Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут»

М. В. КОРЖИК

МОДЕЛЮВАННЯ ОБ'ЄКТІВ ТА СИСТЕМ КЕРУВАННЯ ЗАСОБАМИ MATLAB

Затверджено Вченою радою НТУУ «КПІ» як
навчальний посібник для студентів, що навчаються за спеціальністю
«Автоматизація та комп'ютерно-інтегровані технології»

КИЇВ
НТУУ «КПІ»
2016

УДК 519.2:681.3

Гриф надано Вченою радою
Національного технічного університету
України «Київський політехнічний інститут»
протокол № 7 від 6 червня 2016 року.

Рецензенти:
А. П. Ладанюк, д-р техн. наук, проф.,
Національний університет харчових технологій
Б. Я. Корнієнко, д-р техн. наук, доцент,
Національний авіаційний університет

Відповідальний редактор
А. І. Жученко, д-р техн. наук, проф.

Коржик М. В.

Моделювання об’єктів та систем керування засобами MatLab: навч. посіб. для студ. вищ. навч. закл. / М. В. Коржик. – Київ : НТУУ “КПІ”, 2016. – 174 с. : іл.

Видання містить теоретичні відомості та опис основних засобів системи MatLab / Simulink для ідентифікації, аналізу, моделювання, імітації та синтезу систем керування з лінійними стаціонарними об’єктами. Всі теми проілюстровано великою кількістю прикладів, наведено необхідний довідковий матеріал.

Для студентів вищих навчальних закладів, які навчаються за спеціальністю “Автоматизація та комп'ютерно-інтегровані технології” та споріднених їй.

УДК 519.2:681.3

Вступ

Одним з найпоширеніших в світі програмних комплексів для інженерних та наукових розрахунків є система MatLab (The MathWorks, Inc. США).

Програмний комплекс містить декілька складових. Передусім це об'єктно-орієнтована мова програмування – інтерпретатор, що будується на арифметиці з рухомою крапкою та призначена для обробки масивів даних. Крім того це операційне середовище, в якому можуть виконуватись спеціальні програми – інструменти та команди найпоширеніших операційних систем (DOS та інші).

Ядро MatLab складають команди, функції та інструменти для обчислення математичних функцій, реалізації розкладень різних типів, розв'язання задач оптимізації, лінійних та нелінійних рівнянь, роботи з матрицями та масивами великих розмірностей, вирішення диференційних та різницевих рівнянь, роботи з алгебраїчними поліномами, візуалізації даних тощо.

Крім того MatLab дозволяє створювати власні функції та інструменти. Більш як за п'ятнадцять років існування системи для різних наукових та інженерних галузей було розроблено багато подібних засобів, з яких складаються додаткові бібліотеки (Toolboxes). Деякі з цих бібліотек розповсюджуються разом з системою MatLab. До них, наприклад, належать :

Communications Toolbox – містить ряд функцій та інструментів для прискорення конструювання, аналізу та моделювання сучасних систем зв'язку. Бібліотека може використовуватись в таких галузях зв'язку, як телефонія, телекомунікації, сателітний зв'язок та комп'ютерна периферія.

Control System Toolbox – базисна бібліотека MatLab для роботи з системами керування. Бібліотека містить функції та інструменти для моделювання, аналізу та синтезу лінійних стаціонарних систем керування.

Data Acquisition Toolbox – містить засоби для прямого доступу MatLab до великої кількості периферійних пристроїв, що дає змогу, наприклад, обробляти результати вимірювань в реальному часі.

Database Toolbox – дозволяє організувати обмін інформацією (імпорт та експорт) між MatLab та популярними реляційними базами даних.

Financial Toolbox – містить функції та інструменти для кількісного аналізу в економічних та фінансових розрахунках.

Frequency Domain Identification Toolbox – містить функції для моделювання лінійних дискретних та безперервних систем, що базуються на вимірюванні їх частотного відгуку.

Fuzzy Logic Toolbox – містить повний набір інструментів для проектування, моделювання та аналізу систем, побудованих на нечітких висловлюваннях.

Higher-Order Spectral Analysis Toolbox – містить інструменти для обробки сигналів методом спектрального аналізу. Бібліотека особливо корисна при аналізі сигналів, що виникають в нелінійних системах, або сигналів, які зіпсовані негауссовими шумами.

Image Processing Toolbox – містить інструменти для обробки зображень та побудови блок-схем алгоритмів.

LMI Control Toolbox – містить функції для ефективного вирішення лінійних матричних нерівностей (Linear Matrix Inequalities). LMI широко використовуються в багатьох дисциплінах, таких, як теорія керування,

ідентифікація, фільтрація сигналів, структурне проектування, оптимізація, теорія графів та лінійна алгебра.

Mapping Toolbox – містить функції та інструменти для комп'ютерного відображення географічних даних. Бібліотека може застосовуватись в картографії.

Model Predictive Control Toolbox – містить функції для аналізу та проектування систем керування з прогнозуванням (з еталонною моделлю). Бібліотека особлио корисна при роботі з багатовимірними системами де частина входів / виходів має обмеження. Найчастіше MPC-керування застосовують в хімічній інженерії.

Mu-Analysis and Synthesis Toolbox – містить інструменти для H_∞ оптимального керування та μ - аналізу і синтезу. Бібліотека використовується для проектування робастного керування в лінійних багатовимірних системах.

NAG Foundation Toolbox – містить більш як двісті функцій для розрахунків чисельними методами. Функції отримано адаптацією бібліотек fortran-програм Numerical Algorithm Group.

Neural Network Toolbox – містить функції для проектування та моделювання систем, які представлені нейронними мережами. Такий підхід використовується, коли формальний аналіз системи зробити дуже важко або взагалі неможливо. Бібліотека може застосовуватись для розпізнання образів або для ідентифікації та керування складними нелінійними системами.

Optimization Toolbox – містить засоби для оптимізації лінійних та нелінійних функцій в загальному вигляді з обмеженнями.

Partial Differential Equation Toolbox – містить функції та інструменти для попередньої обробки та розв’язання рівнянь в часткових похідних методом скінченних елементів. Бібліотека використовується для фізичних та інженерних розрахунків в таких областях, як теплопередача, будівельна механіка, електро та магнітостатика, дифузія тощо.

QFT Control Design Toolbox – містить засоби для проектування регуляторів для нестійких систем, які базуються на теорії кількісного зворотного зв'язку (Quantitive Feedback Theory) та аналізі частотних характеристик.

Robust Control Toolbox – містить інструменти для аналізу та синтезу робастних систем керування.

Signal Processing Toolbox – містить інструменти для цифрової обробки сигналів. Бібліотека може застосовуватись в телекомунікаціях, медицині, геофізиці, економетрії та для обробки аудіо та відеосигналів.

Spline Toolbox – містить засоби для конструювання та використання сплайнів (для кусково-поліноміальної апроксимації). Сплайнова апроксимація дозволяє уникнути небажаних ефектів, що виникають при використанні інших типів апроксимації.

Statistics Toolbox – містить функції та інструменти для статистичної обробки даних та моделювання (методом Монте Карло). Бібліотека використовується в статистиці, теорії імовірностей, імітаційному моделюванні тощо.

Symbolic Math Toolbox – містить засоби для розрахунків в символьному вигляді та для використання арифметики із змінною точністю.

System Identification Toolbox – містить функції та інструменти для спостереження за динамічними системами та їх ідентифікації, що будується тільки на обробці вхідних та вихідних сигналів системи.

Wavelet Toolbox – містить інструменти для дослідження локальних, багатовимірних нестаціонарних явищ на базі імпульсної декомпозиції сигналів та розкладення Фур'є. Бібліотека може використовуватись в телекомунікаціях, геофізиці, фінансових розрахунках, медицині, для обробки звукових сигналів тощо.

Simulink – особливий інструмент візуального програмування для моделювання динамічних систем. Моделювання відбувається шляхом

графічного конструювання структурних схем моделей. Для цього Simulink має набір елементарних блоків, які за функціональною ознакою організовані в підрозділи бібліотек.

Власні бібліотеки Simulink мають велику кількість блоків лінійних, нелінійних та дискретних елементів, блоків, що реалізують загальні математичні, трансцендентні, табличні та логічні функції, генераторів та приймачів сигналів, та блоків для створення підсистем і ефективної роботи з сигналами.

Крім того Simulink має засоби для створення користувачем власних блоків, що розміщуються в окремих бібліотеках. Зокрема багато бібліотек MatLab містять блоки для Simulink (наприклад Control System Toolbox). Інші блоки у вигляді додаткових бібліотек (Blocksets) розповсюджуються разом з системою Simulink. До них належать:

Simulink Extras – містить кілька підрозділів, в яких розташовані додаткові дискретні та лінійні блоки і блоки відображення, блоки тригерів та перетворення чисельних величин, та блоки для полегшення лінеаризації моделей.

DSP Blockset – містить багато підрозділів з блоками для цифрової обробки сигналів (Digital Signal Processing).

Fixed Point Blockset – містить блоки для побудови моделей дискретних систем з використанням арифметики з фіксованою крапкою.

Nonlinear Control Design Blockset – містить блоки для проектування робастних нелінійних систем керування в часовій області.

Power System Blockset – містить багато підрозділів з блоками для побудови електро-енергетичних систем.

Dials & Gauges Blockset – містить віртуальні інструменти для побудови інтерактивних моделей.

Stateflow – містить засоби для моделювання подій в термінах теорії скінченних автоматів.

Серед інших можливостей системи MatLab можна згадати комплекс засобів для прискорення розрахунків та генерації виконуваних кодів, організації інтерфейсу з програмами MS Word та MS Excel, підтримки роботи в Internet тощо.

Вказані властивості системи MatLab дозволяють вважати її найбільш придатною (серед інших універсальних математичних систем) для використання при вивченні таких курсів, як теорія автоматичного керування, моделювання та ідентифікація об’єктів та систем, проектування систем автоматизації, оптимізація, метрологія, теорія імовірностей та багато інших.

В цьому посібнику описано основні засоби системи MatLab / Simulink для ідентифікації, аналізу, моделювання, імітації та синтезу лінійних стаціонарних систем керування. Всі теми проілюстровано великою кількістю прикладів (текст виводу у вікно MatLab в прикладах надруковано `непропорційним` шрифтом). Наведено необхідний довідковий матеріал.

Посібник призначено для читачів, що володіють основами роботи з системою MatLab. Бажаючим детальніше ознайомитись з базовими можливостями системи можна звернутися до літератури [20 – 34].

Розділ 1. Моделі динамічних систем

В цьому розділі наведено способи зображення лінійних стаціонарних динамічних систем (linear time invariant, надалі lti-систем) та операції з ними в MatLab за допомогою бібліотеки Control System Toolbox (надалі CST), а також представлені засоби їх дослідження [23, 27].

Бібліотека CST підтримує новий об'єктний клас: lti-система, який включає дискретні або в безперервному часі, одновимірні (з одним входом та одним виходом, надалі SISO) та багатовимірні (з кількома входами та/або виходами, надалі MIMO) системи. Таким чином моделі динамічних систем можна зобразити в MatLab як змінні класу lti-система та оперувати ними за допомогою команд та функцій бібліотеки CST, повний список яких наведено в додатку 1.2.

1.1. Загальні положення

Математичні моделі описують співвідношення між вхідними та вихідними сигналами системи. Виходи моделі частково визначені її входами. Як правило, на виходи системи впливає більше сигналів ніж вимірених входів. Такі "невимірені входи" називаються сигналами збурення або шумом. Якщо позначити входи, виходи та збурення як u, y та e, відповідно, то їх співвідношення можна зобразити так (див. рис. 1.1.).

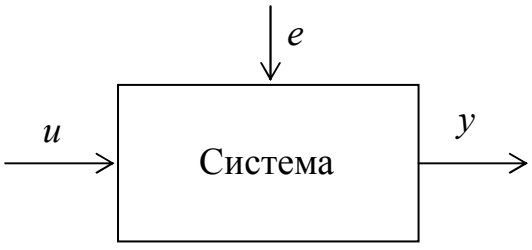

Рис. 1.1.

Математичну модель подібної lti-системи в загальному вигляді можна представити таким чином:

$$y = G \cdot u + H \cdot e, \tag{1.1}$$

де G та H – це оператори перетворення відповідних вхідних сигналів у вихідний.

1.1.1. Моделі в безперервному часі

Динаміка lti-систем з одним входом та одним виходом (SISO) в безперервному часі описується лінійними диференційними рівняннями виду:

$$\begin{aligned} a_0 \frac{d^{Na} y(t)}{dt^{Na}} + \ldots + a_{Na-1} \frac{dy(t)}{dt} + a_{Na} y(t) = \\ = b_0 \frac{d^{Nb} u(t)}{dt^{Nb}} + \ldots + b_{Nb-1} \frac{du(t)}{dt} + b_{Nb} u(t), \end{aligned} \tag{1.2}$$

де a_i та b_i – коефіцієнти диференційного рівняння, Na та Nb – порядки диференційного рівняння.

