraine, 2019, Vol. 2654. P. 122–131.
28. New Technique for Data Hiding in Cover Images Using Adaptively Generated Pseudorandom Sequences. In Proceedings of the International Workshop on Cyber Hygiene (CybHyg-2019) co-located with 1st International Conference on Cyber Hygiene and Conflict Management in Global Information Networks (CyberConf 2019) / Kuznetsov A., Smirnov O., Kovalchuk D., Kuznetsova T. Kyiv, Ukraine, 2019, Vol. 2654. P. 1–14.
29. D. Câmpeanu, A. Câmpeanu. PEAQ – An Objective Method to Assess the Perceptual Quality of Audio Compressed Files. Proceedings of the International

Symposium on System Theory, 2005. P. 487–492.

30. ITU Recommendation ITU-R BS.1387: Method for objective measurements of perceived audio quality, Nov. 2001.
31. Adaptive Pseudo-Random Sequence Generation for Spread Spectrum Image Steganography Proceedings / A. Kuznetsov et al. Kyiv: IEEE 11th International Conference on Dependable Systems, Services and Technologies, DESSERT, 2020. P. 161-166. DOI**:** 10.1109/DESSERT50317.2020.9125032
32. O. Smirnov, A. Kuznetsov, L. Gorbacheva. Hiding information in images using pseudo-random sequences. Kharkiv, 2020. 52 p. DOI: 10.26565/2519-2310-2020-1-01
33. Data Hiding Using Shuffle Algorithm and LSB Method. In Proceedings of the 2018 26th Signal Processing and Communications Applications Conference (SIU) / Akbay K.et al. Turkey, 2018; P. 1–4.
34. Latypov, R.K.; Stolov, E.L. Ternary Echo Hiding in Audio Files. In Proceedings of the 2020 28th Telecommunications Forum ($^{\text{TEL}}$FOR). Belgrade, Serbia, 2020. P. 1–4.
35. Data Hiding Scheme Based on Spread Sequence Addressing. In Proceedings of the 1st International Workshop on Computational & Information Technologies for Risk-Informed Systems (CITRisk 2020) co-located with XX International scientific and technical conference on Information Technologies in Education and Management (ITEM 2020) / Kuznetsov A., Kiian A., Kuznetsova K., Smirnov A. Kherson, Ukraine, 2020. P. 44–58.

ДОДАТОК А

Код для дослідження аудіоконтейнерів методом PEAQ.

```
import numpy as np
from PQEval import PQEval
import time

class PEAQ(object):
        def __init__(self, Amax = 1, Fs = 48000, NF = 2048):
                # Amax = maximum signal amplitude
                # Fs = sampling frequency
                # NF = Length of analysis window

                self.NF = NF
                self.Fs = Fs
                self.Amax = Amax

                #Step forward in half window lengths:
                self.Nadv = self.NF / 2

                #Number of critical bands:
                self.Nc = 109
```

Рисунок А.1 – Клас PEAQ та функція ініціалізації

```
    def process(self, referenceSignal, testSignal):
            #Preform basic procssing (Section 2 in Kabal.)
            # sigR = reference signal
            # sigT = test signal

            sigR = referenceSignal
            sigT = testSignal

            #Number of frames:
            self.Np = np.floor(len(sigR)/self.Nadv)-1

            #Scale audio:
            if np.amax(abs(sigR)) != self.Amax:
                    sigRS = self.Amax*sigR/float(np.amax(abs(sigR)))
                    sigTS = self.Amax*sigT/float(np.amax(abs(sigT)))
                    print ('Signals scaled, max reference value = ' + str(np.amax(abs(sigRS))) + ',')
                    print ('and max test value = ' + str(np.amax(abs(sigTS))) +'.')

            #Instantiate Object to process single frames of data:
            self.PQE = PQEval(Amax = self.Amax, Fs = self.Fs, NF = self.NF)

            print ('Processing Audio...')

            #Create empty matrices:
            X2 = np.zeros((2,self.NF/2+1))

            self.X2MatR = np.zeros((self.Np, self.NF/2+1))
            self.X2MatT = np.zeros((self.Np, self.NF/2+1))

            self.EbNMat = np.zeros((self.Np, self.Nc))
            self.EsMatR = np.zeros((self.Np, self.Nc))
            self.EsMatT = np.zeros((self.Np, self.Nc))
```

Рисунок А.2 – Функція початку розрахунків

```
                self.EhsR = np.zeros((self.Np, self.Nc))
                self.EhsT = np.zeros((self.Np, self.Nc))

                previousFrameR = np.zeros(self.Nc)
                previousFrameT = np.zeros(self.Nc)

                #Maybe take this out later, but useful in debugging:
                self.xMatR = np.zeros((self.Np, self.NF))
                self.xMatT = np.zeros((self.Np, self.NF))

                startS = 0

                startTime = time.clock()

                for i in np.arange(self.Np):
                    xR = sigRS[startS:self.NF+startS]
                    xT = sigTS[startS:self.NF+startS]
                    startS = startS+self.Nadv

                    #Store unmodified windows of audio:
                    self.xMatR[i, :] = xR
                    self.xMatT[i, :] = xT

                    #Process Frame:
                    X2[0,:] = self.PQE.PQDFTFrame(xR)
                    X2[1,:] = self.PQE.PQDFTFrame(xT)

                    self.X2MatR[i,:] = X2[0,:]
                    self.X2MatT[i,:] = X2[1,:]

```

Рисунок А.3 – Продовження функції розрахунків

```
                # Critical band grouping and frequency spreading
                self.EbN, self.Es = self.PQE.PQ_excitCB(X2)

                self.EbNMat[i,:] = self.EbN
                self.EsMatR[i,:] = self.Es[0,:]
                self.EsMatT[i,:] = self.Es[1,:]

                #Time domain spreading
                self.EhsR[i,:], previousFrameR = self.PQE.PQ_timeSpread(self.EsMatR[i,:], previousFrameR)
                self.EhsT[i,:], previousFrameT = self.PQE.PQ_timeSpread(self.EsMatT[i,:], previousFrameT)

            print ('Audio Processed! (Kabal, section 2), ' + str(self.Np) + ' windows processed ' + ' in ' + str(time.clock()

    def computeBW(self, X2MatR, X2MatT):
            #Kabal Section 5.4
            BWRef = np.zeros(self.Np)
            BWTest = np.zeros(self.Np)

            for i in range(int(self.Np)):
                    BWRef[i], BWTest[i] = self.PQmovBW(np.vstack((self.X2MatR[i,:], self.X2MatT[i,:])))

            return BWRef, BWTest

```

Рисунок А.4 – Функції тестів №1

```
    def computeBW(self, X2MatR, X2MatT):
            #Kabal Section 5.4
            BWRef = np.zeros(self.Np)
            BWTest = np.zeros(self.Np)

            for i in range(int(self.Np)):
                    BWRef[i], BWTest[i] = self.PQmovBW(np.vstack((self.X2MatR[i,:], self.X2MatT[i,:])))

            return BWRef, BWTest

    def PQmovBW(self, X2):
        # Bandwidth tests for a single frame of reference and test signal
        # X2 must be of size 2xNF/2+1

        # So this is MOSTLY the same as the Octave implementation, but every now and then it
        # Give an answer that is different by 1, I'm not sure why - SW 3.12.15

        # fx and fl will always be the same values
        fx = 21586
        kx = int(round(self.NF * float(fx)/self.Fs)) # 921
        fl = 8109
        kl = int(round(self.NF * float(fl)/self.Fs)) # 346
        FRdB = 10 # Ref. signal to exceed threshold level by 10dB
        FR = 10**(FRdB/10.) #added dot to make floating point - SW
        FTdB = 5 # Test signal to exceed threshold level by 5dB
        FT = 10**(FTdB/10.) #added dot to make floating point - SW
        N = self.NF/2

        # This is the method Kabal uses to find the threshold level.
        # The for loop seems unneccessary and not as efficient as
        # simply using a max operation on an array of the subset of DFT square mag.
        # values above 21.6kHz.
```

Рисунок А.5 –Функції тестів №2

```
            # Using the max operation on the subset of array values.
            # When tested, was about 0.1ms faster, produces same result,
            # and is cleaner, so I'll stick with this version.
            Xth = np.amax(X2[1,kx:N])

            # BWRef and BWTest remain negative if the BW of the test signal
            # does not exceed FR * Xth for kx-1 <= k <= kl+1
            BWRef = -1
            XthR = FR * Xth # Reference signal threshold level
            for k in xrange(kx-1,kl,-1):
                if (X2[0,k+1] >= XthR):
                    BWRef = k+1
                    break

            BWTest = -1
            XthT = FT * Xth # Test signal threshold level
            for k in xrange(BWRef-1,-1,-1):
                if (X2[1,k+1] >= XthT):
                    BWTest = k+1
                    break

            return BWRef, BWTest

        def computeNMR(self, EbNMat, EhsR):
                #Kabal Section ...
                #Compute NRM for whole time series.

                NMRavg = np.zeros(self.Np)
                NMRmax = np.zeros(self.Np)

                for i in range(int(self.Np)):
                        NMR = self.PQmovNMRB(EbNMat[i,:], EhsR[i,:])
                        NMRavg[i] = NMR['NMRavg']
                        NMRmax[i] = NMR['NMRmax']

                return NMRavg, NMRmax
```

Рисунок А.6 –Функції тестів №3

```
180         def PQmovNMRB(self, EbN, Ehs):
181             # Noise-to-mask ratio
182             # NMR['NMRavg'] = average NMR
183             # NMR['NMRmax'] = max. NMR
184 
185             NMR = dict()
186 
187             # Get ___
188             Nc, fc, fl, fu, dz = self.PQE.PQCB()
189             gm = self.PQ_MaskOffset(dz, Nc)
190 
191             NMRmax = 0
192             NMRm = 0
193             s = 0
194 
195             # Don't need to store the values in an array,
196             # since in the end we mainly care about the average
197             # and maximum values. But for debugging and validation
198             # purposes, I'll include this.
199             R_NM = np.zeros(Nc)
200 
201             for k in xrange(Nc):
202                 NMRm = EbN[k] / (gm[k] * Ehs[k])
203                 R_NM[k] = NMRm # Remove later!
204                 s = s + NMRm
205 
206                 if (NMRm > NMRmax):
207                     NMRmax = NMRm
208 
209             NMR['NMRmax'] = NMRmax
210             NMR['NMRavg'] = float(s)/Nc
211 
212             return NMR
213 
```

Рисунок А.7 –Функції тестів №4

```
214         def PQ_MaskOffset(self, dz, Nc):
215             gm = np.zeros(Nc)
216             for k in xrange(Nc):
217                 if (k <= 12./dz):
218                     mdB = 3
219                 else:
220                     mdB = 0.25*k*dz
221                 gm[k] = 10**(-1*float(mdB)/10)
222             return gm
```

Рисунок А.8 –Функції тестів №5

```
224
225         ## ---------------- Averaging ------------------- ##
226         ## Time averaging functions for MOVs
227         ## Same naming convention as Kabal
228         ##
229
230         def PQ_avgBW(self, BWRef, BWTest):
231                 # I think this is just an average of all the
232                 # positive values, as far as I can tell...
233                 # Our implementation is simpler too, becuase we aren't worried about stereo
234                 # Ok, so these values don't exactly match Octave, but they are pretty close (+
235                 BandwidthRefB = np.mean(BWRef[BWRef >=0])
236                 BandwidthTestB = np.mean(BWTest[BWTest >=0])
237
238                 return BandwidthRefB, BandwidthTestB
239
240         def PQ_avgNMRB(self, NMRavg, NMRmax):
241                 #Average NMR values, we also get another MOV here for free - RelDistFramesB
242                 #This has been validated against Octave, appears to match very well.
243
244                 totalNMRB = 10*np.log10(np.mean(NMRavg))
245
246                 #Threshold:
247                 Tr = 10**(1.5/10)
248                 relDistFramesB = np.mean(NMRmax>Tr)
249
250                 return totalNMRB, relDistFramesB
251
```

