МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ ЦИФРОВИХ СИСТЕМ. ЧАСТИНА 1. ЦИФРОВА ОБРОБКА СИГНАЛІВ

ЛАБОРАТОРНИЙ ПРАКТИКУМ

Навчальний посібник

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для здобувачів ступеня бакалавра
за освітньою програмою
«Системи керування літальними апаратами та комплексами»
спеціальності 173 Авіоніка

Укладач: Ю.В. Бобков

Електронне мережне навчальне видання

Київ
КПІ ім. Ігоря Сікорського
2023

Рецензент *Смирнов С. А.,* канд. фіз.-мат. наук, с.н.с., заступник директора ННФТІ з методичних питань, доцент кафедри інформаційної безпеки КПІ ім. Ігоря Сікорського

Відповідальний редактор *Чепілко М. М.*, докт. фіз.-мат. наук, проф.

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського

(протокол № 6 від 30.03.2023 р.)

за поданням Вченої ради Навчально-наукового інституту аерокосмічних технологій (протокол № 3 від 27.03.2023 р.)

Навчальний посібник призначений для забезпечення лабораторних занять з кредитного модуля «Математичне забезпечення цифрових систем. Частина 1. Цифрова обробка сигналів», що належить до нормативних навчальних дисциплін циклу загальної підготовки для здобувачів ступеня бакалавра спеціальності 173 «Авіоніка», за освітньою програмою «Системи керування літальними апаратами та комплексами».

Основними завданнями лабораторного практикуму є закріплення знань, отриманих при вивченні теоретичного курсу, та отримання практичних умінь і досвіду з побудови математичних моделей цифрових сигналів, дослідження операцій дискретизації сигналів, лінійного та нелінійного квантування в системах цифрової обробки сигналів за допомогою програмних середовищ MatLab та Simulink.

Матеріали посібника можуть бути корисними здобувачам інших спеціальностей, а також фахівцям, що займаються проектуванням і дослідження систем цифрової обробки сигналів та інформаційно-вимірювальних систем із застосуванням програмних середовищ MatLab та Simulink.

Реєстр. № НП 22/23-624. Обсяг 3,9 авт. арк.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
проспект Перемоги, 37, м. Київ, 03056
https://kpi.ua
Свідоцтво про внесення до Державного реєстру видавців, виготовлювачів і розповсюджувачів видавничої продукції ДК № 5354 від 25.05.2017 р.

ЗМІСТ

ВСТУП

Навчальна дисципліна «Математичне забезпечення цифрових систем» належить до нормативних навчальних дисциплін циклу загальної підготовки здобувачів першого (бакалаврського) рівня вищої освіти за спеціальністю 173 «Авіоніка» за освітньо-професійною програмою «Системи керування літальними апаратами та комплексами».

На сучасному етапі розвитку систем авіоніки покращення їх характеристики та підвищення ефективності відбувається за рахунок новітніх досягнень в області цифрової техніки. Переважна кількість сучасних систем керування будується з розвиненою цифровою системою, одним з суттєвих елементів якої є саме цифрова обробка сигналів. Тому вивчення дисципліни «Математичне забезпечення цифрових систем» є необхідним та обов'язковим для здобувачів спеціальності 173 «Авіоніка».

Кредитний модуль «Математичне забезпечення цифрових систем. Частина 1. Цифрова обробка сигналів» формує у здобувачів вищої освіти теоретичні знання та практичний досвід у галузі цифрової обробки сигналів.

При виконанні лабораторних робіт здобувачі отримують необхідні практичні уміння та досвід з побудови математичних моделей цифрових сигналів за допомогою програмних середовищ MatLab та Simulink. Значна увага приділяється дослідженню таких важливих операцій систем цифрової обробки сигналів як операції дискретизації та квантування сигналів. Дослідження квантування розглядається на прикладі лінійного квантування, що є найбільш поширеним та реалізовано в звичайних аналого-цифрових перетворювачах, та нелінійного квантування, що є важливим для систем передачі цифрової інформації.

В теоретичних відомостях до лабораторних робіт наведено достатньо прикладів, що ілюструють застосування технологій та стандартних елементів MatLab та Simulink. Наведено необхідні алгоритми та детально викладені вимоги до порядку виконання робіт, підготовки звітів. Все це сприяє засвоєнню

матеріалу та дає змогу здобувачам самостійно виконувати завдання лабораторних занять.

В додатках до посібника наведені всі необхідні для виконання лабораторних робіт відомості щодо команд та операторів системи MatLab, а саме: арифметичні та логічні оператори мови MATLAB, стандартні змінні та константи; оператори організації циклів, математичні функції, команди побудови графіків.

Матеріали посібника можуть бути корисними здобувачам інших спеціальностей, а також фахівцям, що займаються проектуванням і дослідження систем цифрової обробки сигналів та інформаційно-вимірювальних систем із застосуванням програмних пакетів MatLab та Simulink

Лабораторна робота № 1

Дослідження основних методів побудови математичних моделей сигналів у системах MatLab та Simulink

Ціль роботи: вивчити основні методи побудови математичних моделей сигналів у системах MatLab і Simulink.

1.1 Теоретичні відомості

Система MATLAB (скорочення від MATRIX LABAROTATORY) є інтерактивною системою виконання інженерних і наукових розрахунків, що орієнтована працювати з масивами даних. Система допускає можливість звернення до програм, написаних мовами FORTRAN, C та C++.

Система MatLab підтримує виконання операцій з векторами, матрицями та масивами даних, реалізує сингулярне та спектральне розкладання, підтримує роботу з алгебраїчними поліномами, інтегрування у квадратурах, вирішення диференціальних та різницевих рівнянь. Слід відзначити, що ситема має необхідні засоби для реалізації апарату нечіткої логіки та нейромереж, забезпечує побудову різних видів графіків , тривимірних поверхонь та ліній рівня. [1]

У MatLab реалізовано зручне операційне середовище, яке дозволяє формулювати проблеми та отримувати рішення у звичній математичній формі, не вдаючись до рутинного програмування.

1.1.1 Програмування у середовищі MATLAB

Файли програми MATLAB поділяються на файли-сценарії та файли-процедури. Файл-сценарій – це зовнішній файл, що містить послідовність команд, операторів та функцій системи MATLAB. Після введення такого файлу в MATLAB починається послідовна інтерпретація його операторів. У файл-сценарій можна підключати файли–процедури. Відмінною особливістю файла-сценарію є наявність у першому його рядку команди *script*. Імена файлів–сценаріїв не можна використовувати як операнди у виразах або в якості аргументів функцій. [1,2]

Всі оператори, як у файлі-сценарії, так і у файлі-процедурі, відокремлюються один від одного крапкою з комою.

Файл-процедура (файл-функція) – це зовнішній файл, в якому реалізується виконання окремих процедур або функцій, і які призначені для виконання лише через файл-сценарій.

Стандартна структура файлу–процедури наступна:

1. Перший рядок оголошує ім'я функції, а також її вхідні та вихідні аргументи. Без такого рядка цей файл був би файлом-сценарієм.

2. Після оголошення імені функції повинні бути записані рядки коментарів, що містять довідкову інформацію про цю функцію, її призначення. На початку рядка коментаря має стояти знак %.

3. Змінні у тілі функції – це *локальні* змінні, на відміну змінних файлів – сценаріїв, які є *глобальними*.

Файли-процедури використовуються у файлі–сценарії так само, як і звичайні вбудовані функції системи MATLAB.

Синтаксис оголошення функції наступний:

Function [<список вихідних змінних>] = <ім'я функції>(<список вхідних змінних>).

Типове оформлення процедури-функції у MATLAB:

```
Function w = rmovel(v);
% RMOVEL видаляє з вектора v елементи, що збігаються;
%вхідні змінні: v – вхідний вектор;
%вихідні змінні: w – вектор, який не містить однакових елементів.
w = [ ];
while length(v)>0
w = [w; v(1)]; I = find(abs(v-v(1))  < 1e-6); v(I) = [ ]; I = [ ];
end
```

1.1.2 Оператори середовища MatLab

До операторів середовища MatLab відносяться:

1. Арифметичні та логічні оператори.
2. Стандартні змінні та константи.
3. Оператори організації циклів.
4. Математичні функції.
5. Оператори для побудови та роботи з графіками.
6. Оператори для роботи з файлами.

Опис цих операторів представлений у додатку А.

1.1.3 Приклад реалізації амплітудної модуляції сигналів

Для дослідження можливостей системи MatLab для побудови математичних моделей сигналів розглянемо приклад амплітудної модуляції синусоїдального сигналу за допомогою файлу-сценарію та файлу-процедури.

1.1.3.1 Амплітудна модуляція сигналу з використанням файлу-сценарію

Амплітудно-модульований сигнал визначається як добуток двох гармонічних сигналів, один з яких є модулюючим (огинаюча), а другий – модулюємим (несуча). При цьому частота модулюємого сигналу повинна бути більше частоти модулюючого сигналу. [4]

Алгоритм обчислення амплітудно-модульованого сигналу:

1 Визначити кількість відтворюваних періодів модулюючого сигналу, його частоту та амплітуду.

2 Визначити кількість відтворюваних періодів модулюємого сигналу, його частоту та амплітуду, а також крок дискретизації. Крок дискретизації вибирається з урахуванням коректного відображення форми всіх сигналів. Крок дискретизації для модулюємого та модулюючого сигналів повинен бути однаковим.

3 Задати модулюючий сигнал за допомогою математичної функції sin або cos.

4 Задати модулюємий сигнал за допомогою математичної функції sin або cos.

5 Перемножити два сигнали за допомогою множення масивів. При цьому довжини масивів мають бути однаковими.

6 Побудувати графіки заданих та отриманих сигналів.

Діапазон часу, в якому задаються сигнали, може задаватися як у радіанах, використовуючи стандартну константу π, так і в секундах, але в цьому випадку необхідно правильно вибрати частоту дискретизації.

Побудова графіків здійснюється за допомогою відповідних операторів (див. додаток А).

1.1.3.2 Амплітудна модуляція сигналу з використанням файлу-процедури

Як вже зазначалося вище, файл-процедура - це зовнішній файл, у якому реалізується виконання окремих процедур чи функцій, і який призначені до виконання лише через файл–сценарій. Для його створення мають бути задані вхідні та вихідні змінні.

Вхідними та вихідними змінними для нашого прикладу можуть бути:

- вхідні змінні: число відтворюваних періодів, крок дискретизації, амплітуда та частота модулюючого та модульованого сигналів;

- вихідні змінні: амплітудно-модульований сигнал.

При цьому в тілі файла-процедури може здійснюватися визначення модулюючого і модулюємого сигналів, та обчислення їх добутку. Алгоритм отримання амплітудно-модульованого сигналу аналогічний алгоритму, наведеному у п. 1.1.3.1.

Рекомендується задавати імена вхідних та вихідних змінних, які використовуються в тілі файлу-процедури, відмінними від аналогічних змінних, які використовуються в тілі файлу-сценарію.

1.1.3.3 Моделювання сигналів у Simulink

Розглянемо особливості моделювання гармонічних сигналів в **Simulink**. [1,2]

Візьмемо з бібліотеки блоків **Simulink** два блоки: **Sine Wave** (з розділу **Sources**) та **Scope** (з розділу **Sinks**). Поєднавши їх, отримаємо наступну схему (рисунок 1.1):

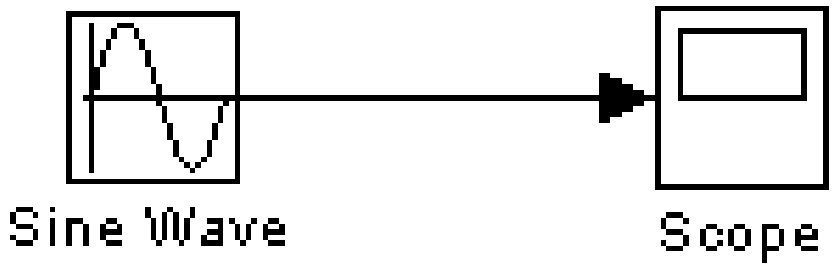


Рисунок 1.1 – Схема моделювання синусоїдального сигналу **Simulink**

Активізуємо вікно осцилоскопа подвійним кліком по блоку **Scope**. Запустимо модель (кнопка **Start simulation**).

В результаті отримаємо у вікні осцилоскопа зображення відрізка синусоїди (рисунок 1.2).

Розглянемо регулювання параметрами гармонічного сигналу. Для виклику вікна налаштування параметрів двічі клікнемо по блоку **Sine Wave** (рисунок 1.3).

Клікнувши по кнопці **Help**, отримаємо інструкцію по даному блоку, сутність якої зводиться до того, що в даному блоці виконується операція:

$$y = Amplitude \times \sin(frequency \times time + phase) + bias$$

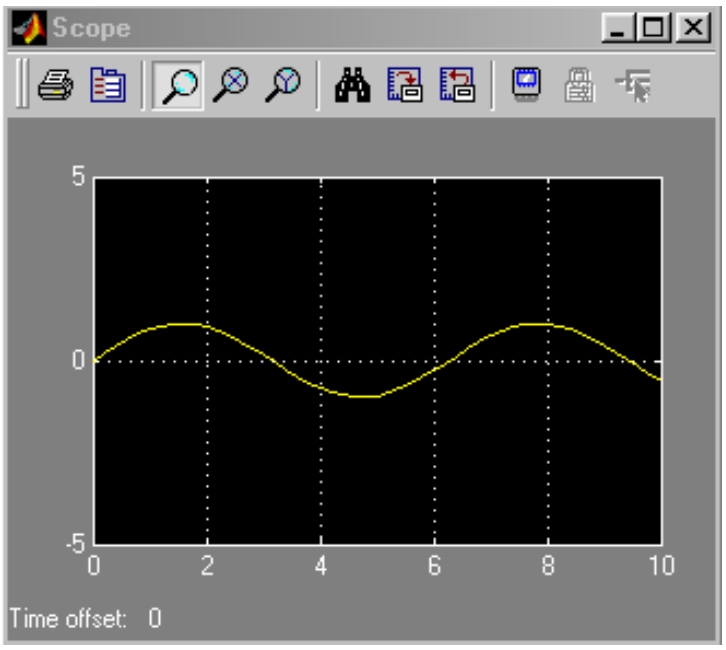


Рисунок 1.2 – Відображення синусоїдального сигналу в **Simulink**

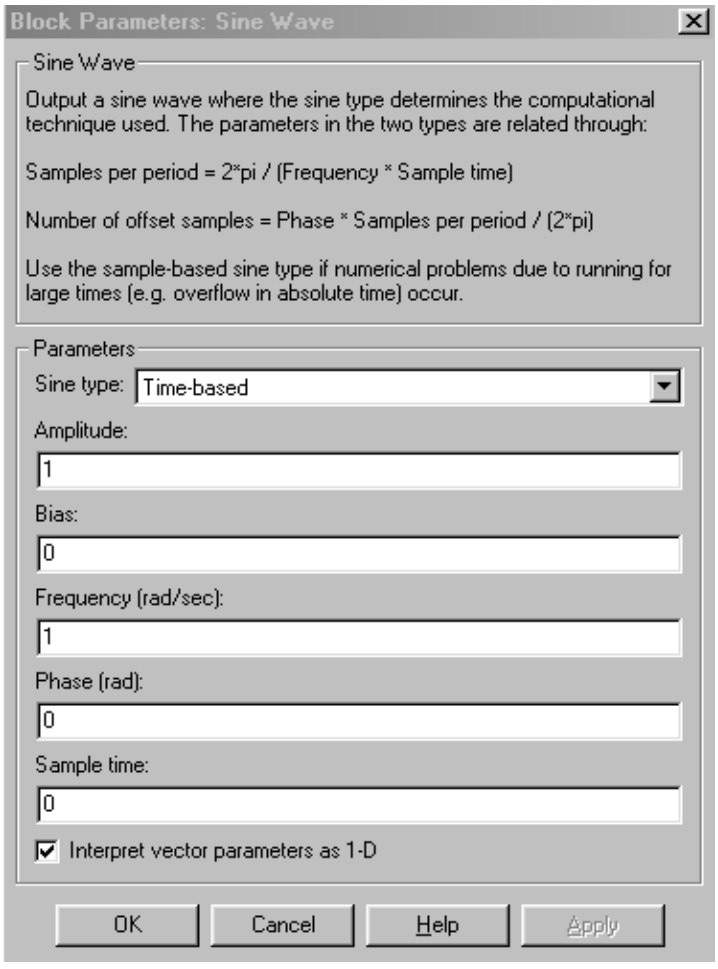


Рисунок 1.3 – Вікно параметрів блоку **Sine Wave**

З наведеної формули та написів на рисунку 1.3 зрозумілий сенс чотирьох змінних: амплітуди, кутової частоти, початкової фази та постійної складової (зміщення).

Для розуміння сенсу змінної “час” слід зазначити відмінність понять “час” і “модельний час”.

Так, генерування відрізка сигналу тривалістю 1 с (модельний час) може тривати значно коротший проміжок часу, наприклад, 0.1 с (реальний час). Швидкість генерування залежить від обсягу обчислень, швидкодії комп'ютера, обраного алгоритму моделювання і т. д. Цілком можливий зворотний ефект - для складного алгоритму процедура моделювання відрізка сигналу тривалістю 0.1 с може становити кілька секунд.

Сигнал може генеруватися двох типів: безперервний **time-based** та дискретний **sample-based**. Для моделювання роботи безперервних систем рекомендують використовувати безперервний тип **time-based**, а моделювання роботи дискретних систем – дискретний тип **sample-based**.

Для вирішення завдань даної лабораторної роботи використовуватимемо безперервний тип **time-based**. Тоді параметр **Sample time** може набувати значення:

- 0 (за замовчуванням) – блок працює у безперервному режимі;
- >0 - блок працює у дискретному режимі;
- -1 – блок наслідує той самий режим, що і блок, який приймає.

Робота в безперервному режимі може призводити до великих похибок генерації на великих проміжках модельного часу, що пов'язано з кроком дискретизації сигналу.

Крім зазначених вище параметрів, необхідно налаштувати параметри моделювання: задати початок і кінець модельного часу, а також вибрати алгоритм моделювання (тип «вирішувача»). На рисунку 1.4 показано вікно

параметрів моделювання, що активізується при виборі позиції меню **Simulation/Simulation parameters**.

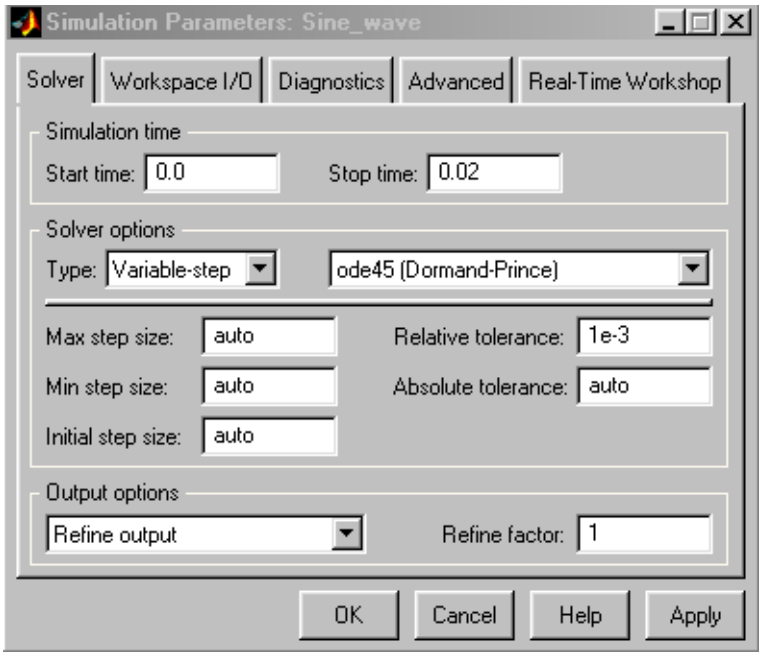
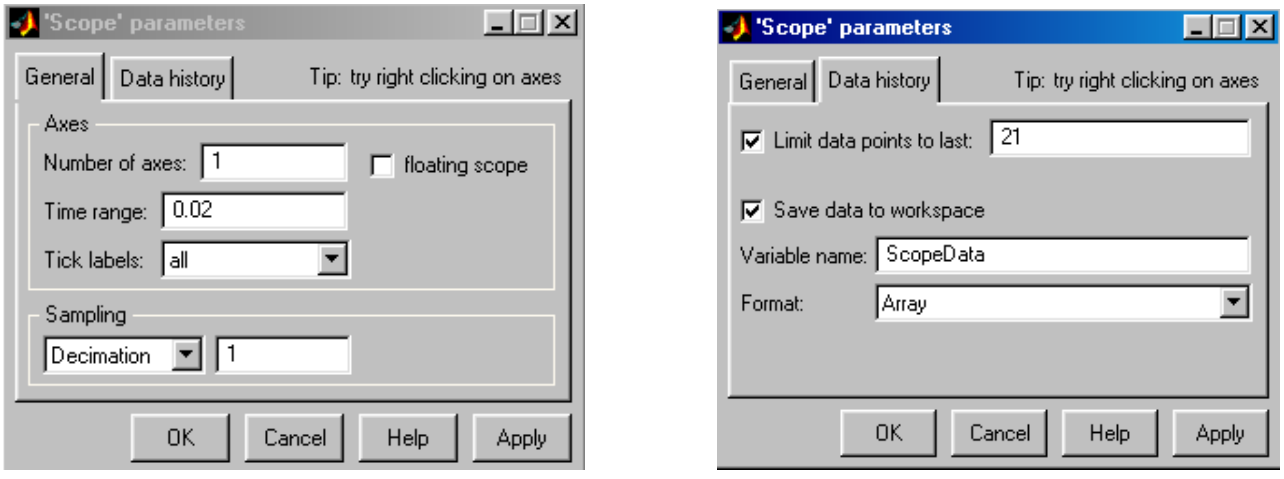


Рисунок 1.4 – Вікно параметрів у меню **Simulation/Simulation parameters**

Для налаштування параметрів осцилоскопа потрібно натиснути кнопку **Parameters** на вікні **Scope** (рисунок 1.5). Після цього можуть бути задані необхідні параметри.



a) b)

Рисунок 1.5 – Вікно параметрів блоку **Scope**

1.2 Порядок роботи із середовищем MatLab

1.2.1 Для запуску середовища MatLab необхідно викликати на виконання файл **matlab.exe**, який знаходиться в каталозі *MatLab*, підкаталозі *bin* (*:\Matlab\bin\matlab.exe).

1.2.2 Після завантаження середовища MatLab з'явиться вікно MatLab Command Window, в якому здійснюється виклик програм на виконання.

1.2.3 Щоб написати програму, необхідно відкрити нову сторінку в текстовому редакторі середовища MatLab. Для цього увійти в меню File, далі New та вибрати рядок M-file. З'явиться нове вікно редактора середовища MatLab, призначене для написання програм.

1.2.4 Для завантаження збереженої на диску програми необхідно увійти в меню File, вибрати рядок Open. Знайти потрібну програму на диску та натиснути на кнопку «Відкрити».

1.2.5 Для запуску програми на виконання необхідно зберегти програму, що знаходиться у вікні редактора, для чого натиснути клавіші Ctrl+S. Далі перейти у вікно MatLab Command Window, де написати назву файлу програми без розширення, який викликається на виконання, і натиснути клавішу Enter.

1.2.6 Після відкриття основного вікна програми **MATLAB** програму **Simulink** можна запустити одним із трьох способів:

– натиснути кнопку  (**Simulink**) на панелі інструментів командного вікна **MATLAB**;

– у командному рядку головного вікна **MATLAB** набрати команду **Simulink** та натиснути клавішу **Enter** на клавіатурі;

– виконати команду **Open…** у меню **File** та відкрити файл моделі (**mdl –** файл).

1.3 Підготовка до лабораторної роботи

1.3.1 Ознайомитись з теоретичними відомостями та порядком роботи із середовищем MatLab.

1.3.2 Написати програму у вигляді файла-сценарію обчислення амплітудно-модульованого сигналу з поданням результатів роботи програми у вигляді одного графіка з відображенням на ньому вхідних гармонічних та отриманого амплітудно-модульованого сигналів. Параметри для гармонічних сигналів вибираються з таблиці 1.1, згідно з заданим варіантом n, де n – номер студента у списку групи (або у послідовно об'єднаному списку груп у порядку зростання їх номерів).

Таблиця 1.1- Параметри сигналів

№ варіанту, n	Модулюючий сигнал		Модулюємий сигнал	
	Амплітуда, В	Частота, Гц	Амплітуда, В	Частота, Гц
1…5	n	n*100	n*4	n*800
6…10	n/2	n*70	n*2	n*600
11…15	n/3	n*50	n	n*400
16…20	n/4	n*40	n/2	n*200
21…25	n/6	n*30	n/2,5	n*250
26…30	n/8	n*10	n/3	n*50

1.3.3 Написати програму у вигляді файлу-функції, що запускається за допомогою файлу-сценарію, для обчислення того ж амплітудно-модульованого сигналу. Результат роботи програм повинен представлятися у вигляді відображення на двох окремих графіках вхідних і отриманого сигналів: на першому графіку відображаються вхідні гармонічні сигнали, на другому - отриманий амплітудно-модульований сигнал.

1.3.4 Змоделювати амплітудно-модульований сигнал у середовищі Simulink для заданого варіанта n. Вивести графіки вихідних гармонічних сигналів та результуючого амплітудно-модульованого сигналу на екран осцилоскопа. Зберегти побудовану модель файлу (mdl - файл).

Примітка. Модель у середовищі Simulink повинна працювати автономно, незалежно від параметрів у **Workspace**, тобто при «очищеному» **Workspace.**

1.4 Порядок виконання лабораторної роботи

1.4.1 Увімкніть ПЕОМ та запустіть програму MatLab.

1.4.2 Наберіть програму, написану за п. 1.3.2.

1.4.3 Запустіть програму виконання.

1.4.4 Отримані результати занесіть до звіту з лабораторної роботи у вигляді скріншотів.

1.4.5 У програмі за п. 1.3.2 змініть значення амплітуди в 1,5 рази, а значення частот у 0,7 разів.

1.4.6 Запустіть програму на виконання.

1.4.7 Отримані результати занесіть до звіту з лабораторної роботи у вигляді скріншотів.

1.4.8 Наберіть програму, написану за п. 1.3.3.

1.4.9 Запустіть програму на виконання.

1.4.10 Отримані результати занесіть до звіту з лабораторної роботи у вигляді скріншотів.

1.4.11 Відкрити файл моделі (mdl - файл), побудованої за п. 1.3.4 для амплітудно-модульованого сигналу в середовищі Simulink.