Рівняння (1.2) можна переписати в компактній формі:

$$\begin{aligned} a_0 p^{Na} y(t) + \ldots + a_{Na-1} p y(t) + a_{Na} y(t) = \\ = b_0 p^{Nb} u(t) + \ldots + b_{Nb-1} p u(t) + b_{Nb} u(t), \end{aligned} \tag{1.3}$$

де p – оператор диференцювання:

$$\frac{d^i}{dt^i} = p^i. \tag{1.4}$$

Тоді оператор перетворення "вхід-вихід" з (1.1) матиме вид:

$$G(p) = \frac{B(p)}{A(p)}, \tag{1.5}$$

де $A(p) = a_0 p^{Na} + \ldots + a_{Na-2} p^2 + a_{Na-1} p + a_{Na}$;

$B(p) = b_0 p^{Nb} + \ldots + b_{Nb-2} p^2 + b_{Nb-1} p + b_{Nb}$.

Імпульсна перехідна (або вагова) функція $g(t)$ визначається як відгук системи в довільний момент часу на імпульсне збурення, що виникло на вході системи за τ одиниць часу до цього моменту [2].

Динамічні властивості lti-системи можна описати за допомогою вагової функції $g(t)$ (див. п. 1.7.2.). Для довільного вхідного сигналу $u(t)$ вихід системи $y(t)$ описується інтегралом згортки:

$$y(t) = \int_{-\infty}^{\infty} g(\tau)u(t-\tau)\, d\tau. \qquad (1.6)$$

Для фізичної реалізовності системи необхідно, щоб $g(\tau) = 0$ при $\tau < 0$. Отже для реальних систем нижня межа інтегрування в (1.6) фактично дорівнює 0.

На практиці властивості динамічних ситем прийнято описувати, як правило, не самою ваговою функцією $g(t)$, а деяким її лінійним перетворенням, вид якого залежить від конкретної задачі.

В часовій області лінійну стаціонарну систему можна описати передатною функцією $G(s)$, яка визначається як перетворення Лапласа функції $g(t)$:

$$G(s) = \int_{0}^{\infty} g(\tau)e^{-s\tau}\, d\tau, \qquad (1.7)$$

де $s = \alpha + j\cdot\beta$ – комплексна змінна.

Для стаціонарних систем передатну функцію в формі зображення Лапласа можна отримати з оператора перетворення (див. (1.1)), якщо в останньому зробити підстановку $p = s$. (Диференцюванню оригіналу при нульових початкових умовах відповідає множення зображення на комплексне число s.)

Якщо lti-система є стійкою, то її динамічні властивості можна також описати через її частотну характеристику $G(j\omega)$, яка визначається як перетворення Фур'є функції $g(t)$:

$$G(j\omega) = \int_{-\infty}^{\infty} g(\tau)e^{-j\omega\tau}\, d\tau, \qquad (1.8)$$

де $\omega = 2\pi f$ – кутова частота, рад/с; $f = 1/T_f$ – циклічна частота, Гц; T_f – період коливань. Фактично частотна характеристика – це частковий випадок передатної функції, коли $s = 0 + j\cdot\omega$. Тобто для фізично реалізовних стійких lti-систем частотна характеристика містить ту ж інформацію, що і передатна функція і може бути отримана простою підстановкою $s = j\omega$.

1.1.2. Дискретні моделі

Якщо система є дискретною, або при наявності дискретної інформації про систему в безперервному часі, модель доцільно розглядати в дискретному часі: $t = kT$, $k = \overline{1, \infty}$, де T – період дискретизації [1].

Динаміка SISO lti-систем в дискретному часі описується різницевими рівняннями виду:

$$\begin{aligned} a_{Na}y(k-Na)+...+a_1y(k-1)+a_0y(k)= \\ =b_{Nb}u(k-Nk-Nb)+...+b_1u(k-Nk-1), \end{aligned} \tag{1.9}$$

де Nk – чисте запізнювання системи; Na – кількість полюсів; Nb – кількість нулів.

Введемо оператор зсуву назад по часу q^i, який визначається як:

$$q^i\, y(k) = y(k–i). \tag{1.10}$$

Перепишемо (1.9) в компактній формі:

$$\begin{aligned} a_{N_a}q^{Na}y(k)+...+a_1qy(k)+a_0y(k)= \\ =q^{Nk}(b_{Nb}q^{Nb}u(k)+...+b_1qu(k)), \end{aligned} \tag{1.11}$$

Тоді оператор перетворення "вхід-вихід" матиме вид:

$$G(q) = q^{Nk}\frac{B(q)}{A(q)}, \tag{1.12}$$

де $A(q) = a_0 + a_1q + a_2q^2 + ... + a_{Na}q^{Na}$;

$B(q) = b_1q + b_2q^2 + ... + b_{Nb}q^{Nb}$.

Дискретна вагова функція $g(k)$ визначається як відгук системи на одиничний імпульс. Вихідний сигнал системи можна визначити як:

$$y(k) = \sum_{i=1}^{\infty} g(i)u(k-i). \tag{1.13}$$

Тоді передатна функція дискретної системи визначається як z-перетворення вагової функції:

$$G(z) = \sum_{i=1}^{\infty} g(i)z^{-i}, \tag{1.14}$$

де $z = e^{T\cdot s}$ – комплексна змінна.

Якщо вагова функція системи задана дискретним набором ординат $g(k)$, то частотну характеристику системи можна отримати підстановкою $z = e^{j\omega T}$:

$$G(j\omega) = \sum_{i=1}^{\infty} g(i) e^{-j\omega i} . \qquad (1.15)$$

1.2. Зображення моделей

1.2.1. Моделі в безперервному часі

Передатну функцію SISO системи (з одним входом та одним виходом) зручно записувати в поліноміальній формі:

$$G(s) = \frac{B(s)}{A(s)}, \qquad (1.16)$$

де $A(s) = a_0 s^{Na} + ... + a_{Na-1} s + a_{Na}$;

$B(s) = b_0 s^{Nb} + ... + b_{Nb-1} s + b_{Nb}$.

Корені полінома в чисельнику (1.16) називаються нулями, а корені полінома в знаменнику – полюсами системи. Така форма представлення моделі в CST називається tf-формою.

Для зображення lti-системи в tf-формі використовується функція tf.

Приклад 1.1.

Зобразити систему з передатною функцією:

$W_1(s) = \frac{1.2(1-4s)}{(1+4s)(1+8s)} = \frac{-4.8s+1.2}{32s^2+12s+1}$. Для такої системи вектори коефіцієнтів поліномів $B(s)$ та $A(s)$ (див. (1.16)) мають вигляд:

$[b_0 \quad b_1] = [-4.8 \quad 1.2]$; $[a_0 \quad a_1 \quad a_2] = [32 \quad 12 \quad 1]$.

```
» W1=tf([-4.8 1.2],[32 12 1],'var','p')
Transfer function:
  -4.8 p + 1.2
----------------
32 p^2 + 12 p + 1
```

Теж саме можна зробити в інщий спосіб:

```
» s=tf('s');
» W1=(-4.8*s+1.2)/(32*s^2+12*s+1)
Transfer function:
  -4.8 s + 1.2
-----------------
32 s^2 + 12 s + 1
```

Кінець приклада.

Добре відомо [10], що за відсутності комплексних полюсів та нулів (1.16) можна розкласти на елементарні дроби (розкладення Хевісайда):

$$G(s) = \frac{k_1}{s - s_1} + \frac{k_2}{s - s_2} + ... + \frac{k_{Na}}{s - s_{Na}}, \qquad (1.17)$$

де $s_1, s_2, ..., s_{Na}$ – полюси, а $k_1, k_2, \ldots, k_{Na}$ – лишки $G(s)$. Іншу форму запису можна отримати факторизацією чисельника та знаменника передатної функції:

$$G(s) = \frac{K(s - c_1)(s - c_2)...(s - c_{Nb})}{(s - s_1)(s - s_2)...(s - s_{Na})}, \qquad (1.18)$$

де $c_1, c_2, \ldots, c_{Nb}$ – нулі $G(s)$, а K – узагальнений коефіцієнт передачі.

Представлення передатної функції через її нулі, полюси та коефіцієнти передачі в CST називається zpk-формою. Для зображення lti-системи в zpk-формі використовується функція zpk.

Приклад 1.2.

Зобразити систему з передатною функцією:

$W_2(s) = \dfrac{0.3(s + 0.4)(s + 0.6)}{(s + 0.6)(s + 0.7)(s + 0.3)}$. Для такої системи вектори нулів та полюсів (див. (1.18)) мають вид: $\mathbf{c} = \begin{bmatrix} -0.4 & -0.6 \end{bmatrix}$; $\mathbf{s} = \begin{bmatrix} -0.6 & -0.7 & -0.3 \end{bmatrix}$.

Узагальнений коефіцієнт передачі системи $K = 0.3$.

```
» W2=zpk([-0.4 -0.6],[-0.6 -0.7 -0.3],0.3)
 Zero/pole/gain:
  0.3 (s+0.4) (s+0.6)
----------------------
(s+0.6) (s+0.7) (s+0.3)
```

Кінець приклада.

Якщо $G(s)$ має комплексно-спряжені полюси та/або нулі (наприклад $s_i = \alpha + j\cdot\beta$ та $s_{i+1} = \alpha - j\cdot\beta$), то наведене розкладення повинно мати члени другого порядку.

Приклад 1.3.

Зобразити систему з передатною функцією: $W_3(s) = \dfrac{0.22(s+0.4)}{(s+s_1)(s+s_2)} = \dfrac{0.22(s+0.4)}{s^2+0.24s+0.072}$. Така системи має комплексно-спряжені полюси: $\mathbf{s} = \begin{bmatrix} -0.12 + j0.24 & -0.12 - j0.24 \end{bmatrix}$.

```
» W3=zpk(-0.4,[-0.12+j*0.24 -0.12-j*0.24],0.22)
 Zero/pole/gain:
    0.22 (s+0.4)
----------------------
(s^2 + 0.24 s + 0.072)
```

Або в інщий спосіб:

```
» s=zpk('s');          % В такій формі можна визначати передатні
                       % функції відносно 's' або 'z'
» W3=0.22*(s+0.4)/(s^2+0.24*s+0.072)
 Zero/pole/gain:
    0.22 (s+0.4)
----------------------
(s^2 + 0.24 s + 0.072)
```

Кінець приклада.

Отримати мінімальну реалізацію моделі в будь-якій формі можна за допомогою функції minreal. Ця функція скорочує однакові нулі та полюси, а також нормалізує поліноми чисельника та знаменники відносно старшого степеня.

Приклад 1.4.

Отримати мінімальну реалізацію передатної функції $W_2(s)$ (див. приклад 1.2.).

```
» W2
 Zero/pole/gain:
  0.3 (s+0.4) (s+0.6)
----------------------
(s+0.6) (s+0.7) (s+0.3)
» W2=minreal(W2)
Zero/pole/gain:
  0.3 (s+0.4)
--------------
(s+0.7) (s+0.3)
```

Отримати мінімальну реалізацію передатної функції $W_1(p)$ (див. приклад 1.1.).

```
» W1
Transfer function:
  -4.8 p + 1.2
-----------------
32 p^2 + 12 p + 1
» W1=minreal(W1)
Transfer function:
   -0.15 p + 0.0375
-----------------------
p^2 + 0.375 p + 0.03125
```

Кінець приклада.

1.2.2. Дискретні моделі

В поліноміальній формі дискретна передатна функція SISO-системи матиме вигляд:

$$G(z) = \frac{B(z)}{A(z)}, \tag{1.19}$$

де $A(z) = a_0 z^{Na} + ... + a_{Na-1} z + a_{Na}$;

$B(z) = b_0 z^{Nb} + ... + b_{Nb-1} z + b_{Nb}$.

Приклад 1.5.

Зобразити дискретну систему з передатною функцією:

$G_1(z) = \dfrac{-0.11z + 0.14}{z^2 - 1.66z + 0.687}$ та періодом дискретизації $T = 1$. Для такої системи вектори коефіцієнтів поліномів $B(z)$ та $A(z)$ (див. (1.19)) мають вигляд:

$[b_0 \quad b_1] = [-0.11 \quad 0.14]$; $[a_0 \quad a_1 \quad a_2] = [1 \quad -1.66 \quad 0.687]$.

```
» G1=tf([-0.11 0.14],[1 -1.66 0.687],1)
Transfer function:
   -0.11 z + 0.14
-------------------
z^2 - 1.66 z + 0.687
Sampling time: 1
```

Кінець приклада.