Рисунок А.9 –Функції тестів №6 та №7

ДОДАТОК Б

Таблиці додаткових параметрів методу PEAQ

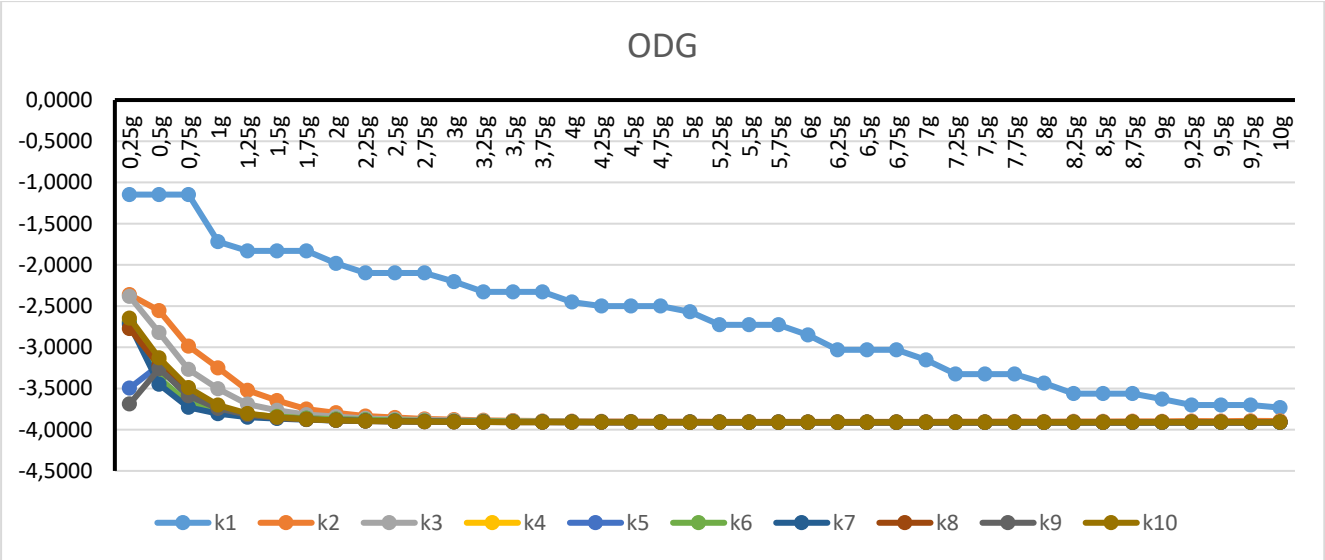

Рисунок Б.1 – Показник ODG для способу формування сигналів №1 при різних параметрах k для методу Лізи Марвел

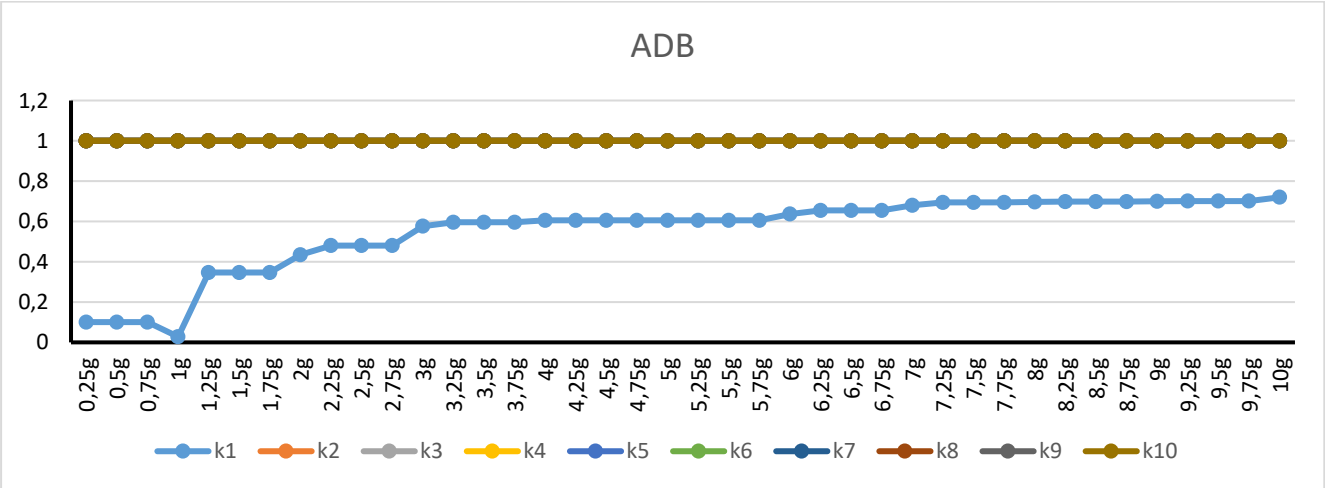

Рисунок Б.2 –Показник ADB для способу формування сигналів №1 при різних параметрах k для методу Лізи Марвел

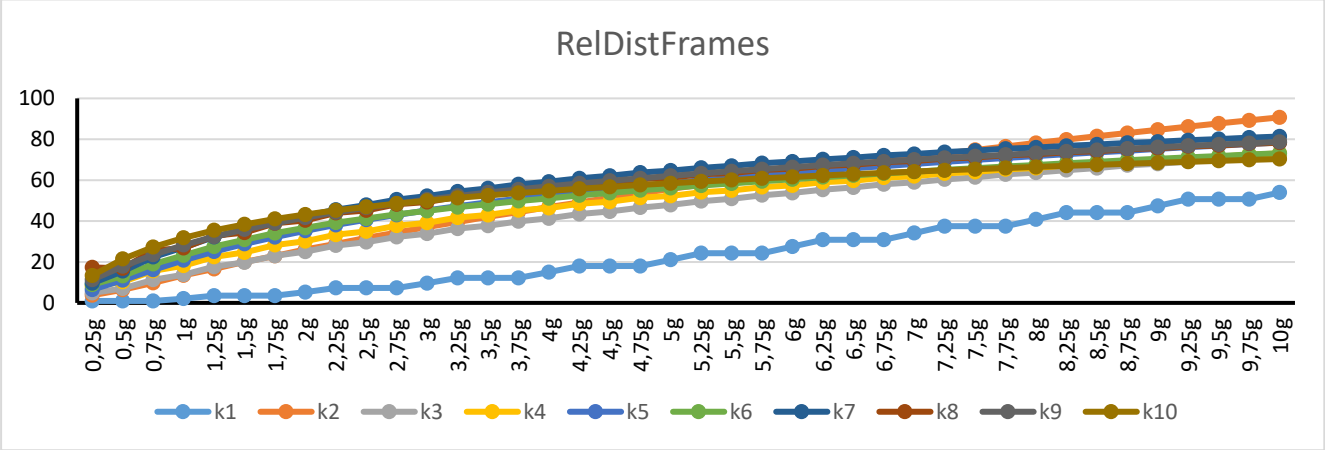

Рисунок Б.3 –Показник Relative Disturbed Frames для способу формування сигналів №1 при різних параметрах k для методу Лізи Марвел

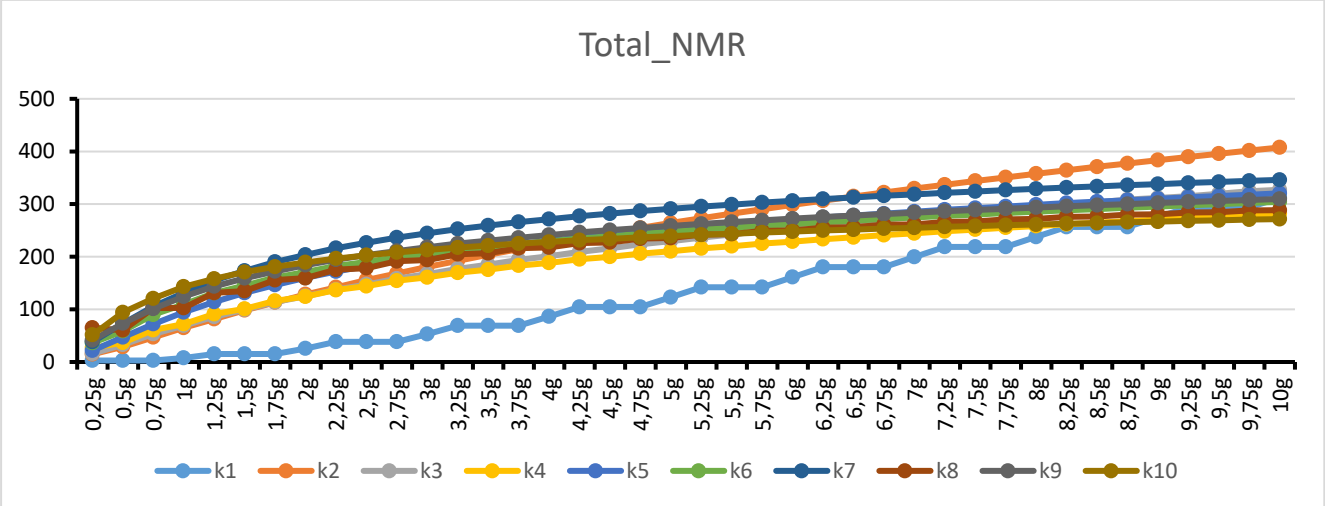

Рисунок Б.4 –Показник NMR для способу формування сигналів №1 при різних параметрах k для методу Лізи Марвел

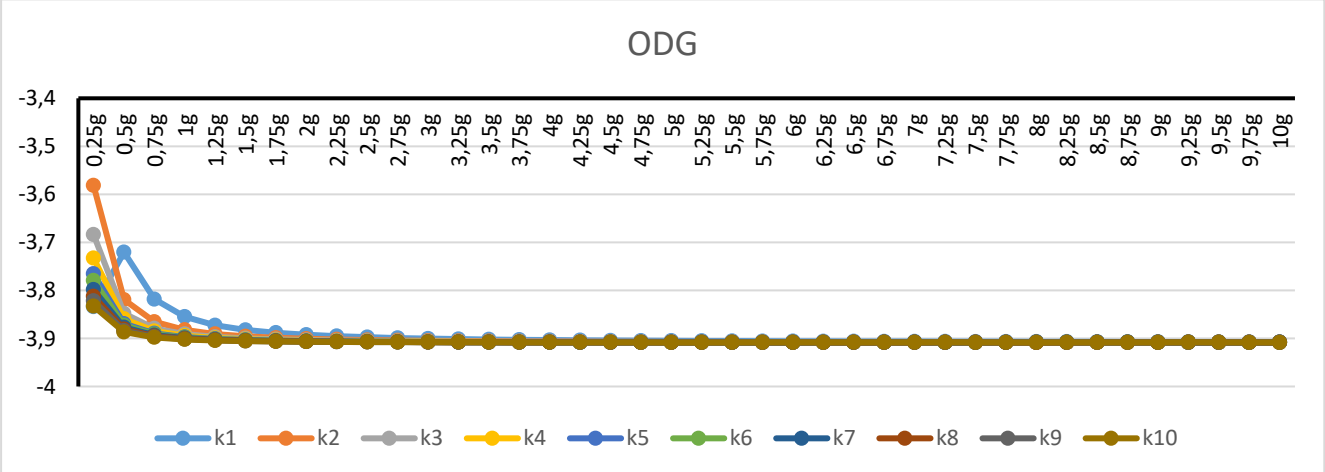

Рисунок Б.5 –Показник ODG для способу формування сигналів №2 при різних параметрах k для методу Лізи Марвел

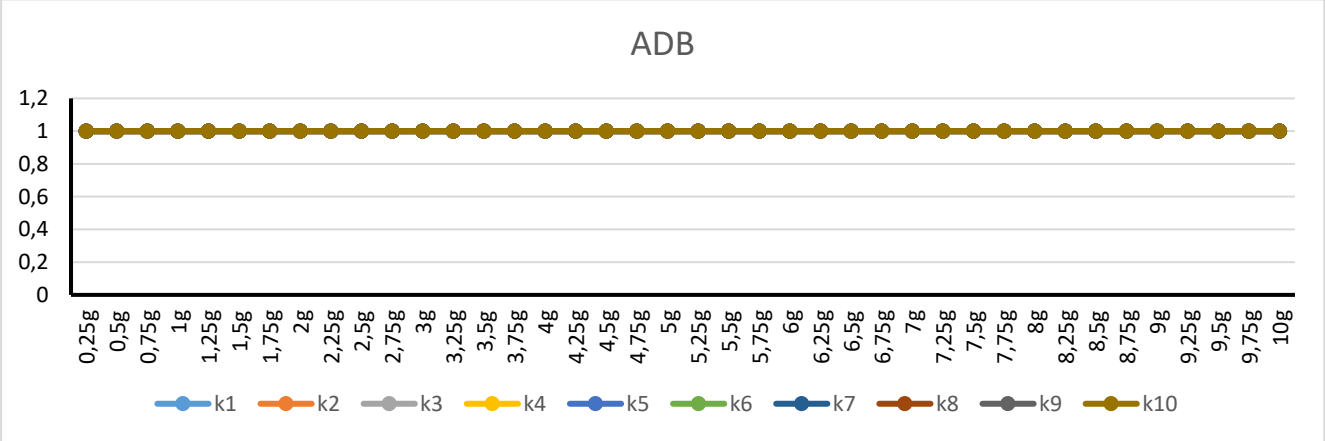

Рисунок Б.6 –Показник ADB для способу формування сигналів №2 при різних параметрах k для методу Лізи Марвел