1.4.12 Запустіть модель, побудовану за п. 1.3.4 на виконання.

1.4.13 Отримані результати роботи моделі та вікно встановлених параметрів занесіть до звіту з лабораторної роботи у вигляді скріншотів.

1.4.14 У моделі за п. 1.3.4 змініть значення амплітуди в 1,5 рази, а значення частот у 0,7 разів.

1.4.15 Запустіть модель на виконання.

1.4.16 Отримані результати роботи моделі та вікно встановлених параметрів занесіть до звіту з лабораторної роботи у вигляді скріншотів.

1.4.17 Порівняйте результати побудови моделей сигналів у MatLab і Simulink, і зробіть висновки.

1.5 Зміст звіту з лабораторної роботи

Звіт з лабораторної роботи повинен містити:

1.5.1 Під час допуску до лабораторної роботи

1.5.1.1 Ціль лабораторної роботи.

1.5.1.2 Теоретичні відомості.

1.5.1.3 Написані за пунктами 1.3.2, 1.3.3 та програми з коментарями для середовища MatLab, а також підготовлену за п. 1.3.4 модель для Simulink із зазначенням параметрів налаштування.

1.5.2 При здачі лабораторної роботи до матеріалів за п. 1.5.1 мають бути додані результати виконання роботи.

1.5.2.1 Результати виконання програми за п. 1.4.4.

1.5.2.2 Результати виконання програми за п. 1.4.7.

1.5.2.3 Результати виконання програми за п. 1.4.10.

1.5.2.4 Результати виконання програми за п. 1.4.13.

1.5.2.5 Результати виконання програми за п. 1.4.16.

1.5.2.6 Висновки.

1.6 Контрольні питання

1.6.1 Дайте загальну характеристику MatLab.

1.6.2 У чому відмінність файлу-сценарію від файла-процедури?

1.6.3 Охарактеризуйте оператори для побудови графіків.

1.6.4 Як задається амплітудно-модульований сигнал за допомогою файлу-сценарію?

1.6.5 Як задається амплітудно-модульований сигнал за допомогою файла-процедури?

1.6.8 Як реалізується амплітудно-модульований сигнал за допомогою системи Simulink?

Лабораторна робота № 2

Дослідження дискретизації сигналів за допомогою системи Matlab та Simulink

Ціль роботи: дослідити операцію дискретизації сигналів за допомогою системи Matlab та Simulink

2.1 Теоретичні відомості

2.1.1 Дискретизація сигналів

Перетворення аналогового сигналу в дискретний прийнято називати «дискретизацією». [1,2,3,5,7]

Можливі варіанти сигналів у цифрових системах показані рисунку 2.1. Сигнал, зображений на рисунку 2.1,а, називатимемо *вхідним аналоговим.* На рисунку 2.1,б представлена дискретна версія вхідного сигналу, що зазвичай називають *даними, оцифрованими природним способом,* або *даними з амплітудно-імпульсною модуляцією* (pulse amplitude modulation – PAM). Дані на рисунку 2.1,б ще несумісні з цифровою системою, оскільки амплітуда кожної природної вибірки все ще може приймати безліч можливих значень, а цифрова система працює з кінцевим набором значень. Якщо значення імпульсів утворюють незліченну множину, вони називаються *дискретними відліками.*

Далі ці імпульси можна подати на пристрій квантування, який перетворює імпульси так, що їх значення утворюють зліченну множину – такі імпульси називаються *квантованими відліками.* Дані в такому форматі сумісні з цифровою

системою і представлені на рисунку 2.1,в і 2.1,г. Імпульси на рисунку 2.1,г відрізняються від імпульсів на рисунку 2.1,в тим, що повністю заповнюють проміжок між моментами оновлення значення сигналу. Такий спосіб дискретизації, іменований «вибірка-зберігання», найефективніший з точки зору завадостійкості. [1,2,5]

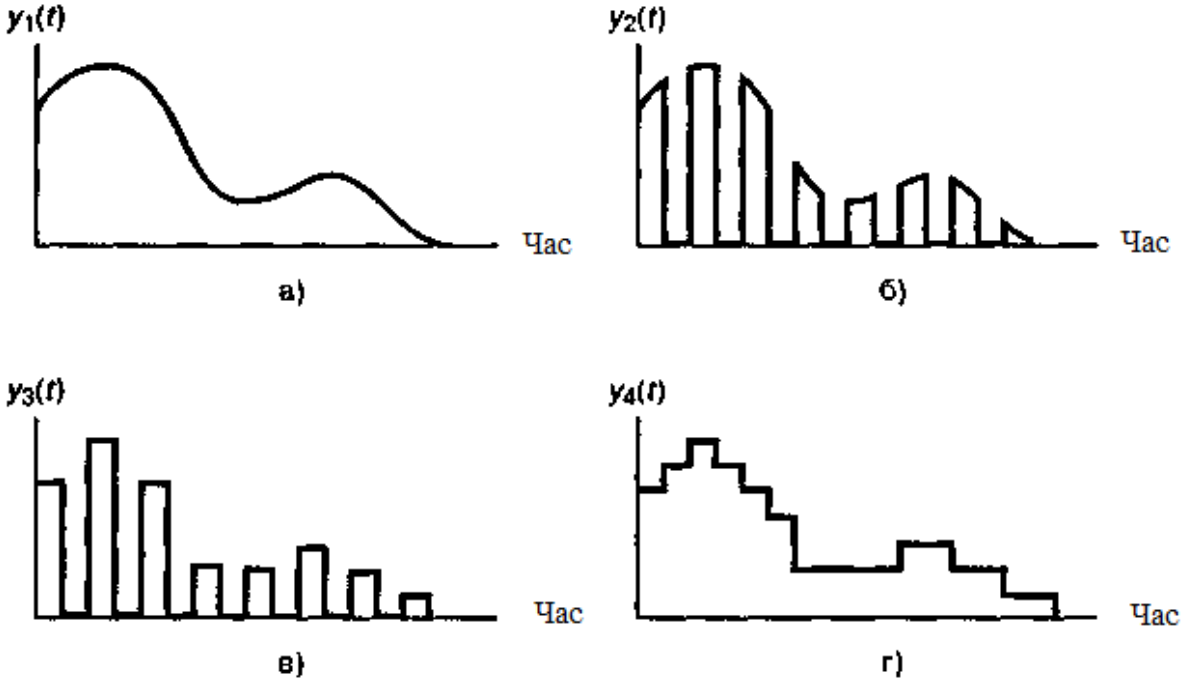


Рисунок 2.1 – Сигнали в системі координат «час-амплітуда»:
а) вхідний аналоговий сигнал; б) дані у природній дискретизації;
в) квантовані вибірки; г) вибірка-зберігання

2.1.2 Моделювання дискретних сигналів у Matlab

Генерувати сигнали в **Matlab** можна трьома способами [1,2]:

1) у діалоговому режимі, за допомогою послідовності команд у командному вікні;

2) в автоматичному режимі, шляхом створення та запуску на виконання m-скрипта;

3) в автоматичному режимі, шляхом створення та виклику m-функції.

Генерування сигналів у діалоговому режимі. Цей спосіб найбільш трудомісткий, оскільки вимагає кожну команду набирати з клавіатури в командному вікні. Щоб підвищити продуктивність праці, можна всю послідовність команд попередньо набрати у будь-якому текстовому редакторі (зазвичай це **Notebook** чи **Word**), а потім, скопіювавши текст у буферну пам'ять (**Clipboard**), вставити його у командне вікно. Недолік цього способу в тому, що необхідно одночасно тримати активними дві програми – **Matlab** і текстовий редактор. Перевага даного способу проявляється тоді, коли роботу в **Matlab** виконують, дотримуючись деякої інструкції, в якій теоретичні відомості чергуються з практичними завданнями у вигляді фрагментів текстів m-скриптів.

Наприклад, так виглядає в текстовому редакторі послідовність команд генерування **N** відліків гармонічного сигналу амплітудою **A**, частотою **f0**, початковою фазою **Fi0**, з частотою дискретизації **fs**:

```
% гармонічний сигнал
A=1; f0=100; Fi0=pi/2; fs=1000; N=20;  % параметри сигналу
t=(0:N-1)/fs;                          % моменти часу
s=A*sin(2*pi*f0*t+Fi0);        % обчислення відліків сигналу
plot(t,s)                              % виведення графіка
title('Гармонічний сигнал')    % заголовок
xlabel('Час, с'); ylabel('Рівень');   % написи вздовж осей
grid on                                % координатна сітка
```

Отриманий графік відображається у спеціальному вікні (рисунок 2.2) з написом Figure#1 (якщо це перший графік, що будується). Графік зручно зберігати шляхом експорту в економному форматі *.jpg.

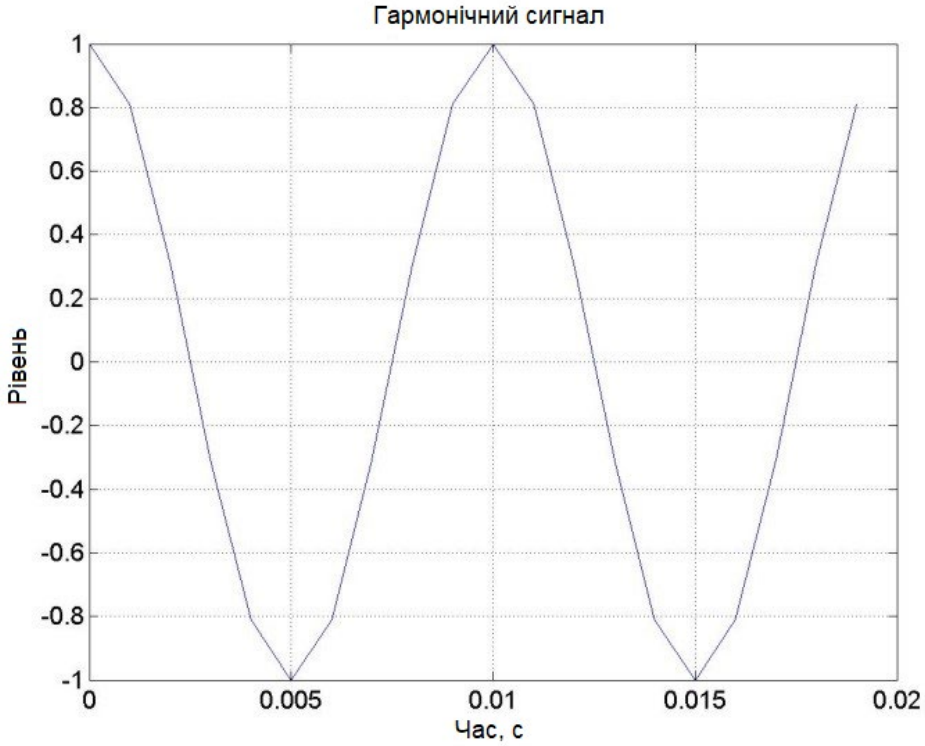


Рисунок 2.2 – Графік гармонічного сигналу

Примітка: при використанні символів кирилиці в тексті команд (коментарі, заголовки тощо) слід враховувати особливості «ставлення» кожної конкретної версії Matlab до кирилиці.

Генерування сигналів шляхом створення m-скрипта. Даний спосіб відрізняється тим, що всі команди набираються у спеціальному вікні редактора m-файлів (рисунок 2.3).

Даний спосіб хороший тим, що замість сторонніх програмних продуктів використовується власний інструментарій Matlab, спеціалізований для написання та налагодження m-скриптів.

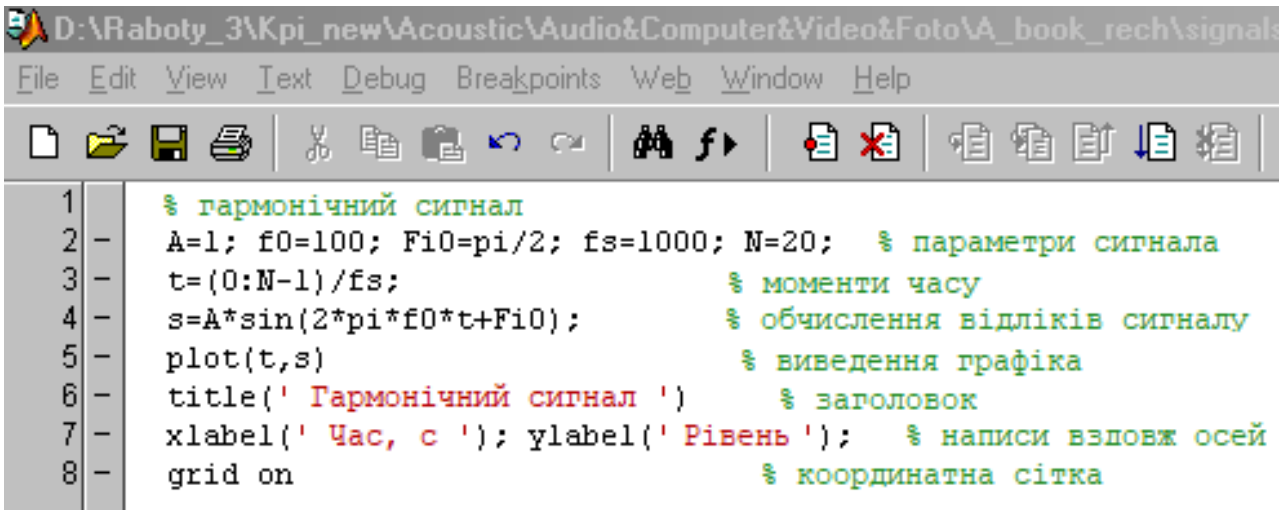


Рисунок 2.3 – Вікно редактора m-файлів

Генерування сигналів шляхом створення m-функції. Цей спосіб відрізняється тим, що вхідні дані записують як аргумент деякої функції $y = f(x)$, а вихідні – як значення цієї функції. Зручність у тому, що символьні позначення даних можуть відрізнятись від позначень, що використовуються у тілі функції. Більш того, числові значення вхідних даних можна просто задавати в найменуванні функції, що викликається. Останню обставину продемонструємо на покроковому прикладі.

Створимо підпрограму – m-скрипт **ton.m** виду:

```
% скрипт ton
s=A*sin(2*pi*f0*t+Fi0);        % обчислення відліків сигналу
```

Команду виконання цього скрипту потрібно супроводжувати командами підготовки вхідних даних та виведення вихідних даних:

```
A=1; f0=100; Fi0=pi/2; fs=1000; N=20;  % параметри сигнала
t=(0:N-1)/fs;                % моменти часу
ton;                         % обчислення відліків сигналу
```

```
plot(t,s)                    % виведення графіка
title('Гармонічний сигнал')  % заголовок
xlabel('Час, с'); ylabel('Рівень');  % написи вздовж осей
grid on                      % координатна сітка
```

Очевидно, позначення вхідних і вихідних даних програми, яка викликає, повинні співпадати з позначеннями відповідних даних підпрограми, що викликається.

У цьому випадку доцільніше представити дві наведені вище частини у вигляді одного m-скрипта.

Тепер зробимо інакше – напишемо і збережемо m-функцію під ім'ям **ton_sig.m**:

```
%-------функція ton_sig.m -----------
% [s,t]=ton_sig(B,f1,Fi1,Fs,N1)
%---------------------------------------
% генерування гармонічного сигналу
% y = B * sin(2*pi*f1*x + Fi1),
% B - амплітуда;
% N1 - кількість відліків сигналу;
% f1 - частота;
% Fs - частота дискретизації;
% Fi1 - початкова фаза сигналу
%---------------------------------------
function [y,x] = ton_sig( B, f1, Fi1, Fs, N1 )
%---------------------------------------
x = (0:N1-1)/Fs;    % моменти часу
y = B * sin( 2*pi*f1*x + Fi1 );
%-------- кінець функції ton_sig.m ----------
```

Тепер m-скрипт генерування того ж відрізка гармонічного сигналу виглядатиме так:

```
% гармонічний сигнал
[s,t]=ton_sig(1,100,pi/2,1000,20)  % обчислення відліків сигналу
plot(t,s)                  % виведення графіка
title('Гармонічний сигнал')    % заголовок
xlabel('Час, с'); ylabel('Рівень');   % написи вздовж осей
grid on                    % координатна сітка
```

Як бачимо, тепер числові значення вхідних даних задаються як аргументи m-функції **ton_sig.m**. Вихідні дані функції застосовуються для побудови графіка.

Очевидно, застосування m-функцій вигідно тоді, коли алгоритм формування значень функції досить складний: міститься багато команд та звернень до різноманітних бібліотечних функцій із непростим синтаксисом.

Очевидним є і недолік m-функцій – необхідно пам'ятати їх синтаксис. Втім, отримати потрібну інформацію можна, якщо у командному вікні задати команду **help**:

```
>> help ton_sig
```

В результаті на моніторі з'явиться коментар, з якого починається m-функція. Для наведеного раніше прикладу текст допомоги має такий вигляд:

```
%-------функція ton_sig.m -----------
% [s,t]=ton_sig(B,f1,Fi1,Fs,N1)
%---------------------------------------
% генерування гармонічного сигналу
% y = B * sin(2*pi*f1*x + Fi1),
% B - амплітуда;
% N1 - кількість відліків сигналу;
% f1 - частота;
% Fs - частота дискретизації;
% Fi1 - початкова фаза сигналу
%---------------------------------------
```

Тому дуже важливо при програмуванні m-функцій забезпечувати їх якісним та докладним коментарем.

2.1.3 Моделювання дискретних сигналів у Simulink

Генерування сигналів в Simulink має свої особливості [1,2]. Розглянемо їх.

Візьмемо з бібліотеки блоків **Simulink** два блоки: **Sine Wave** (з розділу **Sources**) та **Scope** (з розділу **Sinks**). З'єднавши їх, отримаємо наступну схему (рисунок 2.4).

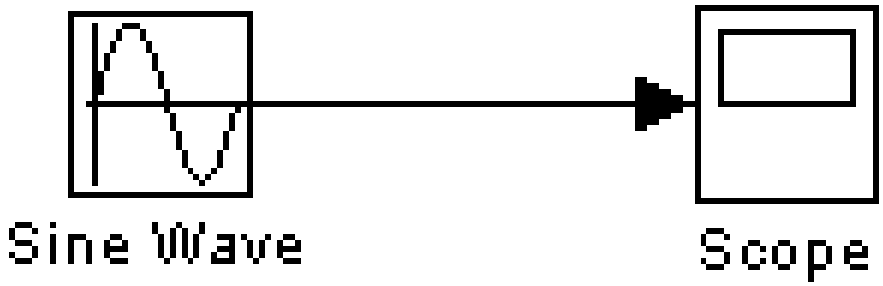


Рисунок 2.4 – Схема моделювання гармонічного сигналу

Далі подвійним клацанням по блоку осцилоскопа активізуємо вікно, що імітує екран осцилоскопа, і запустимо модель (кнопка **Start simulation**). В результаті отримаємо зображення відрізка синусоїди (рисунок 2.5).

Складається враження, що генерувати гармонічний сигнал в середовищі Simulink навіть простіше, ніж в середовищі Matlab. Але при цьому важливо вміти керувати параметрами гармонічного сигналу.

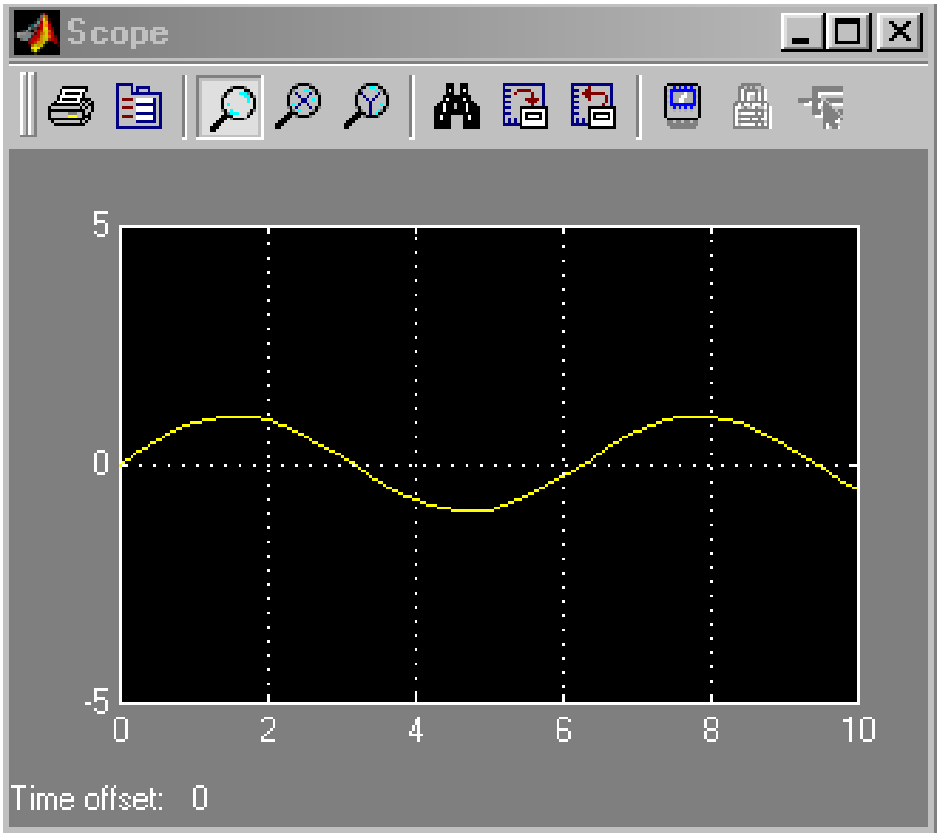


Рисунок 2.5 – Екран осцилоскопа

Двічі клацнемо по блоку **Sine Wave** – в результаті з'явиться вікно налаштувань параметрів (рисунок 2.6).

Натиснувши на кнопку **Help**, отримаємо інструкцію до даного блоку, сутність якої зводиться коротко до того, що в даному блоці виконується операція:

$$y = Amplitude \times \sin(frequency \times time + phase) + bias$$

З наведеної формули та написів на рисунку 2.6 стає зрозумілим сенс чотирьох змінних: амплітуди, кутової частоти, початкової фази та постійної складової. Залишається поки що зашифрованим сенс змінної "час".

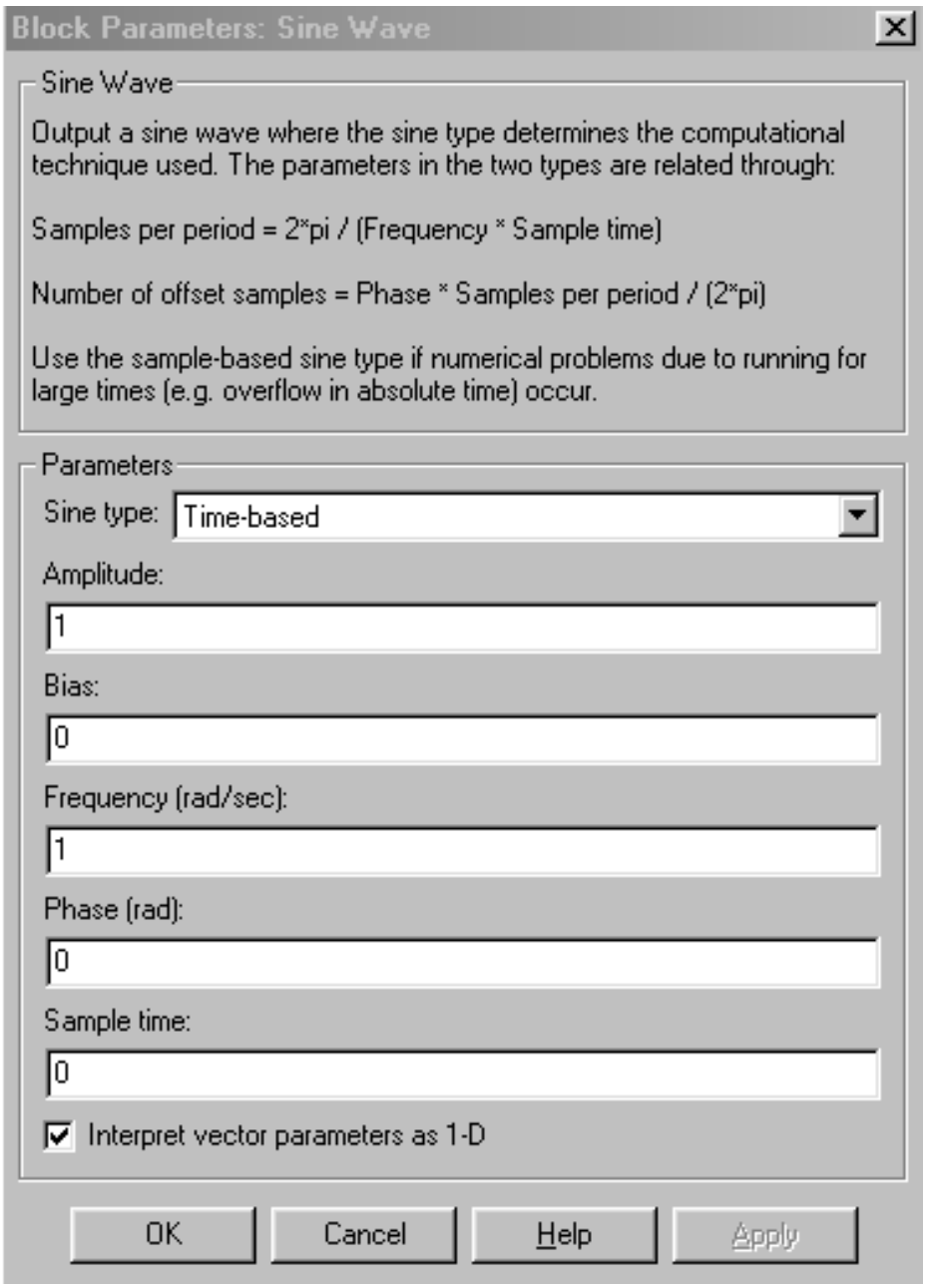


Рисунок 2.6 – Вікно налаштувань параметрів блока **Sine Wave**

Зупиняючись на цьому важливому питанні, відзначимо відмінність понять “час” та “модельний час”. Так, генерування відрізка сигналу тривалістю 1 с (модельний час) може тривати значно коротший проміжок часу, наприклад, 0.1 с (реальний час). Швидкість генерування залежить від обсягу обчислень, швидкодії комп'ютера, від обраного "вирішувача", тобто алгоритму моделювання, тощо. Цілком можливий зворотний ефект – для складного

алгоритму процедура моделювання відрізка сигналу тривалістю 0.1 с може розтягнутися на декілька секунд.