Якщо ексцес системи $Nk = Na - Nb$, то через зворотні поліноми можна записати:

$$G(z^{-1}) = z^{-Nk}\frac{B(z^{-1})}{A(z^{-1})}, \tag{1.20}$$

де $A(z^{-1}) = a_0 + a_1 z^{-1} + ... + a_{Na} z^{-Na}$;

$B(z^{-1}) = b_0 + b_1 z^{-1} + ... + b_{Nb} z^{-Nb}$.

В загальному випадку для фізично реалізовних дискретних систем керування $b_0 = 0$, тобто $u(k)$ не впливає на $y(k)$. Тоді для lti-систем можна вважати $q = z^{-1}$ (див. (1.12)).

Приклад 1.6.

Ексцес системи з приклада 1.5. $Nk = 1$. Тоді передатна функція цієї системи матиме вигляд: $G_1(z^{-1}) = \dfrac{-0.11z^{-1} + 0.14z^{-2}}{1 - 1.66z^{-1} + 0.687z^{-2}}$. Відповідні вектори-рядки коефіцієнтів поліномів $B(z^{-1})$ та $A(z^{-1})$ (див. (1.20)) мають вигляд: $[b_0 \quad b_1 \quad b_2] = [0 \quad -0.11 \quad 0.14]$; $[a_0 \quad a_1 \quad a_2] = [1 \quad -1.66 \quad 0.687]$.

```
» G1=tf([0 -0.11 0.14],[1 -1.66 0.687],1,'var','z^-1')
Transfer function:
  -0.11 z^-1 + 0.14 z^-2
-------------------------
1 - 1.66 z^-1 + 0.687 z^-2
Sampling time: 1
```

Кінець приклада.

Дискретну передатну функцію можна також виразити через її нулі, полюси та коефіцієнт передачі:

$$G(z) = \frac{K(z-c_1)(z-c_2)...(z-c_{Nb})}{(z-s_1)(z-s_2)...(z-s_{Na})}, \tag{1.21}$$

або

$$G(z^{-1}) = z^{-Nk}\frac{K(1-c_1z^{-1})(1-c_2z^{-1})...(1-c_{Nb}z^{-1})}{(1-s_1z^{-1})(1-s_2z^{-1})...(1-s_{Na}z^{-1})}, \tag{1.22}$$

Якщо $G(z)$ має комплексно-спряжені полюси та/або нулі, то наведені розкладення повинні мати члени другого порядку.

Приклад 1.7.

Зобразити систему з передатною функцією:

$G_2(z) = \dfrac{-0.15z(z-1.25)}{(z-0.87)(z-0.56)(z-0.73)}$ та періодом дискретизації $T = 1$. Для такої системи вектори-рядки нулів та полюсів (див. (1.21)) мають вигляд: $\mathbf{c} = [0 \quad 1.25]$; $\mathbf{s} = [0.87 \quad 0.56 \quad 0.73]$. Узагальнений коефіцієнт передачі системи $K = -0.15$.

```
» G2=zpk([0 1.25],[0.87 0.56 0.73],-0.15,1)
 Zero/pole/gain:
     -0.15 z (z-1.25)
-------------------------
(z-0.87) (z-0.73) (z-0.56)

Sampling time: 1
```

Кінець приклада.

1.2.3. Багатовимірні системи

Динамічна система, що має декілька входів та/або виходів (MIMO-система) може описуватись, як лінійне відображення за допомогою передатної матриці:

$$\mathbf{y} = \mathbf{G} \cdot \mathbf{u}, \tag{1.23}$$

де **u** – вектор вхідних сигналів розміром *Nu;* **y** – вектор вихідних сигналів розміром *Ny;* **G** – передатна матриця розміром *Ny*×*Nu; Ny*, *Nu* – кількість виходів та входів системи відповідно. Таким чином кожен елемент матриці **G** є передатною функцією системи по відповідному каналу "вхід – вихід".

З точки зору MatLab передатна функція багатовимірної системи є масивом, що складається з передатних функцій одновимірних систем.

Приклад 1.8.

Зобразити MIMO систему з двома входами та двома виходами. Передатні функції кожного з каналів керування такі:

$$W_{11}(p) = \frac{\Delta y_1(t)}{\Delta u_1(t)} = \frac{p+2}{p^2+2p+1}; \quad W_{12}(p) = \frac{\Delta y_1(t)}{\Delta u_2(t)} = 3;$$

$$W_{21}(p) = \frac{\Delta y_2(t)}{\Delta u_1(t)} = \frac{2}{p+2}; \qquad W_{22}(p) = \frac{\Delta y_2(t)}{\Delta u_2(t)} = 4.$$

Тоді систему в цілому можна описати як:

$$\begin{bmatrix} \Delta y_1(t) \\ \Delta y_2(t) \end{bmatrix} = \begin{bmatrix} W_{11}(p) & W_{12}(p) \\ W_{21}(p) & W_{22}(p) \end{bmatrix} \cdot \begin{bmatrix} \Delta u_1(t) \\ \Delta u_2(t) \end{bmatrix}.$$

```
» Wm=tf({[1 2],3;2,4},{[1 2 1],1;[1 2],1},'var','p')
Transfer function from input 1 to output...
         p + 2
#1:  -------------
     p^2 + 2 p + 1
       2
#2:  -----
     p + 2
Transfer function from input 2 to output...
#1:  3
```

```
#2:  4
```

Передатну функцію кожного з каналів керування можна розглядати окремо. Наприклад, передатна функція по каналу перший вхід – другий вихід ($\Delta y_2(t) = W_{21}(p) \cdot \Delta u_1(t)$) визначається так:

```
» Wm(2,1)
Transfer function:
  2
-----
p + 2
```

Кінець приклада.

Інший спосіб задання багатовимірних систем див. в п. 1.5.5. та прикладі 1.18.

1.2.4. Моделі в просторі станів

В просторі станів моделі описуються системами диференційних (або різницевих для дискретних систем) рівнянь першого порядку, які називаються рівняннями стану [6].

Систему в безперервному часі в просторі станів можна представити так:

$$\begin{aligned} \frac{d\mathbf{x}(t)}{dt} &= \mathbf{A} \cdot \mathbf{x}(t) + \mathbf{B} \cdot \mathbf{u}(t); \\ \mathbf{y}(t) &= \mathbf{C} \cdot \mathbf{x}(t) + \mathbf{D} \cdot \mathbf{u}(t), \end{aligned} \tag{1.24}$$

де **A**, **B**, **C** та **D** – матричні оператори: **A** – матриця стану розміром *Nx*×*Nx;* **B** – матриця керування розміром *Nx*×*Nu;* **C** – матриця виходу розміром *Ny*×*Nx;* **D** – матриця розміром *Ny*×*Nu;* **x**, **u** та **y** – вектори сигналів системи: **x** – вектор стану розміром *Nx;* **y** – вектор вимірювань розміром *Ny;* **u** – вектор керування розміром *Nu; Nx*, *Ny*, *Nu* – кількість станів, виходів та входів системи відповідно. Таке представлення моделей в CST називається ss-формою.

Для зображення lti-системи в ss-формі використовується функція `ss`.

Приклад 1.9.

MIMO систему з приклада 1.8. в просторі станів можна описати за допомогою матриць (див. (1.24)):

$$\mathbf{A}=\begin{bmatrix}-2 & -0.5 & 0\\ 2 & 0 & 0\\ 0 & 0 & -2\end{bmatrix};\quad \mathbf{B}=\begin{bmatrix}1 & 0\\ 0 & 0\\ 1 & 0\end{bmatrix};$$

$$\mathbf{C}=\begin{bmatrix}1 & 1 & 0\\ 0 & 0 & 2\end{bmatrix};\qquad \mathbf{D}=\begin{bmatrix}0 & 3\\ 0 & 4\end{bmatrix}.$$

```
» A=[-2,-0.5,0;2,0,0;0,0,-2];
» B=[1,0;0,0;1,0];
» C=[1,1,0;0,0,2];
» D=[0,3;0,4];
» W=ss(A,B,C,D)
a =
                    x1          x2          x3
          x1        -2        -0.5           0
          x2         2           0           0
          x3         0           0          -2
b =
                    u1          u2
          x1         1           0
          x2         0           0
          x3         1           0
c =
                    x1          x2          x3
          y1         1           1           0
          y2         0           0           2
d =
                    u1          u2
          y1         0           3
          y2         0           4
Continuous-time model.
```

Кінець приклада.

Крім того CST підтримує, так звану, дескрипторну форму (неявну форму Коші) моделі в просторі станів. В цьому випадку перше рівняння з (1.24) має вигляд:

$$\mathbf{E} \cdot \frac{d\mathbf{x}(t)}{dt} = \mathbf{A} \cdot \mathbf{x}(t) + \mathbf{B} \cdot \mathbf{u}(t), \tag{1.25}$$

де **E** – невироджена матриця розміром *Nx*×*Nx*, тобто модель має еквівалентну явну форму:

$$\begin{aligned} &\frac{d\mathbf{x}(t)}{dt} = (\mathbf{E}^{-1}\mathbf{A})\mathbf{x}(t) + (\mathbf{E}^{-1}\mathbf{B})\mathbf{u}(t); \\ &\mathbf{y}(t) = \mathbf{C} \cdot \mathbf{x}(t) + \mathbf{D} \cdot \mathbf{u}(t). \end{aligned} \tag{1.26}$$

Дескрипторну форму доцільно застосовувати, коли **E** – погано обумовлена відносно операції інверсії.

Для зображення ss-системи в неявній формі Коші використовується функція dss.

Дискретні системи можна описати в подібний спосіб:

$$\begin{aligned} &\mathbf{x}(k+1) = \mathbf{A_d} \cdot \mathbf{x}(k) + \mathbf{B_d} \cdot \mathbf{u}(k); \\ &\mathbf{y}(k) = \mathbf{C} \cdot \mathbf{x}(k) + \mathbf{D} \cdot \mathbf{u}(k), \end{aligned} \tag{1.27}$$

Зображують дискретні системи за допомогою функцій ss та dss з вказуванням періода дискретизації.

Зв'язок між дискретними моделями та моделями в безперервному часі можна встановити, прийнявши $kT \le t < (k+1)T$

$$\mathbf{A_d} = e^{\mathbf{A} \cdot T}; \quad \mathbf{B_d} = \int_0^T e^{\mathbf{A} \cdot \tau} \mathbf{B}\, d\tau, \tag{1.28}$$

де *T* – період дискретизації.

1.2.5. Моделі в частотній області

В CST лінійну стаціонарну модель можна задати, як послідовність зафіксованих комплексних частотних відгуків системи для заданої кількості частот (frd-форма). Такі моделі не мають параметричного представлення та їх аналіз обмежений частотними методами.

Для зображення моделі в frd-формі використовується функція frd.

1.3. Властивості lti-об’єктів

Як і будь-який інший об’єктний клас MatLab, lti має властивості (properties), які, з точки зору структури даних, можна вважати полями (fields) змінних, що належать класу lti-модель. Одержати загальну інформацію про властивості lti-об’єктів можна за допомогою команди ltiprops. Список властивостей моделі *sys* конкретної форми можна одержати також за допомогою команди set(*sys*). Повний список властивостей наведено в додатку 2.

Значення кожної окремої властивості *PropertyValue* системи *sys* можна отримати за допомогою команди get такого формату:

PropertyValue = get(*sys*, *PropertyName*),

де *PropertyName* – повна назва властивості або її скорочення з кількістю символьв, достатньою для однозначної ідентифікації.

Задати значення кожної окремої властивості можна різними способами:

1) При створенні lti-системи (див приклади 1.6. та 1.8.).

2) За допомогою команди set такого формату:

set(*sys*, *PropertyName*, *PropertyValue*, …).

3) Прямим привласненням значення відповідному полю.

Приклад 1.10.

Розглянемо змінну G_1 з приклада 1.5.

```
» G1
Transfer function:
   -0.11 z + 0.14
--------------------
z^2 - 1.66 z + 0.687
Sampling time: 1

» set(G1)                % Список властивостей моделі
```

```
            num: Ny-by-Nu cell of row vectors Nu = no.of inputs)
            den: Ny-by-Nu cell of row vectors Ny = no.of outputs)
       Variable: [ 's' | 'p' | 'z' | 'z^-1' | 'q' ]
             Ts: scalar
     InputDelay: Nu-by-1 vector
    OutputDelay: Ny-by-1 vector
  ioDelayMatrix: Ny-by-Nu array (I/O delays)
      InputName: Nu-by-1 cell array of strings
     OutputName: Ny-by-1 cell array of strings
     InputGroup: M-by-2 cell array if M input groups
    OutputGroup: P-by-2 cell array if P output groups
          Notes: array or cell array of strings
       UserData: arbitrary
Type "ltiprops tf" for more details.
» get(G1,'Ts')
ans = 1
```

Слід відзначити, що примусова зміна властивості *Ts* не призводить до зміни періоду дискретизації системи (див. п. 1.6.3.).

```
» get(G1,'Variable')     % Змінна передатної функції
ans = z
» set(G1,'Var','q')      % Змінемо цю властивість
» get(G1,'InputName')    % Назва входу системи
ans = {''}
» get(G1,'OutputName')   % Назва виходу системи
ans = {''}
% Назви входу та виходу не визначені. Задамо їх:
» G1.InputName='Витрата');
» G1.OutputName='Тиск')
Transfer function from input "Витрата" to output "Тиск":
  -0.11 q + 0.14 q^2
----------------------
1 - 1.66 q + 0.687 q^2
Sampling time: 1
```

Кінець приклада.