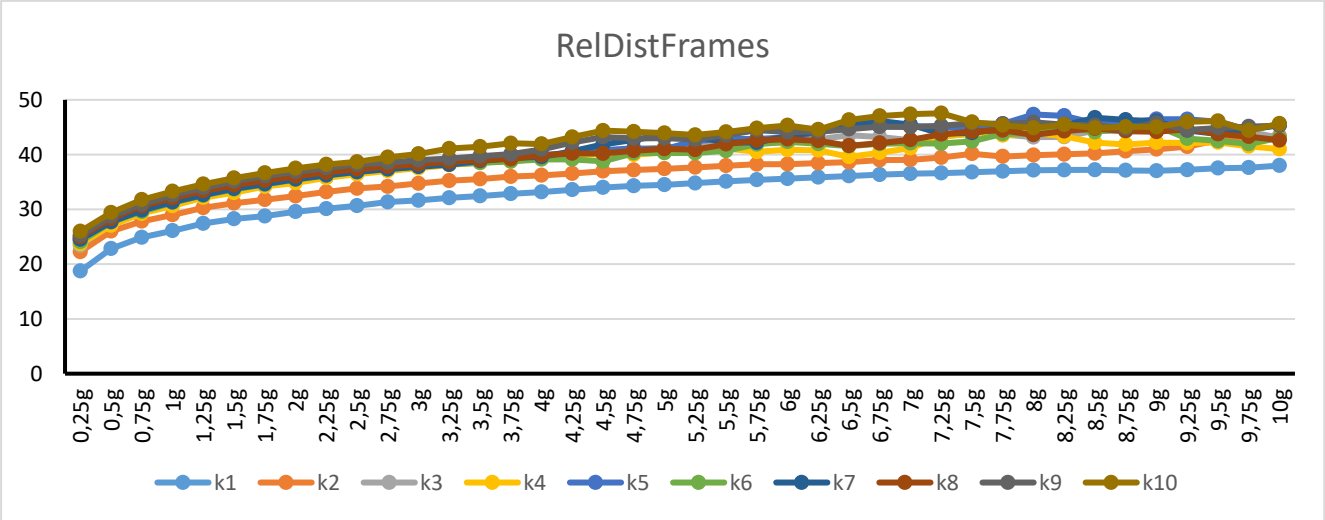

Рисунок Б.7 –Показник Relative Disturbed Frames для способу формування сигналів №2 при різних параметрах k для методу Лізи Марвел

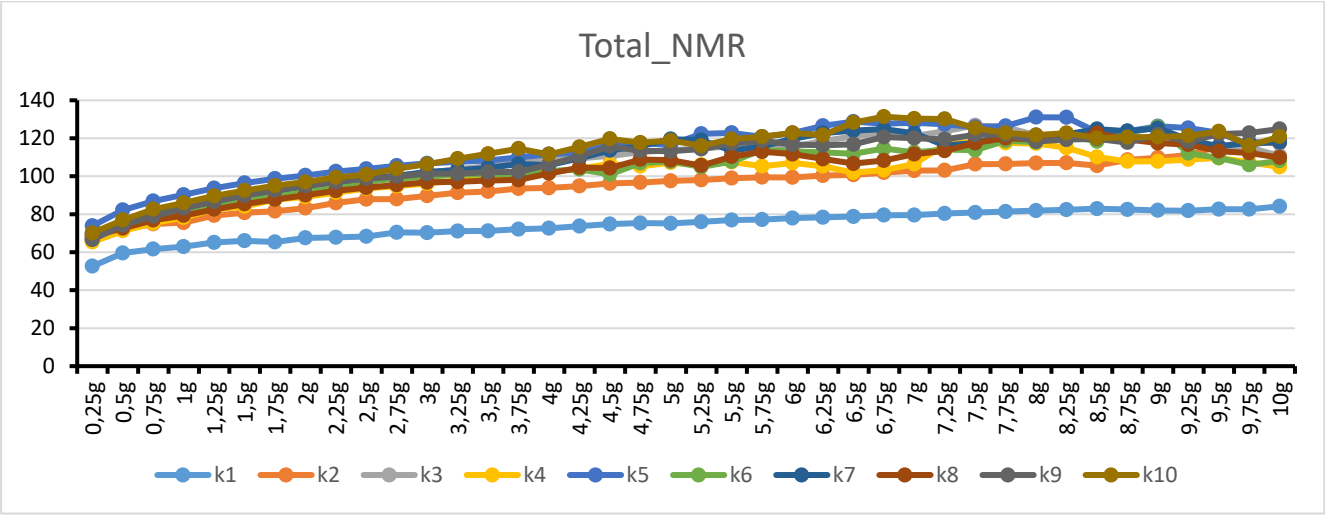

Рисунок Б.8 –Показник NMR для способу формування сигналів №2 при різних параметрах k для методу Лізи Марвел

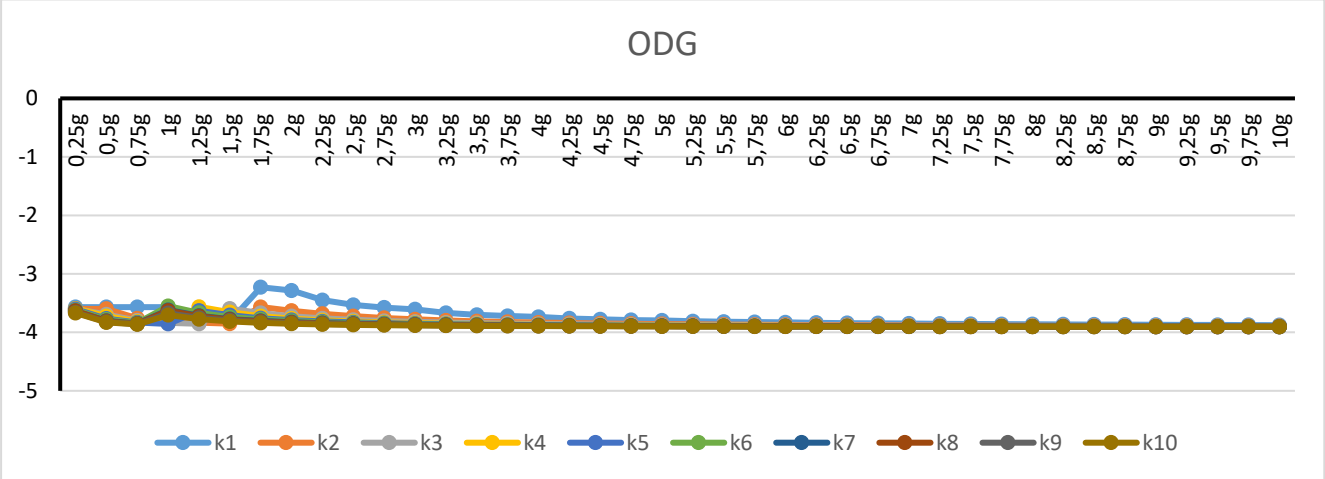

Рисунок Б.9 –Показник ODG для способу формування сигналів №3 при різних параметрах k для методу Лізи Марвел

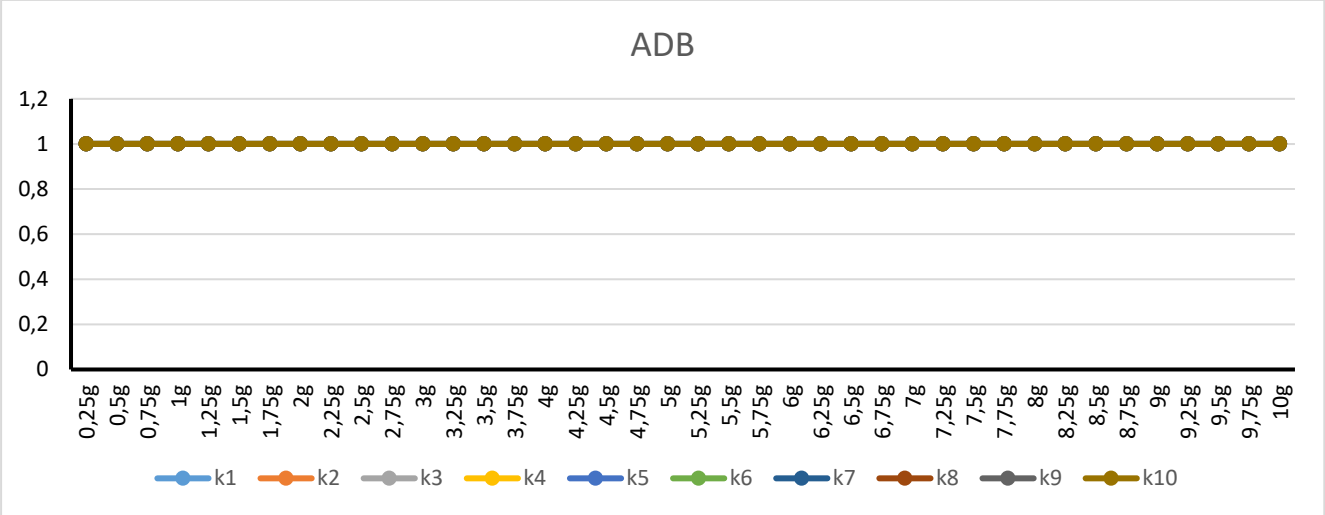

Рисунок Б.10 –Показник ADB для способу формування сигналів №3 при різних параметрах k для методу Лізи Марвел

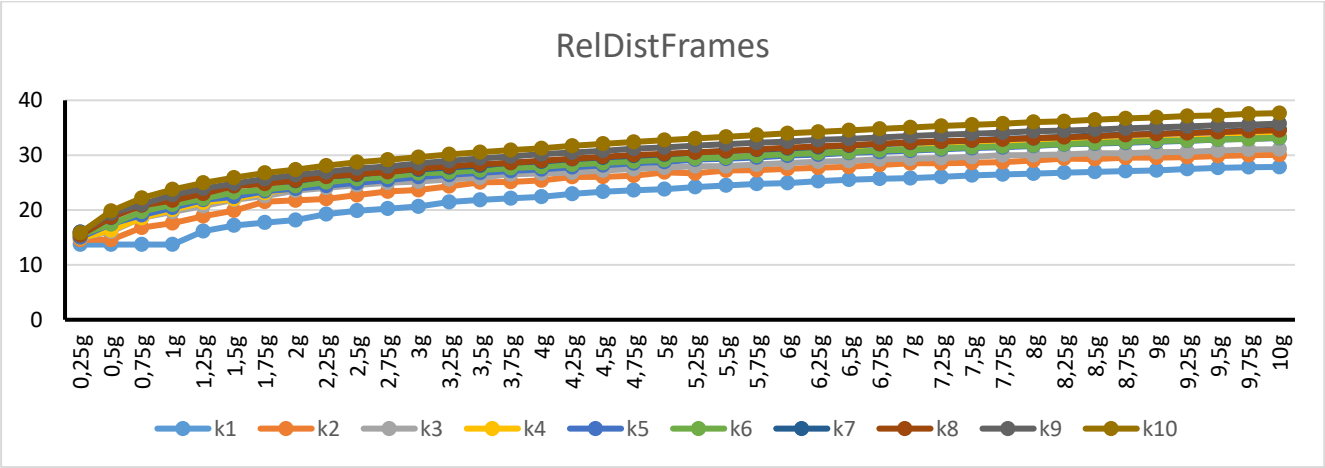

Рисунок Б.11 –Показник Relative Disturbed Frames для способу формування сигналів №3 при різних параметрах k для методу Лізи Марвел