Сигнал може генеруватися двох типів: неперервний **time-based** та дискретний **sample-based**. Для моделювання роботи неперервних систем рекомендують використовувати неперервний тип **time-based**, а моделювання роботи дискретних систем – дискретний тип **sample-based**.

Якщо встановлено тип **time-based**, тоді параметр **Sample time** може приймати значення:

1) **0** (за замовчуванням) – блок працює в неперервному режимі;

2) **>0** – блок працює у дискретному режимі;

3) **-1** – блок наслідує той самий режим, що й приймаючий блок.

Як вказується в **Help**, робота в неперервному режимі може призводити до великих похибок генерації на великих проміжках модельного часу.

Робота в дискретному режимі змушує блок поводитися так, наче до виходу неперервного генератора був приєднаний блок **Zero-Order Hold**. Дійсно, зібравши дві схеми (рисунок 2.7) і задавши в обох випадках значення параметра **Sample time**, що дорівнює 0.5 (вікно налаштування блоку **Zero-Order Hold** показано на рисунку 2.8), отримуємо ідентичні результати (рисунок 2.9).

Таким чином, блок **Zero-Order Hold** можна трактувати як “дискретизатор”, тобто частину аналого-цифрового перетворювача (АЦП), відповідальну за дискретизацію сигналу. Іноді блок **Zero-Order Hold** називають АЦП. Однак це некоректно, оскільки дискретизований сигнал в реальному АЦП піддається ще й квантуванню за рівнем. У блоці **Zero-Order Hold** квантування не здійснюється.

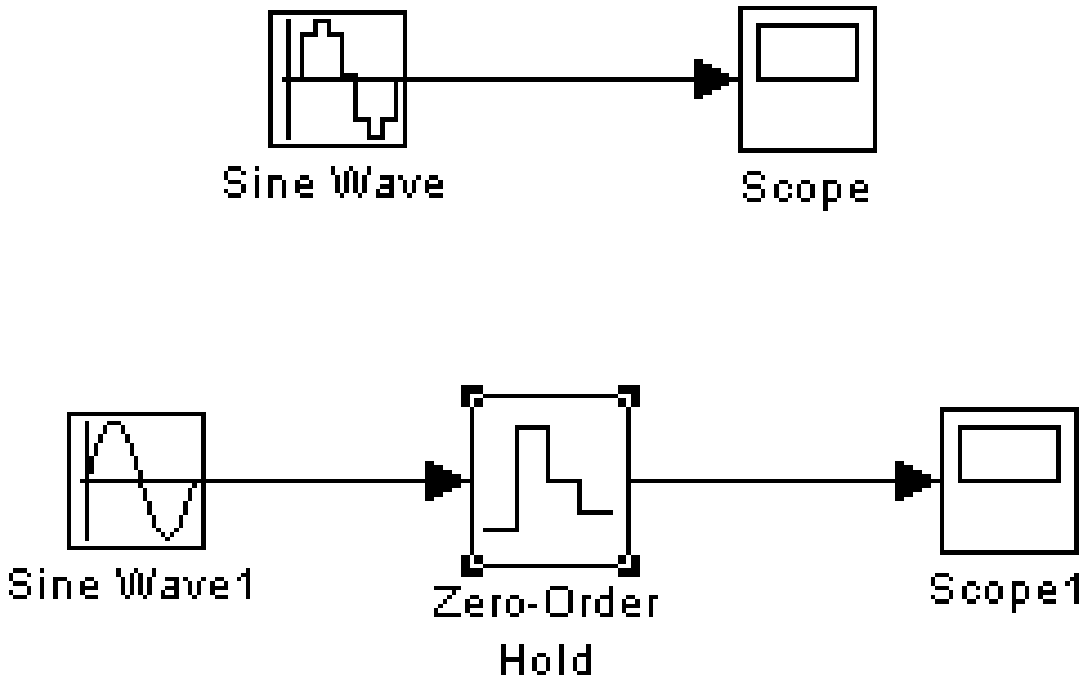


Рисунок 2.7 – Схеми моделювання дискретного гармонічного сигналу

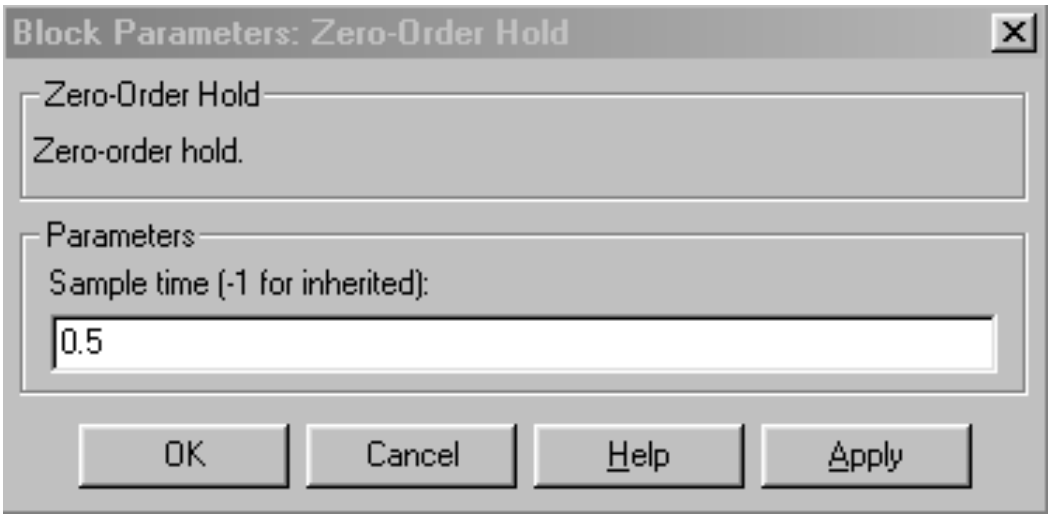


Рисунок 2.8 – Вікно налаштування блока **Zero-Order Hold**

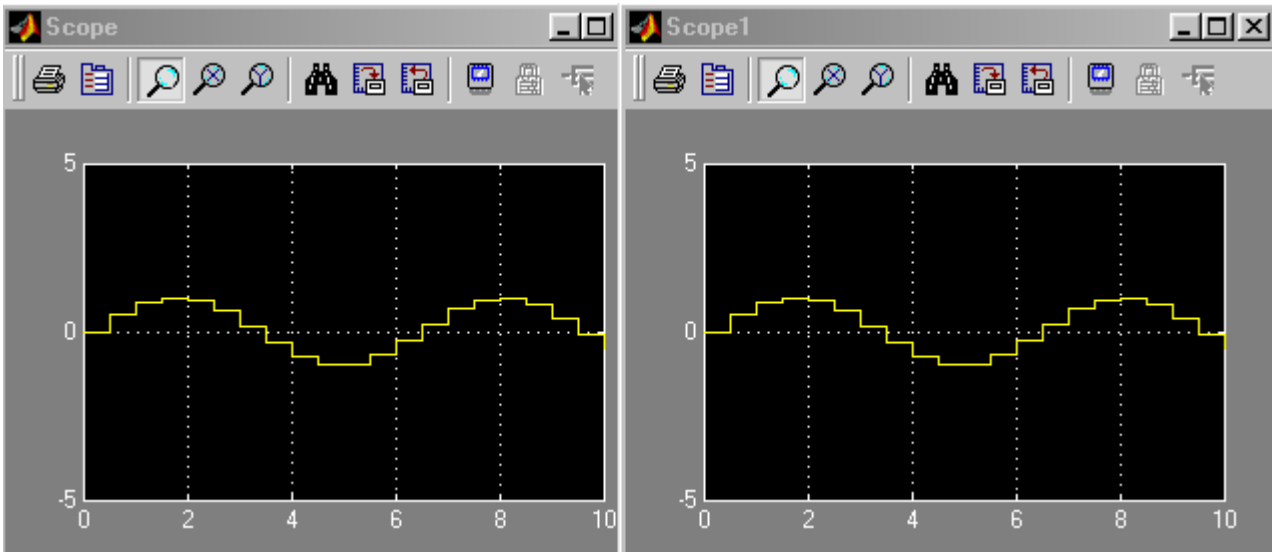


Рисунок 2.9 – Результати моделювання дискретного гармонічного сигналу

Окрім блоку **Scope**, графік можна побудувати і за допомогою блоку **X-Y-Graph**, на верхній вхід X якого потрібно подати послідовність моментів часу за допомогою блоку **Clock** (годинник), а на нижній вхід Y – значення сигналу, що генерується (рисунок 2.10).

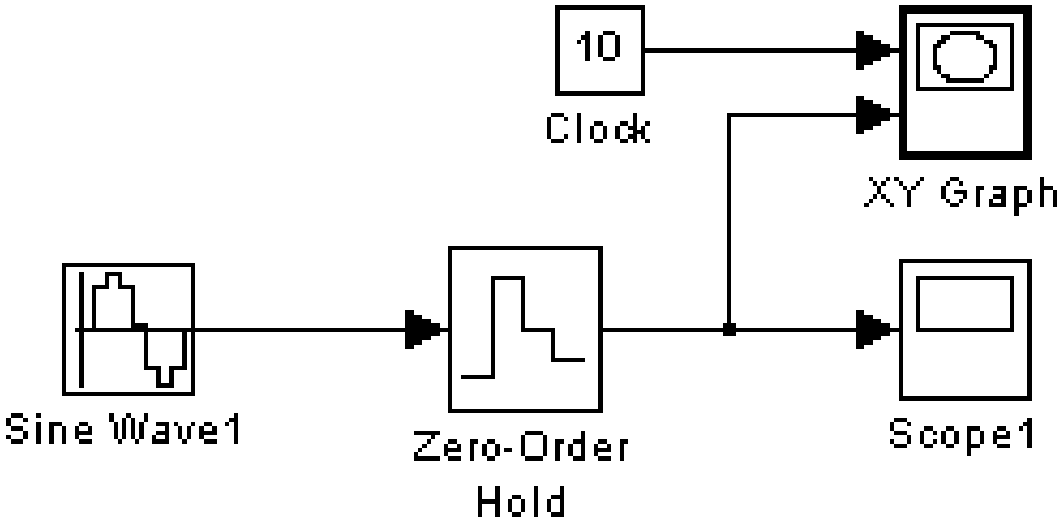


Рисунок 2.10 – Схема моделювання з використанням блоку **X-Y-Graph**

В результаті попередньо налаштований (у відповідному вікні налаштування задаються граничні значення аргументу і функції, а також вказується значення параметра **Sample time**) графобудівник видасть показаний на рисунку 2.11 графік, якщо для блоку **X-Y-Graph** задано **Sample time=-1** (тобто період дискретизації наслідується).

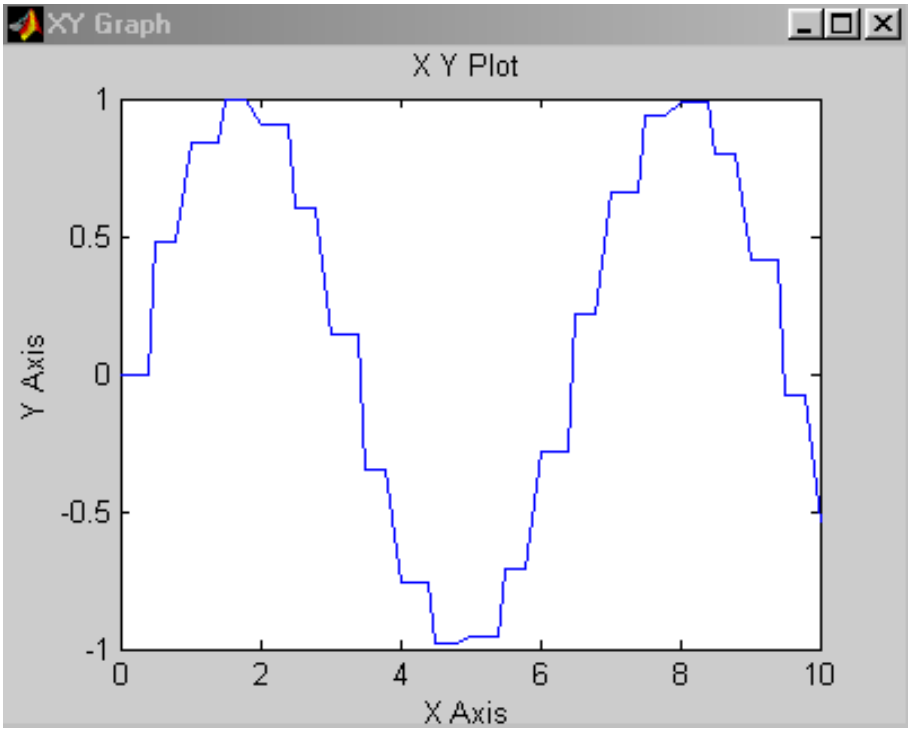


Рисунок 2.11 – Графік, побудований за допомогою блока **X-Y-Graph (Sample time=-1)**

Графік буде дещо іншим (рисунок 2.12), якщо для блока **X-Y-Graph** задано **Sample time=0.5**.

Ще один спосіб побудови графіків. Масиви відліків моментів часу та відповідних значень сигналу можна за допомогою блоку **To Workspace** експортувати з середовища **Simulink** в середовище **Matlab** (рисунок 2.13).

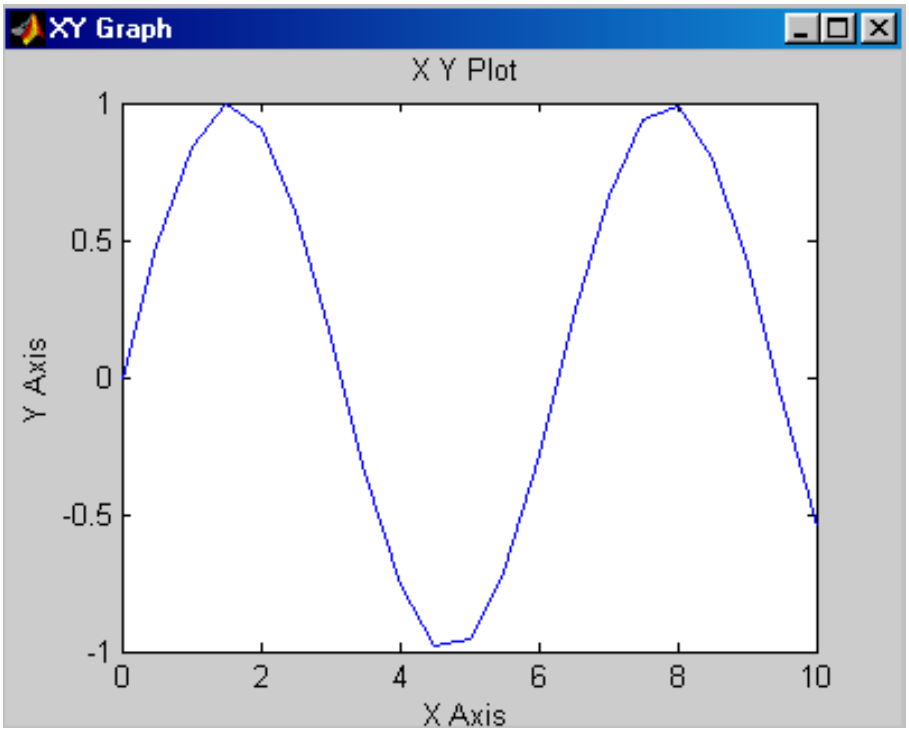


Рисунок 2.12 – Графік, побудований за допомогою блока **X-Y-Graph (Sample time=0.5)**

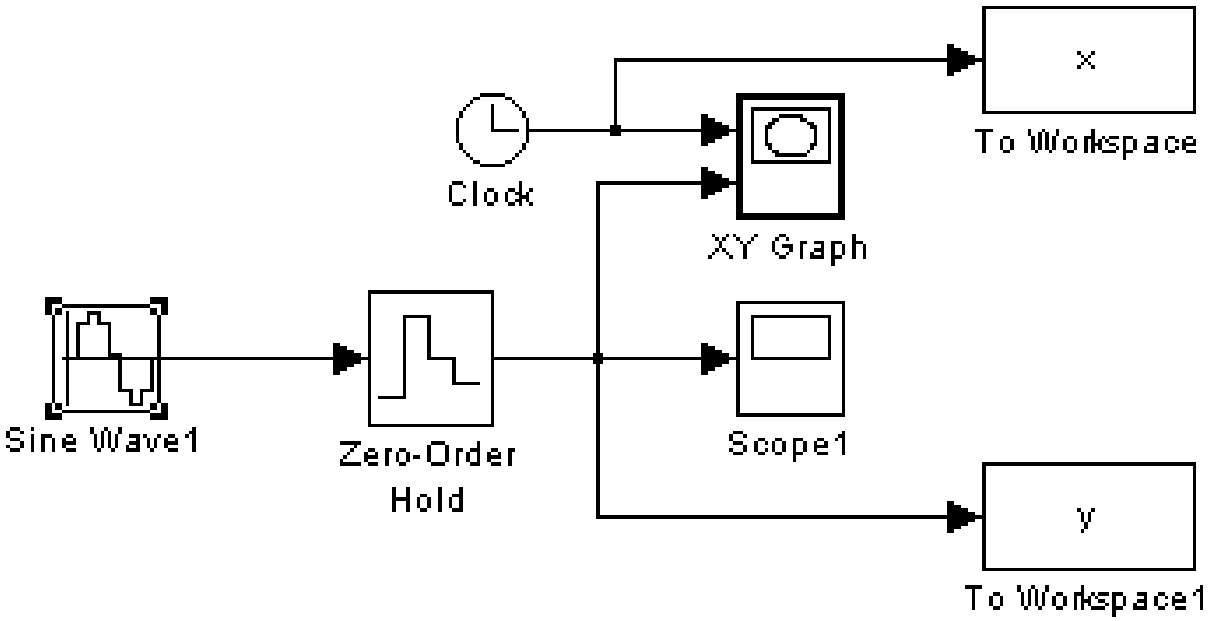


Рисунок 2.13 – Схема моделювання з використанням експорту даних із середовища **Simulink** в середовище **Matlab**

При цьому, як показує практика, краще задати формат **array** для даних, що експортуються (рисунок 2.14).

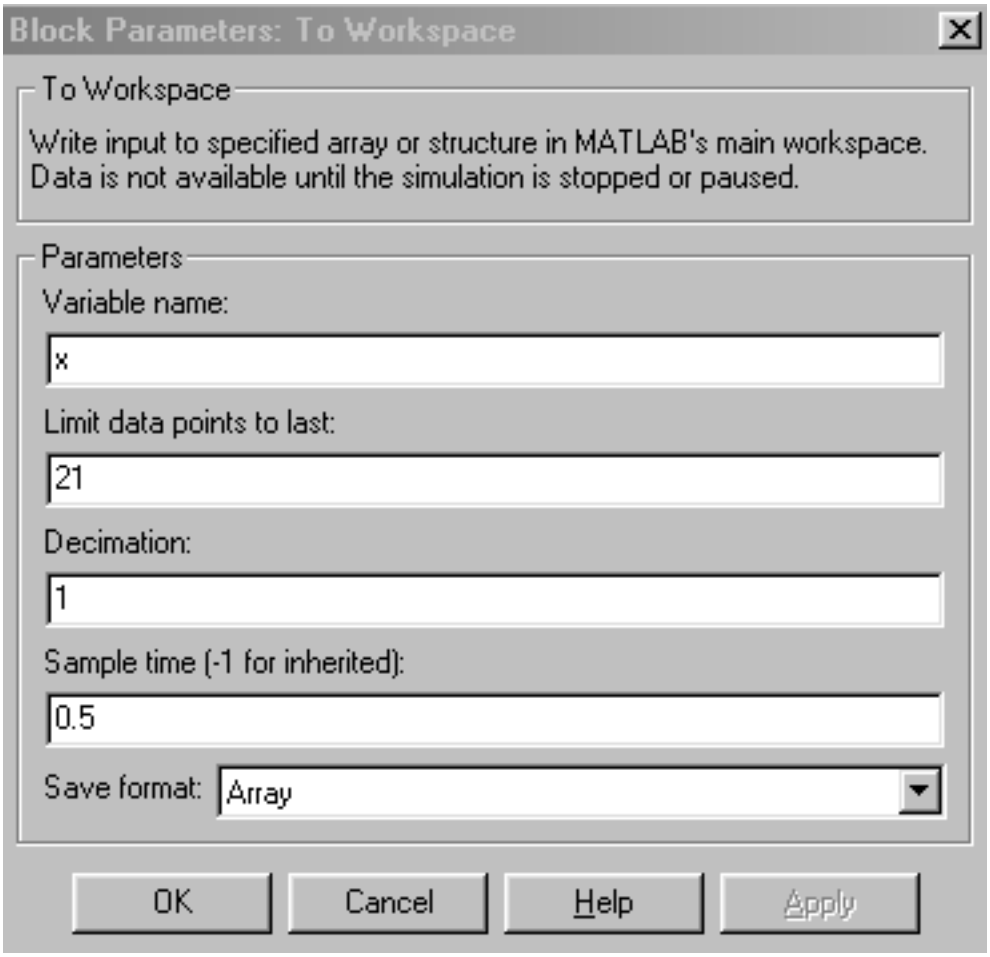


Рисунок 2.14 – Вікно налаштування блоку **To Workspace**

Подальша побудова графіка в середовищі **Matlab** за допомогою команди **plot(x,y)** не складає жодних труднощів (рисунок 2.15).

Таким чином, сигнал типу **time-based** при роботі блоку генерації в режимі неперервного часу має вигляд плавної функції часу, а в режимі дискретного часу – вид ступінчастого сигналу, такого, як наче до виходу генератора плавного сигналу був приєднаний блок **Zero-Order Hold**, що є дискретизатором типу

"відлік-зберігання". Іншими словами, задаючи режим дискретного часу, ми уникаємо необхідності використання блоку **Zero-Order Hold**.

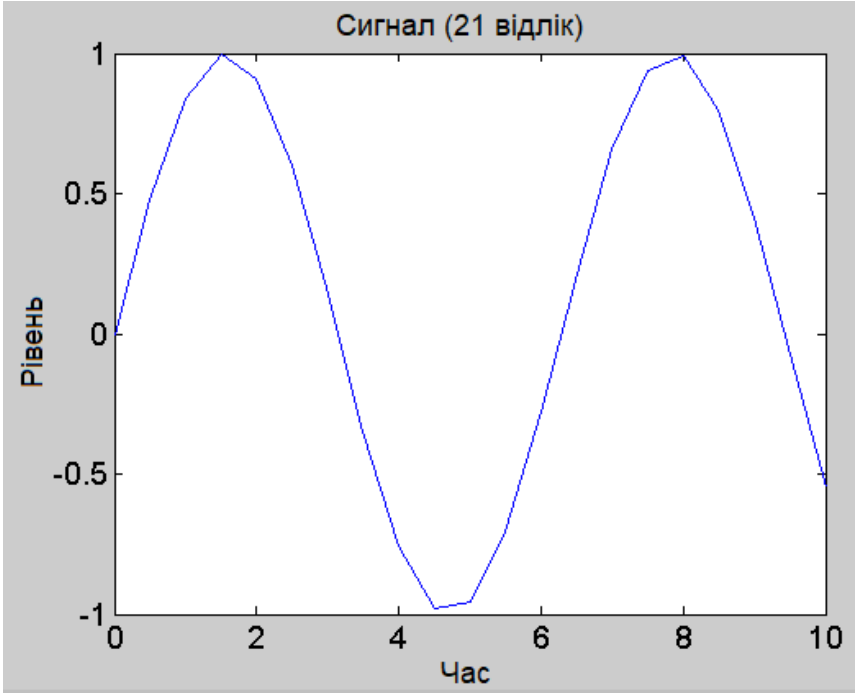


Рисунок 2.15 – Графік гармонічного сигналу побудований в середовищі **Matlab**

А тепер згенеруємо в **Simulink** відрізок дискретного гармонічного сигналу з тими ж параметрами, що були задані в **Matlab**: амплітуда 1, частота 100 Гц, частота дискретизації 1000 Гц, початкова фаза $\pi / 2$, кількість відліків 20.

Збираємо знову схему з генератора та осцилоскопа (рисунок 2.4). У вікні-масці налаштувань генератора вказуємо необхідні числові значення параметрів, задаємо тип **time-based** та присвоюємо значення Sample time = 0.001 (рисунок 2.16).

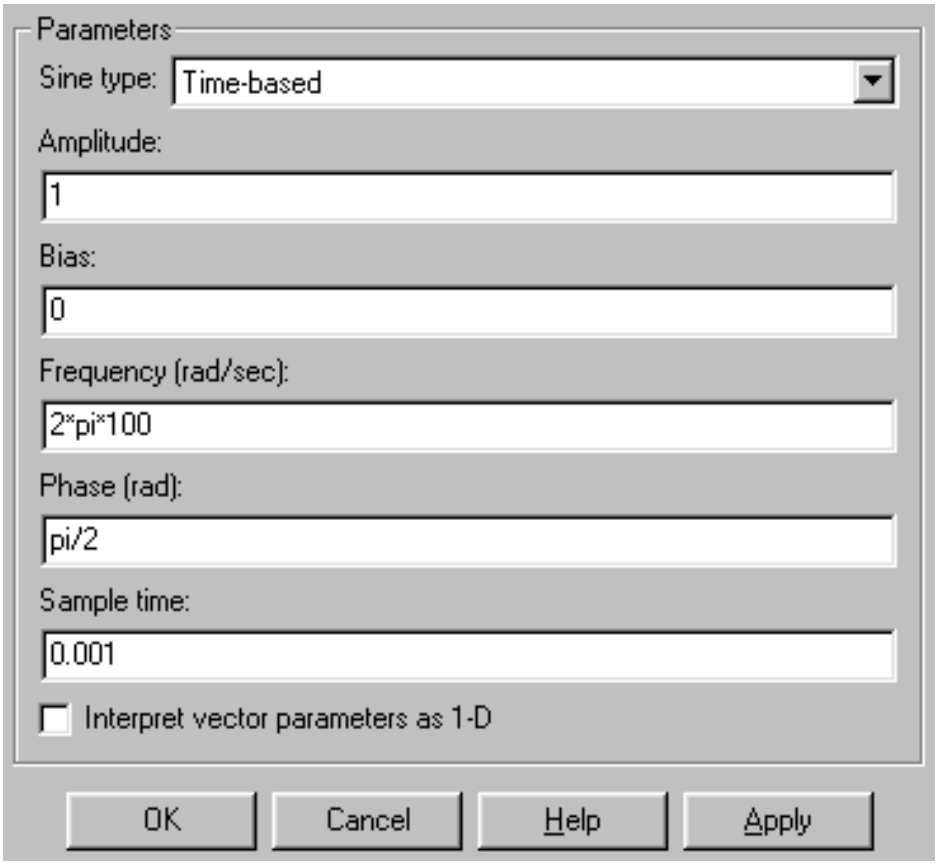


Рисунок 2.16 – Вікно налаштувань параметрів блока **Sine Wave** (Sample time=0.001)

Після запуску моделі отримуємо на екрані осцилоскопа зовсім не ту картину, яку очікували (рисунок 2.17).