1.4. Моделі систем з запізнюванням

1.4.1. Системи в безперервному часі

Передатну функцію одновимірної системи (SISO) з запізнюванням в безперервному часі можна представити так:

$$W(s) = G(s)\cdot e^{-s\cdot\tau}, \tag{1.29}$$

де $G(s)$ – передатна функція системи без запізнювання; τ – час запізнювання між входом та виходом системи.

Багатовимірна система (MIMO) може мати відмінні запізнювання в кожному каналі передачі. Передатна матриця такої системи визначається як:

$$\mathbf{W}(s) = [G_{ij}(s)\cdot e^{-s\cdot\tau_{ij}}], \tag{1.30}$$

де $G_{ij}(s)$ – передатна функція відповідного каналу системи; τ_{ij} – запізнювання між j-м входом та i-м виходом системи; $i = \overline{1,\ Ny}$; $j = \overline{1,\ Nu}$.

Запізнювання для моделі в tf, zpk або frd-формі задається шляхом привласнення властивості *ioDelayMatrix* відповідної змінної, що є матрицею запізнювань виду **Tau =** [τ_{ij}].

Приклад 1.11.

Задамо запізнювання системи безпосередньо при створенні SISO моделі. Зобразимо систему з передатною функцією $W_4(s) = \dfrac{e^{-2s}}{s+2}$.

Властивість *ioDelayMatrix* (скорочено - '*io*') такої моделі є скаляром.

```
» W4=tf(1,[1 2],'io',2)
Transfer function:
             1
exp(-2*s) * -----
            s + 2
```

Кінець приклада.

Для моделей в просторі станів (ss-форма) введення однакового запізнювання на вході чи виході каналу дає різні траєкторії змінних стану. Для

задання запізнювання таких систем слід використовувати властивості *InputDelay* (привласнити вектор виду **Tau** = [τ_j]) та *OutputDelay* (привласнити вектор виду **Tau** = [τ_i]) в залежності від конкретної задачі.

Якщо MIMO модель в tf, zpk або frd-формі має однакові запізнювання τ у всіх каналах, то бажано також використовувати властивість *InputDelay* (або *OutputDelay*) шляхом привласнення їй скалярного значення τ.

В загальному випадку для моделей в tf, zpk або frd-формі, якщо властивість *InputDelay* має значення [α_1 α_2 … α_{Nu}] а властивість *OutputDelay* має значення [β_1 β_2 … β_{Ny}], то це рівнозначно тому, що властивість *ioDelayMatrix* має значення:

$$\mathbf{Tau} = \begin{bmatrix} \alpha_1+\beta_1 & \alpha_2+\beta_1 & \cdots & \alpha_{Nu}+\beta_1 \\ \alpha_1+\beta_2 & \alpha_2+\beta_2 & \cdots & \alpha_{Nu}+\beta_2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1+\beta_{Ny} & \alpha_2+\beta_{Ny} & \cdots & \alpha_{Nu}+\beta_{Ny} \end{bmatrix}. \qquad (1.31)$$

1.4.2. Дискретні системи

Для дискретних систем запізнювання задається цілою кількістю періодів дискретизації (див. (1.12) та (1.20)). В CST дискретне запізнювання можна задати різними способами.

Приклад 1.12.

Розглянемо знов дискретну SISO модель G_1 з приклада 1.5.

```
» G3=G1
Transfer function:
   -0.11 z + 0.14
-------------------
z^2 - 1.66 z + 0.687
Sampling time: 1
» get(G3,'io')
ans = 0                  % Система без запізнювання
» set(G3,'io',2); G3     % Встановлюємо запізнювання в два T
```

```
Transfer function:
            -0.11 z + 0.14
z^(-2) * --------------------
         z^2 - 1.66 z + 0.687
Sampling time: 1
```

Кінець приклада.

При необхідності запізнювання дискретних систем можна врахувати зміною характеристичного полінома (функція delay2z).

Інший спосіб роботи з запізнюванням описано в п. 1.6.3.

Приклад 1.13.

Модель G_3 з приклада 1.12. має запізнювання в два інтервали дискретизації. Врахуємо це запізнювання як додаткові полюси:

```
» G4=delay2z(G3)
Transfer function:
      -0.11 z + 0.14
-------------------------
z^4 - 1.66 z^3 + 0.687 z^2
Sampling time: 1
```

Кінець приклада.

1.5. З'єднання моделей

CST дозволяє будувати lti-системи будь-якої складності за допомогою операцій з'єднання моделей, які можуть мати різні форми. Результатом таких операцій є lti-система у формі, що визначається за правилом пріоритета: frd > ss > zpk > tf. Дискретні системи можна з'єднувати тільки, якщо вони мають однаковий або невизначений (–1) період дискретизації.

1.5.1. Послідовне з'єднання моделей

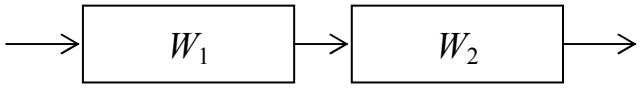

Рис. 1.2.

Таке з’єднання (див. рис. 1.2.) відповідає множенню моделей:

$$W = W_1 \times W_2, \tag{1.32}$$

і може також здійснюватись за допомогою функції series.

Приклад 1.14.

З’єднаємо послідовно об’єкти з передатними функціями W_1 та W_2 (див. приклади 1.1. та 1.2.):

```
» W1,W2
Transfer function:
   -0.15 s + 0.0375
-----------------------
s^2 + 0.375 s + 0.03125
Zero/pole/gain:
  0.3 (s+0.4)
---------------
(s+0.7) (s+0.3)
» W=W1*W2
Zero/pole/gain:
     -0.045 (s-0.25) (s+0.4)
---------------------------------
(s+0.25) (s+0.3) (s+0.125) (s+0.7)
```

Такий самий результат дає команда:

```
» W=series(W1,W2)
```

Кінець приклада.

1.5.2. Паралельне з’єднання моделей

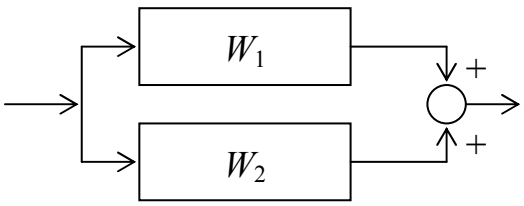

Рис. 1.3.

Таке з’єднання (див. рис. 1.3.) відповідає складанню моделей:

$$W = W_1 + W_2, \tag{1.33}$$

і може також здійснюватись за допомогою функції parallel.

Приклад 1.15.

З'єднаємо паралельно об'єкти з передатними функціями W_1 та W_2 (див. попередній приклад):

```
» W=parallel(W1,W2)
Zero/pole/gain:
0.15 (s+0.309) (s^2 + 0.491s + 0.2508)
--------------------------------------
  (s+0.25) (s+0.3) (s+0.125) (s+0.7)
```

Такий самий результат дає команда:

```
» W=W1+W2
```

Кінець приклада.

1.5.3. Зустрічно-паралельне з'єднання моделей

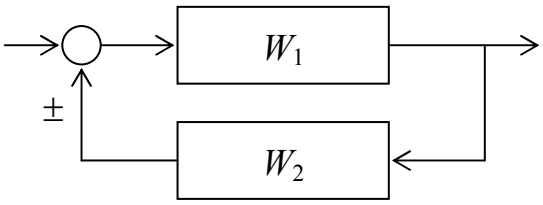

Рис. 1.4.

Це з'єднання (див. рис. 1.4.) відповідає формулі:

$$W = \frac{W_1}{1 \mp W_1 \times W_2}. \tag{1.34}$$

Для додатного зворотного зв'язку в знаменнику (1.34) ставлять знак "–" а для від'ємного – знак "+". Таке з'єднання може також здійснюватись за допомогою функції feedback (параметр *sign* приймає значення +1 для додатного зворотного зв'язку і –1 для від'ємного).

Приклад 1.16.

З'єднаємо об'єкти з передатними функціями W_1 та W_2 (див. попередній приклад) зустрічно-паралельно з від'ємним зворотним зв'язком:

```
» W=feedback(W1,W2,-1)
```

```
Zero/pole/gain:
        -0.15 (s+0.7) (s+0.3) (s-0.25)
-------------------------------------------
(s+0.8015) (s+0.3737) (s^2 + 0.1998s + 0.03694)
```

З'єднаємо об'єкти з передатними функціями W_1 та W_2 зустрічно-паралельно із додатним зворотним зв'язком:

```
» W=W1/(1-W1*W2);
» W=minreal(W)
 Zero/pole/gain:
       -0.15 (s-0.25) (s+0.3) (s+0.7)
-------------------------------------------
(s+0.5235) (s+0.0198) (s^2 + 0.8317s + 0.199)
```

Кінець приклада.

1.5.4. Довільне з'єднання моделей

На практиці важко побудувати модель в просторі станів по її структурній схемі навіть для систем помірної складності. Істотно спрощує цю задачу застосування функцій append та connect.

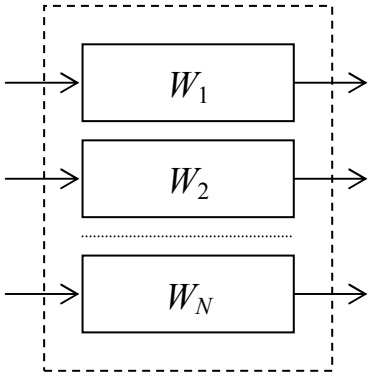

Рис. 1.5.

Спочатку за допомогою функції append необхідно створити діагонально-блочну модель **S** без врахування перехресних зв'язків (див. рис. 1.5.) виду:

$$\mathbf{S} = \mathrm{diag}[\, W_1\ W_2 \ldots W_N\,]. \qquad (1.35)$$

Потім треба задати матрицю перехресних зв'язків **Q**, кожен рядок якої відповідає одному входу системи **S**. Перший елемент рядка – це номер входу, а інші елементи вказують на номери виходів, що алгебраїчно підсумовуються на

цьому вході, причому від'ємні елементи вказують на підсумовування зі знаком "–". Крім того треба задати вектори входу **In** та виходу **Out**, що вказують які входи та виходи системи **S** є зовнішніми. Після цього функція connect(**S**,**Q**,**In**,**Out**) будує модель заданої структури в ss-формі.

Приклад 1.17.

З'єднаємо об'єкти з передатними функціями W_1 та W_2 зустрічно-паралельно з від'ємним зворотним зв'язком (див. попередній приклад та рис. 1.4.). Така схема характеризується матрицями:

$$\mathbf{S}=\begin{bmatrix} W_1 & 0 \\ 0 & W_2 \end{bmatrix};\ \mathbf{Q}=\begin{bmatrix} 1 & -2 \\ 2 & 1 \end{bmatrix};$$

$$\mathbf{In}=1; \qquad \mathbf{Out}=1.$$

```
» S=append(W1,W2);
» Q=[1 -2;2 1];
» W=connect(S,Q,1,1)
a =
                    x1          x2          x3          x4
          x1       -0.375      -0.125    -0.42426    -0.16971
          x2         0.25           0           0           0
          x3     -0.10607     0.10607          -1       -0.21
          x4            0           0           1           0
b =
                    u1
          x1       0.7746
          x2            0
          x3            0
          x4            0
c =
                    x1          x2          x3          x4
          y1     -0.19365     0.19365           0           0
d =
                    u1
          y1            0
```

Кінець приклада.

1.5.5. Конкатенація моделей

Побудувати багатовимірну систему (MIMO) з кількох одновимірних (SISO) можна за допомогою операцій конкатенації.

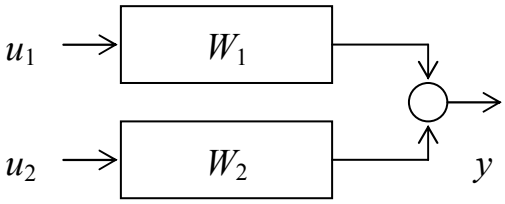

Рис. 1.6.

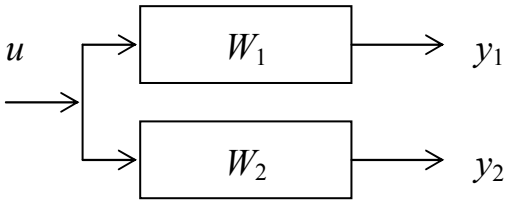

Рис. 1.7.