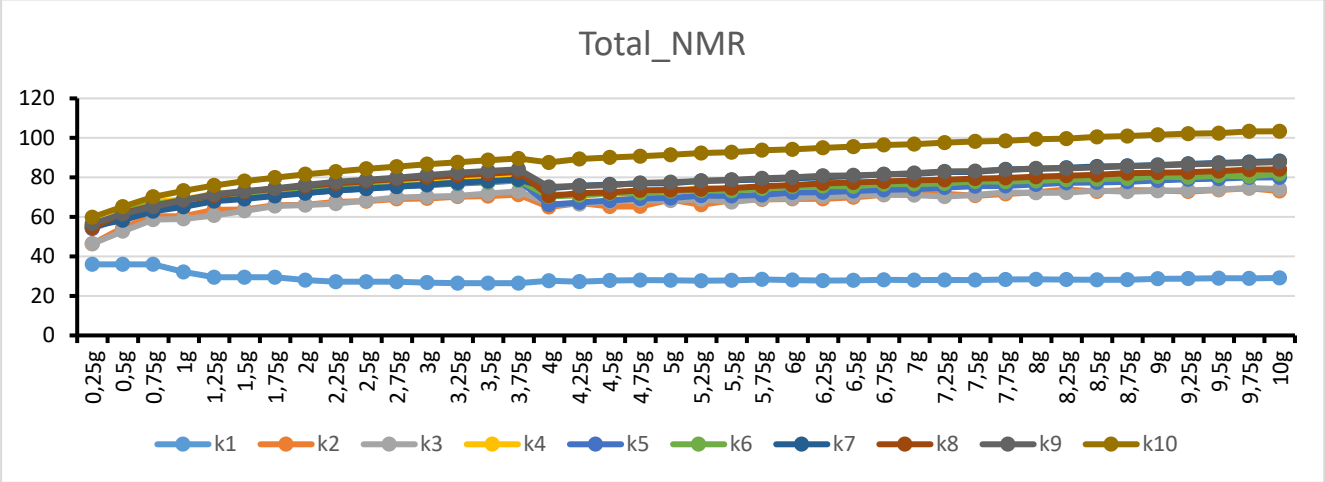

Рисунок Б.12 –Показник NMR для способу формування сигналів №3 при різних параметрах k для методу Лізи Марвел

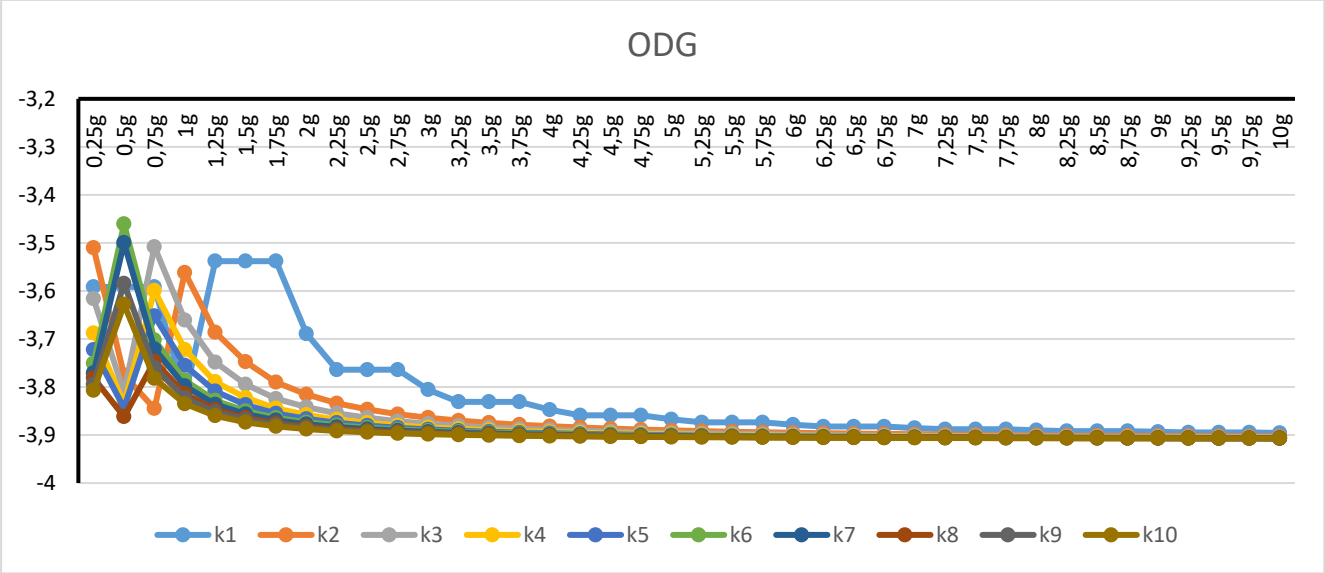

Рисунок Б.13 –Показник ODG для способу формування сигналів №4 при різних параметрах k для методу Лізи Марвел

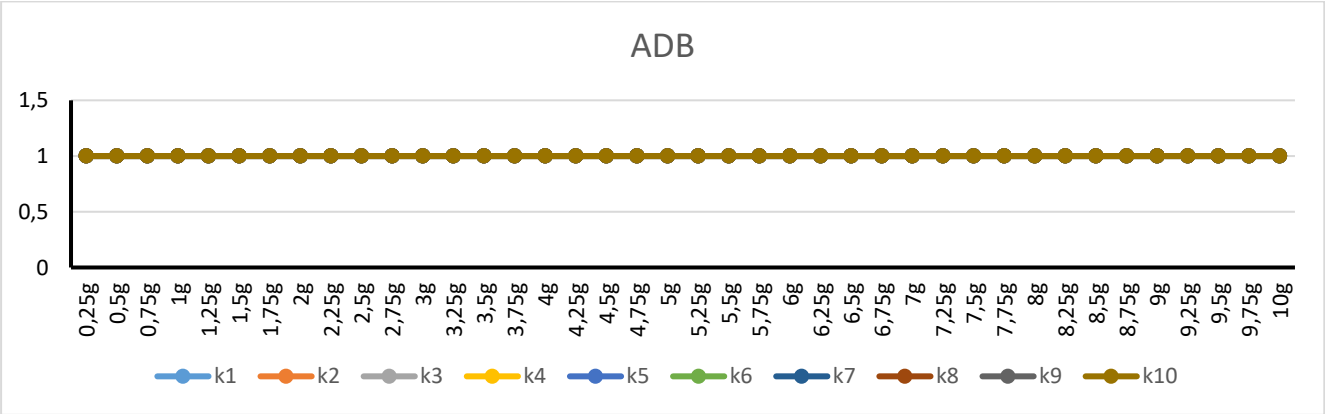

Рисунок Б.14 –Показник ADB для способу формування сигналів №4 при різних параметрах k для методу Лізи Марвел

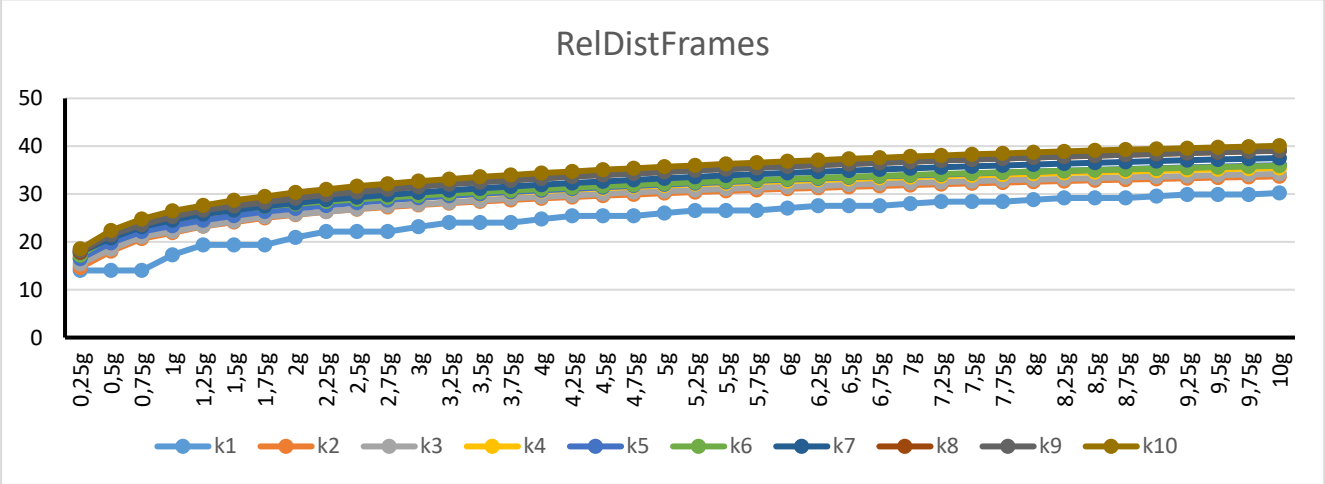

Рисунок Б.15 –Показник Relative Disturbed Frames для способу формування сигналів №4 при різних параметрах k для методу Лізи Марвел

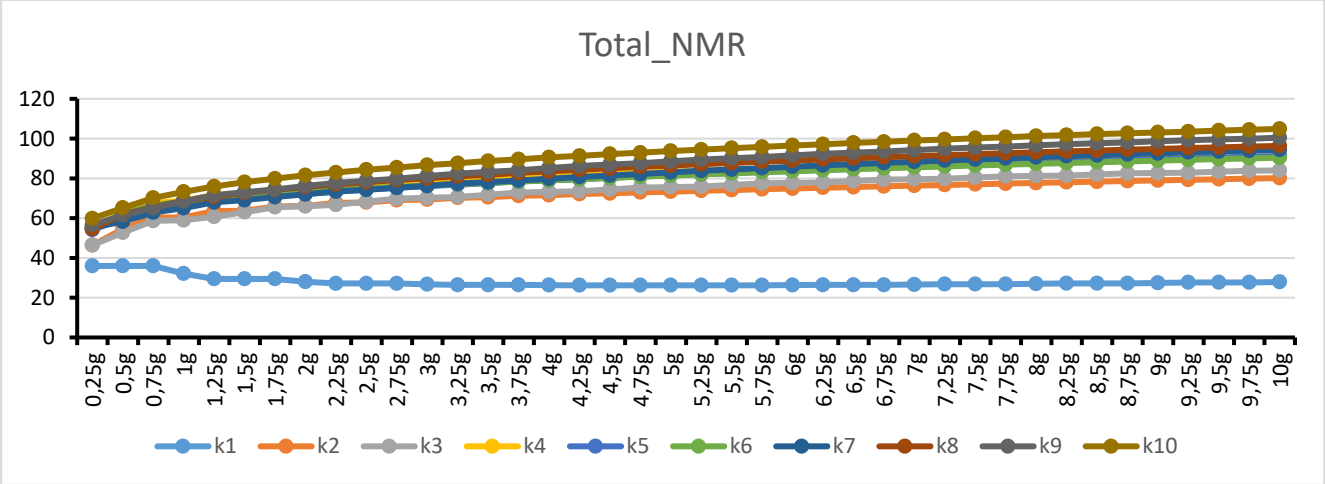

Рисунок Б.16 –Показник NMR для способу формування сигналів №4 при різних параметрах k для методу Лізи Марвел

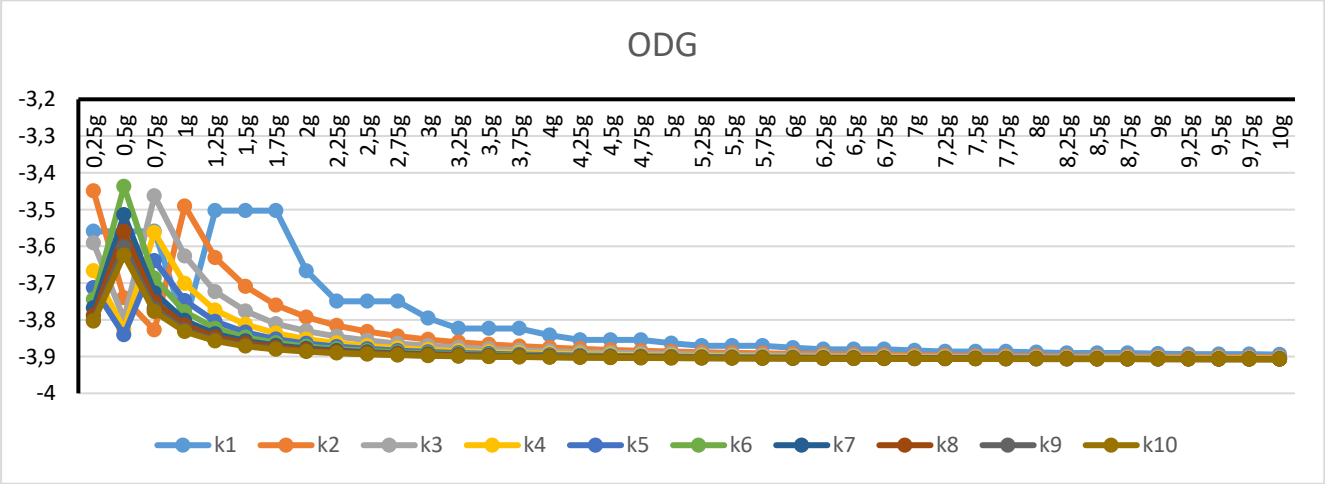

Рисунок Б.17 – Показник ODG для способу формування сигналів №5 при різних параметрах k для методу Лізи Марвел