Причина проста – потрібно ще налаштувати параметри моделювання: задати початок та кінець модельного часу (у нашому випадку це 0 та 0.02 с, відповідно), а також вибрати алгоритм моделювання (тип «вирішувача»).

На рисунку 2.18 показано вікно налаштувань параметрів моделювання, що активізується при виборі позиції меню **Simulation/Simulation parameters**.

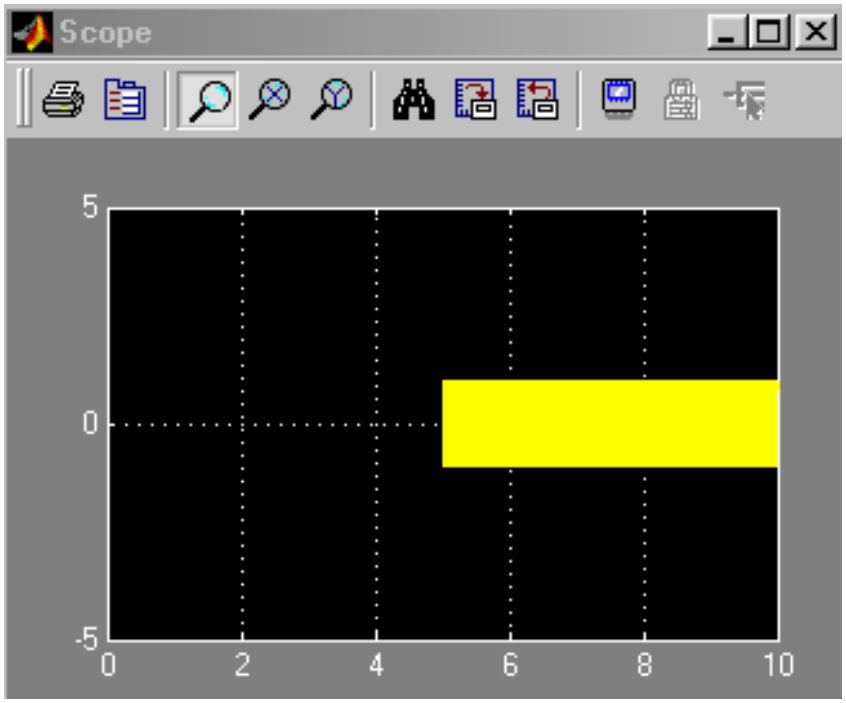


Рисунок 2.17 – Екран осцилоскопа (Sample time=0.001)

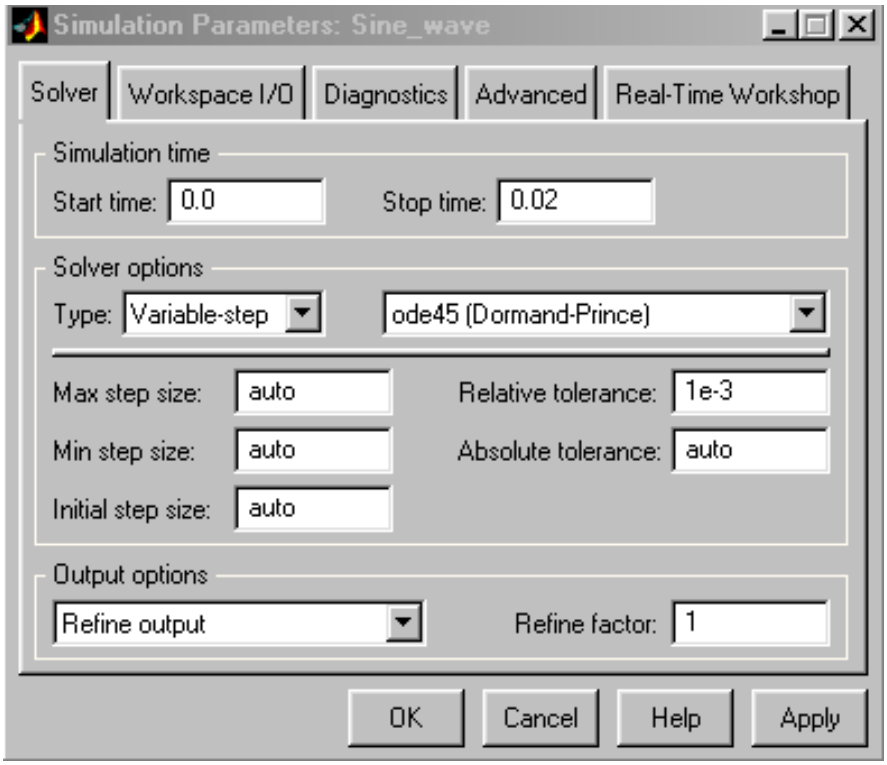
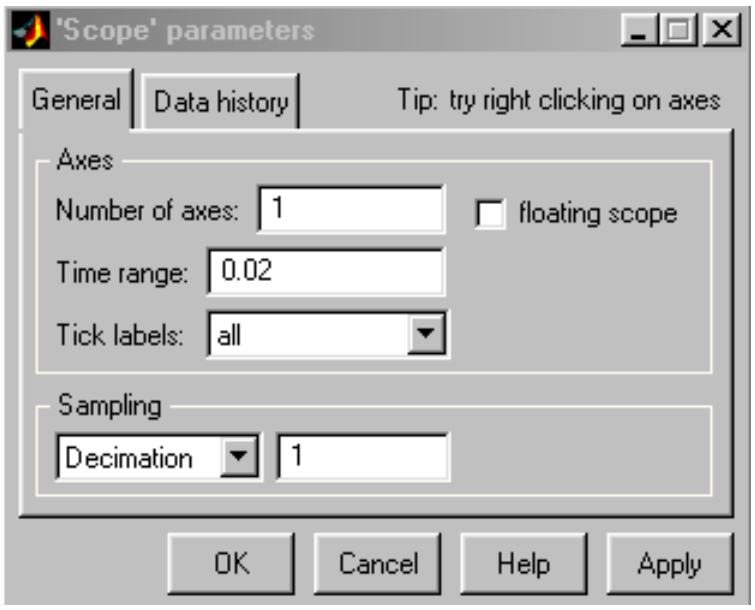
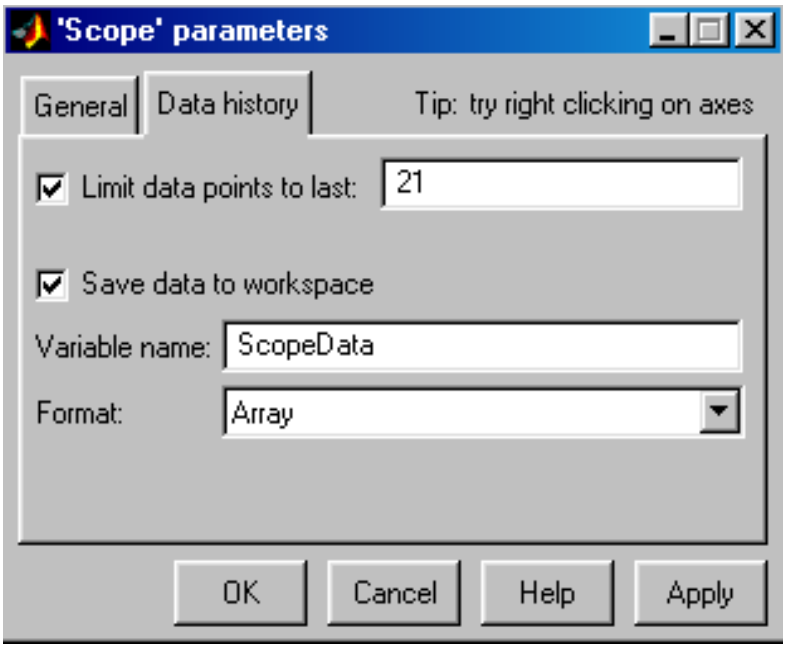


Рисунок 2.18 – Вікно налаштувань параметрів моделювання в меню **Simulation/Simulation parameters**.

Крім того, налаштуємо параметри осцилоскопа, клацнувши по кнопці **Parameters** у вікні **Scope** (рисунок 2.19, а, б).



а)



б)

Рисунок 2.19 – Вікна налаштувань параметрів осцилоскопа

Після встановлення параметрів (рисунок 2.19,а,б) та запуску моделі на екрані осцилоскопа з'явиться зображення (рисунок 2.20).

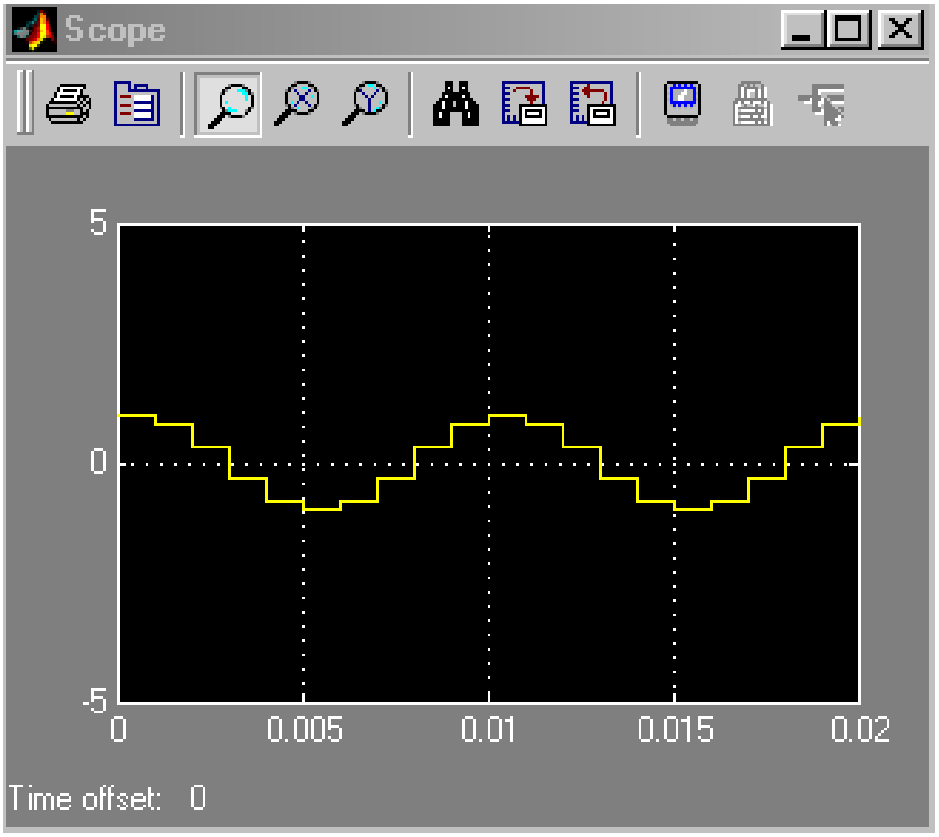


Рисунок 2.20 – Екран осцилоскопа після налаштування параметрів

Оскільки параметри осцилоскопа були задані так, щоб у робочий простір виводився двовимірний масив **ScopeData** значень аргументу і функції, за допомогою команд

```
>> y1=ScopeData(:,1);
>> y2=ScopeData(:,2);
>> plot(y1,y2)
```

можна побудувати графік згенерованої функції засобами **Matlab** (рисунок 2.21).

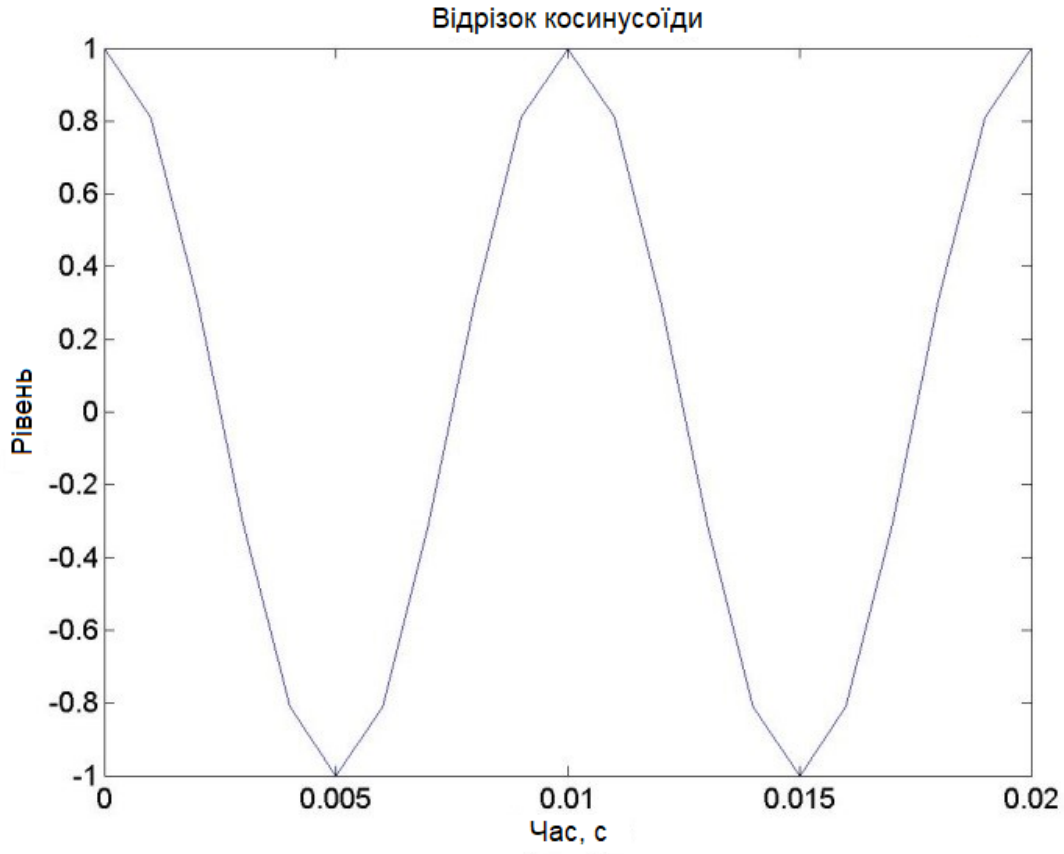


Рисунок 2.21 – Побудова графіка функції засобами **Matlab**

Порівнюючи рисунок 2.21 і рисунок 2.2, помічаємо лише одну відмінність – при моделюванні в **Simulink** згенерована 21 точка, тоді як в **Matlab** генерувалося 20 точок. Причина відмінності проста: на інтервалі модельного часу T при частоті дискретизації F_s знаходиться TF_s+1 моментів часу, для яких буде згенерований сигнал. Очевидно, цю обставину легко врахувати, досягнувши повного збігу результатів моделювання в середовищах **Matlab** і **Simulink**.

2.1.4 Висновки

Дискретизація аналогових сигналів – перший крок на шляху вирішення задачі поєднання аналогових пристроїв і систем із дискретними даними.

Моделювання дискретних сигналів можна проводити або в середовищі **Matlab**, або в середовищі **Simulink**. Можливе сумісне використання цих середовищ, що збільшує гнучкість інструментарію.

Генерувати сигнали в **Matlab** можна трьома способами:

1) у діалоговому режимі (послідовність команд у командному вікні);

2) в автоматичному режимі, шляхом створення та запуску m-скрипта;

3) в автоматичному режимі, шляхом створення та виклику m-функції.

Моделювання сигналів у **Simulink** зручне завдяки своїй наочності, проте вимагає додаткових навичок задання параметрів блоків, з яких конструюється модель.

Важливою особливістю моделювання в **Simulink** є очевидна відмінність понять «реальний час» і «модельний час». Реальний час – це час, необхідний для проведення обчислень (моделювання). Модельний час – тривалість процесу, що моделюється.

Модельний час може бути неперервним (**time-based**) та дискретним (**sample-based**). Неперервний час рекомендується використовувати при моделюванні неперервних (аналогових) систем, дискретний – дискретних (цифрових). Контроль результатів моделювання в **Simulink** можна здійснювати як шляхом побудови графіків, так і шляхом роздрукування значень числових масивів.

Побудову графіків зручніше проводити у середовищі **Simulink**. Аналіз числових даних зручніше проводити, експортуючи їх у робочий простір **Matlab**.

2.2 Підготовка до лабораторної роботи

2.2.1 Ознайомитись з теоретичними відомостями.

2.2.2 Написати програму у вигляді файлу-сценарію (m-скрипту, m-файлу) для моделювання дискретизованого сигналу, що заданий функцією $S(t)$, побудови його графіка та графіка гармонічних компонентів, які є його складовими:

$$S(t)=A_1 \sin(2\pi f_1 t+\varphi_1)+A_2 \sin(2\pi f_2 t+\varphi_2);$$

$$0\le t\le T.$$

Параметри для гармонічних сигналів обираються з таблиці 2.1, згідно з заданим варіантом n, де n – номер студента в списку групи (або послідовно об'єднаному списку груп у порядку зростання їх номерів).

Таблиця 2.1 – Параметри гармонічних сигналів

№ варіанта, n	A_1, В	A_2, B	f_1, Гц	f_2, Гц	φ_1, рад	φ_2, рад
1…5	n	n*1,5	n*100	n*300	0	π/2
6…10	n/2	n	n*70	n*200	π/4	2π/4
11…15	n/3	n/2	n*50	n*100	π/2	π/4
16…20	n/4	n/5	n*40	n*200	π	2π/6
21…25	n/6	n/4	n*30	n*250	2π/4	π
26…30	n/7	n/5	n*10	n*50	2π/6	0

Тривалість T обрати рівною одному або двом періодам сигналу нижчої частоти.

Частоту дискретизації $\boldsymbol{F_s}$ задати двома способами:

а) відповідно до теореми Котельникова - Найквіста;

б) збільшивши обране за п. а) значення в 5-6 разів.

Для побудови графіків функцій використати дві команди: **plot** і **stem**.

2.2.3 Написати програму у вигляді файлу-функції для моделювання того ж дискретизованого сигналу *S(t)* та побудови його графіка та графіка гармонічних компонентів, які є його складовими. Запуск файлу-функції повинен здійснюватися за допомогою окремого файлу-сценарію (m-скрипту, m-файлу).

Для побудови графіків функцій використати дві команди: **plot** і **stem**.

2.2.4 Змоделювати сигнал *S(t)* у середовищі Simulink з використанням режиму генерації типу **sample-based** для двох заданих за п. 2.2.2 значень частоти дискретизації та з виведенням графіків сигналу *S(t)* та гармонічних компонентів, що є його складовими на екран осцилоскопа.

Примітка. Модель у середовищі Simulink повинна працювати автономно, незалежно від параметрів у **Workspace**, тобто при «очищеному» **Workspace.**

2.2.5 Змоделювати сигнал *S(t)* у середовищі Simulink з використанням неперервного режиму генерації типу **time-based** і блоку **Zero-Order Hold** для двох заданих за п. 2.2.2 значень частоти дискретизації та з виведенням графіків сигналу *S(t)* та гармонічних компонентів, що є його складовими на екран осцилоскопа.

Примітка. Модель у середовищі Simulink повинна працювати автономно, незалежно від параметрів у **Workspace**, тобто при «очищеному» **Workspace.**

2.3 Порядок виконання лабораторної роботи

2.3.1 Увімкніть ПЕОМ та запустіть програму MatLab.

2.3.2 Наберіть програму, написану за п. 2.2.2 та запустіть її на виконання для двох заданих за п. 2.2.2 значень частоти дискретизації.

2.3.3 Отримані результати занесіть до звіту з лабораторної роботи у вигляді скріншотів.

2.3.4 В написаній програмі змініть значення амплітуд у 1,5 рази, а значення частот у 0,7 разів та запустіть її на виконання для двох заданих за п. 2.2.2 значень частоти дискретизації.

2.3.5 Отримані результати занесіть до звіту з лабораторної роботи у вигляді скріншотів.

2.3.6 Наберіть програми, написані за п. 2.2.3 та запустіть програму (m-файл) на виконання для двох заданих за п. 2.2.2 значень частоти дискретизації.

2.3.7 Отримані результати занесіть до звіту з лабораторної роботи у вигляді скріншотів.

2.3.8 Запустіть модель Simulink, підготовлену за п. 2.2.4, на виконання для двох заданих за п. 2.2.2 значень частоти дискретизації.

2.3.9 Отримані результати роботи моделі та вікно встановлених параметрів занесіть до звіту з лабораторної роботи у вигляді скріншотів.

2.3.10. Запустіть модель Simulink, підготовлену за п. 2.2.5 на виконання для двох заданих за п. 2.2.2 значень частоти дискретизації.

2.3.11 Отримані результати роботи моделі та вікно встановлених параметрів занесіть до звіту з лабораторної роботи у вигляді скріншотів.

2.4 Зміст звіту з лабораторної роботи

Звіт з лабораторної роботи повинен містити:

2.4.1 При допуску до лабораторної роботи:

2.4.1.1 Ціль лабораторної роботи.

2.4.1.2 Короткі теоретичні відомості.

2.4.1.3 Написані за пунктами 2.2.2 та 2.2.3 програми з коментарями для середовища MatLab.

2.4.1.3 Блок-схеми моделей у середовищі Simulink за пунктами 2.2.4 і 2.2.5 із зазначенням значень параметрів, що задаються для них.

2.4.2 При здачі лабораторної роботи до матеріалів за п. 2.4.1 повинні бути додані результати виконання роботи.

2.4.2.1 Результати виконання програми за п. 2.3.3.

2.4.2.2 Результати виконання програми за п. 2.3.5.

2.4.2.3 Результати виконання програми за п. 2.3.7.

2.4.2.4 Результати виконання програми за п. 2.3.9.

2.4.2.5 Результати виконання програми за п. 2.3.11.

2.4.2.6 Висновки.

2.5 Контрольні питання

2.5.1 Що таке дискретизація сигналу?

2.5.2 У чому відмінність файлу-сценарію від файлу-процедури (файлу-функції)?

2.5.3 Як здійснюється генерація неперервного та дискретного сигналу в середовищі Simulink?

2.5.4 Що таке «модельний час»?

2.5.5 Як обирається частота дискретизації?

Лабораторна робота № 3

Дослідження методів лінійного квантування за допомогою системи Matlab та Simulink

Ціль роботи: дослідити властивості лінійного квантування у середовищах Matlab та Simulink.

3.1 Теоретичні відомості

3.1.1 Квантування сигналів

Квантування сигналів являє собою перетворення величини, що безперервно змінюється, у величину, що змінюється ступінчасто із заданими розмірами ступенів. [1,2,6,7]

Ступенем квантування називається різниця між двома сусідніми заданими значеннями квантованої величини $q_к$.

Квантування широко застосовується в процесі управління за необхідності впливу на технологічний процес сигналом з параметром точно заданого розміру.

Квантований сигнал у функції часу може бути виражений за допомогою одиничної ступінчатої функції $qk1(t-t_{Ni})$, що дорівнює одиниці, якщо аргумент додатній, або дорівнює нулю — якщо від'ємний:

$$X_{K.B.}(t) = N_i(t_{Ni}) * qk1(t-t_{Ni}),$$

де $N_i(t_{Ni})$ – коди сигналу, i = 1, 2, …

Квантування може виконуватися з рівномірними (рисунок 3.1, *а, б*) та нерівномірними (рисунок 3.1, *в, г*) ступенями квантування. Ступінь квантування виражається в одиницях даної величини та у відносних одиницях.

Рівномірне квантування застосовується частіше, зокрема під час вимірювань. Як правило, при цьому кількість рівнів або ступенів квантування достатньо велика.

Нерівномірне квантування застосовується зазвичай при невеликому числі ступенів квантування, наприклад в статистичних аналізаторах, при передачі голосових сигналів та ін.

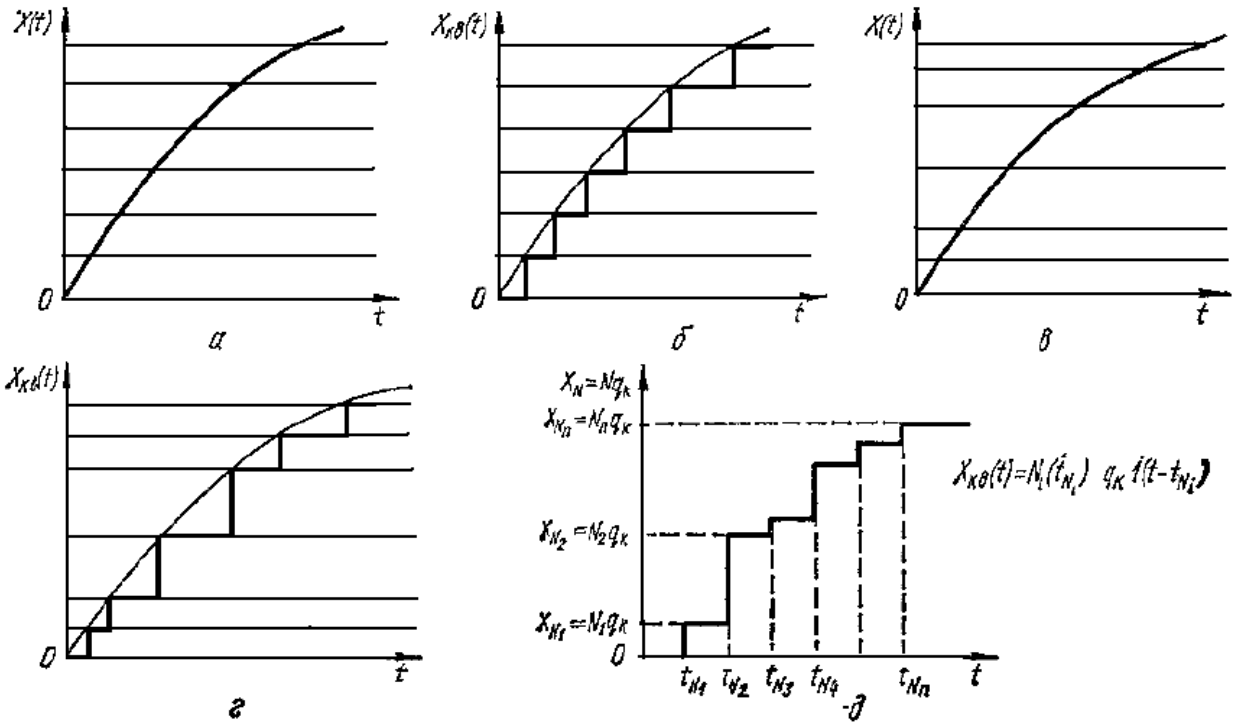


Рисунок 3.1 - Квантування аналогових сигналів:
а, б — з рівномірними ступенями; *в, г* — з нерівномірними ступенями; д — квантування відомої величини на виході міри, що регулюється

Квантування є однією з найвідповідальніших операцій процесу вимірювання, перетворення сигналу. Квантування зразкової величини здійснюється за допомогою багатоканальної або регульованої міри. При цьому на виході міри отримуються величини заданого розміру зі значеннями, що виражаються в одиницях даної фізичної величини (рисунок 3.1, д). Квантування в одиницях відносних величин може бути виконаним в багатоканальному або регульованому масштабному перетворювачі (відносне квантування).