Горизонтальна конкатенація зображена на рис. 1.6. та визначається рівнянням виду:

$$y = \begin{bmatrix} W_1 & W_2 & \dots & W_{Nu} \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{Nu} \end{bmatrix}. \tag{1.36}$$

Вертикальна конкатенація зображена на рис. 1.7. та визначається рівнянням виду:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{Ny} \end{bmatrix} = \begin{bmatrix} W_1 \\ W_2 \\ \vdots \\ W_{Ny} \end{bmatrix} \cdot u\,. \tag{1.37}$$

Приклад 1.18.

Побудуємо MIMO систему з двома входами та двома виходами, що визначена в прикладі 1.8. (див. п. 1.2.3.), за допомогою операцій конкатенації:

```
» s=tf('s');
» W11=(s+2)/(s^2+2*s+1);        % Передатні функції
» W12=0*s+3;                    % окремих каналів регулювання
» W22=0*s+4;
» W21=2/(s+2);
» W=[W11 W12;W21 W22]           % Передатна функція MIMO системи
```

```
Transfer function from input 1 to output...
         s + 2
#1:  ------------
     s^2 + 2 s + 1
       2
#2:  -----
     s + 2
Transfer function from input 2 to output...
#1:  3
#2:  4
```

Кінець приклада.

1.6. Перетворення моделей

1.6.1. Перетворення форм моделей

Як відзначалось раніше, в CST lti-модель може існувати в чотирьох формах: tf, zpk, ss та frd. Перші три форми можна явно перетворити в будь-яку іншу за допомогою функцій сворення моделей в такому форматі:

systf = tf(*sys*);

syszpk = zpk(*sys*);

sysss = ss(*sys*);

sysfrd = frd(*sys*, *frequency*),

де *frequency* – вектор частот, для яких розраховується частотний відгук.

Приклад 1.19.

Модель в прикладі 1.1. (або 1.4.) визначена, як поліноміальна передатна функція W_1 (тобто в tf-формі):

```
» W1
Transfer function:
   -0.15 s + 0.0375
-----------------------
s^2 + 0.375 s + 0.03125
```

Перетворимо її в zpk-форму:

```
» W1z=zpk(W1)
Zero/pole/gain:
  -0.15 (s-0.25)
------------------
(s+0.25) (s+0.125)
```

Та ж модель в просторі станів матиме вид:

```
» W1s=ss(W1)
a =
                      x1            x2
           x1       -0.375       -0.125
           x2         0.25             0
b =
                      u1
           x1          0.5
           x2            0
c =
                      x1            x2
           y1         -0.3           0.3
d =
                      u1
           y1            0
Continuous-time model.
```

Кінець приклада.

Треба відзначити, що frd-модель не може бути перетворена в іншу форму.

1.6.2. Перетворення моделей типу дискретний/безперервний час

Для такого типу перетворень моделей в tf zpk та ss-формах існують дві функції: c2d – для перетворення моделей з безперервного часу в дискретний, та d2c – для зворотного перетворення.

Приклад 1.20.

Модель W_1 (див. попередній приклад) визначена в безперервному часі. В дискретному часі з періодом дискретизації $T = 0.5$ модель матиме вид:

```
» W1d=c2d(W1,0.5)
```

```
Transfer function:
-0.06389 z + 0.07244
---------------------
z^2 - 1.822 z + 0.829
Sampling time: 0.5
```

Для перевірки зробимо зворотне перетворення:

```
» d2c(W1d)
Transfer function:
   -0.15 s + 0.0375
-----------------------
s^2 + 0.375 s + 0.03125
```

Кінець приклада.

Наведені методи дискретизації та екстраполяції побудовані на використанні екстраполятора нульового порядку – zoh (zero-order hold). Принцип дії zoh полягає в тому, що при отриманні на вхід дискретного сигналу $u(k)$ він генерує безперервний сигнал $u(t)$, екстраполюючи дискретне значення постійним рівнем впродовж всього періоду дискретизації:

$$u(t) = u(k),\ kT \le t \le (k+1)T. \tag{1.38}$$

Крім того в функції c2d можна використовувати метод екстраполятора першого порядку – foh (first-order hold). Для цього параметру *method* треба привласнити значення ‘foh’. Екстраполятор foh відрізняється від zoh способом апроксимації вхідного сигналу. Він використовує лінійну екстраполяцію на періоді дискретизації:

$$u(t) = u(k) + \frac{t-kT}{T}(u(k+1)-u(k)),\ kT \le t \le (k+1)T. \tag{1.39}$$

Про інщі методи дискретизації та екстраполяції в функціях c2d та d2c дивись в [23, 27].

1.6.3. Зміна періоду дискретизації моделі

Змінити період дискретизації дискретних моделей в tf zpk та ss-формах можна за допомогою функції d2d, що використовує екстраполятора нульового порядку (zoh).

Приклад 1.21.

В попередньому прикладі отримана дискретна передатна функція W_{1d} моделі з періодом дискретизації $T = 0.5$. Змінемо період дискретизації моделі:

```
» d2d(W1d,0.3)           % Для T=0.3
Transfer function:
 -0.04092 z + 0.04411
----------------------
z^2 - 1.891 z + 0.8936
Sampling time: 0.3
» d2d(W1d,1)             % Для T=1
Transfer function:
  -0.1079 z + 0.1391
----------------------
z^2 - 1.661 z + 0.6873
Sampling time: 1
```

Кінець приклада.

За допомогою тієї ж функції в форматі:

sysk = d2d(*sys*, [], *Nk*),

можна задати запізнювання дискретної системи у вигляді додаткових полюсів (див. п. 1.4.2.).

Приклад 1.22.

В прикладі 1.12. з дискретної передатної функції G_1 без запізнювання отримано передатну функцію G_3 з запізнюванням в два інтервали дискретизації. В прикладі 1.13. отримано передатну функцію G_4, де запізнювання враховано у вигляді додаткових полюсів. Те ж можна зробити однією командою:

```
» G4=d2d(G1,[],2)
Transfer function:
      -0.11 z + 0.14
-------------------------
z^4 - 1.66 z^3 + 0.687 z^2
Sampling time: 1
```

Кінець приклада.

1.7. Дослідження моделей

1.7.1. Дослідження нулів та полюсів системи

Характеристики у вигляді діаграм нулів та полюсів є універсальними для lti-моделей у будь-якій формі. CST містить багато функцій для розрахунків, сортування та візуалізації розташування нулів та полюсів на комплексній площині (див. додаток 1.2.).

Розрахувати нулі та полюси системи можна за допомогою функцій zero та pole.

Приклад 1.23.

Розрахуємо нулі та полюси системи в безперервному часі з передатною функцією W_1 (див. приклади 1.4 та 1.19):

```
» zero(W1)
ans = 0.2500
» pole(W1)
ans = -0.2500
      -0.1250
```

Повторимо розрахунки для дискретного аналога цієї системи W_{1d} з періодом дискретизації $T = 0.5$ (див. приклад 1.20.):

```
» zpk(W1d)
Zero/pole/gain:
 -0.063894 (z-1.134)
--------------------
(z-0.9394) (z-0.8825)
Sampling time: 0.5
» zero(W1d)
ans = 1.1337
» pole(W1d)
ans = 0.9394
      0.8825
```

Кінець приклада.

Розглянемо команду побудови діаграми нулів та полюсів pzmap разом з допоміжними командами нанесення сітки сталих ліній коефіцієнта згасання і частоти sgrid та zgrid.

Приклад 1.24.

Побудуємо діаграму нулів та полюсів для системи в безперервному часі з передатною функцією W_1 (див. попередній приклад):

```
» pzmap(W1)
» sgrid
```

Результат побудови зображено на рис. 1.8.

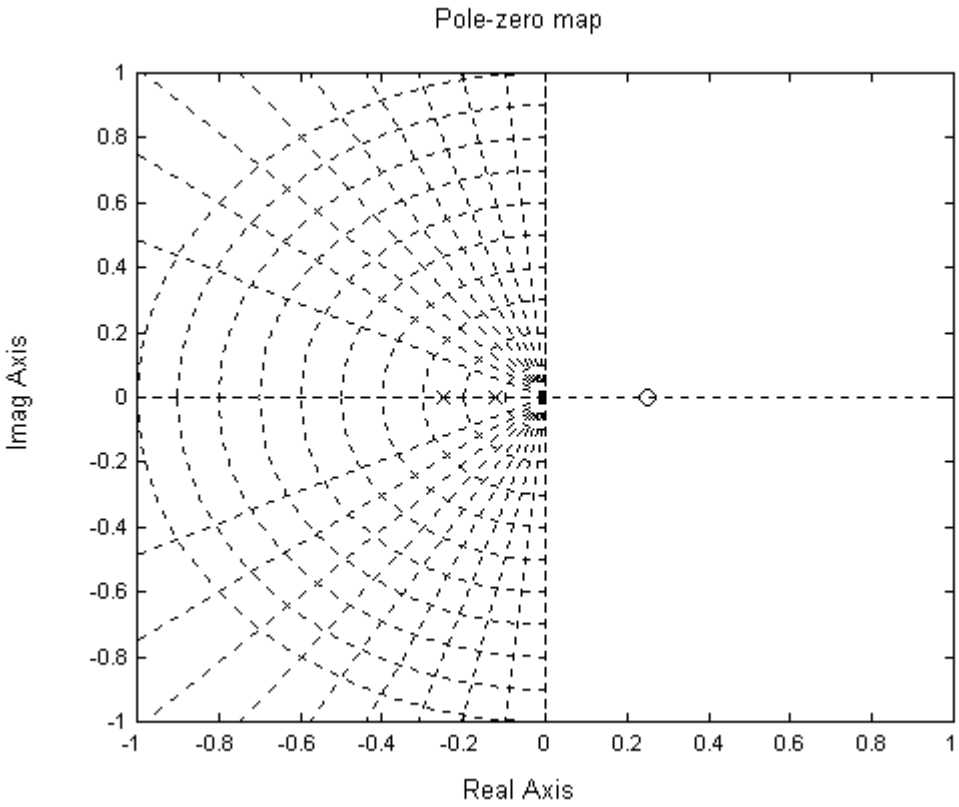

Рис. 1.8.

Побудуємо тепер діаграму нулів та полюсів для дискретної системи з передатною функцією W_{1d} (див. попередній приклад):

```
» pzmap(W1d)
» zgrid
```

Результат побудови зображено на рис. 1.9.

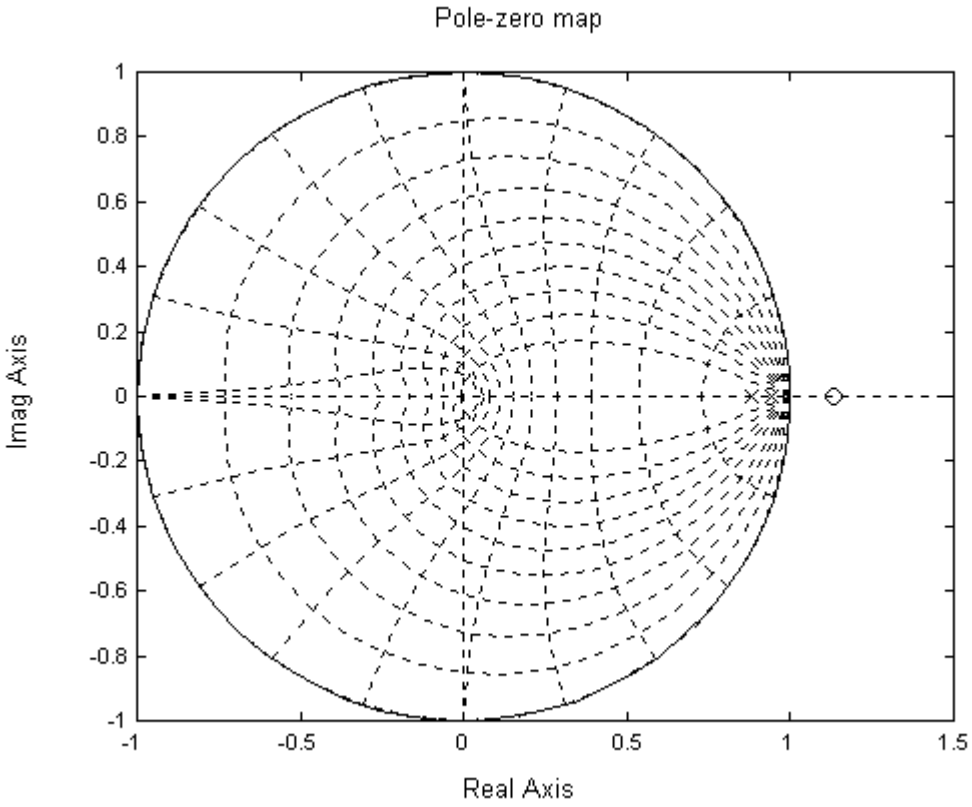

Рис. 1.9.