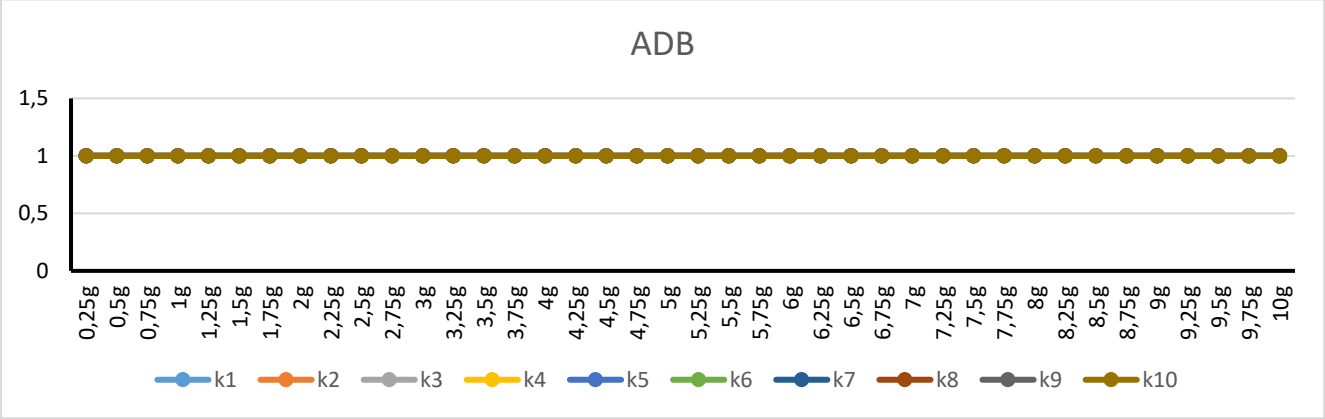

Рисунок Б.18 – Показник ADB для способу формування сигналів №5 при різних параметрах k для методу Лізи Марвел

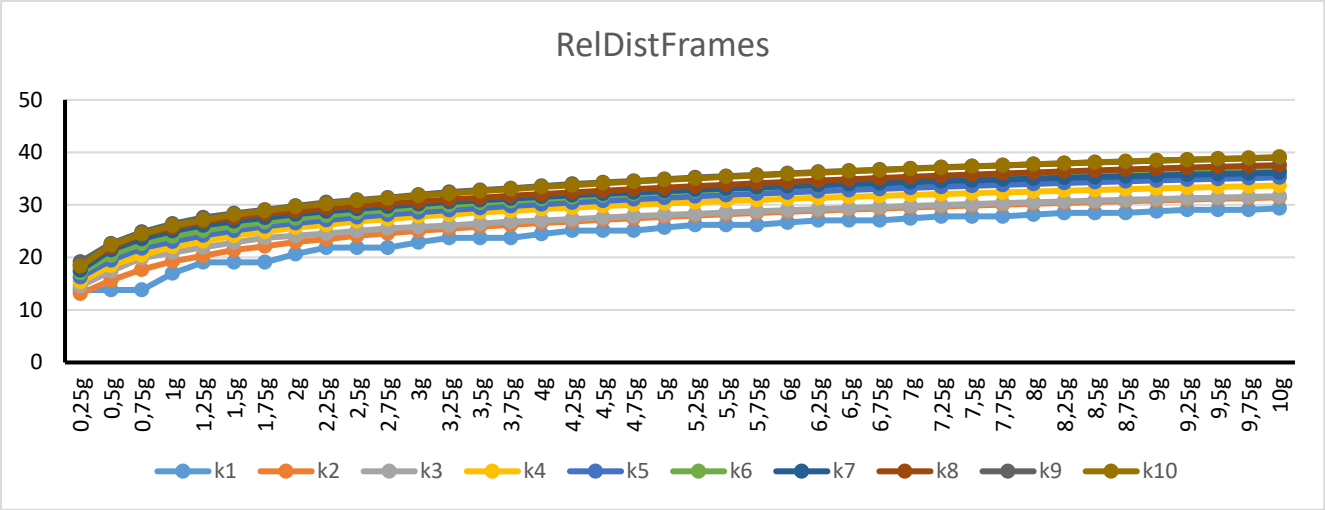

Рисунок Б.19 – Показник Relative Disturbed Frames для способу формування сигналів №5 при різних параметрах k для методу Лізи Марвел

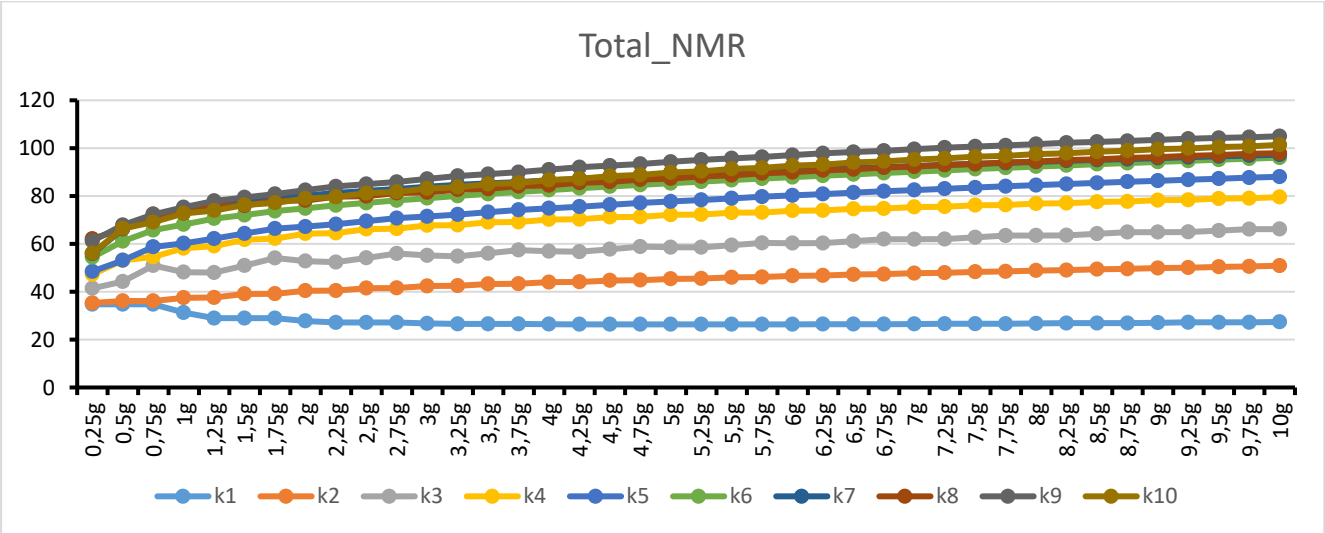

Рисунок Б.20 – Показник NMR для способу формування сигналів №5 при різних параметрах k для методу Лізи Марвел

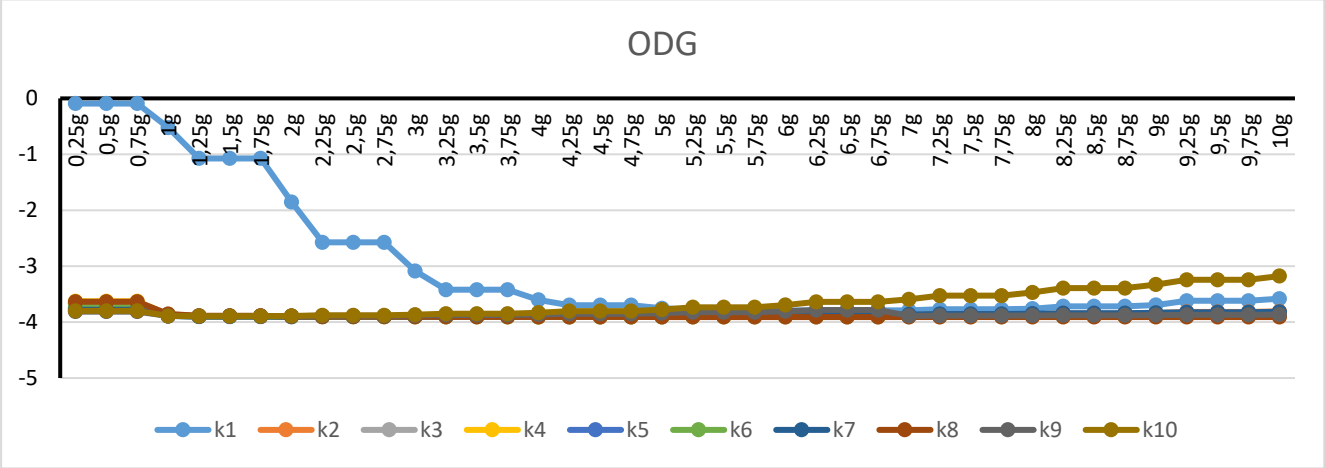

Рисунок Б.21 – Показник ODG для способу формування сигналів №1 при різних параметрах k для методу адресації

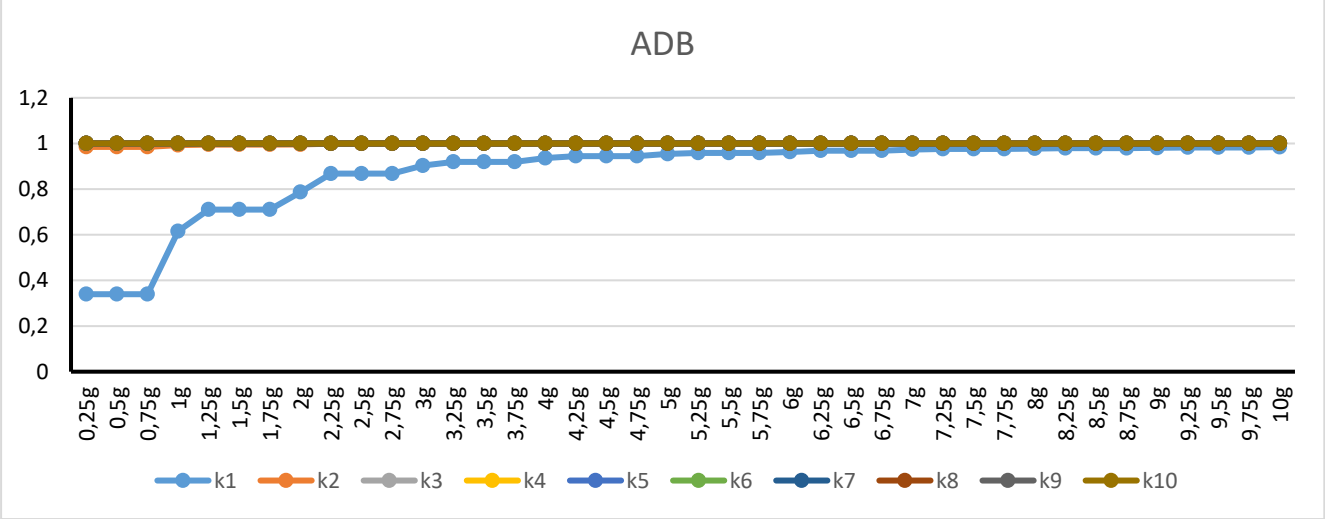

Рисунок Б.22 – Показник ADB для способу формування сигналів №1 при різних параметрах k для методу адресації

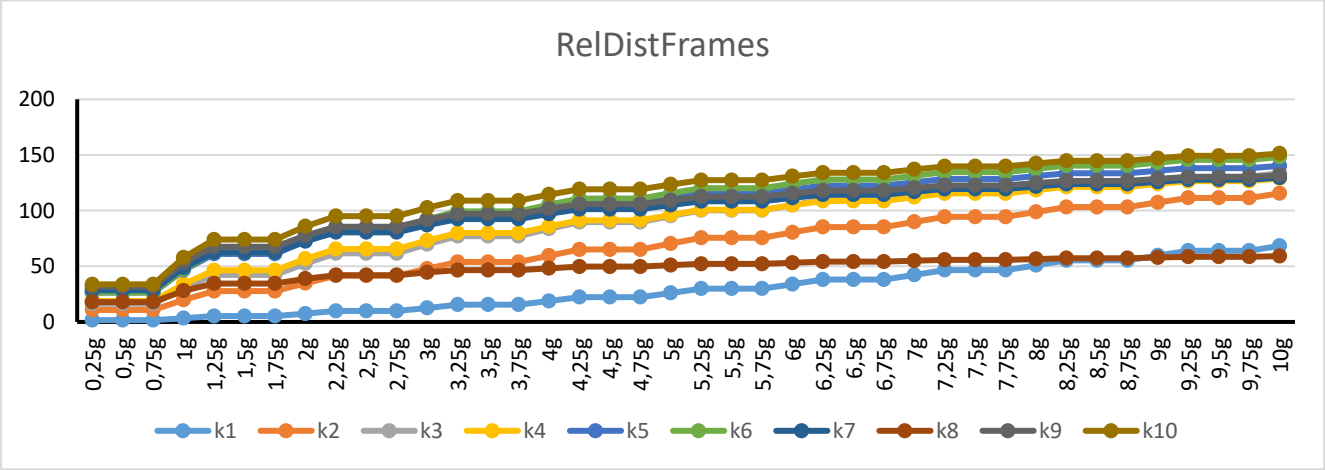

Рисунок Б.23 – Показник Relative Disturbed Frames для способу формування сигналів №5 при різних параметрах k для методу адресації