Похибка від квантування Δ_k є методичною похибкою методу відображення неперервної за розміром величини обмеженим по числу розрядів числом, тобто методичною похибкою самого вимірювання. Похибка від квантування дорівнює різниці між результатом вимірювання без попереднього перетворення x_n та істинним значенням величини X, якщо похибки міри та компаратора дорівнюють нулю:

$$\Delta_k = x_n - X.$$

Похибка від квантування Δ_k по характеру зміни може бути систематичною постійною по величині (при $X = const$) або випадковою (при $X = var$ по випадковому закону).

Рівномірне (лінійне) квантування. Розглянемо рисунок 3.2, на якому зображені L рівнів квантування аналогового сигналу з повним діапазоном напруг, що дорівнює (у вольтах):

$$V_{PP} = V_P - (-V_P) = 2V_P .$$

Квантовані імпульси можуть мати додатні та від'ємні значення. Крок між рівнями квантування, що називаються *ступенем квантування*, дорівнює q вольт. Якщо рівні квантування рівномірно розподілені по всьому діапазону, то прилад для квантування називається *рівномірним*, або *лінійним*. При квантуванні кожне дискретне значення аналогового сигналу апроксимується квантованим

імпульсом: апроксимація дає похибку, що не перевищує $q/2$ в додатному напрямку або $-q/2$ у від'ємному (при введені поправки на половину ступеня квантування). Таким чином, похибка сигналу у результаті квантування обмежена половиною квантованого інтервалу, $\pm q/2$ вольт.

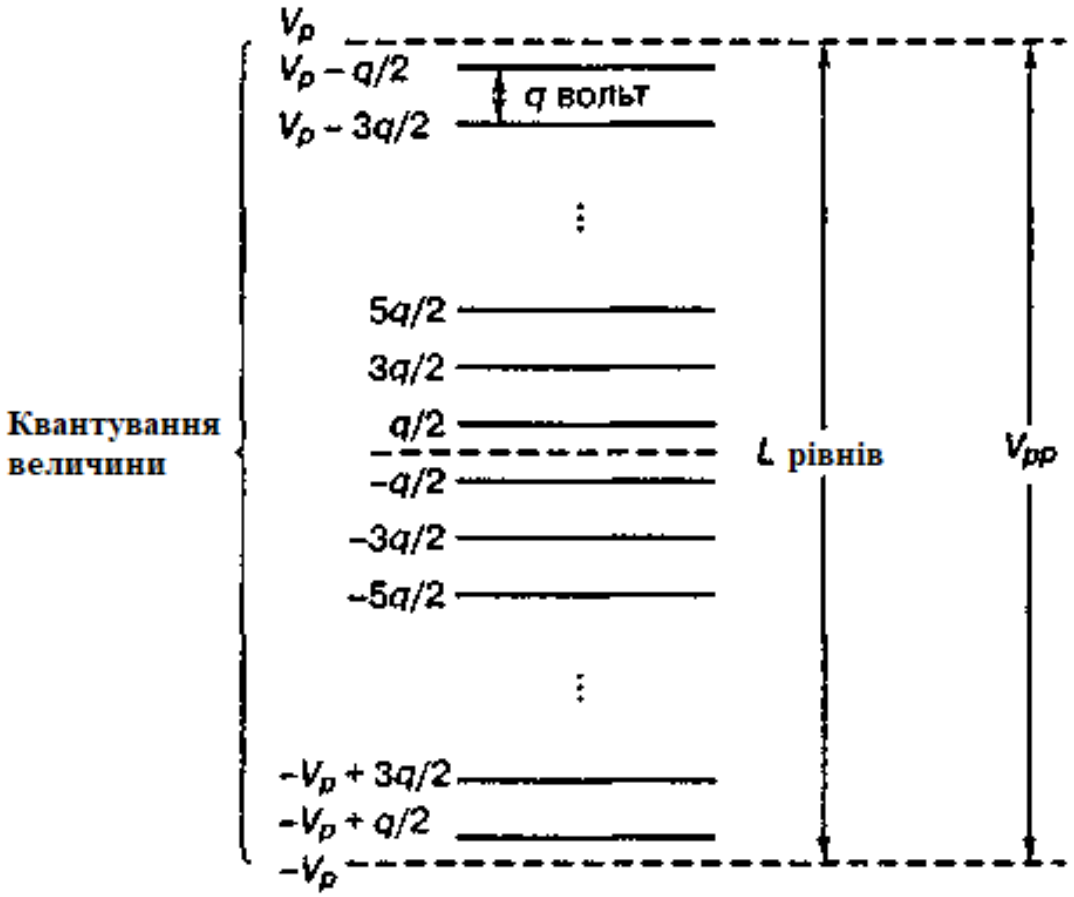


Рисунок 3.2 – Рівні квантування

Хорошим критерієм якості рівномірного пристрою квантування є його дисперсія - середній квадрат похибки при передбачуваному нульовому середньому:

$$e_i = \tilde{x}_i - x_i,$$

де $\tilde{x}_i$ - квантований відлік,

x_i - відлік, що квантується.

Якщо вважати, що похибка від квантування e рівномірно розподілена в межах інтервалу квантування шириною q (тобто приймає всі можливі значення з однаковою ймовірністю), то дисперсія похибки для квантуючого приладу дорівнює:

$$\sigma^2 = \int_{-q/2}^{q/2} e^2 p(e) de = = \int_{-q/2}^{q/2} e^2 \frac{1}{q} de = \frac{q^2}{12}, \qquad (3.1)$$

де $p(e)$ – густина ймовірностей похибки квантування;

$p(e)=1/q$ при $-q/2<q<q/2$ та 0 для інших значень q.

Дисперсія σ^2 відповідає *середній потужності шуму квантування*. Пікову потужність аналогового сигналу (нормовану на 1 Ом) можна виразити як:

$$V_P^2 = \left(\frac{V_{PP}}{2} \right)^2 = \left(\frac{q(L-1)}{2} \right)^2 \approx \frac{q^2 L^2}{2}, \qquad (3.2)$$

де L - число рівнів квантування.

Об'єднання виразів (3.1) та (3.2) дає відношення *пікової* потужності сигналу до *середньої* потужності квантового шуму $(S/N)_q$:

$$(S/N)_q = \frac{q^2 L^2 / 4}{q^2 / 12} = 3L^2, \qquad (3.3)$$

де $N = \sigma^2$ — середня потужність шуму квантування.

Очевидно, що співвідношення $(S/N)_q$ квадратично росте з числом рівнів квантування. В граничному значенні ($L \to \infty$) сигнал стає аналоговим

(нескінченна кількість рівнів квантування та нульовий шум квантування). Відзначимо, що для випадкових сигналів в параметр $(S/N)_q$ входить не максимальна, а середня потужність сигналу. В цьому випадку для отримання середньої потужності сигналу необхідно знати функцію густини ймовірності.

Лінійні прилади для квантування легко реалізувати та легко зрозуміти – в цьому їх очевидна перевага. До того ж, вибір параметрів пристроїв рівномірного квантування не вимагає жодних знань про статистику амплітуд та кореляційних властивостей вхідного сигналу.

3.1.2 Моделювання лінійного квантування сигналів в Matlab

Математичний опис процедури квантування. Процедуру квантування описують за допомогою вектора (розмірністю $L-1$) розділень області визначення сигналу $P_1, P_2, \ldots, P_{L-1}$ (vector partition) та вектора (розмірністю L) значень кодованого сигналу $C_1, C_2, \ldots, C_L$ (codebook). [1] Якщо x - поточне значення сигналу, тоді на вихід квантувача видаються значення по правилу, що показано в таблиці 3.1:

Таблиця 3.1

Вхід x	Вихід y
$x \le P_1$	C_1
$P_1 \le x \le P_2$	C_2
$P_2 \le x \le P_3$	C_3
…	…

$x \geq P_{L-1}$	C_L

М-функція квантування в **Matlab** має вигляд:

[ind, xQ, D] = quantiz(x, partition, codebook),

де **x** – вектор відліків сигналу; **partition** – вектор границь інтервалів $P_1, P_2, \ldots, P_{L-1}$; **codebook** – вектор $C_1, C_2, \ldots, C_L$ значень кодової книги; **ind** – вектор-стовбець індексів (номерів) интервалів; **xQ** – вектор-строка квантованих значень сигналу; **D** – середній квадрат похибки квантування:

$$D = \frac{1}{N} \sum_{n=1}^{N_t} (x_n - \tilde{x}_n)^2 .$$

Приклад застосування m-функції:

```
% квантування гармонічного (50 Гц) сигналу одиничної амплітуди
% частота дискретизації 1000 Гц
i=1:20;
x=cos(2*pi*50*i/1000);
partition=[-.75 -.25 .25 .75];
codebook=[-.825 -.5 0 .5 .825];
[ind, xQ, D] = quantiz(x, partition, codebook);
plot(i,x), hold on, stem(i,xQ)
legend('Вихідний сигнал','Результат квантування')
for n=1:4
prt(n,i)=partition(n); plot(i,prt,'k')
end
for m=1:5
cdbk(m,i)=codebook(m); plot(i,cdbk,'r-.')
end
figure
stem(i,ind), title('Індекси')
ind, xQ, D
```

Вхідний та квантований сигнали показані на рисунку 3.3, графік значень індексів показаний на рисунку 3.4.

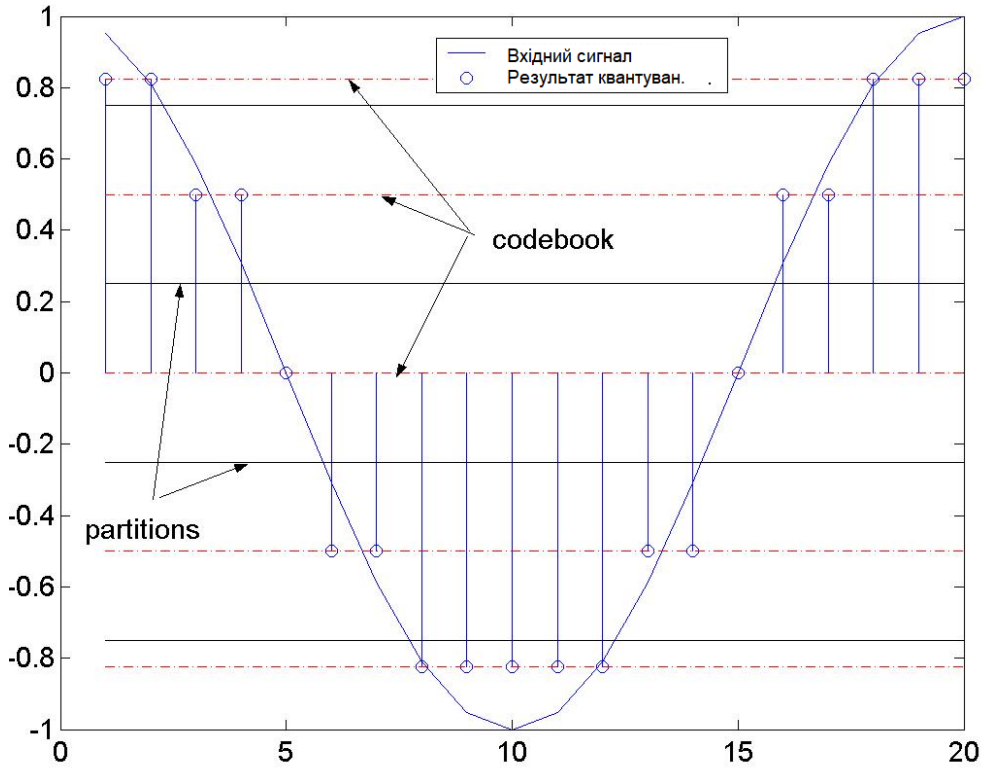


Рисунок 3.3 – Вхідний та квантований сигнали

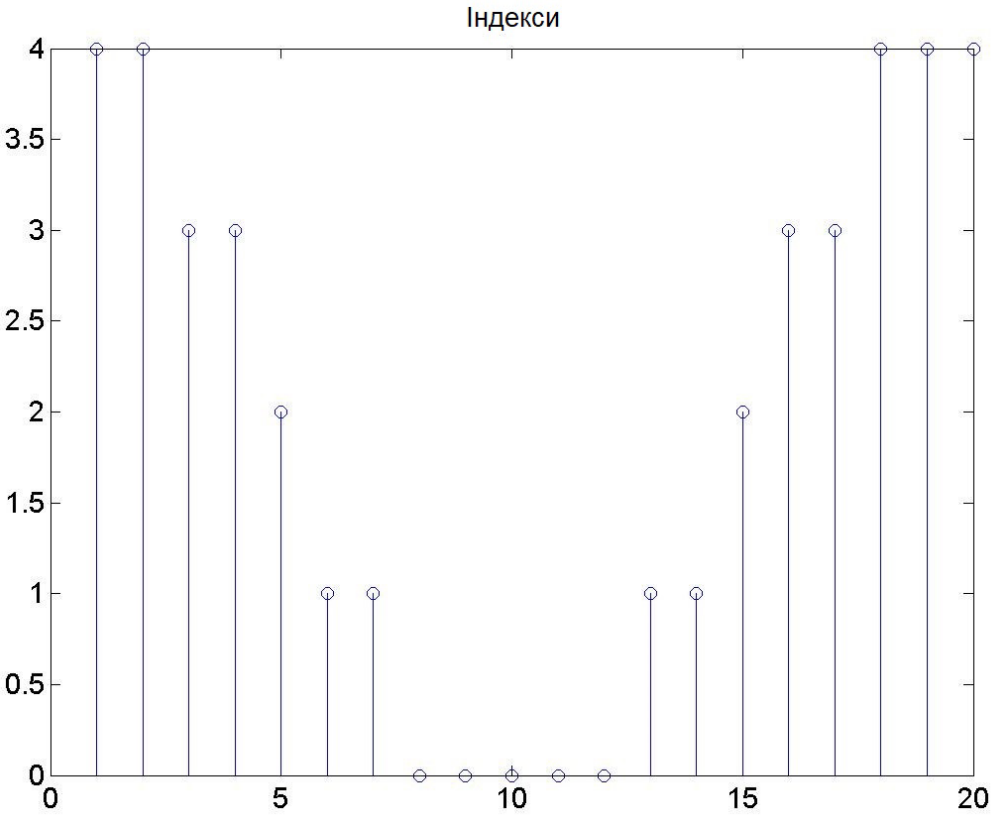


Рисунок 3.4 – Графік значень індексів

Розглянемо особливості лінійного квантування реальних сигналів. Як приклад обираємо звуковий файл в форматі «***.mp3**», наприклад, файл **vorona.mp3**, що містить сигнал звуків ворони.

Програма лінійного квантування сигналу, що міститься в файлі **vorona.mp3**, має наступний вигляд:

```
% lin_quant.m - програма лінійного квантування
[y,Fs]=audioread('vorona.mp3'); % читання сигналу (Fs – частота дискретизації)
ymax = max(abs(y)); y = y/ymax; % нормалізація сигналу
T = length(y)/Fs; disp('тривалість сигналу T = '); disp(T);  % 'тривалість сигналу в секундах
t = 1/Fs:1/Fs:length(y)/Fs; figure; plot(t,y);
title(['Вхідний сигнал, ' num2str(bits) ' біт, ' num2str(Fs) ' Гц'],'FontSize', 14, 'FontName', 'MS Sans Serif');
xlabel('Час, с','FontSize', 14, 'FontName', 'MS Sans Serif');grid on;
audioplayer(y,Fs); pause;
b = 8; % нова кількість біт
L = 2^b; % нова кількість рівнів
q = 2/(L-1); % квант рівня
partition = -1+q/2:q:1-q/2;
codebook = -1:q:1;
[ind, ylinq, D] = quantiz(y, partition, codebook);
figure; plot(t,ylinq);
title(['Рез-тат лін. квантування, ' num2str(b) ' біт, ' num2str(Fs) ' Гц'],'FontSize', 14, 'FontName', 'MS Sans Serif');
xlabel('Час, с','FontSize', 14, 'FontName', 'MS Sans Serif');grid on;
audioplayer(ylinq,Fs);
```

Результат лінійного 4-бітного квантування реального сигналу показаний на рисунку 3.5.

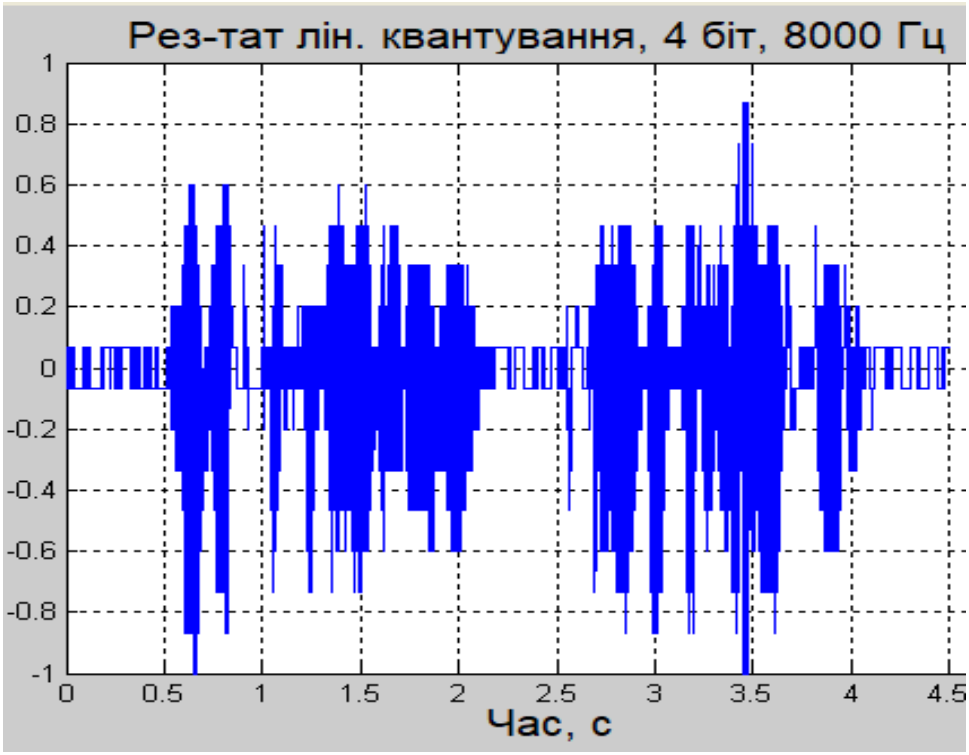


Рисунок 3.5 - Результат лінійного 4-бітного квантування реального сигналу

На графіку (рисунок 3.5) чітко видно, що малі рівні голосового сигналу суттєво спотворені.

3.1.3 Моделювання лінійного квантування сигналів в Simulink

Блок **Sampled quantizer encode** (рисунок 3.6) знаходиться (для Matlab) в розділі Communications Blockset\Source Coding та здійснює лінійне квантування. [1]

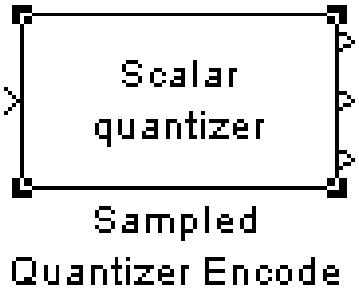


Рисунок 3.6 – Блок **Sampled quantizer encode**

Даний блок має один вхід та три виходи. На вхід подається сигнал, що підлягає квантуванню. На виходах, згори вниз: вектор **ind**, вектор **xQ**, вектор **квадратів** похибки квантування (а не **середній квадрат** похибки квантування, як у випадку m-функції **quantiz**).

Приклад застосування блоку **Sampled quantizer encode** показаний на рисунку 3.7.

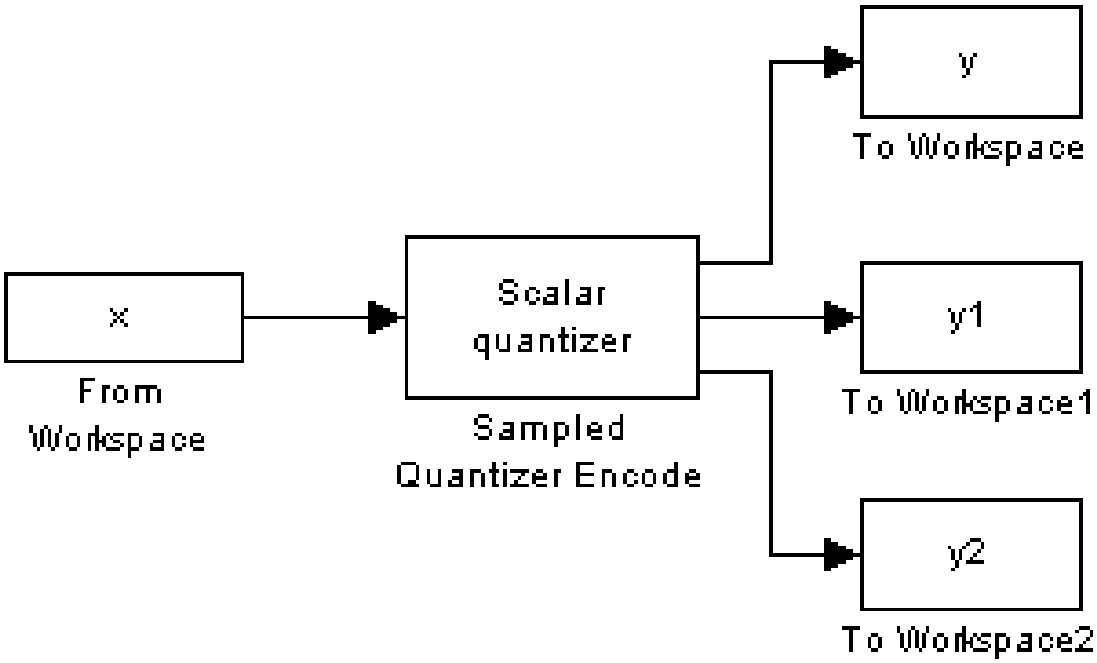


Рисунок 3.7 – Квантування сигналу блоком **Sampled quantizer encode**

Налаштування блоку **From Workspace** показані на рисунку 3.8, а блоку **Sampled quantizer encode** - на рисунку 3.9.

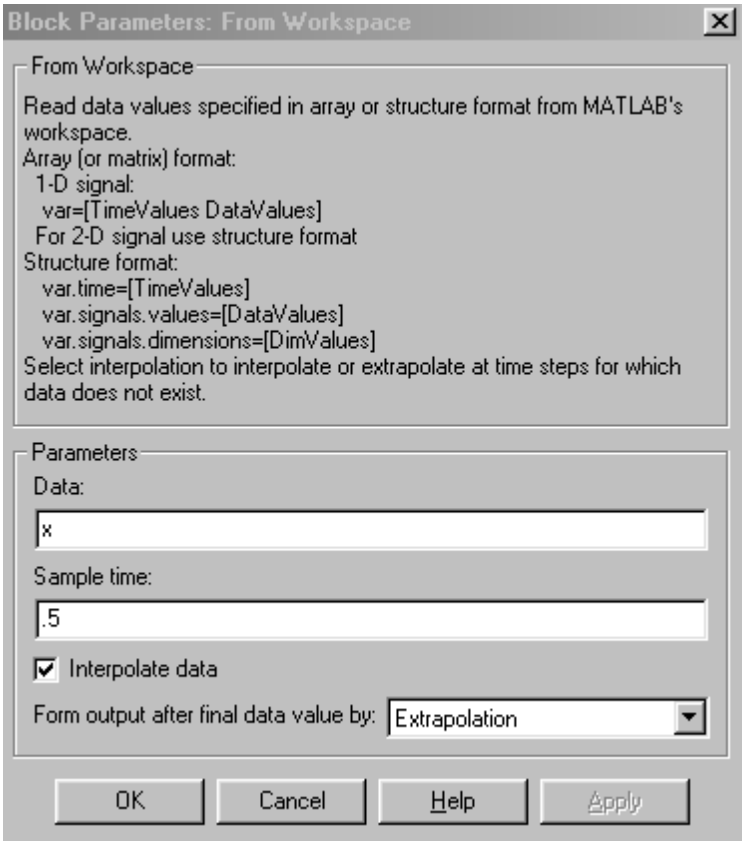


Рисунок 3.8 – Вікно-маска для налаштування блоку **From Workspace**

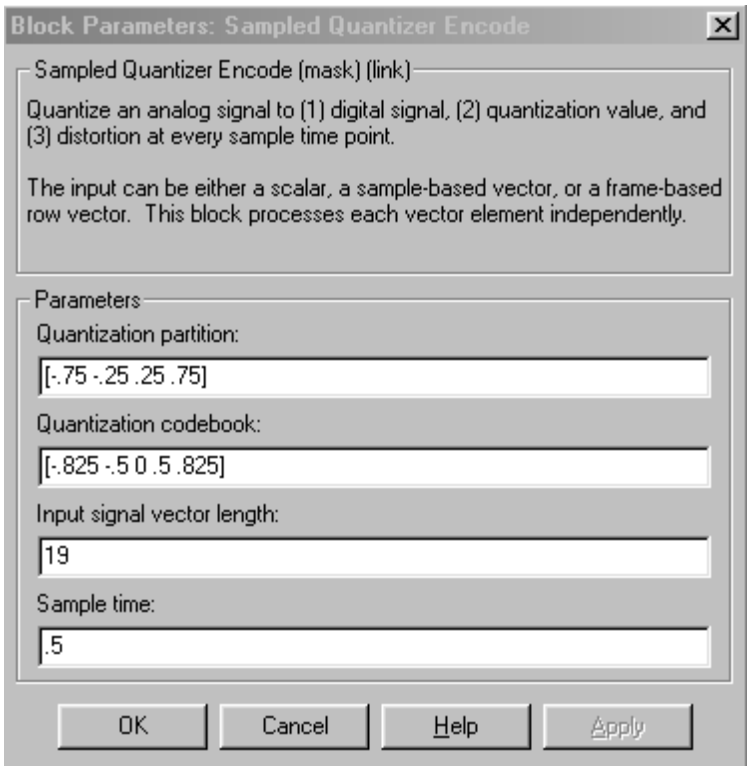


Рисунок 3.9 – Вікно-маска для налаштування блоку **Sampled quantizer encode**

Варто додати, що довжина вхідного вектора блоку **Sampled quantizer encode** береться не 20, а 19, що необхідно для нормальної роботи схеми (мається на увазі, що в кожному з варіантів лабораторної роботи довжина вхідного вектора може бути іншою).