Кінець приклада.

1.7.2. Дослідження моделей в часовій області

Важливими характеристиками динамічних систем є їх перехідні функції. По виду перехідного процесу можна визначити коефіцієнт передачі системи, час встановлення процесу, перерегулювання і статичну помилку та інші характеристики.

Перехідною функцією $h(t)$ називають функцію, що описує відгук системи на одиничний ступінчастий вхід при нульових початкових умовах. Аналітично одиничний ступінчастий вхід можна описати одиничною функцією:

$$u(t) = 1(t) = \begin{Bmatrix} 1 \text{ при } t \geq 0; \\ 0 \text{ при } t < 0. \end{Bmatrix} \tag{1.40}$$

Для побудови перехідної характеристики використовується команда step.

Приклад 1.25.

Побудуємо перехідні характеристики систем з передатними функціями W_1 та W_3 (див. приклади 1.3. та 1.4.):

```
» step(W1,W3)
```

Результат побудови зображено на рис. 1.10.

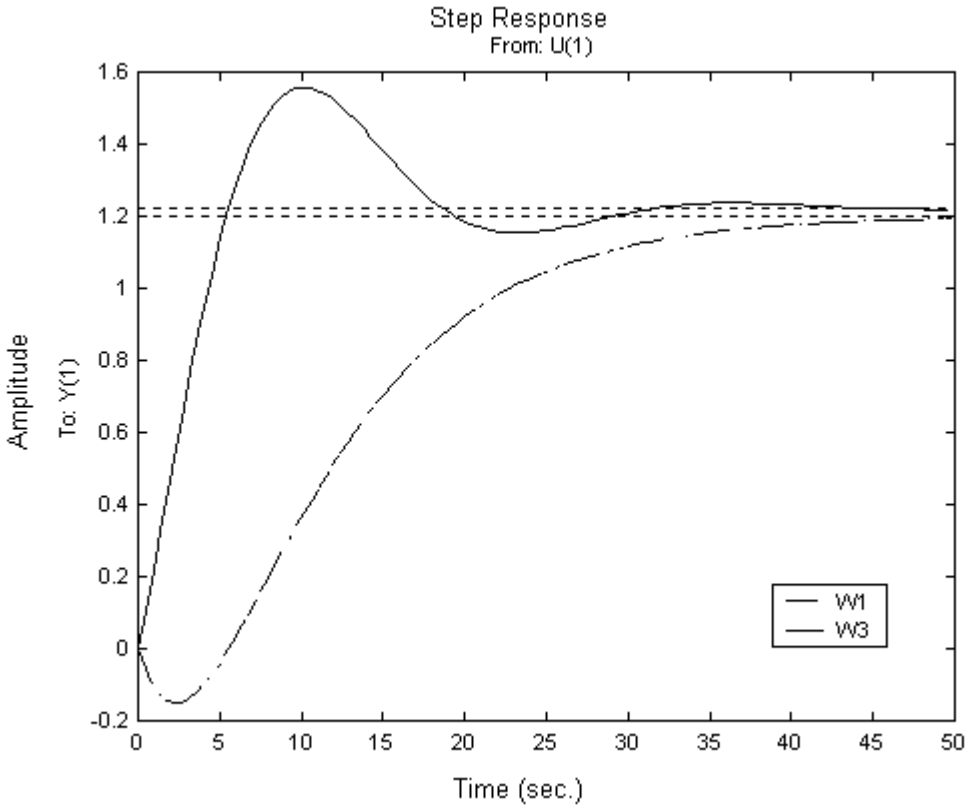

Рис. 1.10.

По характеру перехідних процесів можна сказати, що об’єкт з передатною функцією W_1 є немінімально-фазовим, а об’єкт з передатною функцією W_3 – коливальним [5].

Кінець приклада.

Імпульсна перехідна функція $g(t)$ визначається як відгук системи в довільний момент часу на імпульсне збурення. Під імпульсним входом розуміють одиничну імпульсну функцію :

$$u(t) = \delta(t), \tag{1.41}$$

яка має такі властивості:

$$\delta(t) = 0, \text{ ï ðè } t \neq 0, \quad \int_{-\infty}^{\infty} \delta(t)\, d(t) = 1, \quad \frac{dl(t)}{dt} = \delta(t). \tag{1.42}$$

Для побудови імпульсної перехідної характеристики використовується команда impulse.

Дискретна імпульсна функція $g(k)$ визначається як відгук системи на одиничний імпульс:

$$u(k) = \begin{cases} 1, \text{ при } k = 0; \\ 0, \text{ при } k \neq 0. \end{cases} \tag{1.43}$$

Приклад 1.26.

Побудуємо імпульсні перехідні характеристики системи з передатною функцію W_1 та її дискретного аналога з передатною функцію W_{1d} (див. попередній приклад):

```
» impulse(W1,c2d(W1,1))
```

Результат побудови зображено на рис. 1.11.

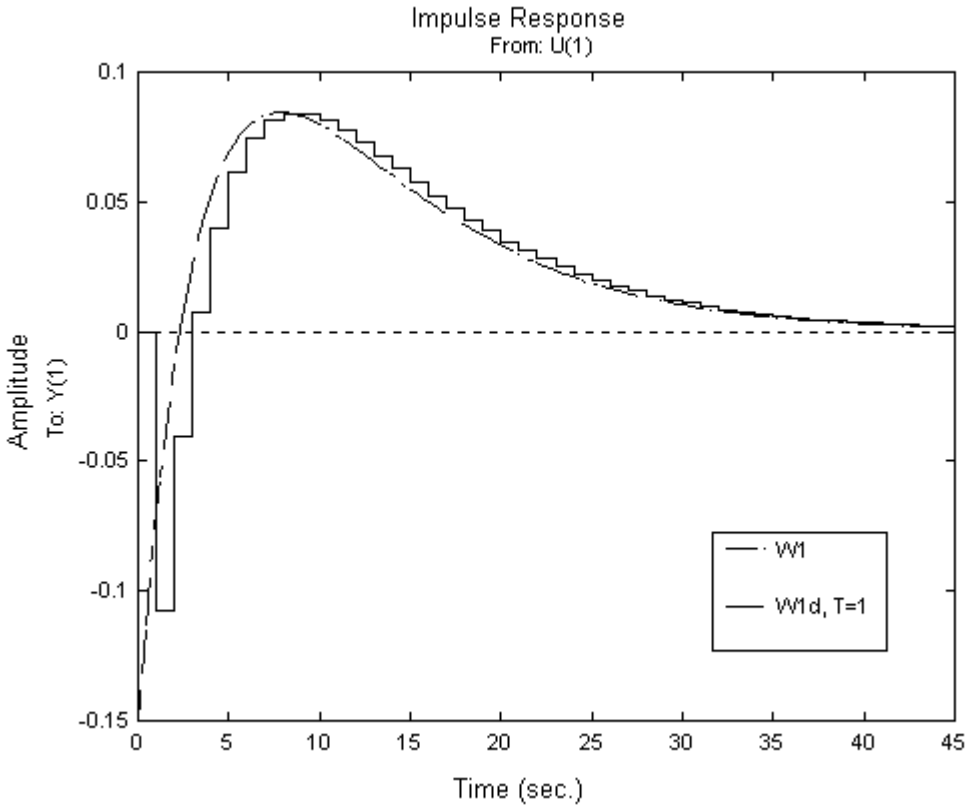

Рис. 1.11.

Кінець приклада.

Побудувати реакцію системи на довільний вхід можна за допомогою команди lsim. При цьому вхід задається двома параметрами: t – задає інтервал моделювання, а u – це матриця, кожен рядок якої $u(i, :)$ задає значення вхідних сигналів для всіх входів системи в момент часу $t(i)$.

Сформувати значення параметрів *t* та *u* для періодичних (синусоїдальних, прямокутних або імпульсних) сигналів одиничної амплітуди можна за допомогою функції gensig.

Приклад 1.27.

Побудуємо відгук системи з передатною функцію W_1 на синусоїдальний вхідний сигнал з амплітудою 4 та періодом 5:

```
» [u,t]=gensig('sin',5);
» grid on
» lsim(W1,4*u(1:200),t(1:200))
```

Результат побудови зображено на рис. 1.12.

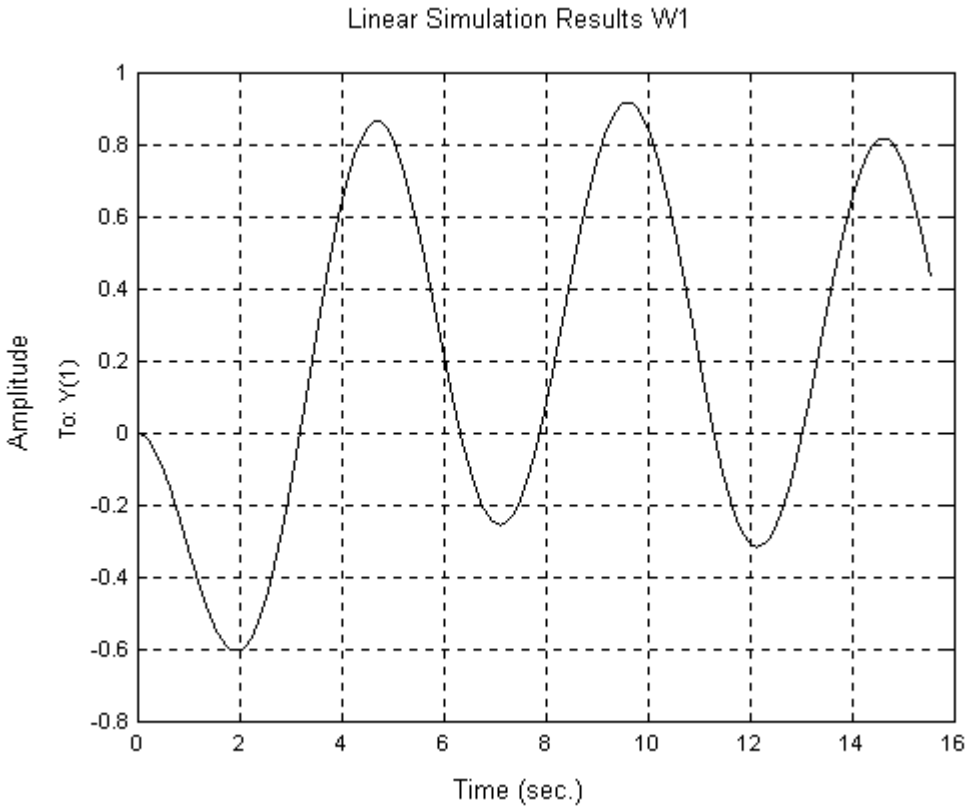

Рис. 1.12.

Кінець приклада.

1.7.3. Частотні характеристики систем

Такі властивості систем, як стійкість замкненого контуру, запас стійкості по амплітуді та фазі, ширина смуги пропускання, резонансні частоти,

коефіцієнт передачі по постійному сигналу можна визначити за частотними характеристиками.

Частотна характеристика системи в безперервному часі визначається рівнянням (1.8). В загальному випадку частотна характеристика є комплексною функцією частоти і може бути зображена у вигляді:

$$G(j\omega) = G_{\mathrm{Re}}(\omega) + jG_{\mathrm{Im}}(\omega), \tag{1.44}$$

де $G_{\mathrm{Re}}(\omega)$ та $G_{\mathrm{Im}}(\omega)$ – дійсна та уявна частини функції $G(j\omega)$ відповідно, які визначаються рівняннями:

$$G_{\mathrm{Re}}(\omega) = \int_0^\infty g(\tau)\cos(\omega\cdot\tau)d\tau, \tag{1.45}$$

$$G_{\mathrm{Im}}(\omega) = \int_0^\infty g(\tau)\sin(\omega\cdot\tau)d\tau. \tag{1.46}$$

Частотну характеристику можна також записати в полярній формі:

$$G(j\omega) = \left|G(j\omega)\right|\cdot\exp(j\varphi(\omega)), \tag{1.47}$$

де

$$\left|G(j\omega)\right| = \sqrt{G_{\mathrm{Re}}^2(\omega) + G_{\mathrm{Im}}^2(\omega)}, \tag{1.48}$$

$$\varphi(\omega) = \mathrm{Arg}\,G(j\omega) = \mathrm{arctg}\left(\frac{G_{\mathrm{Im}}(\omega)}{G_{\mathrm{Re}}(\omega)}\right), \text{ ï ðè } \left|\mathrm{Arg}\,G(j\omega)\right| \le \frac{\pi}{2}. \tag{1.49}$$

Модуль $\left|G(j\omega)\right|$ називається амплітудною характеристикою, а аргумент $\varphi(\omega)$ – фазовою характеристикою.

Приклад 1.28.