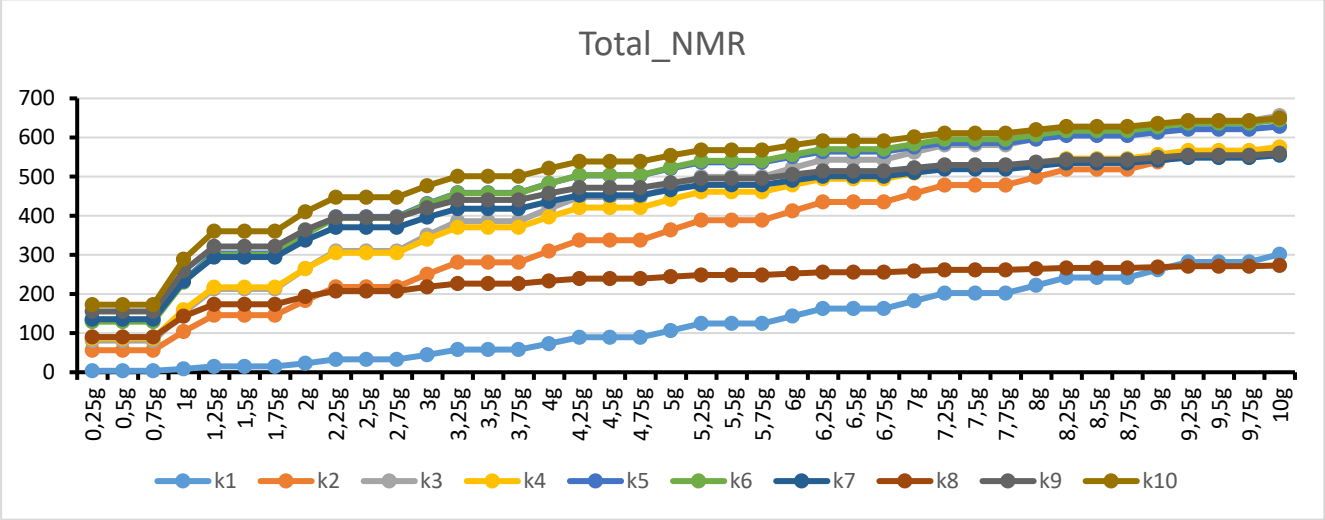

Рисунок Б.24 – Показник NMR для способу формування сигналів №1 при різних параметрах k для методу адресації

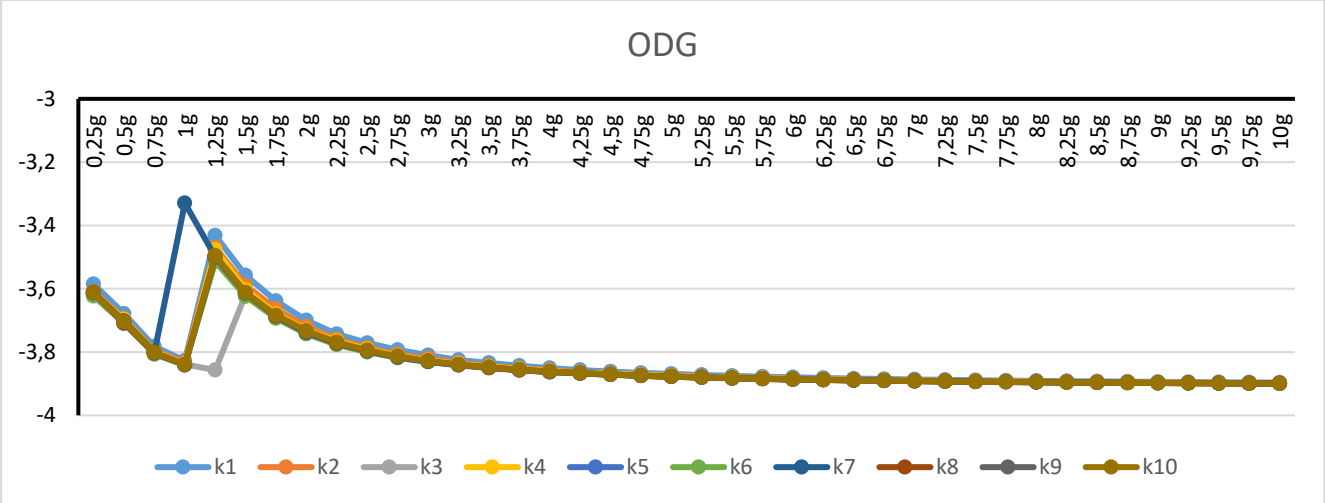

Рисунок Б.25 – Показник ODG для способу формування сигналів №2 при різних параметрах k для методу адресації

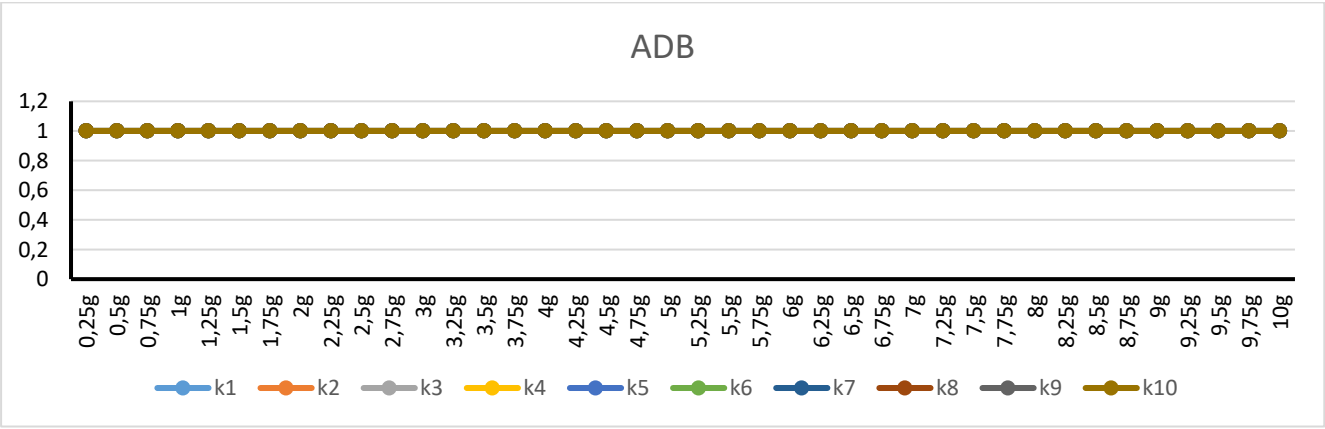

Рисунок Б.26 – Показник ADB для способу формування сигналів №2 при різних параметрах k для методу адресації

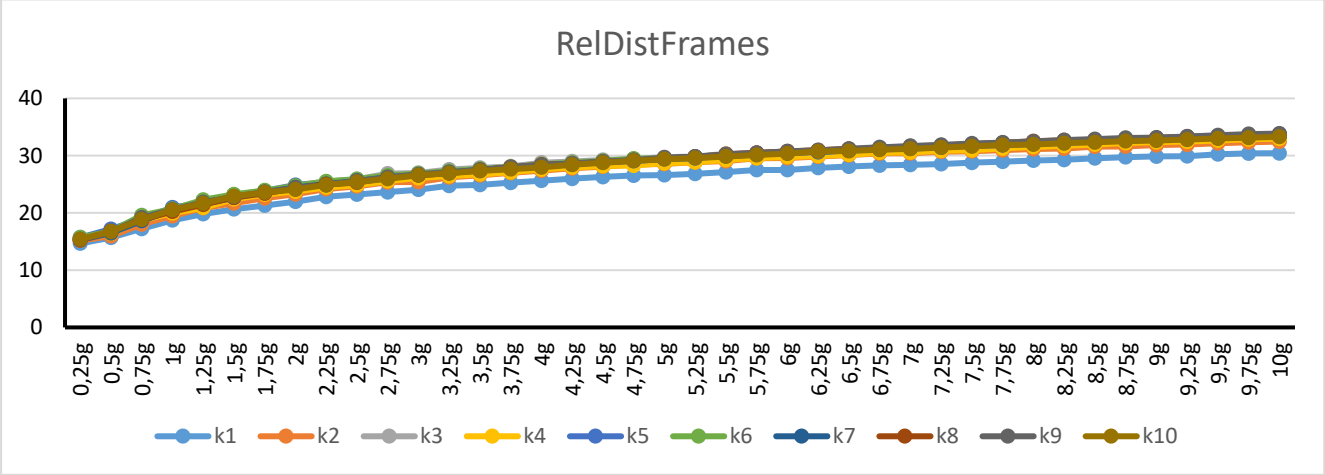

Рисунок Б.27 – Показник Relative Disturbed Frames для способу формування сигналів №2 при різних параметрах k для методу адресації

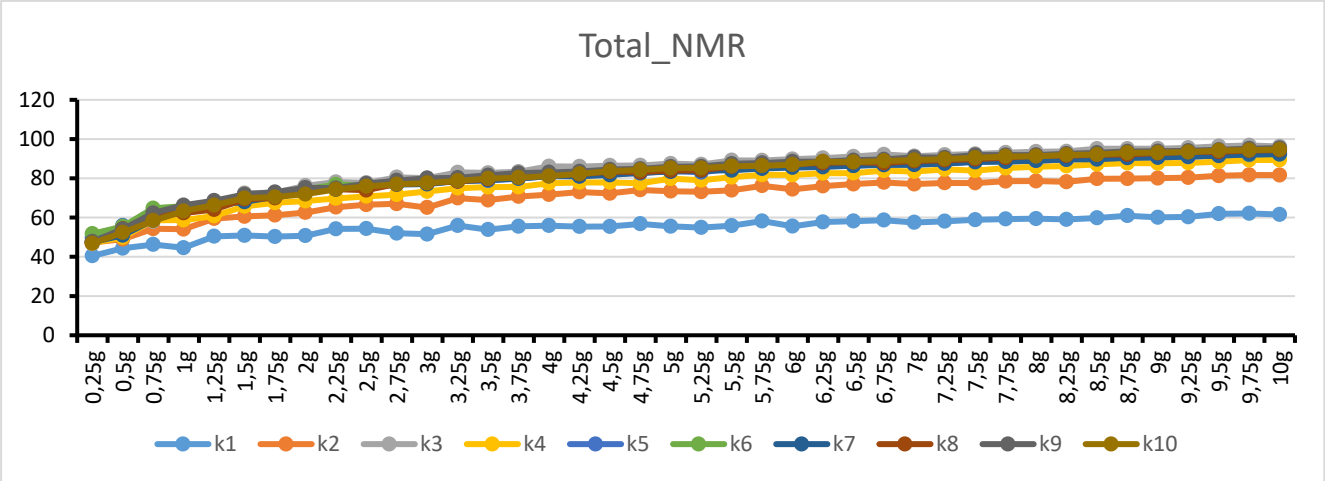

Рисунок Б.28 – Показник NMR для способу формування сигналів №2 при різних параметрах k для методу адресації

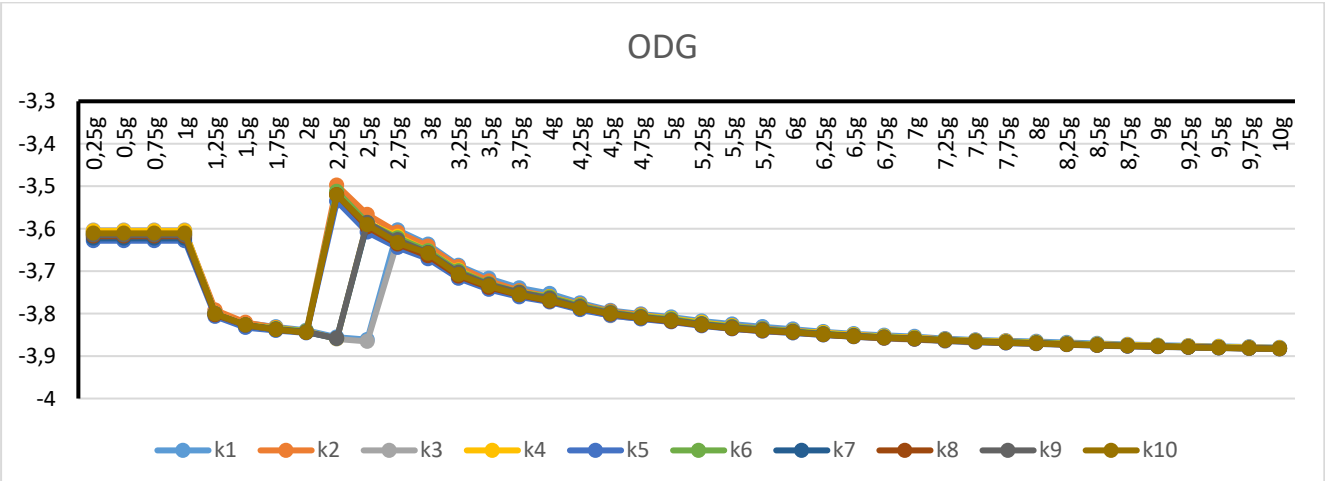

Рисунок Б.29 – Показник ODG для способу формування сигналів №3 при різних параметрах k для методу адресації