Налаштування блоку **To Workspace** показані на рисунку 3.10.

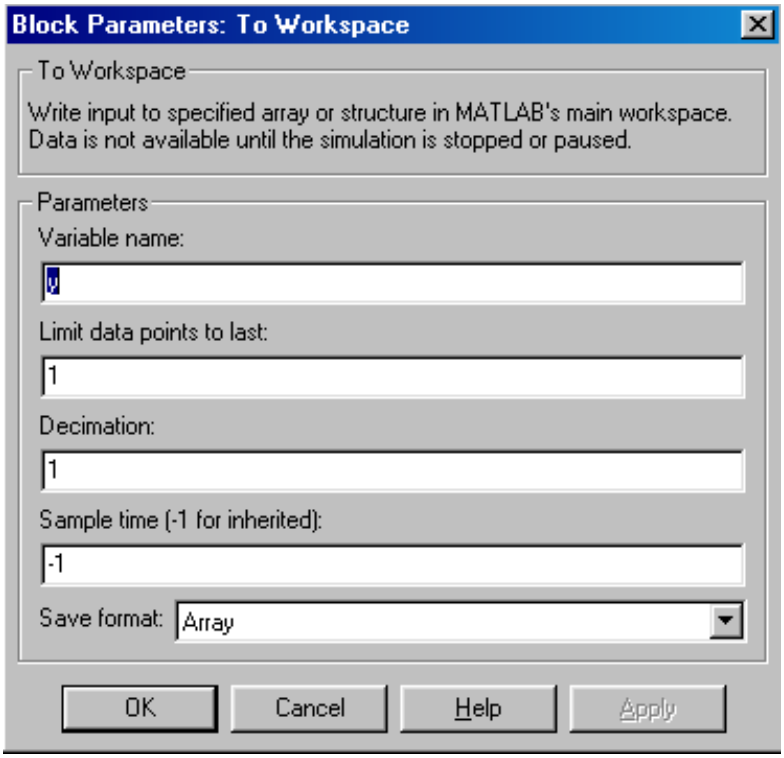


Рисунок 3.10 – Вікно-маска для налаштування блоку **To Workspace**

Зверніть увагу (рисунок 3.10): формат результату – масив (**array**), а не структура (**structure**). Дані в форматі масиву (**array**) зручніше аналізувати, оскільки не потрібно витрачати час на їх вилучення із структури (**structure**) за допомогою спеціальних доволі великих команд.

Далі в командному вікні **Matlab** будується графік (рисунок 3.11) масиву **y** за допомогою команди **stem(y)**:

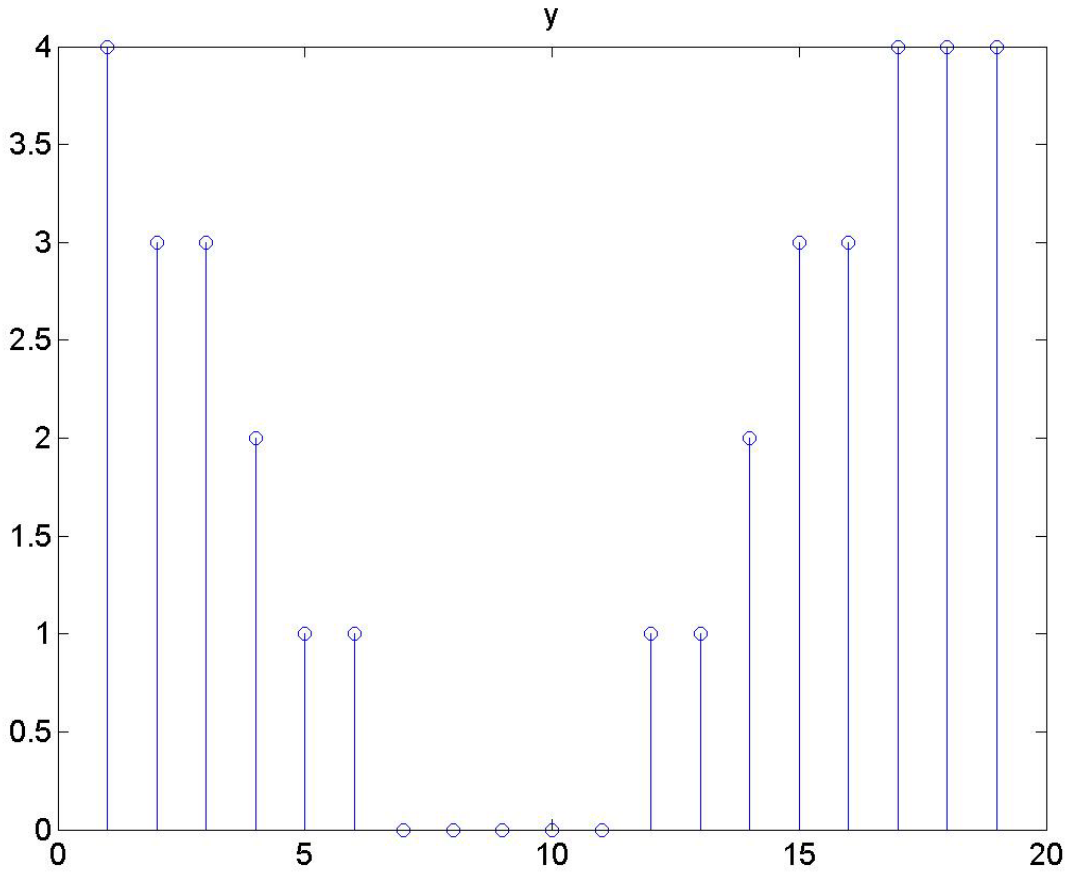


Рисунок 3.11 – Графік масиву **y**

Порівнюючи рисунок 3.11 з рисунком 3.4, бачимо, що на рисунку 3.11 зображений масив індексів, але без першого елементу. Таким чином, «загубився» один (перший) елемент. Це зумовлено тим, що розмір вхідної вибірки ми задавали 19.

Аналогічно будуємо графік вихідного сигналу **y1** з другого виходу (рисунок 3.12).

Порівнюючи рисунок 3.12 з рисунком 3.3, бачимо, що на рисунку 3.12 зображений квантований сигнал, але також без першого відліку.

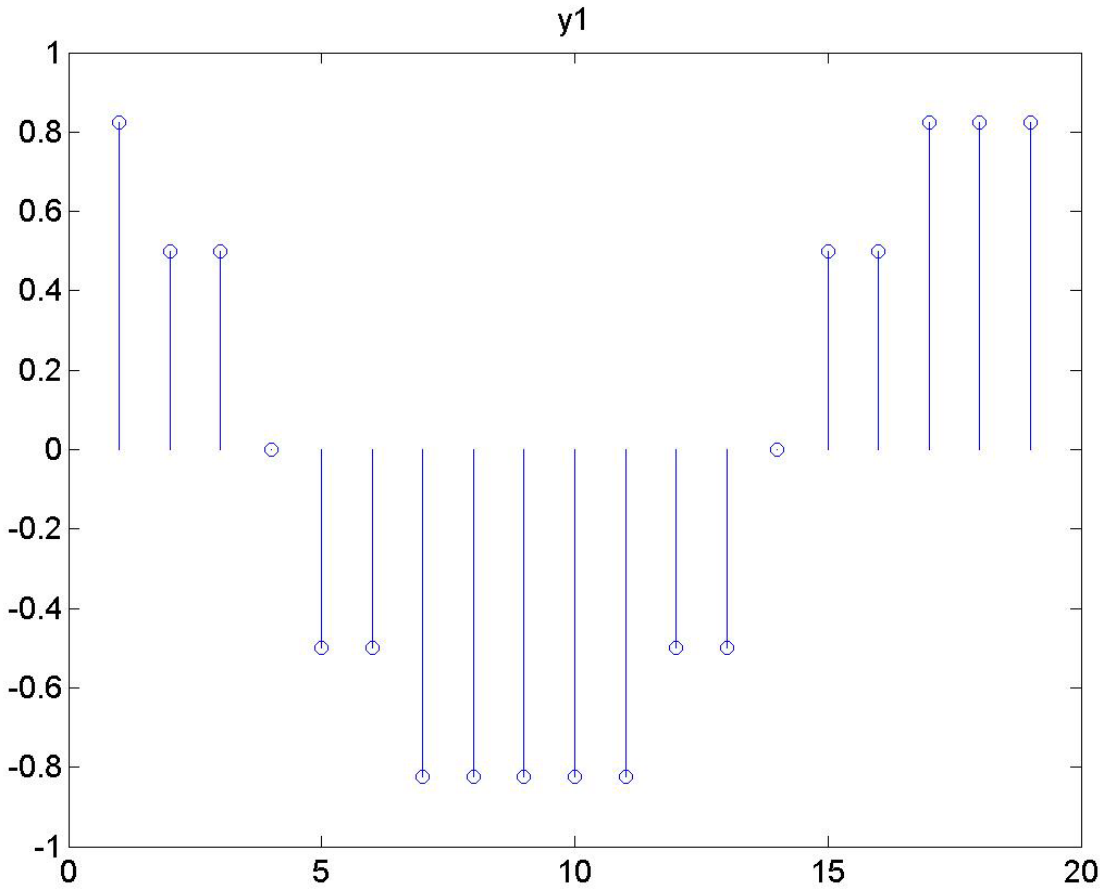


Рисунок 3.12 – Графік вихідного сигналу **y1**

Аналогічно будуємо графік масиву **y2** (рисунок 3.13).

Порівнюючи рисунок 3.13 з рисунком 3.3, можна зробити висновок, що на рисунку 3.13 зображений квадрат похибки квантування (масив), а не середній квадрат похибки квантування (число).

Таким чином, реалізація квантування за допомогою **m**-функції **quantiz** та за допомогою блоку **Sampled quantizer encode** приводить до результатів, які дещо відрізняються одне від одного.

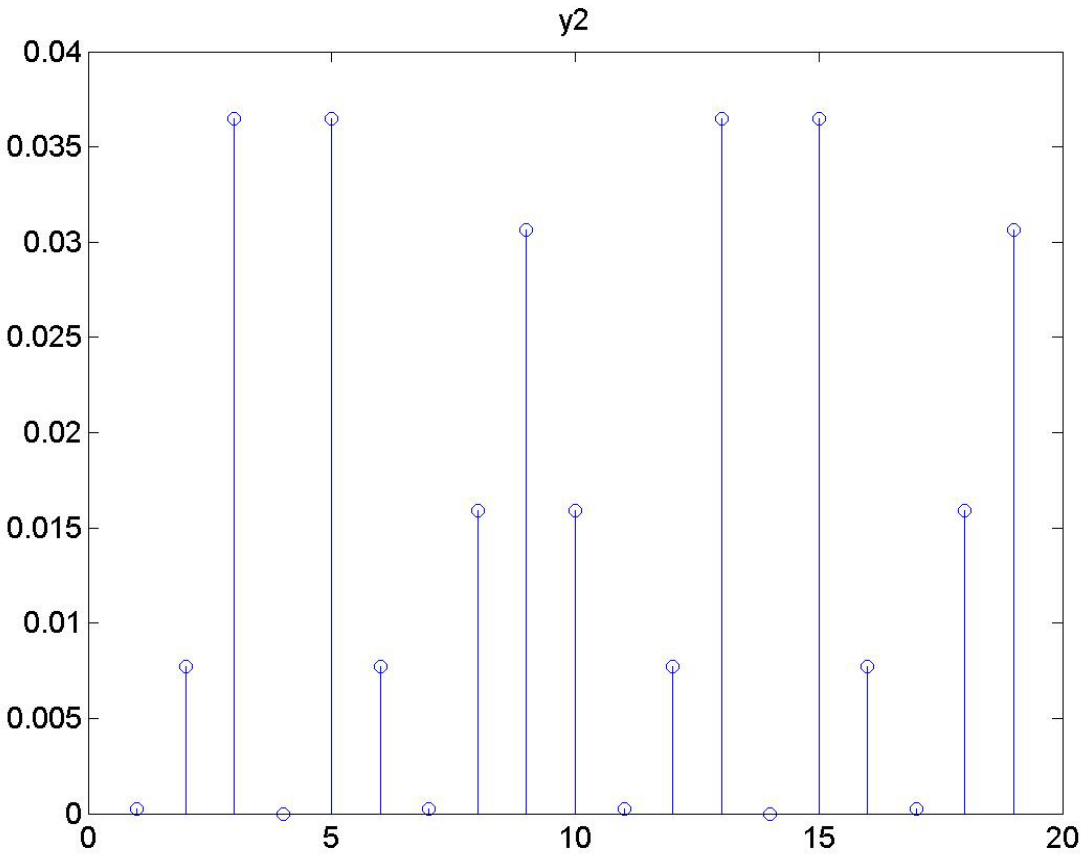


Рисунок 3.13– Графік вихідного сигналу **y2**

3.2 Підготовка до лабораторної роботи

3.2.1 Ознайомитися з короткими теоретичними відомостями та порядком роботи з середовищами MatLab та Simulink.

3.2.2 Написати програму у вигляді файлу-сценарію (m-файлу) для моделювання гармонічного сигналу, який заданий функцією $x(t)$:

$$x(t)=X_m \cos(2\pi f_0 t+\varphi).$$

Параметри для гармонічного сигналу обираються з таблиці 3.2, згідно з заданим варіантом **n**, де **n** – номер студента у списку групи (або у послідовно об'єднаному списку груп у порядку зростання їх номерів).

Тривалість T реалізації сигналу обрати рівною двом періодам сигналу.

Частоту дискретизації F_s задати в 20 разів більше частоти сигналу.

Таблиця 3.2 – Параметри гармонічного сигналу

№ варіанту, **n**	X_m, В	f_0, Гц	φ, рад	L
1…5	n	n*300	0	8
6…10	n/2	n*200	π/4	7
11…15	n/3	n*100	π/2	6
16…20	n/4	n*200	π	5
21…25	n/6	n*250	π/3	4
26…30	n/7	n*50	π/5	9

3.2.3 Написати програму у вигляді файлу-сценарію для моделювання лінійного квантування сигналу *x(t)*, що був заданий за п.3.2.2, та побудови графіків: вихідного сигналу; квантованого сигналу; значень індексів; а також виводу середнього квадрату похибки квантування.

Параметри команд **codebook** та **partition** задаються виходячи із заданого числа L рівнів квантування (таблиця 3.2).

3.2.4 Написати програму у вигляді файлу сценарію для моделювання лінійного квантування реального звукового сигналу, що міститься в заданому викладачем файлі (наприклад, **vorona.mp3**). Вивести на екран результат лінійного квантування реального сигналу.

Параметри команд **codebook** та **partition** задаються виходячи з числа L_{new} рівнів квантування, яке визначається за формулою:

$$L_{new} = 2^L ,$$

де значення параметра L (для даного пункту дорівнює числу біт) обираються з таблиці 3.2 згідно з заданим варіантом, але не перевищує значення розрядності (числа біт) вхідного сигналу.

Моделювання провести для нормованих значень сигналу.

3.2.5 Змоделювати в середовищі Simulink процедуру лінійного квантування гармонічного сигналу, параметри якого вибираються з таблиці 3.2 згідно з заданим варіантом.

На екран вивести: вектор-стовпець індексів (номерів) інтервалів **(ind)**, вектор-рядок квантованих значень сигналу **(xQ)**, вектор **квадратів** похибки квантування.

3.2.6 Змоделювати в середовищі Simulink процедуру лінійного квантування реального звукового сигналу, який міститься в заданому викладачем файлі (наприклад, **vorona.mp3**). Необхідні параметри обираються згідно із заданим варіантом аналогічно п. 3.2.4.

Вивести на екран осцилоскопа вихідні сигнали всіх блоків та загальний результат лінійного квантування реального сигналу.

Примітка. Моделі у середовищі Simulink повинна працювати автономно, незалежно від параметрів у **Workspace**, тобто при «очищеному» **Workspace**.

3.3 Порядок виконання лабораторної роботи

3.3.1 Увімкніть ПЕОМ та запустіть програму MatLab.

3.3.2 Наберіть програму лінійного квантування гармонічного сигналу, написану за п. 3.2.3 та запустіть її на виконання. Отримані результати занесіть у протокол лабораторної роботи у вигляді скріншотів.

3.3.3 Наберіть програму лінійного квантування реального звукового сигналу, написану за п. 3.2.4 і запустіть її виконання. Отримані результати занесіть у протокол лабораторної роботи у вигляді скріншотів.

3.3.4 Запустіть модель, підготовлену за п. 3.2.5, що реалізує процедуру лінійного квантування гармонічного сигналу, на виконання. Отримані результати занесіть у протокол лабораторної роботи у вигляді скріншотів.

3.3.5 Запустіть модель, підготовлену за п. 3.2.6, що реалізує процедуру лінійного квантування реального звукового сигналу на виконання. Отримані результати занесіть у протокол лабораторної роботи у вигляді скріншотів.

3.4 Зміст звіту з лабораторної роботи

Звіт з лабораторної роботи повинен містити:

3.4.1 При допуску до лабораторної роботи:

3.4.1.1 Ціль лабораторної роботи.

3.4.1.2 Короткі теоретичні відомості.

3.4.1.3 Написані за пунктами 3.2.2 - 3.2.4 **програми з коментарями** для середовища MatLab.

3.4.1.3 Блок-схеми моделей в середовищі Simulink за пунктами 3.2.5, 3.2.6 з вказуванням **значень параметрів,** що задаються для блоків.

3.4.2 При здачі лабораторної роботи до матеріалів за п. 3.4.1 мають бути додані результати виконання роботи.

3.4.2.1 Результати виконання програм за пунктами 3.3.2, 3.3.3.

3.4.2.2 Результати роботи моделей у середовищі Simulink за пунктами 3.3.4, 3.3.5.

3.4.2.3 Висновки.

3.5 Контрольні питання

3.5.1 Що таке квантування сигналу?

3.5.2 Як здійснюється лінійне квантування?

3.5.3 Як визначається похибка від квантування?

3.5.4 Які особливості процедури **quantiz** MatLab?

3.5.5 У чому відмінність моделювання квантування в Simulink від процедури **quantiz** MatLab?

Лабораторна робота № 4

Дослідження методів нелінійного квантування за допомогою системи Matlab і Simulink

Ціль роботи: дослідити властивості нелінійного квантування в середовищах Matlab і Simulink

4.1 Теоретичні відомості

4.1.1 Квантування сигналів

Квантування сигналів являє собою перетворення величини, що безперервно змінюється, у величину, що змінюється ступінчасто із заданими розмірами ступенів. [1,2,6,7]

Ступенем квантування називається різниця між двома сусідніми заданими значеннями квантованої величини $q_к$.

Квантування широко застосовується в процесі управління за необхідності впливу на технологічний процес сигналом з параметром точно заданого розміру.

Квантований сигнал у функції часу може бути виражений за допомогою одиничної ступінчатої функції $qk1(t-t_{Ni})$, що дорівнює одиниці, якщо аргумент додатній, або дорівнює нулю — якщо від'ємний:

$$X_{K.B.}(t) = N_i(t_{Ni}) * qk1(t-t_{Ni}),$$

де $N_i(t_{Ni})$ – коди сигналу, i = 1, 2, …

Квантування може виконуватися з рівномірними (рисунок 4.1, *а, б*) та нерівномірними (рисунок 4.1, *в, г*) ступенями квантування. Ступінь квантування виражається в одиницях даної величини та у відносних одиницях.

Рівномірне квантування застосовується частіше, зокрема під час вимірювань. Як правило, при цьому кількість рівнів або ступенів квантування достатньо велика.

Нерівномірне квантування застосовується зазвичай при невеликому числі ступенів квантування, наприклад в статистичних аналізаторах, при передачі голосових сигналів та ін.

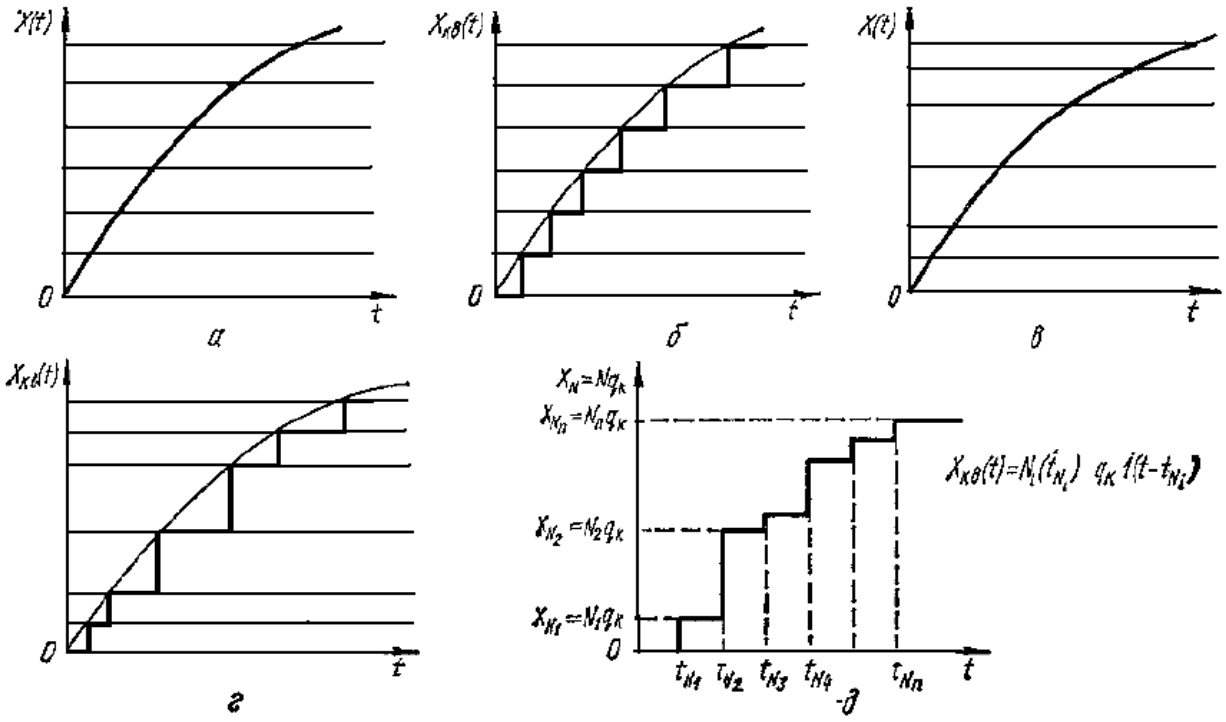


Рисунок 4.1 - Квантування аналогових сигналів:
а, б — з рівномірними ступенями; *в, г* — з нерівномірними ступенями; д — квантування відомої величини на виході міри, що регулюється

Квантування є однією з найвідповідальніших операцій процесу вимірювання, перетворення сигналу. Квантування зразкової величини

здійснюється за допомогою багатоканальної або регульованої міри. При цьому на виході міри отримуються величини заданого розміру зі значеннями, що виражаються в одиницях даної фізичної величини (рисунок 3.1, д). Квантування в одиницях відносних величин може бути виконаним в багатоканальному або регульованому масштабному перетворювачі (відносне квантування).

Похибка від квантування Δ_k є методичною похибкою методу відображення неперервної за розміром величини обмеженим по числу розрядів числом, тобто методичною похибкою самого вимірювання. Похибка від квантування дорівнює різниці між результатом вимірювання без попереднього перетворення x_n та істинним значенням величини X, якщо похибки міри та компаратора дорівнюють нулю:

$$\Delta_k = x_n - X.$$

Похибка від квантування Δ_k по характеру зміни може бути систематичною постійною по величині (при $X = const$) або випадковою (при $X = var$ по випадковому закону).

Нерівномірне (нелінійне) квантування. Нелінійні пристрої квантування, що забезпечують нерівномірне квантування, застосовуються тоді, коли виникає необхідність врахувати статистику амплітуд і кореляційні властивості вхідного сигналу.

Існують додатки, для яких рівномірні пристрої квантування найкращі. Це – обробка музичних сигналів, обробка зображень, контроль процесів, вимірювання та ряд інших. Для деяких інших додатків більш прийнятні нерівномірні квантуючі пристрої. Найважливішим прикладом такого роду є обробка мовних сигналів у системах зв’язку.

Людська мова характеризується унікальними статистичними властивостями, одна з яких проілюстрована на рисунку 4.2.

Рисунок 4.2 – Статистичний розподіл амплітуд мови однієї людини

На осі абсцис відкладено амплітуди мовного сигналу, нормовані за середньоквадратичним значенням амплітуд, а на осі ординат – ймовірність. Для більшості каналів мовного зв'язку домінують дуже низькі рівні сигналів: 50% часу напруга, що характеризує енергію виявленої мови, складає менше чверті середньоквадратичного значення. Значення з великими амплітудами зустрічаються відносно рідко: тільки 15% часу напруга перевищує середньоквадратичне значення.

При передачі мови система з рівномірним квантування буде неекономною – багато рівнів квантування будуть використовуватися доволі рідко. Крім того, в такій системі шум квантування буде однаковим для всіх амплітуд сигналу. Отже, при такому квантування відношення сигнал/шум буде гірше для сигналів низьких рівнів, ніж для сигналів високих рівнів.

Нерівномірне квантування може забезпечити найкраще квантування слабких сигналів і грубе квантування сильних сигналів. Значить, в цьому випадку шум квантування може бути пропорційним сигналу. Результатом є підвищення

загального відношення сигнал/шум – зменшення шуму для домінуючих слабких сигналів за рахунок підвищення шуму для сильних сигналів, які зустрічаються рідко. На рисунку 4.3 порівнюється квантування слабкого та сильного сигналів при рівномірному і нерівномірному квантуванні. Ступінчасті сигнали представляють собою апроксимації аналогових сигналів (після введення спотворення внаслідок квантування).