Побудуємо амплітудно-фазову частотну характеристику (годограф) для системи з передатною функцію W_1 (див. попередній приклад) для 80 значень частоти в рад/с, логарифмічно розподілених в діапазоні $\omega = \overline{0,\ 2\pi}$:

```
» fr=logspace(-4,0.8,80);          % Визначення вектора частот
» resp=squeeze(frdata(frd(W1,fr)));% Визначення вектора частотного
                                   % відгуку системи W1
» plot(real(resp),imag(resp))      % Або просто plot(resp)
» grid on
```

```
» title('Годограф АФХ W1')
» xlabel('Дійсна вісь')
» ylabel('Уявна вісь')
```

Результат побудови зображено на рис. 1.13.

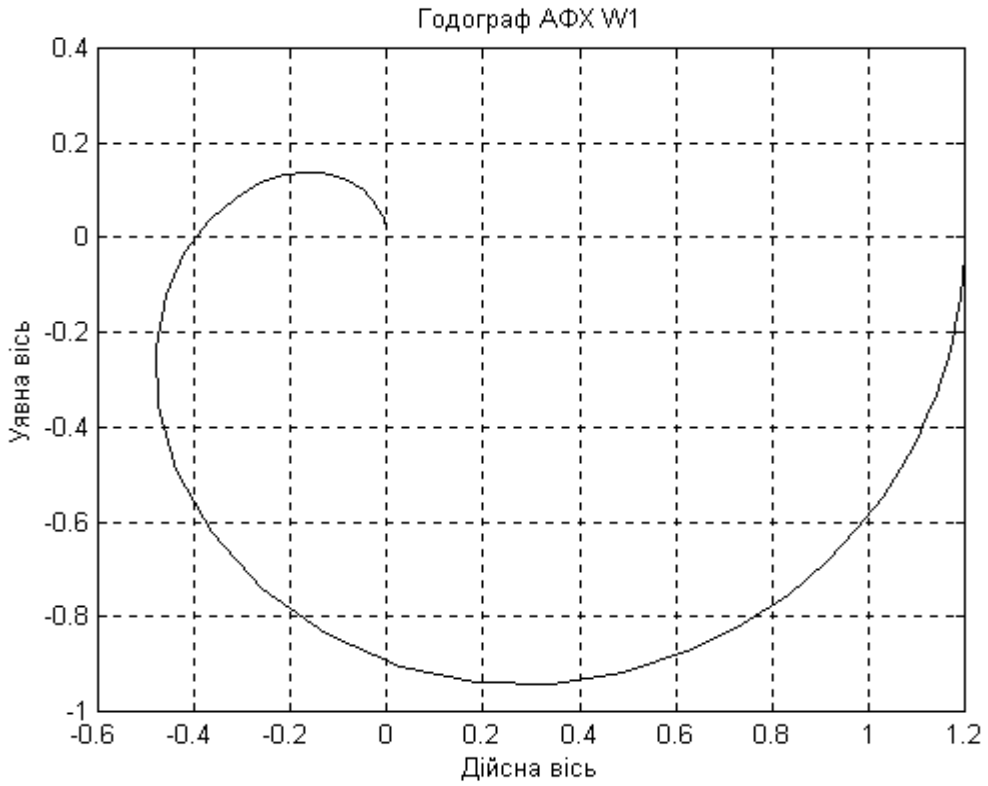

Рис. 1.13.

Те ж саме можна зробити в полярних координатах (див. (1.48) та (1.49)):

```
» polar(angle(resp),abs(resp))
» title('Годограф АФХ W1')
```

Результат побудови зображено на рис. 1.14.

Кінець приклада.

Побудувати амплітудно-фазову частотну характеристику у вигляді годографа Найквіста можна також за допомогою команди nyquist.

Приклад 1.29.

Побудуємо годограф Найквіста для системи з передатною функцію W_1 (див. попередній приклад). Результат побудови зображено на рис. 1.15.

```
» grid on
» nyquist(W1)
```

Кінець приклада.

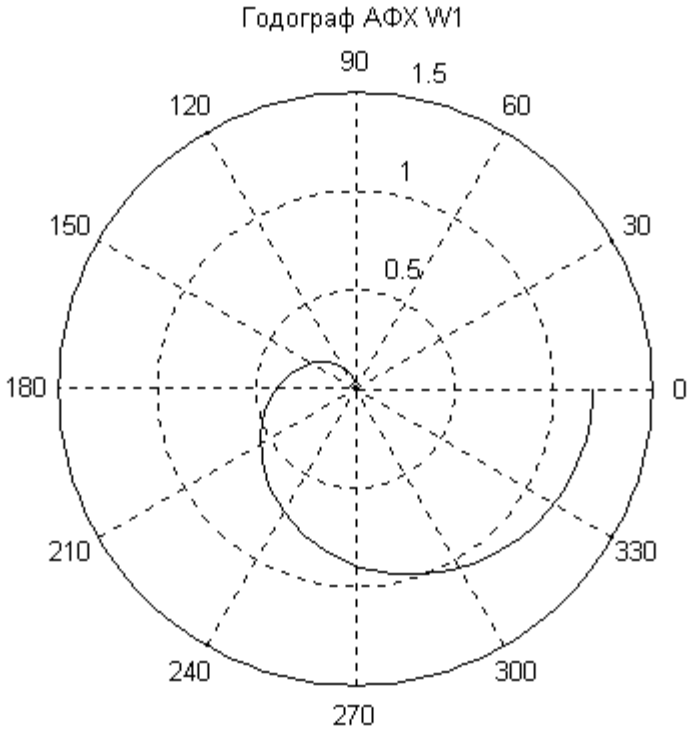

Рис. 1.14.

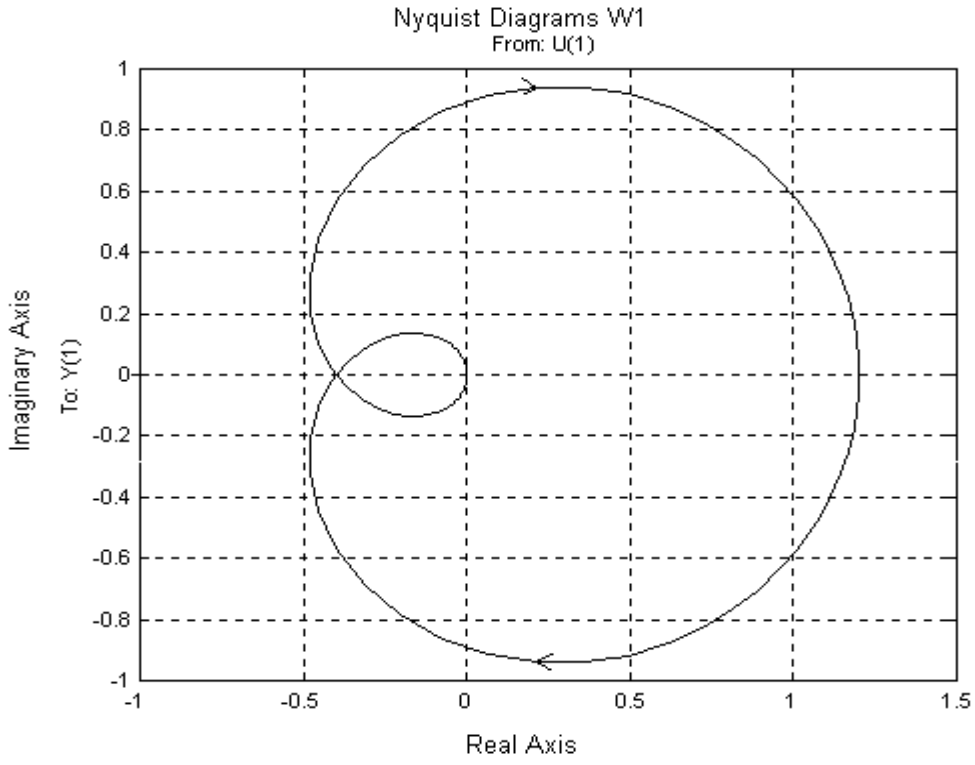

Рис. 1.15.

1.7.4. Логарифмічні частотні характеристики

Частотні характеристики системи зручно розглядати в логарифмічному масштабі, тому що:

– амплітудну характеристику (1.48) можна апроксимувати кусково-лінійною ломаною;

– такі діаграми охоплюють широкий діапазон значень частоти;

– множення амплітуди частотної характеристики в логарифмічному масштабі зводиться до простого додавання.

Логарифмічна амплітудна частотна функція визначається як:

$$L(\omega) = 20\lg\left|G(j\omega)\right|. \qquad (1.50)$$

Одиницями вимірювання $L(\omega)$ є децибел (дБ).

Графік залежності $L(\omega)$ від логарифму частоти називають логарифмічною амплітудною частотною характеристикою. Логарифмічною фазовою частотною характеристикою називають графік залежності фазової частотної функції $\varphi(\omega)$ від логарифму частоти [4, 11]. Разом вони складають діаграму Боде системи, яку можна побудувати командою bode.

Приклад 1.30.

Побудуємо логарифмічні частотні характеристики системи з передатною функцію W_1 (див. попередній приклад):

```
» grid on
» bode(W1)
```

Результат побудови зображено на рис. 1.16.

Кінець приклада.

Перепишемо передатну функцію системи (1.18) в іншій формі:

$$G(s) = \frac{K(1+\tau_1^{[c]}s)(1+\tau_2^{[c]}s)...(1+\tau_{Nb}^{[c]}s)}{(1+\tau_1^{[s]}s)(1+\tau_2^{[s]}s)...(1+\tau_{Na}^{[s]}s)}, \qquad (1.51)$$

де $\tau_i^{[s]} = -\frac{1}{s_i}$ та $\tau_i^{[c]} = -\frac{1}{c_i}$ – сталі часу елементарних ланок.

Слід пам’ятати, що при наявності комплексних нулів або полюсів розкладення (1.51) матиме і члени другого порядку.

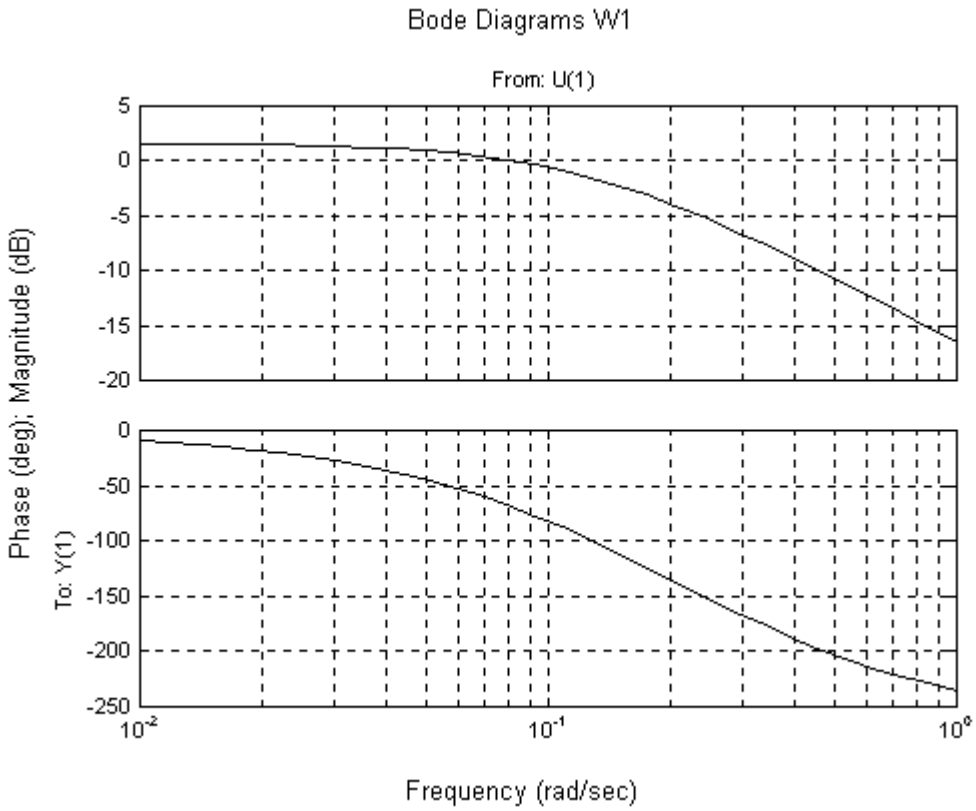

Рис. 1.16.

Тоді відповідні логарифмічні частотні характеристики можна визначити як:

$$\begin{aligned} L(\omega) &= \sum_{i=1}^{Na+Nb+1} 20 \cdot \lg\left|W_i(j\omega)\right|; \\ \varphi(\omega) &= \sum_{i=1}^{Nb} \operatorname{arctg}(T_i^{[c]}\omega) - \sum_{i=1}^{Na} \operatorname{arctg}(T_i^{[s]}\omega), \end{aligned} \tag{1.52}$$

де $W_i(j\omega)$ – частотні характеристики елементарних ланок, на які можна розкласти систему.