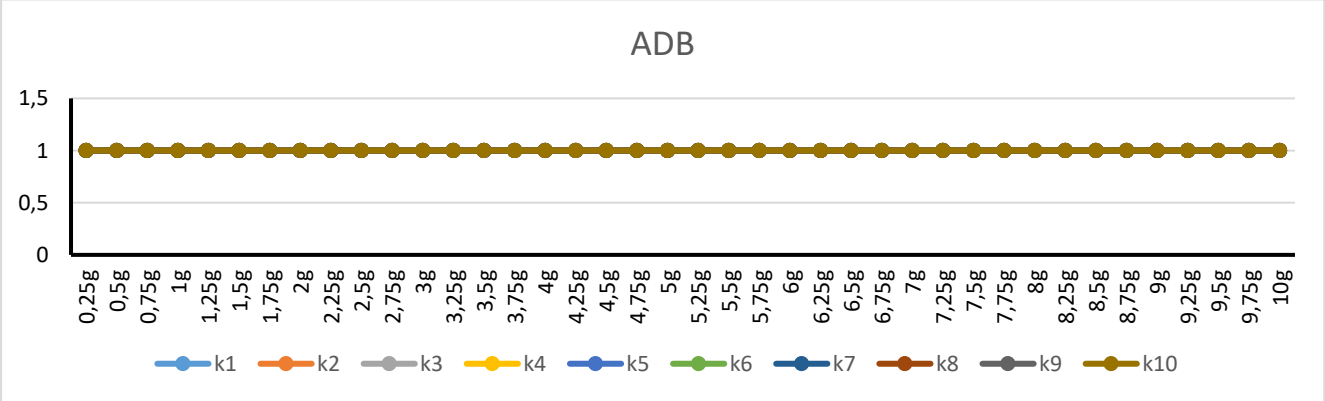

Рисунок Б.30 – Показник ADB для способу формування сигналів №3 при різних параметрах k для методу адресації

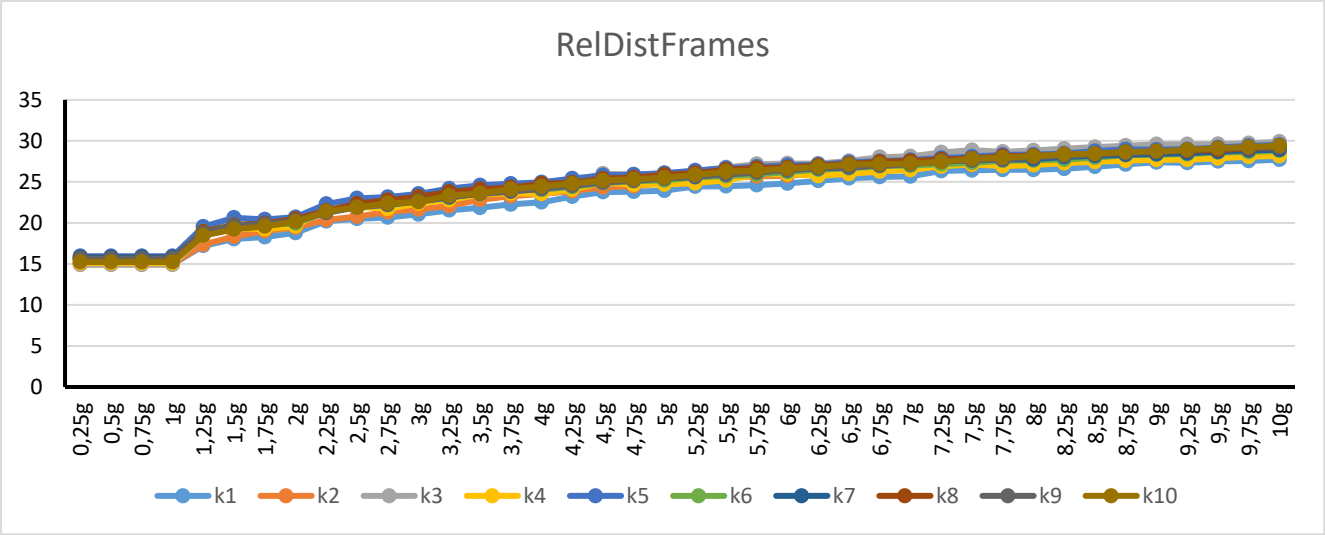

Рисунок Б.31 – Показник Relative Disturbed Frames для способу формування сигналів №3 при різних параметрах k для методу адресації

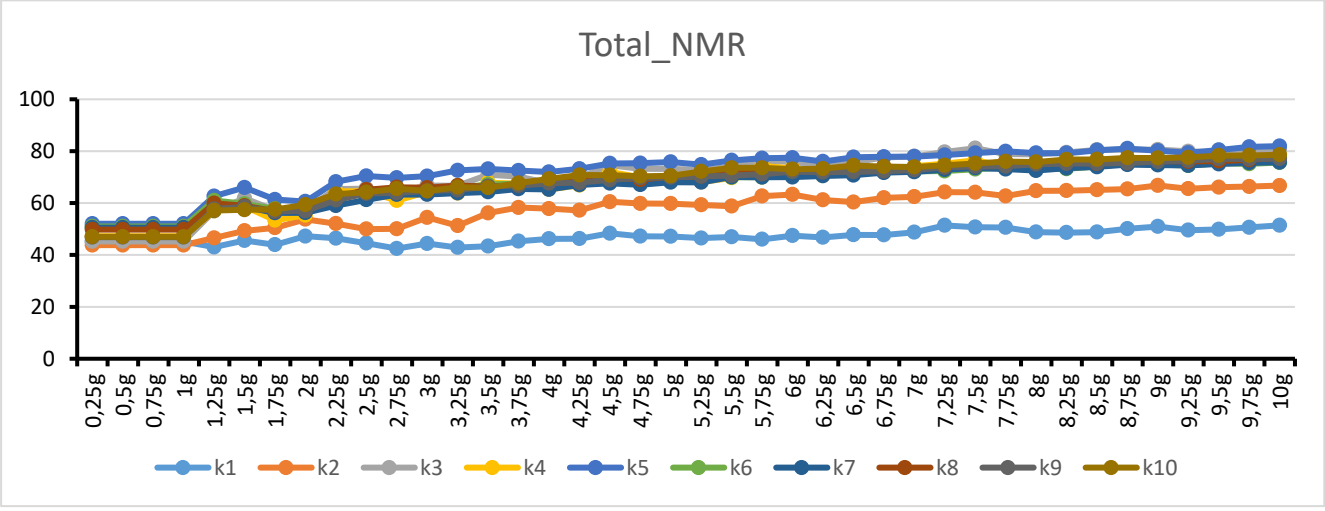

Рисунок Б.32 – Показник NMR для способу формування сигналів №3 при різних параметрах k для методу адресації

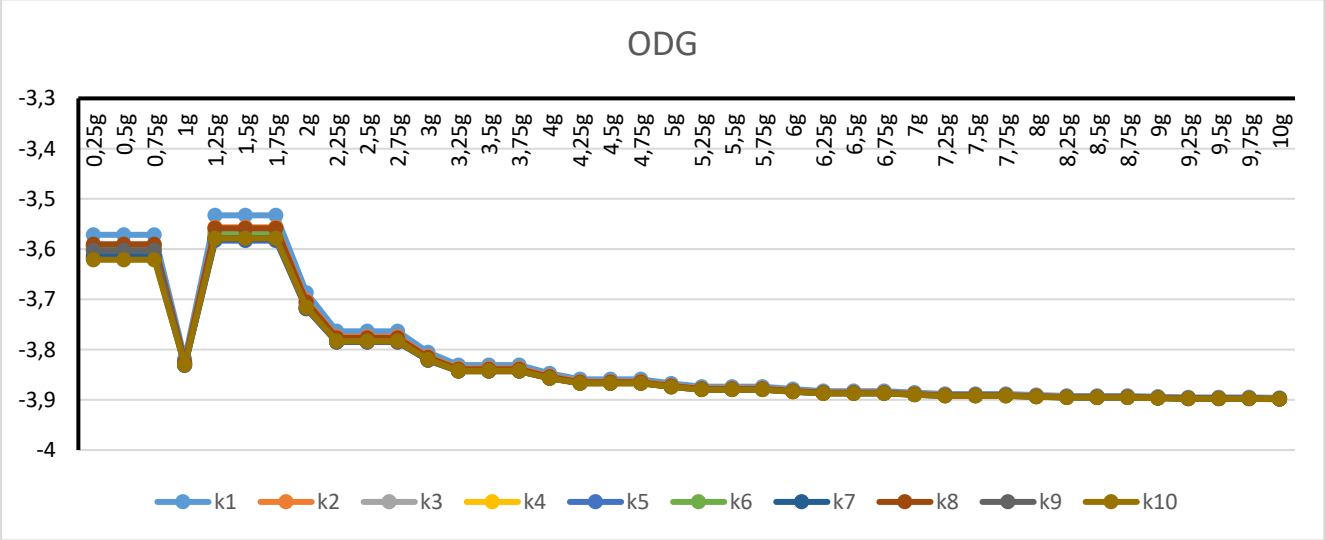

Рисунок Б.33 – Показник ODG для способу формування сигналів №4 при різних параметрах k для методу адресації

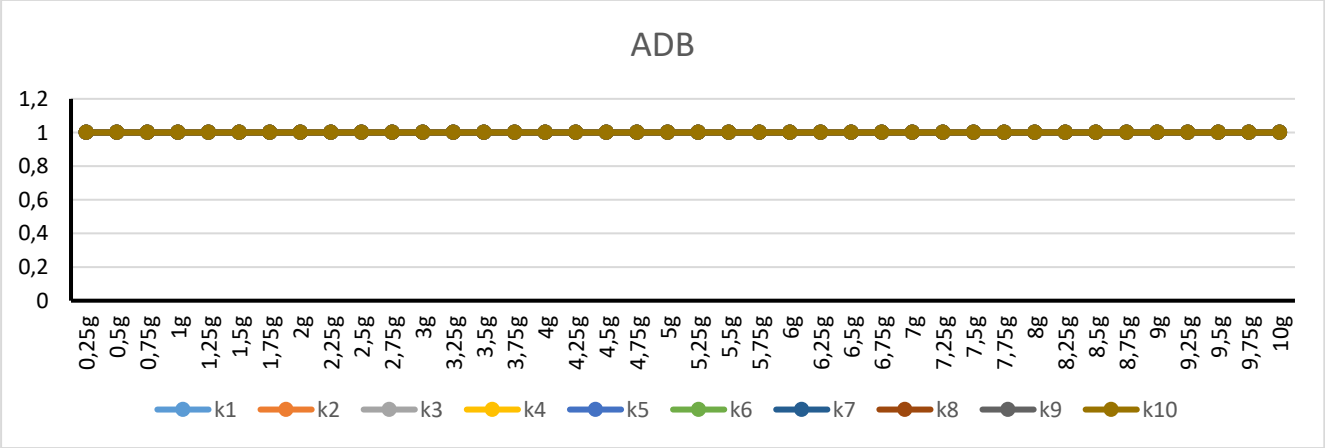

Рисунок Б.34 – Показник ADB для способу формування сигналів №4 при різних параметрах k для методу адресації