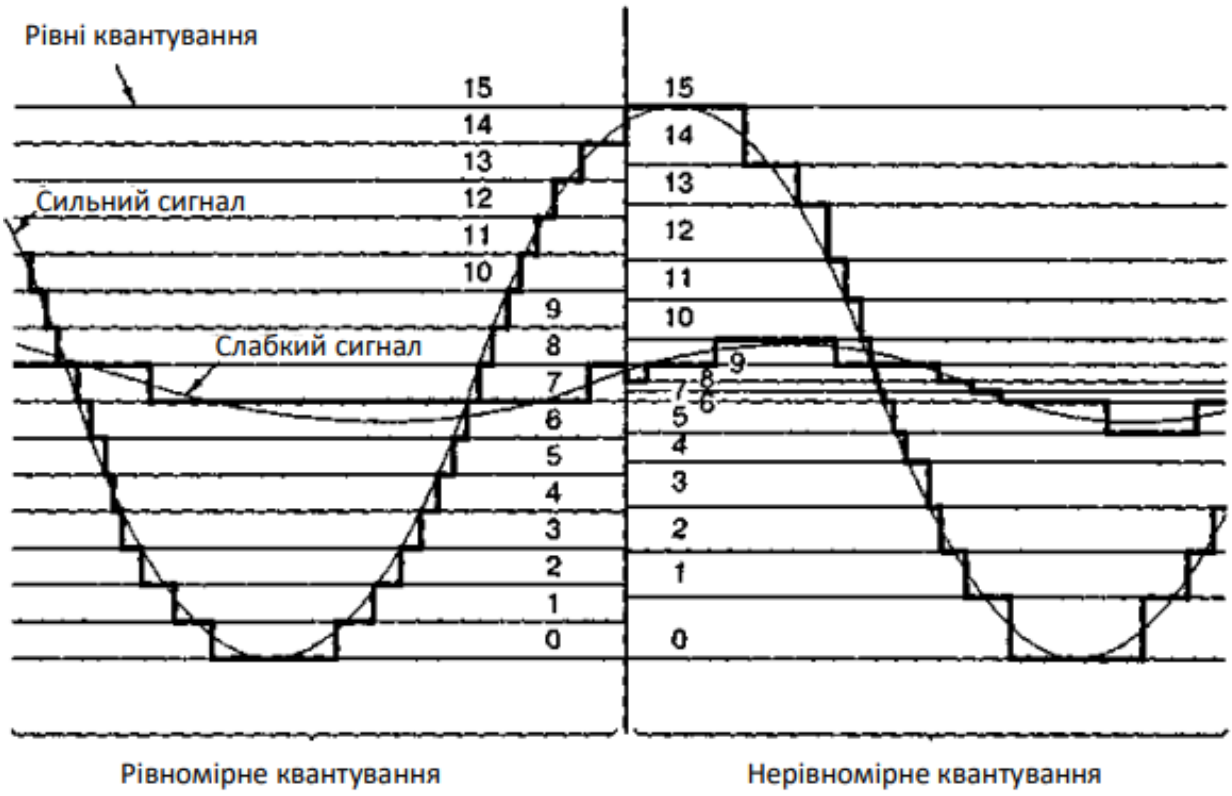


Рисунок 4.3 – Рівномірне і нерівномірне квантування сигналів

Одним зі способів отримання нерівномірного квантування є використання пристрою з нерівномірним квантуванням з характеристикою, показаною на рисунку 4.4, *а.* Набагато частіше нерівномірне квантування реалізується наступним чином: спочатку вихідний сигнал деформується за допомогою пристрою, що має логарифмічну характеристику стиснення, що показана на рисунку 4.4, *б,* а потім використовується пристрій квантування з рівномірним кроком. Для сигналів малої амплітуди характеристика стиснення має більш

крутий фронт, ніж для сигналів більшої амплітуди. Отже, зміна даного сигналу при малих амплітудах торкнеться більшої кількості рівномірно розміщених рівнів квантування, ніж та сама зміна при великих амплітудах. Характеристика стиснення ефективно змінює розподіл амплітуд вхідного сигналу, отож на виході системи стиснення вже не існує переваги сигналів *малих* амплітуд. Після стиснення деформований сигнал подається на вхід рівномірного (лінійного) пристрою квантування з характеристикою, показаною на рисунку 4.4, *в.* Після прийому сигнал пропускається через пристрій з характеристикою, оберненою до показаної на рисунку 4.4, *б,* яка називається *розширенням*, тож загальна передача не є деформованою. Описана пара етапів обробки сигналу (стиснення і розширення) в сукупності зазвичай називається *компандуванням*.

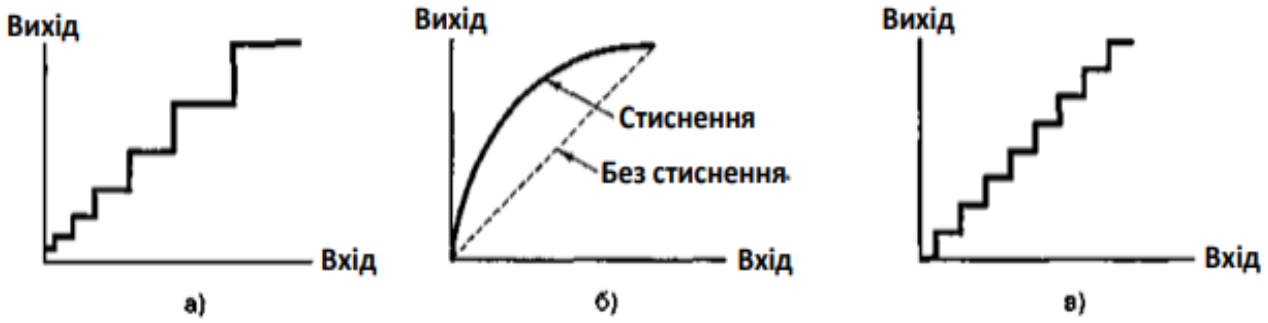


Рисунок 4.4 – Характеристики пристроїв:
а) нерівномірного квантування; б) стиснення; в) рівномірного квантування

В Північній Америці μ-рівнева характеристика пристрою стиснення описується наступним законом:

$$y = y_{\max} \frac{\ln\left[1 + \mu(|x|/x_{\max})\right]}{\ln(1+\mu)} \operatorname{sgn} x, \qquad (4.1)$$

де

$$\operatorname{sgn} x = \begin{cases} +1 \text{ при } x \geq 0 \\ -1 \text{ при } x < 0 \end{cases},$$

μ - додатна константа;

x і y - напруги на вході та виході;

x_{max} і y_{max}- максимальні амплітуди напруг на вході та виході, відповідно.

В Європі для опису характеристики пристрою стиснення використовується трохи інший закон:

$$y = \begin{cases} y_{max} \dfrac{A(|x|/x_{max})}{1+\ln A} \operatorname{sgn} x, & 0 < \dfrac{|x|}{x_{max}} \leq \dfrac{1}{A}; \\ y_{max} \dfrac{1+\ln\left[A(|x|/x_{max})\right]}{1+\ln A} \operatorname{sgn} x, & \dfrac{1}{A} < \dfrac{|x|}{x_{max}} \leq 1; \end{cases} \quad (4.2)$$

Тут A – додатна константа, а x і y визначені так само, як і в формулі (4.1).

Вигляд характеристик (4.1) і (4.2) наведено на рисунку 4.5.

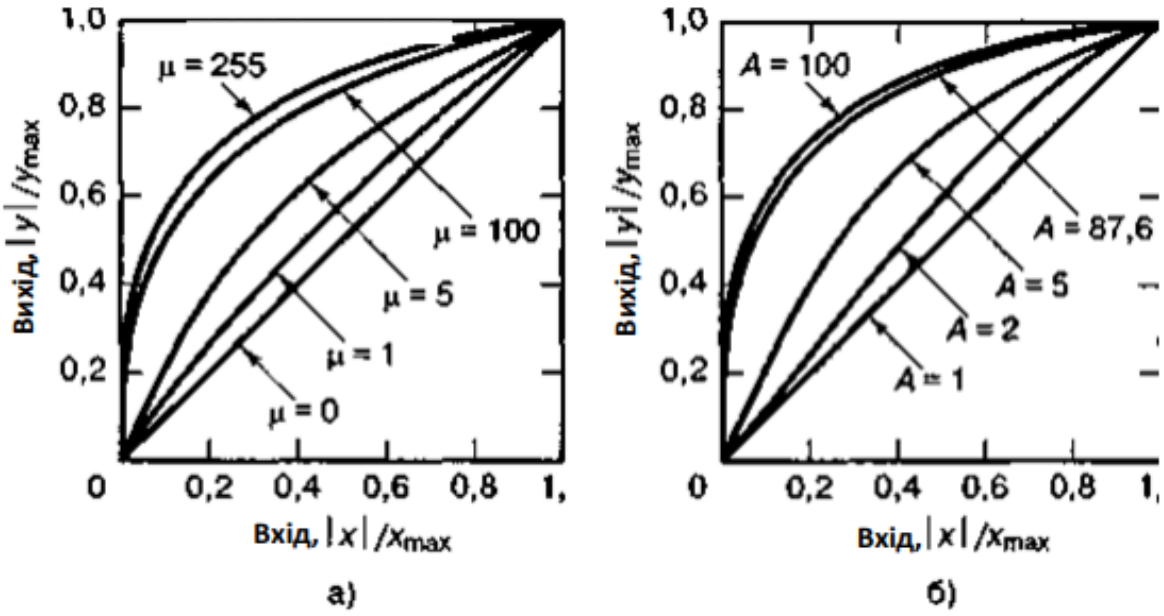


Рисунок 4.5 – Характеристики пристроїв стиснення:

а) для різних значень μ; б) для різних значень A

2.1.2 Моделювання квантування сигналів в Matlab

Математичний опис процедури квантування. Процедуру квантування описують за допомогою вектору (розмірністю $L-1$) поділу області визначення сигналу $P_1, P_2, \ldots, P_{L-1}$ (vector partition) і вектору (розмірністю L) значень кодованого сигналу $C_1, C_2, \ldots, C_L$ (codebook) [1]. Якщо x - поточне значення сигналу, тоді на вихід квантувача видаються значення за правилом, наведеним у таблиці 4.1:

Таблиця 4.1

Вхід x	Вихід y
$x \le P_1$	C_1
$P_1 \le x \le P_2$	C_2
$P_2 \le x \le P_3$	C_3
…	…
$x \ge P_{L-1}$	C_L

М-функція квантування в **Matlab** має вигляд:

[ind, xQ, D] = quantiz(x, partition, codebook),

де **x** – вектор відліків сигналу;

partition – вектор границь інтервалів $P_1, P_2, \ldots, P_{L-1}$;

codebook – вектор $C_1, C_2, \ldots, C_L$ значень кодової книги;

ind – вектор-стовбець індексів (номерів) інтервалів;

xQ – вектор-рядок квантованих значень сигналу;

D – середній квадрат похибки квантування:

$$D = \frac{1}{N} \sum_{n=1}^{N_t} (x_n - \tilde{x}_n)^2 \ .$$

Приклад застосування m-функції:

```
% квантування гармонічного (50 Гц) сигналу одиничної амплітуди
% частота дискретизації 1000 Гц
i=1:20;
x=cos(2*pi*50*i/1000);
partition=[-.75 -.25 .25 .75];
codebook=[-.825 -.5 0 .5 .825];
[ind, xQ, D] = quantiz(x, partition, codebook);
plot(i,x), hold on, stem(i,xQ)
legend('Вхідний сигнал','Результат квантуван.')
for n=1:4
prt(n,i)=partition(n); plot(i,prt,'k')
end
for m=1:5
cdbk(m,i)=codebook(m); plot(i,cdbk,'r-.')
end
figure
stem(i,ind), title('Індекси')
ind, xQ, D
```

Вхідний та квантований сигнали наведені на рисунку 4.6, графік значень індексів наведено на рисунку 4.7.

Розглянемо особливості нелінійного квантування реальних сигналів. Для прикладу оберемо звуковий файл в форматі «***.mp3**», наприклад, файл **vorona.mp3**, що містить сигнал звуків ворони.

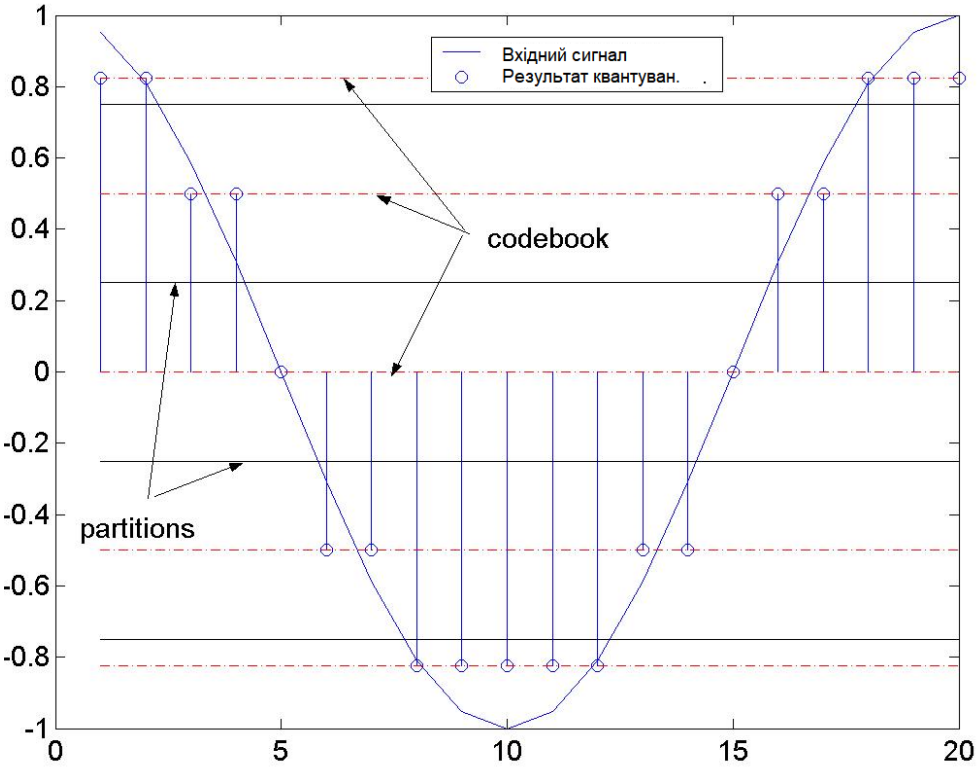


Рисунок 4.6 – Вхідний та квантований сигнали

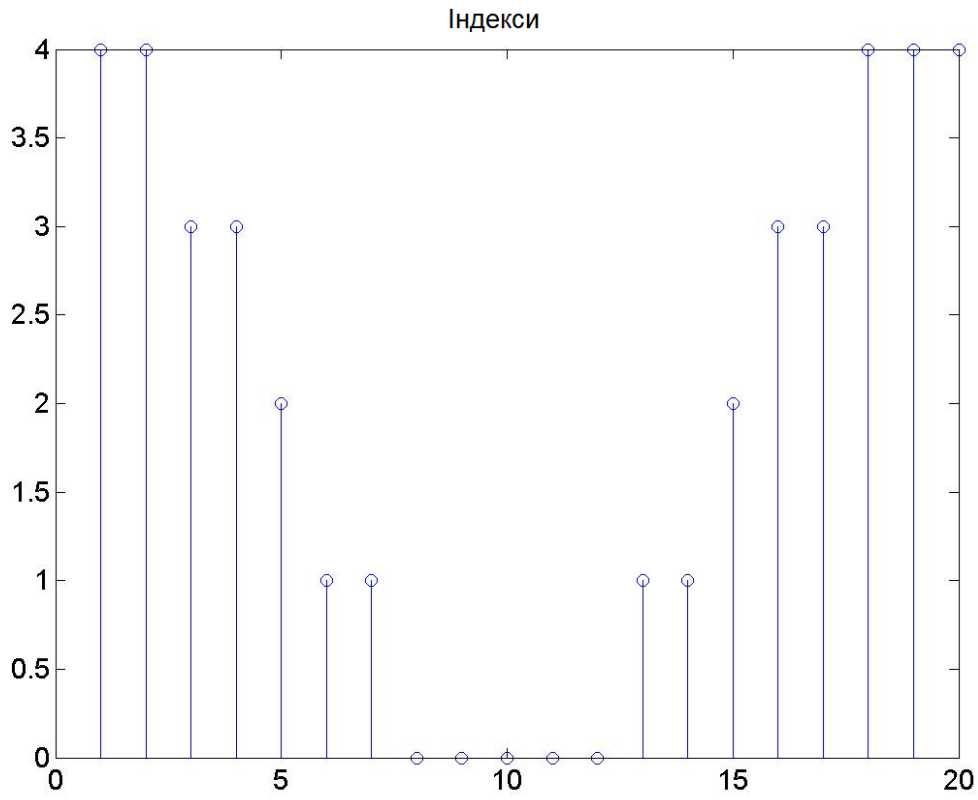


Рисунок 4.7 – Графік значень індексів

Програма нелінійного квантування реального сигналу, що міститься у файлі **vorona.mp3**, за **A**-законом має наступний вигляд:

```
% nonlin_quant.m – програма нелінійного квантування
[y,Fs]=audioread('vorona.mp3'); % читання сигналу (Fs – частота дискретизації)
ymax = max(abs(y)); y = y/ymax; % нормалізація сигналу
T = length(y)/Fs; disp('протяжність сигналу T = '); disp(T);  % протяжність сигналу в секундах
t = 1/Fs:1/Fs:length(y)/Fs; figure; plot(t,y);
title(['Вхідний сигнал, ' num2str(bits) ' бит, ' num2str(Fs) ' Гц'],'FontSize', 14, 'FontName', 'MS Sans Serif');
xlabel('Время, с','FontSize', 14, 'FontName', 'MS Sans Serif');grid on;
audioplayer(y,Fs); pause;
%=== попереднє спотворення сигналу за A-законом ===
A = 100;       % значення параметра A
if 0<abs(y)<=1/A
   yn = A*abs(y).*sign(y)/(1+log(A));
else
   if y == 0
      y = 1e-3 ;
   end
      yn = (1+log(A*abs(y))).*sign(y)/(1+log(A));
end
sum_nan = sum(isnan(yn));   % кількість NaN
yn = yn(~isnan(yn));        % видалення  NaN
t1 = 1/Fs:1/Fs:(length(y)-sum_nan)/Fs;
figure; plot(t1,yn);
title(['Предспотворений сигнал, ' num2str(bits) ' бит, ' num2str(Fs) ' Гц'],'FontSize', 14, 'FontName', 'MS Sans Serif');
xlabel('Час, с','FontSize', 14, 'FontName', 'MS Sans Serif');grid on;
audioplayer(yn,Fs); pause;
%=== квантування попередньо спотвореного сигналу =======
```

```
b = 8; % нова кількість біт
L = 2^b; % нова кількість рівнів
q = 2/(L-1); % квант рівня
partition = -1+q/2:q:1-q/2;
codebook = -1:q:1;
[ind, yq, D] = quantiz(yn, partition, codebook);
figure; plot(t1,yq);
title(['Квантування предспотвор. сигналу, ' num2str(b) ' бит, '
num2str(Fs) ' Гц'],'FontSize', 14, 'FontName', 'MS Sans Serif');
xlabel('Время, с','FontSize', 14, 'FontName', 'MS Sans Serif');grid on;
% === зворотне нелінійне спотворення сигналу ======
if 0<abs(yq)<=1/(1+log(A))
   yv = (1+log(A))*abs(yq).*sign(yq)/A;
else
   yv = exp((1+log(A))*abs(yq)-1).*sign(yq)/A;
end
figure; plot(t1,yv);
title(['Рез-тат нелін. квантування, ' num2str(b) ' бит, ' num2str(Fs) '
Гц'],'FontSize', 14, 'FontName', 'MS Sans Serif');
xlabel('Время, с','FontSize', 14, 'FontName', 'MS Sans Serif');grid on;
audioplayer(yv,Fs);
```

Результат нелінійного 4-бітного квантування за А-законом (А=100) реального сигналу показано на рисунку 4.8.

На графіку (рисунок 4.8) чітко видно, що малі рівні мовного сигналу в цілому спотворені значно менше, однак спостерігаються локальні сплески. Попередні дослідження показали, що причиною виникнення цих сплесків є нульові значення вхідного сигналу.

Рисунок 4.8 - Результат нелінійного (A=100) 4-бітного квантування реального сигналу

4.1.3 Моделювання квантування сигналів в Simulink

Розглянемо реалізацію нелінійного квантування в середовищі Simulink.

Приклад схеми для нелінійного квантування сигналу наведено на рисунку 4.9.

Блок **Sampled quantizer encode** (рисунок 4.10) знаходиться (для Matlab) в розділі Communications Blockset\Source Coding і здійснює лінійне квантування.

Даний блок має один вхід та три виходи. На вхід подається сигнал, що квантується. На виходах, зверху вниз: вектор **ind**, вектор **xQ**, вектор **квадратів** похибки квантування (а не **середній квадрат** похибки квантування, як у випадку m-функції **quantiz**). [1,2]

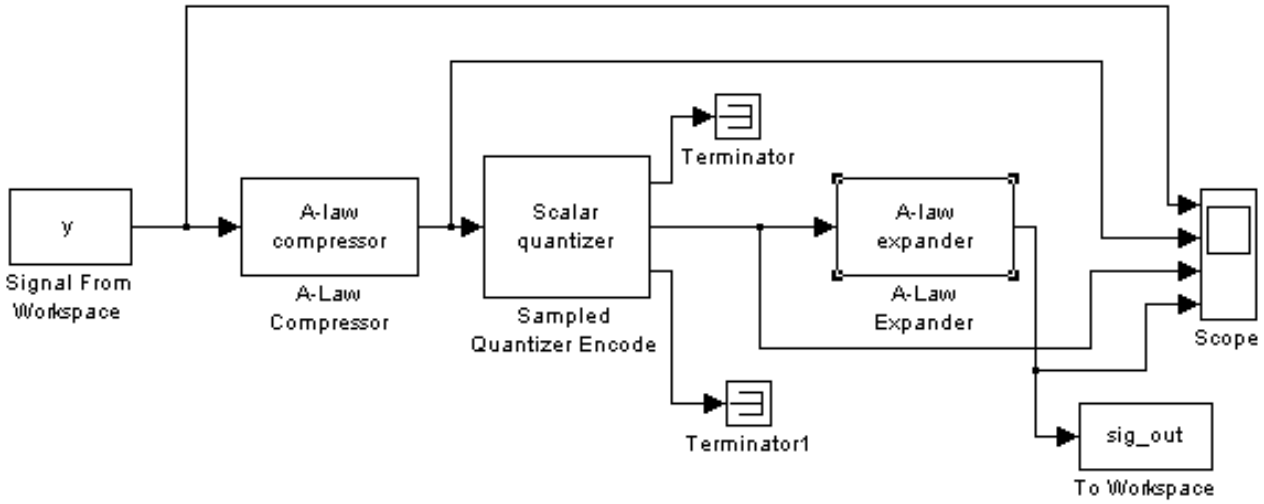


Рисунок 4.9 – Нелінійне квантування сигналу в середовищі Simulink

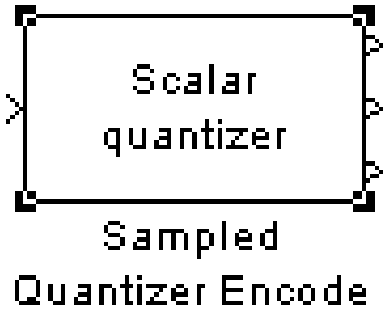


Рисунок 4.10 – Блок **Sampled quantizer encode**

Налаштування блоку **From Workspace** показані на рисунку 4.11, а блоку **Sampled quantizer encode** - на рисунку 4.12.

Відмітимо, що довжина вхідного вектору блоку **Sampled quantizer encode** береться не 20, а 19, що необхідно для нормальної роботи схеми (звісно, в кожному з варіантів лабораторної роботи довжина вхідного вектору може бути іншою).

Налаштування блоку **To Workspace** показані на рисунку 4.13.

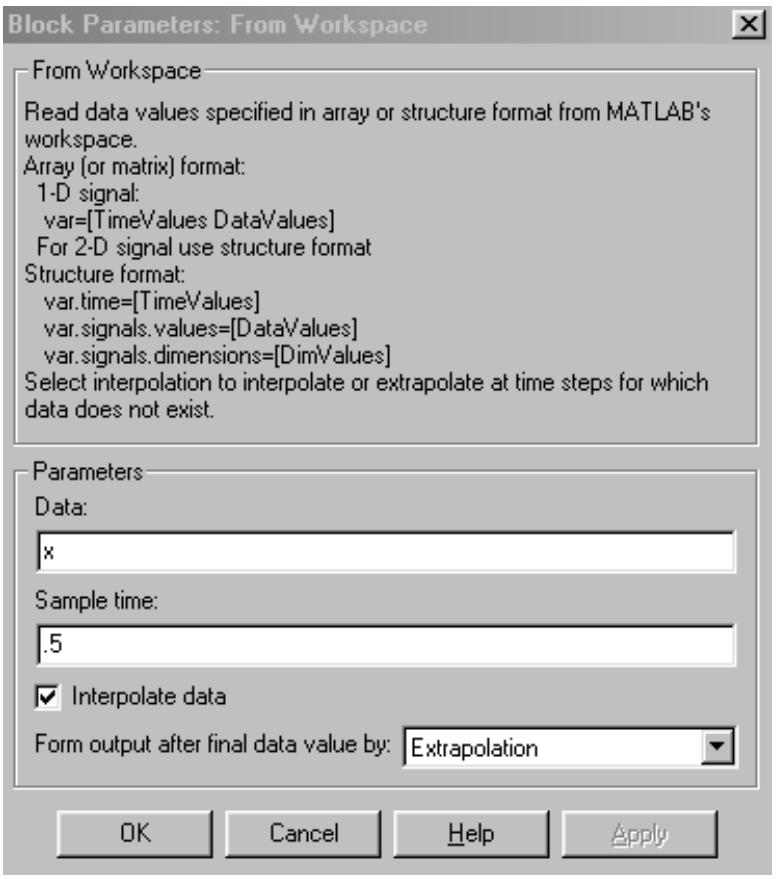


Рисунок 4.11 – Вікно-маска для налаштування блоку **From Workspace**

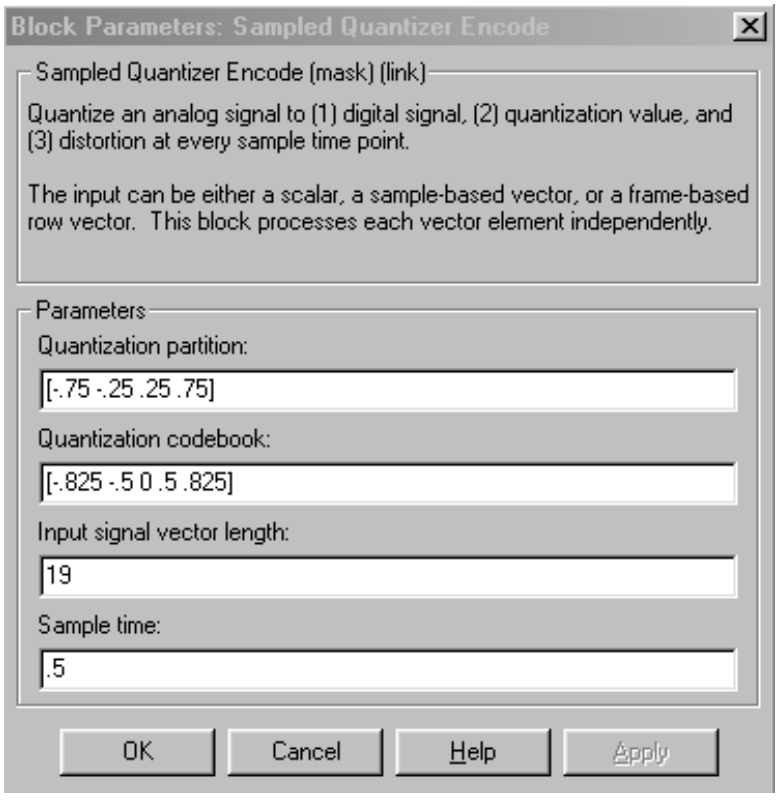


Рисунок 4.12 – Вікно-маска для налаштування блоку **Sampled quantizer encode**

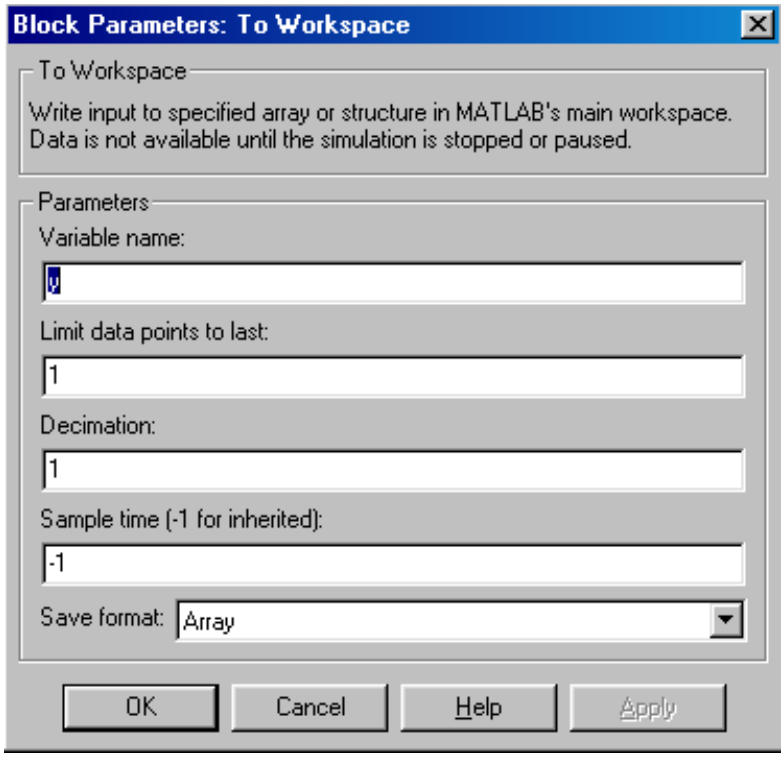


Рисунок 4.13 – Вікно-маска для налаштування блоку **To Workspace**

Зверніть увагу (рисунок 4.13): формат результату – масив (**array**), а не структура (**structure**). Дані в форматі масиву (**array**) зручніше аналізувати, оскільки не потрібно витрачати час на вилучення зі структури (**structure**) за допомогою спеціальних доволі громіздких команд.

Нелінійне квантування реального сигналу в середовищі Simulink при тих самих параметрах (4 біти, A=100) приводить до значно кращих результатів в області малих рівнів сигналу (рисунок 4.14).

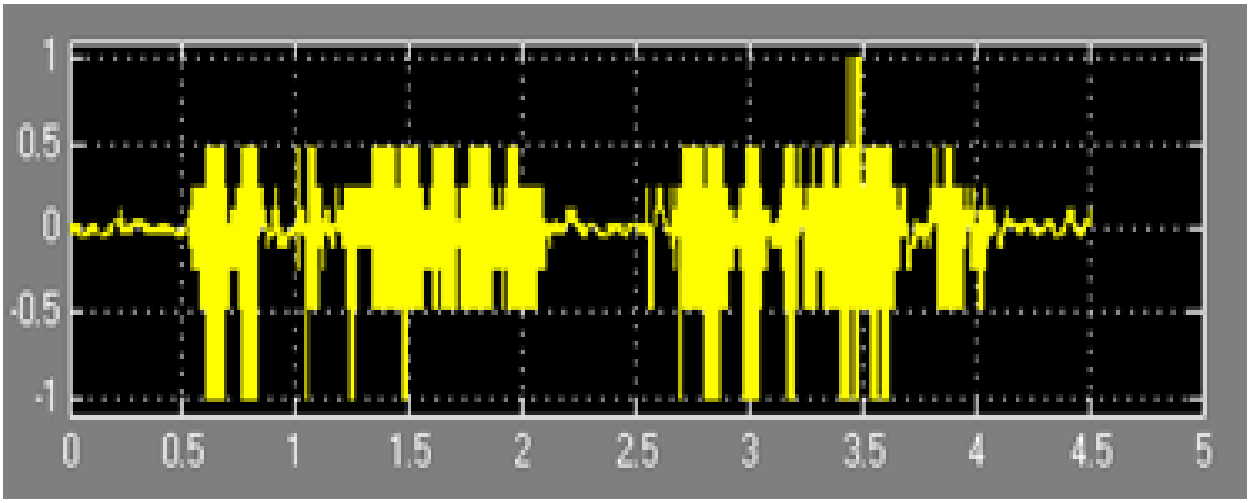


Рисунок 4.14 – Результат нелінійного квантування реального сигналу в середовищі Simulink

4.2 Підготовка до лабораторної роботи

4.2.1 Ознайомитися з короткими теоретичними відомостями і порядком роботи з середовищем MatLab і Simulink.

4.2.2 Написати програму у вигляді файлу-сценарію (m-файлу) для моделювання гармонічного сигналу, який задано функцією $x(t)$:

$$x(t)=X_m cos(2\pi f_0 t+\varphi)$$

Параметри для гармонічного сигналу обираються з таблиці 4.2, відповідно заданому варіанту n, де n – номер студента в списку групи (або в послідовно об’єднаному списку груп в порядку зростання їх номерів).

Тривалість T реалізації сигналу обирати рівною двом періодам сигналу.

Частоту дискретизації F_s задати у 20 разів вище частоти сигналу.

Таблиця 4.2 – Параметри гармонічного сигналу

№ варіанту, **n**	X_m, В	f_0, Гц	φ, рад	L	μ	A
1…5	n	n*300	0	8	n*40	n*20
6…10	n/2	n*200	π/4	7	n*16	n*8
11…15	n/3	n*100	π/2	6	n*10	n*6
16…20	n/4	n*200	π	5	n*6	n*4
21…25	n/6	n*250	π/3	4	n*3	n*2
26…30	n/7	n*50	π/5	9	n	n

4.2. 3 Написати програму у вигляді файлу-сценарію для моделювання нелінійного квантування сигналу *x(t)*, заданого за п.4.2.2, і побудови графіків: вхідного сигналу; квантованого сигналу; значень індексів; а також середнього квадрату похибки квантування.

Параметри команд **codebook** і **partition** задаються виходячи з заданої кількості L рівнів квантування (таблиця 4.2).

Нелінійне квантування здійснити для порівняння двома способами: за прийнятою у Північній Америці та Європі характеристикам пристрою стиснення. Значення параметрів L, μ і A обираються з таблиці 4.2, згідно заданому варіанту. При моделюванні значення амплітуд вхідного і вихідного сигналів пристрою стиснення прийняти однаковими: $Y_m = X_m$.

4.2.4 3 Написати програму у вигляді файлу-сценарію для моделювання нелінійного квантування реального звукового сигналу, що міститься в заданому викладачем файлі (наприклад, **vorona.mp3**). Здійснити зворотне нелінійне перетворення квантованого сигналу, тобто реалізувати процедуру компандування сигналу. Вивести на екран загальний результат нелінійного квантування реального сигналу.

Параметри команд **codebook** і **partition** задаються виходячи з кількості L_{new} рівнів квантування, яке визначається за формулою:

$$L_{new} = 2^L ,$$

де значення параметру L (для цього пункту дорівнює кількості біт) обираються з таблиці 4.2 відповідно заданому варіанту, але не перевищують значення розрядності (кількості біт) вхідного сигналу.

Нелінійне квантування здійснити для порівняння двома способами: за прийнятою у Північній Америці та Європі характеристикам пристрою стиснення. Значення параметрів L, μ і A обираються з таблиці 4.2, згідно заданому варіанту. Моделювання провести для нормованих значень сигналу.

4.2.5 Змоделювати в середовищі Simulink процедуру нелінійного квантування гармонічного сигналу, параметри якого обираються з таблиці 4.2, згідно заданому варіанту.

Нелінійне квантування здійснити для порівняння двома способами: за прийнятою у Північній Америці та Європі характеристикам пристрою стиснення. Значення параметрів μ і A обираються з таблиці 4.2, згідно заданому варіанту

На екран вивести: вектор-стовбець індексів (номерів) інтервалів **(ind)**, вектор-рядок квантованих значень сигналу **(xQ)**, вектор **квадратів** похибки квантування.

4.2.6 Змоделювати в середовищі Simulink процедуру нелінійного квантування реального звукового сигналу, що міститься в заданому викладачем файлі (наприклад, **vorona.mp3**). Здійснити зворотне нелінійне перетворення квантованого сигналу, тобто реалізувати процедуру компандування сигналу.

Нелінійне квантування здійснити для порівняння двома способами: за прийнятою у Північній Америці та Європі характеристикам пристрою

стиснення. Значення параметрів μ і A обираються з таблиці 4.2, згідно заданому варіанту.

Вивести на екран осцилоскопу вихідні сигнали усіх блоків і загальний результат нелінійного квантування реального сигналу.

Примітка. Моделі у середовищі Simulink повинні працювати автономно, незалежно від параметрів у **Workspace**, тобто при «очищеному» **Workspace**.

4.3 Порядок виконання лабораторної роботи

4.3.1 Увімкніть ПЕОМ і запустіть програму MatLab.

4.3.2 Наберіть програму нелінійного квантування гармонічного сигналу, написану за п. 4.2.3 і запустіть її на виконання. Отримані результати занесіть в протокол лабораторної роботи у вигляді скріншотів.

4.3.3 Наберіть програму нелінійного квантування реального звукового сигналу, написану за п. 4.2.4 і запустіть її на виконання. Отримані результати занесіть в протокол у вигляді скріншотів.

4.3.4 Запустіть модель, підготовлену за п. 4.2.5, яка реалізує процедуру нелінійного квантування гармонічного сигналу, на виконання. Отримані результати занесіть в протокол лабораторної роботи у вигляді скріншотів.

4.3.5 Запустіть модель, підготовлену за п. 4.2.6, яка реалізує процедуру нелінійного квантування реального звукового сигналу, на виконання. Отримані результати занесіть в протокол лабораторної роботи у вигляді скріншотів.

4.4 Зміст звіту з лабораторної роботи

Звіт з лабораторної роботи повинен містити:

4.4.1 При допуску до лабораторної роботи:

4.4.1.1 Ціль лабораторної роботи.

4.4.1.2 Короткі теоретичні відомості.

4.4.1.3 Написані за пунктами 4.2.2 - 4.2.4 **програми з коментарями** для середовища MatLab.

4.4.1.3 Блок-схеми моделей в середовищі Simulink за пунктами 4.2.5, 4.2.6 із зазначенням **значень параметрів,** що задаються для блоків.

4.4.2 При здачі лабораторної роботи до матеріалів за п. 4.4.1 повинні бути додані результати виконання роботи.

4.4.2.1 Результати виконання програм за пунктами 4.3.2, 4.3.3.

4.4.2.2 Результати роботи моделей в середовищі Simulink за пунктами 4.3.4, 4.3.5.

4.4.2.3 Висновки.

4.5 Контрольні питання

4.5.1 Як представляється квантований сигнал у функції часу?

4.5.2 Як здійснюється нелінійне квантування?

4.5.3 Що таке компандування сигналу?

4.5.4 Які переваги нелінійного квантування?

4.5.5 Які закони пристроїв стиснення використовуються і чим вони відрізняються?

Додаток А

Основні оператори мови MATLAB

А.1 Арифметичні та логічні оператори мови MATLAB

Мова MATLAB – це мова операторів. Оператори складаються із спеціальних символів, імен функцій та змінних, а також числових констант і можуть закінчуватися комою або крапкою з комою, які керують виведенням результату на екран. Імена змінних та функцій можуть бути складені з будь-яких символів англійського алфавіту та цифр. Ім'я обов'язково повинне починатися з літери.

У виразах можуть використовуватися як скаляри – константи та числа, так і масиви – одномірні (вектори) та двовимірні (матриці). Система MATLAB дозволяє працювати з константами, змінними масивами, що містять як числову, так і символьну інформацію. Для запису дійсних чисел використовується як звичайна десяткова форма з необов'язковою десятковою точкою для цілих чисел, так і показникова форма з мантисою та показником ступеня з основою десять.

Наприклад. Число 0.05 можна записати ще й у такому вигляді: 5E–2 (пробілів перед експоненційною частиною немає).

У системі MATLAB реалізовано два типи арифметичних операцій: операції над матрицями та над масивами (див. таблицю А.1). Операції над матрицями визначені відповідно до правил лінійної алгебри, а операції над масивами виконуються поелементно. Щоб розрізняти ці операції, операції над масивом передує крапка. Операції складання та віднімання над матрицями та масивами дають однакові результати.

Таблиця А.1

Операції над матрицями	**Операції над масивами**
A + B	A + B
A – B	A – B
A * B	A .* B
A / B	A ./ B
A \ B	A .\ B
A ^ B	A .^ B
A’	A.’

Опис особливостей виконання цих операцій наведено нижче.

Додавання: A + B.

Доданки повинні бути однакового розміру за винятком випадку, коли один з них скаляр.

Віднімання: A – B.

Обидва операнди мають бути однакового розміру, крім випадку, коли один з них скаляр.

Множення матриць: A * B.

Число стовпців першого співмножника має дорівнювати числу рядків другого. На скаляр множаться усі елементи співмножника.

Множення масивів: A. * B.

Поелементне перемноження двох масивів однакового розміру. На скаляр множаться усі елементи масиву.

Вирішення систем лінійних рівнянь виду A*X = B: A \ B.

Якщо A – матриця розміру $n_x n$ та B – матриця розміру $n_x k$, то рівняння вирішується методом Гаусса. Якщо A – матриця розміру $m_x n$ та B – матриця

розміру nxk, то система виявляється недовизначеною чи перевизначеною і вирішується на основі мінімізації другої норми нев'язок.

Ліве ділення масивів: A. \ B.

Результатом є матриця з елементами B(i,j)/A(i,j); матриці мають бути однакових розмірів.

Розв'язання систем лінійних рівнянь виду X*A = B: A/B.

Праве ділення масивів: A ./ B.

Результатом є матриця з елементами A(i,j)/A(i,j); матриці повинні бути однакових розмірів, за винятком випадку, коли один з них вироджується в скаляр.

Зведення матриці на ступінь p: A^p.

Якщо p - ціле додатне число, то ступінь матриці обчислюється шляхом перемноження її на себе; якщо p – ціле від'ємне число, те саме відноситься до зворотної матриці.

Ступінь масиву: A.^B.

Результатом є матриця з елементами A(i,j)^B(i,j); матриці мають бути однакових розмірів.

Транспонування матриці: A'.

Для дійсних матриць – результатом є транспонована матриця; для комплексних – транспонування доповнюється комплексним спряженням.

Транспонування масиву: A.'.

Для дійсних та комплексних масивів – рядки просто замінюються стовпцями.

Список логічних операторів наведено у таблиці А.2.

Таблиця А.2.

Операція	Опис	Приклади
<	Менше	A<B
?	Менше або дорівнює	A?B
>	Більше	A>B
?	Більше або дорівнює	A?B
==	Дорівнює	A==B
~=	Не дорівнює	A~=B
&	Логічне «І»	A&B
|	Логічне «АБО»	A | B
~	Логічне заперечення	~A
Xor	Исключающее «ИЛИ»	Xor(A,B)

А.2 Стандартні змінні та константи

У системі MATLAB є наступні константи та змінні, які ініціалізуються за умовчанням:

ANS – результат виконання останньої операції.

При використанні операторів з неявним присвоєнням система MATLAB автоматично створює цю змінну і надає їй значення результату.

Приклад: після введення виразу

Sin(pi/6)

MATLAB видасть результат:

ans = 0.5000000

I або J – уявна одиниця.

Цим константам спочатку присвоюється значення, що дорівнює $\sqrt{-1}$.

INF – нескінченність.

Це спеціальна змінна, що дозволяє фіксувати переповнення розрядної сітки.

Приклад: під час операції ділення на нуль10/0

ломп'ютер видасть повідомлення:

Warning: Divide by zero

ans = Inf

PI - Число p.

A.3 Стандартні змінні та константи

Синтаксис: **for v = <вираз-масив>**

<оператори>

end

Опис: на відміну від універсальних мов програмування, змінна циклу в мові MATLAB є масивом. Тому виконання циклу найбільш загальної формі полягає у тому, що змінній циклу присвоюються значення стовпців масиву і потім виконуються оператори, які мають залежати від змінної v.

Приклад: x = [ ];

For v=[0 2 3 1]

x=[2^v]

end

Ця програма формує вектор, елементи якого є ступенями $2^{v(i)}$.

Але можливе використання спрощеної конструкції виду m:n (змінна змінює своє значення від m до n з кроком 1) або m:<крок>:n.

Приклад: for I=1:n, x(i)=0, end

Синтаксис: **While <логічний вираз>**

<оператори>

end

Опис: Цей цикл виконується, до тих пір, поки вірний логічний вираз.

Приклад: n=1

While n<5.2, n=n+1; end

Синтаксис: **If < логічний вираз >**

<оператори>

else

<оператори>

end

Опис: Оператори умовного виразу виконуються лише під час виконання логічного виразу.

Приклад:

f or i=1:n

for j=1:n

if i==j; a(i,j)=5; else a(i,j)=0; end

end

end

Синтаксис: **BREAK.**

Опис: Ця функція перериває виконання циклів for та while. У разі вкладених циклів переривання можливе лише внутрішнього циклу. Використовується разом із оператором if.

Приклад:

While 1

n=input(‘Введіть n. Зупинка вводу n?0’)

if n<=0, break, end

r=rank(magic)

end

Синтаксис: **Y=ZEROS**

Y=ZEROS(m, n)

Y=ZEROS(size(A))

Опис: Формування масиву нулів. Перша функція формує масив нулів розміру nxn. Друга функція формує масив нулів розміру mxn. Третя функція формує масив нулів пропорційний масивом A.

Синтаксис: **Y=ONES**

Y=ONES(m, n)

Y=ONES(size(A))

Опис: Формування масиву одиниць. Перша функція формує масив одиниць розміру nxn. Друга функція формує масив одиниць розміру mxn. Третя функція формує масив одиниць пропорційний масивом A.

Синтаксис: : **Y=EYE**

Y=EYE(m, n)

Y=EYE(size(A))

Опис: Формування одиничної матриці. Перша функція формує поодиноку матрицю розміру nxn. Друга функція формує одиничну матрицю

розміру mxn. Третя функція формує одиничну матрицю пропорційну матриці A.

Синтаксис: : **Y=RAND**

Y=RAND(m, n)

Y=RAND(size(A))

Опис: Формування масиву елементів, розподілених за рівномірним законом.

Синтаксис: : Y=RANDN(n)

Y=RANDN(m, n)

Y=RANDN(size(A))

Опис: Формування масиву елементів, розподілених за нормальним законом.

A.4 Математичні функції

Y=**ABS**(x)

Опис: Для масиву дійсних чисел X ця функція повертає масив Y абсолютних значень елементів X.

Y=**SIGN**(x)

Опис: Для масиву дійсних чисел X ця функція повертає масив Y тих самих розмірів, в якому на місці додатного числа стоїть 1, на місці нульового– 0, а на місці від'ємного – (-1).

Y=**CEIL**(x)

Опис: Повертає значення, округлені до найближчого цілого ≥X.

Y=**FIX**(x)

Опис: Повертає значення з усіченням дробової частини числа.

Y=**FLOOR**(x)

Опис: Повертає значення, округлені до найближчого цілого ≤X.

Y=**ROUND**(x)

Опис: Повертає значення, округлені до найближчого цілого.

Y=**SQRT**(x)

Опис: Квадратний корінь.

Y=**EXP**(x)

Опис: Експоненціальна функція.

Y=**LOG**(x)

Опис: Функція натурального логарифму.

Y=**SORT**(x)

Опис: Сортування елементів масиву за зростанням.

Y=**MAX**(x)

Опис: Повертає максимальний елемент. У випадку двовимірного масиву – це вектор-рядок, що містить максимальні елементи кожного стовпця.

Y=**MIN**(x)

Опис: Повертає мінімальний елемент. У випадку двовимірного масиву – це вектор-рядок, що містить мінімальні елементи кожного стовпця.

А.5 Математичні функції

Синтаксис: **GRID**

GRID ON

GRID OFF

Опис: Нанесення сітки на графік. Команда grid on наносить координатну сітку на осі. Команда grid off видаляє координатну сітку. Команда grid виконує роль перемикача з однієї функції іншу.

Синтаксис: **HOLD ON**

HOLD OFF

HOLD

Опис: Керування режимом збереження поточного графічного вікна. Команда hold on включає режим збереження поточного графіка, так що наступні команди призведуть до додавання нових графіків у графічному вікні. Команда hold off вимикає режим збереження графіка. Команда hold реалізує перемикання від одного режиму до іншого.

Синтаксис: TITLE('<текст>')

Опис: Ця команда розміщує текст над графіком.

Синтаксис: **XLABEL('<текст>')**

YLABEL('<текст>')

ZLABEL('<текст>')

Опис: Ці команди розміщують текст вздовж осей X, Y, Z.

Синтаксис: **TEXT(x,y, '<текст>')**

TEXT(x, y, z, '<текст>')

Опис: Ця команда додає до поточного графіка вказаного тексту у заданій точці (x, y).

Синтаксис: **PLOT**

Опис: Будує графік елементів одномірного масиву y в залежності від номера елемента. Якщо у двовимірний масив, то будується графік для стовпців.

Синтаксис: **PLOT(x, y)**

Опис: Відповідає побудові звичайної функції, коли одновимірний масив x відповідає значенням аргументу, а одновимірний масив y – значенням функції.

Синтаксис: **PLOT(x, y, s)**

Опис: Ця команда дозволяє виділити графік функції, вказавши спосіб відображення точок, тип ліній та колір ліній за допомогою змінної s (таблиця А.3).

Таблиця А.3

Тип лінії	Тип точки	Колір
Неперервна -	Точка .	Жовтий y
Штрихова --	Плюс +	Фіолетовий m
Подвійний пунктир :	Зірочка *	Блакитний c
Штрих-пунктир -.	Кружечок o	Червоний r
	Хрестик ?	Зелений g
		Синій b
		Білий w
		Чорний k

Синтаксис: **PLOT(x1, y1, s1, x2, y2, s2…)**

Опис: Дозволяє об'єднати кілька функцій на одному графіку, визначивши для кожної з них свій спосіб відображення.

Синтаксис: **X = LOGSPSACE (d1, d2)**

Опис: Ця функція формує вектор-рядок, що містить 50 рівновіддалених в логарифмічному масштабі точок, які покривають діапазон від 10^{d1} до 10^{d2}.

Синтаксис: **LOGLOG(x, y)**

LOGLOG(x, y, s)

Опис: Ця команда рівносильна функції plot, за винятком того, що вона використовує по обох осях логарифмічний масштаб замість лінійного.

Приклад побудови графіка y=exp(x).

x=logspace(-1, 2);

loglog(x, exp(x))

Синтаксис: **POLAR(p, r)**

POLAR(p, r, s)

Опис: Ці команди реалізують побудову графіків у полярних координатах, що задаються кутом p і радіусом r.

Синтаксис: **PLOT3(x, y, z)**

PLOT3(x, y, z, s)

Опис: Ця команда будує тривимірний графік за одновимірними масивами x, y, z. Параметр s, як описано вище, дозволяє вказати спосіб відображення графіка.

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

Основна

1 Ушенко Ю.О., Гавриляк М.С. та ін, Основи та методи цифрової обробки сигналів: від теорії до практики: навч. посібник / уклад. : Ю.О. Ушенко, М.С. Гавриляк, М.В. Талах, В.В. Дворжак. – Чернівці : Чернівецький нац. ун-т ім. Ю. Федьковича, 2021. - 308 с.

2 Заболотній С. В. Цифрове оброблення сигналів: Посібник для студентів напряму підготовки 6.050901 "Радіотехніка" усіх форм навчання [Електронний ресурс] / Авт.-укл. С. В.Заболотній; За ред. проф. Ю. Г. Леги ; М-во освіти і науки України, Черкас. держ. технол. ун-т. – Черкаси: ЧДТУ, 2010. – 119 с.

3 Тотосько О.В., Стухляк П.Д. Цифрова обробка сигналів та зображень / Навчальний посібник. – Тернопіль: ТНТУ, 2016 – 132с.

4 Бабак В.П., Хандецький В.С., Шрюфер Е. Обробка сигналів. / Підручник для студентів технічних спеціальностей вузів. — К.: «Либідь», 1999 – 496 с.

Допоміжна

5 Richard G. Lyons. Understanding Digital Signal Processing, 3rd Edition. - Prentice Hall, 2011. – 954 p.

6 Колєнов С. О. Цифрова обробка інформації: Методичний посібник до лабораторного практикуму для студентів радіофізичного факультету / Колєнов С. О./ – Київ: Радіофізичний факультет Київського національного університету імені Тараса Шевченка, 2008. – 56 с.

7 Метрологія та вимірювальна техніка : Підручник — 2-е вид., доп. та переробл. /Є.С.Поліщук, М.М. Дорожовець, В.О. Яцук, В.М. Ванько, Т.Г. Бойко; 3а ред. Є.С. Поліщука; — Львів: Вид-во Львівської політехніки, 2012. — 544 с.