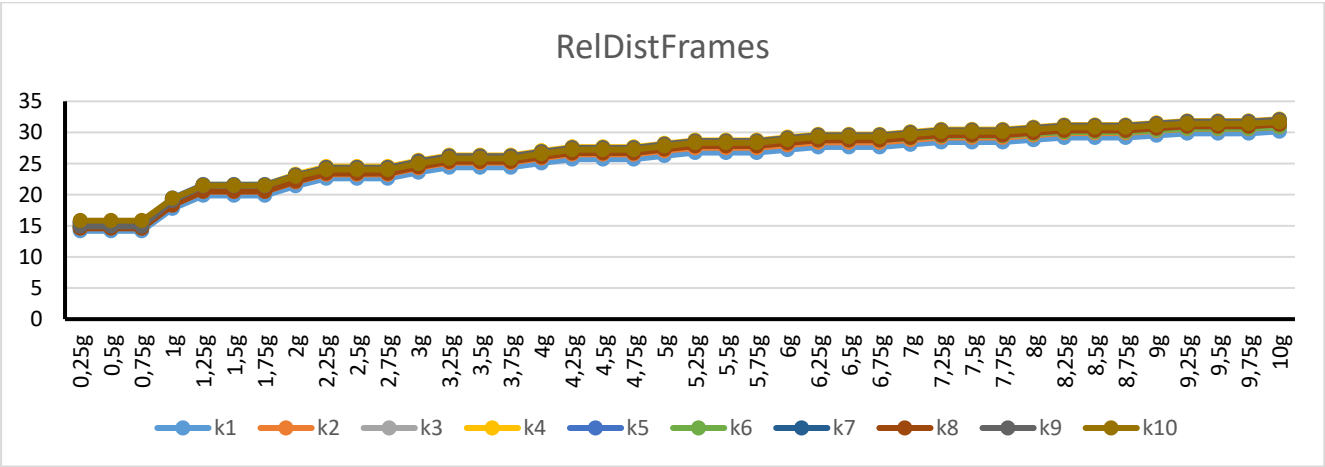

Рисунок Б.35 – Показник Relative Disturbed Frames для способу формування сигналів №4 при різних параметрах k для методу адресації

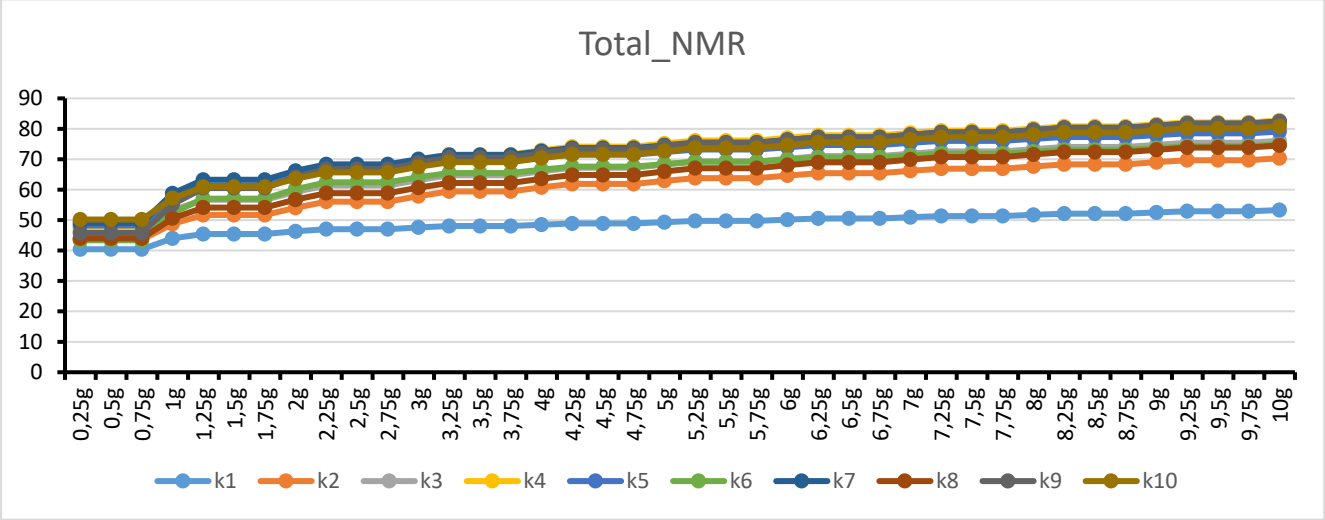

Рисунок Б.36 – Показник NMR для способу формування сигналів №4 при різних параметрах k для методу адресації

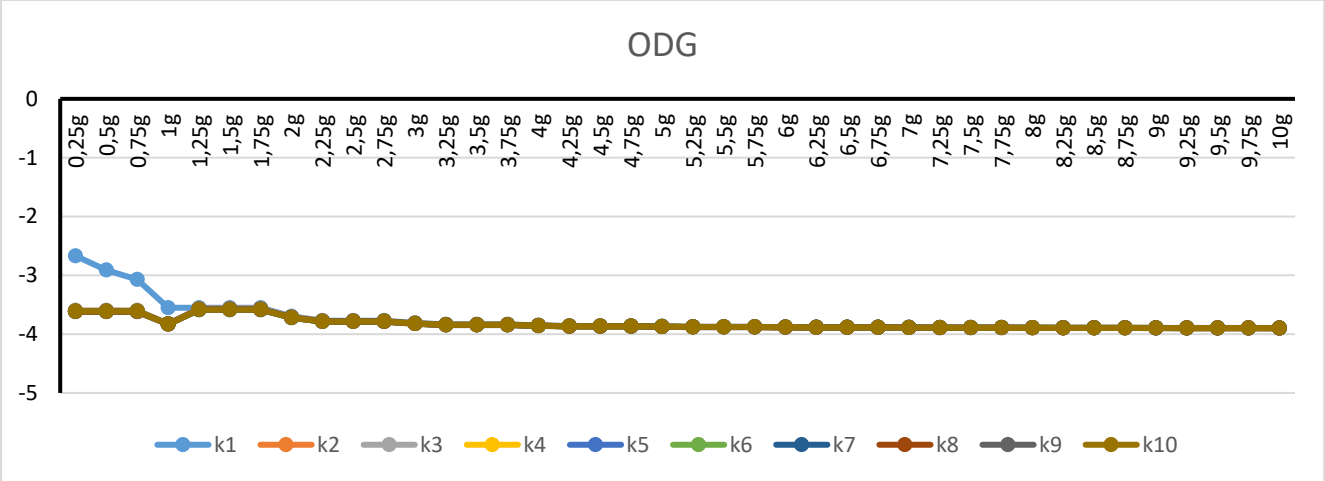

Рисунок Б.37 – Показник ODG для способу формування сигналів №5 при різних параметрах k для методу адресації

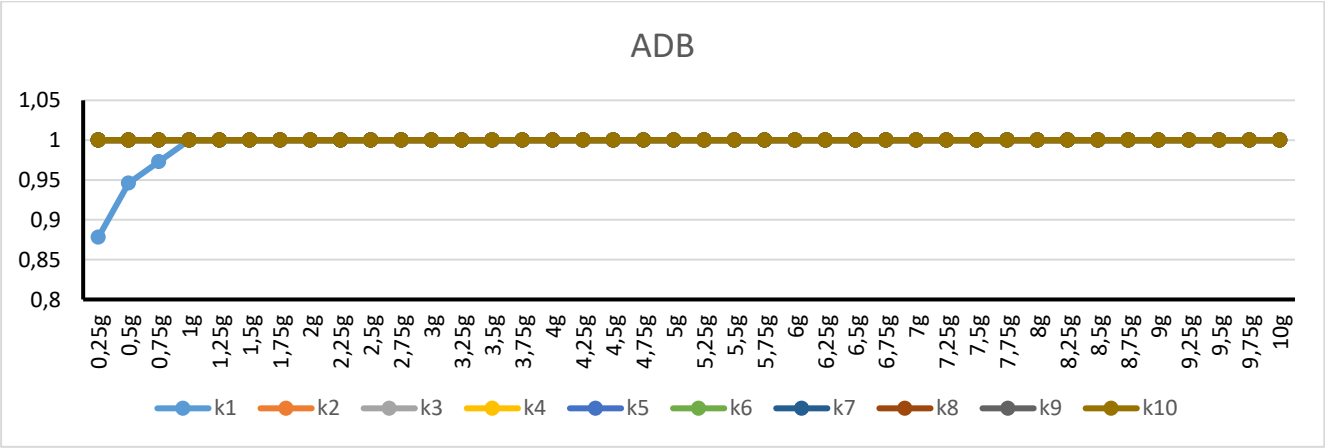

Рисунок Б.38 – Показник ADB для способу формування сигналів №5 при різних параметрах k для методу адресації

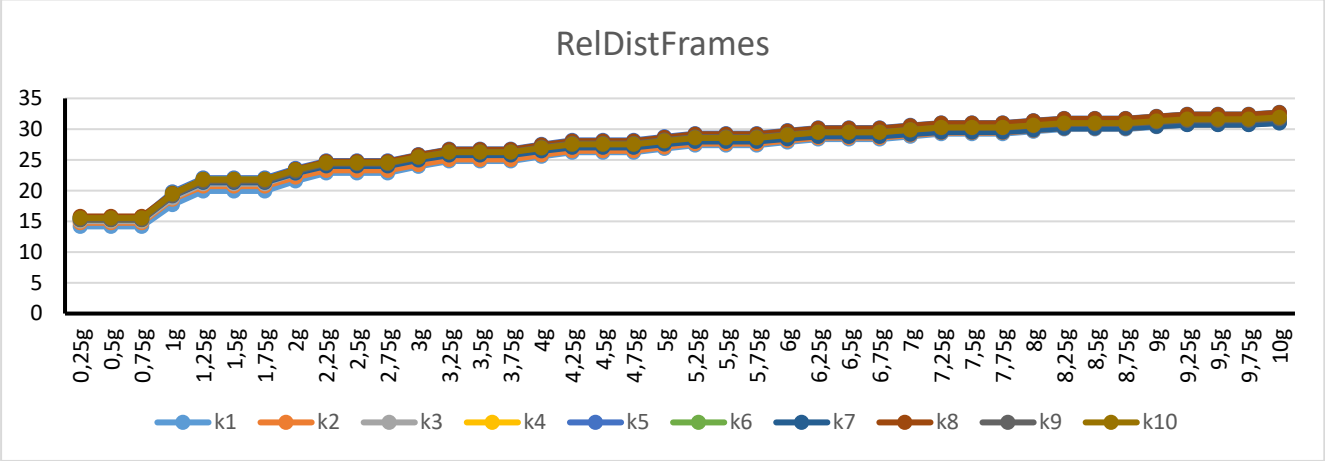

Рисунок Б.39 – Показник Relative Disturbed Frames для способу формування сигналів №5 при різних параметрах k для методу адресації

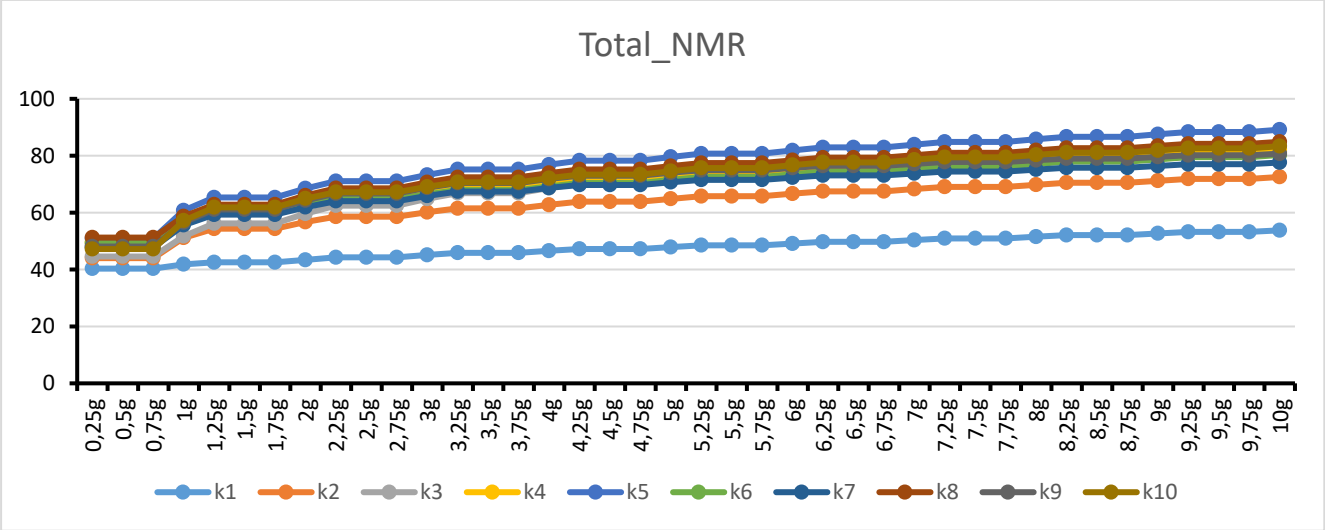

Рисунок Б.40 – Показник NMR для способу формування сигналів №5 при різних параметрах k для методу адресації