Існує чотири типи елементарних ланок [4]:

– Підсилювальна ланка:

$$W(j\omega) = K; \;\; L(\omega) = const; \;\; \varphi(\omega) = 0. \tag{1.53}$$

– Ланка з полюсом (або нулем) в початку координат на комплексній площині:

$$W(j\omega) = (j\omega)^{\pm 1}; \;\; L(\omega) = \pm 20 \cdot \lg(\omega); \;\; \varphi(\omega) = \pm\frac{\pi}{2}. \tag{1.54}$$

– Ланка з полюсом (або нульом) на дійсній осі:

$$W(j\omega)=(1+j\omega\tau)^{\pm 1};\ \ L(\omega)=\pm 10\cdot\lg(1+\omega^2\tau^2);\ \ \varphi(\omega)=\pm\operatorname{arctg}(\omega\tau)\,. \quad (1.55)$$

Частоту $\omega_n=\dfrac{2\pi}{\tau}$ називають частотою зламу.

– Ланка з комплексними полюсами (або нулями):

$$W(j\omega)=(1+j2\zeta\vartheta-\vartheta^2)^{\pm 1};$$
$$L(\omega)=\pm 10\cdot\lg((1-\vartheta^2)^2+4\zeta^2\vartheta^2);\ \ \varphi(\omega)=\pm\operatorname{arctg}\left(\frac{2\zeta\vartheta}{1-\vartheta^2}\right), \quad (1.56)$$

де $\vartheta=\dfrac{\omega}{\omega_n}$ а ζ – коефіцієнт згасання (демпфування), який характеризує відносну зміну суміжних амплітуд перехідного процесу.

Максимального значення амплітуда частотного відгуку набуває на резонансній частоті ω_r :

$$\omega_r=\omega_n\sqrt{1-2\zeta^2},\ \ \text{ï ðè}\ \zeta\le\frac{1}{\sqrt{2}}. \quad (1.57)$$

З діаграм (рис. 1.17.) видно, що асимптотичні характеристики коливальної ланки в області низьких частот співпадають з характеристиками ланки першого порядку, але в області високих частот зсув по фазі вдвічі більший. Крім того, при малих значеннях коефіцієнта згасання спостерігається наближення резонансної частоти ω_r до власної частоти ланки ω_n.

Приклад 1.31.

Дослідимо вплив коефіцієнта згасання ζ системи на характер її частотних характеристик на прикладі об’єкта з передатною функцією $W=\dfrac{5}{s^2+2\zeta s+1}$.

Побудуємо діаграми Боде для значень коефіцієнта згасання: $\zeta=$ `[0.01 0.06 0.15 0.4]`

```
» dzeta=[0.01 0.06 0.15 0.4];
» for i=1:length(dzeta) W(i)=tf(5,[1 2*dzeta(i) 1]); end
» grid on
» bode(W(1),W(2),W(3),W(4),{0.6,1.7})
```

Результат побудови частотних характеристик зображено на рис. 1.17.

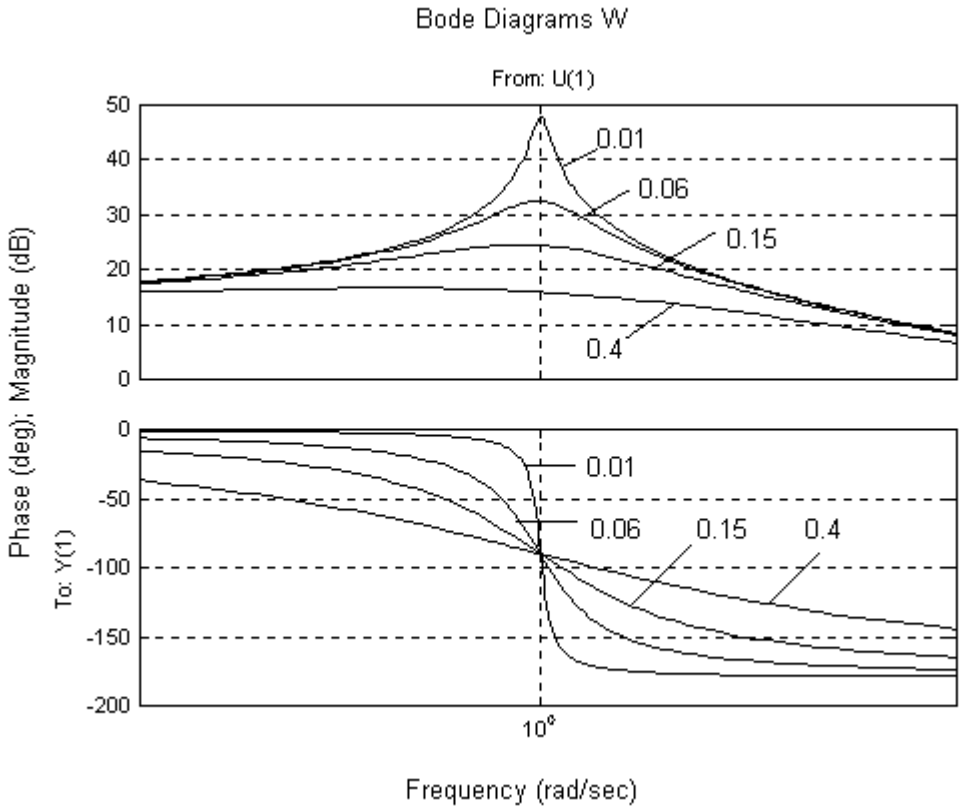

Рис. 1.17.

Кінець приклада.

Реальні системи часто мають дуже малий коефіцієнт згасання ($\zeta << 1$). По суті, такі системи є вузькосмуговими фільтрама [2]. Діапазон частот, що вони пропускають, прийнято визначати шириною смуги пропускання за рівнем половинної енергії:

$$B_r = \omega_2 - \omega_1 \approx 2\zeta\omega_r . \qquad (1.58)$$

1.7.5. LTI-Viewer

Для аналізу динамічних lti-систем в складі бібліотек CST є спеціальний інтерактивний інструмент, що реалізовано на базі графічного інтерфейсу користувача (GUI), і який включає описані засоби дослідження систем та багато інших можливостей. Виклик інструменту здійснюється командою ltiview.

Розділ 2. Ідентифікація динамічних систем

В цьому розділі наведено методи ідентифікації та дослідження динамічних стохастичних систем в MatLab за допомогою бібліотеки System Identification Toolbox (надалі SIT), а також засоби перетворення форм моделей для сумісності з бібліотекою Control System Toolbox (CST).

Повний список команд та функцій бібліотеки SIT наведено в додатку 1.1.

2.1. Загальні положення

Ідентифікація системи - це процес побудови динамічної моделі системи за результатами спостережень за її входами та виходами. В реальних умовах такі спостереження, майже завжди, спотворені оскільки система функціонує у випадковому середовищі (див. рис. 1.1.). Кожна серія вимірювань дає специфічну реалізацію процесу, що навряд чи повториться в майбутньому і яку не можна передбачити з достатньо великою точністю. Іншими словами, при ідентифікації ми маємо справу із стохастичними явищами і змушені враховувати другий доданок в загальній моделі (1.1) як недетерміноване збурення.

Розрізняють параметричну та непараметричну ідентифікацію.

Параметрична ідентифікація – це процес оцінювання параметрів моделі заданої структури. Методи параметричної ідентифікації зводяться до пошуку чисельних значень параметрів, які дають найкращій збіг між виходом моделі і виміреним виходом системи.

Непараметрична ідентифікація – це оцінювання поведінки системи у вигляді дискретних наборів даних без обов’язкового використання попередніх відомостей про модель. Типовими випадками непараметричної ідентифікації є пряме оцінювання імпульсного відгуку системи методом кореляційного аналізу та пряме оцінювання частотного відгуку системи методом спектрального аналізу [26].

2.2. Моделі стохастичних систем

2.2.1. Зображення стохастичних систем

При наявності дискретної інформації про систему в якості загальної (1.1) найчастіше використовують дискретну модель.

Зв'язок між входом $u(k)$ та виходом $y(k)$ лінійної системи можна виразити за допомогою передатної функції $G(q)$:

$$y(k) = G(q)u(k) + v(k) \tag{2.1}$$

де $v(k)$ – збурення системи, q – оператор зсуву назад, а

$$G(q)u(k) = \sum_{i=1}^{\infty} g(i)u(k-i), \tag{2.2}$$

де послідовність $g(k)$ є імпульсним відгуком системи (див. також (1.14)), тобто $g(i)$ це вихід системи в i-й момент часу, якщо входом системи був одиничний імпульс в нульовий момент часу.

Функція комплексної змінної $G(e^{j\omega})$ як функція кутової частоти ω тоді є частотною характеристикою або частотною функцією системи (див. (1.15)).

Збурення $v(k)$ можна розглядати як фільтрований білий шум $e(k)$. Тоді в іншій формі (2.1) можна записати:

$$y(k) = G(q)u(k) + H(q)e(k), \tag{2.3}$$

де $H(q)$ – передатна функція фільтру випадкового сигналу.

2.2.2. Стохастичні характеристики випадкового процесу

Задачу ідентифікації стохастичних динимічних систем за результатами вимірювань їх входів та виходів можна розглядати як процес виявлення лінійних залежностей між кількома сукупностями даних. Такі залежності визначаються через кореляційні функції або спектральні щільності [8].

Міру лінійності зв'язку випадкових процесів $\{v\}$ та $\{y\}$ можна задати так:

$$\sigma_{vy} = \mathrm{E}[(v-\mu_v)(y-\mu_y)] = \lim_{x\to\infty}\frac{1}{N}\sum_{i=1}^{N}(v_i-\mu_v)(y_i-\mu_y), \tag{2.4}$$

де μ_v – математичне сподівання процесу $\{v\}$, що визначається як:

$$\mu_v = \mathrm{E}[v] = \lim_{x\to\infty} \frac{1}{N} \sum_{i=1}^{N} v(i), \tag{2.5}$$

а μ_y – математичне сподівання процесу $\{y\}$, яке визначається подібно (2.5). Якщо розмір виборки N наближається до ∞, то (2.4) визначає кореляцію процесів. Для зручності міру лінійної залежності процесів оцінюють по значенню коефіцєнта кореляції:

$$\rho_{vy} = \frac{\sigma_{vy}}{\sigma_v \sigma_y}, \; -1 \le \rho_{vy} \le 1, \tag{2.6}$$

де σ_v – середньо-квадратичне (або стандартне) відхилення процесу $\{v\}$, що визначається як:

$$\sigma_v = +\sqrt{\sigma_v^2}, \tag{2.7}$$

а дисперсія σ_v^2 визначається як:

$$\sigma_v^2 = \mathrm{E}[v^2] - \mu_v^2 = \mathrm{E}[(v-\mu_v)^2] = \lim_{x\to\infty} \frac{1}{N} \sum_{i=1}^{N} (v(i)-\mu_v)^2, \tag{2.8}$$

середньо-квадратичне відхилення σ_y та дисперсія σ_y^2 процесу $\{y\}$ визначаються аналогічно (2.7) та (2.8). Повна лінійна залежність процесів відповідає: $\rho_{vy} = \pm 1$.

Будемо вважати процеси, що розглядаються, стаціонарними та ергодичними, тобто будемо мати справу з одиничними реалізаціям процесів: $v(k)$ та $y(k)$. Введемо нове поняття: лаг (λ) - запізнювання $y(k)$ відносно $v(k)$. Тоді визначемо кореляційну функцію $v(k)$ та $y(k)$ для довільного λ (або взаємну кореляційну функцію) по аналогії з (2.4):

$$\begin{aligned} C_{vy}(\lambda) &= \mathrm{E}[(v(k)-\mu_v)(y(k+\lambda)-\mu_y)] = \\ &= \lim_{x\to\infty} \frac{1}{N} \sum_{i=1}^{N} (v(i)-\mu_v)(y(i+\lambda)-\mu_y) = R_{vy}(\lambda) - \mu_v \mu_y, \end{aligned} \tag{2.9}$$

де взаємна коваріаційна функція визначається як:

$$R_{vy}(\lambda) = \mathrm{E}[v(k)y(k+\lambda))] = \lim_{x\to\infty} \frac{1}{N} \sum_{i=1}^{N} v(i)y(i+\lambda). \tag{2.10}$$

Взаємна коваріаційна функція дозволяє встановити, яку долю енергії вносить $v(k)$ в $y(k)$. Слід відзначити, що в деяких роботах (наприклад [5]) коваріаційною функцією називають відношення виду (2.9).

Одиничний ергодичний процес можна описати його математичним сподіванням (середнім арифметичним див. (2.5)) та автокореляційною функцією:

$$C_v(\lambda) = \mathrm{E}[(v(k) - \mu_v)(v(k+\lambda) - \mu_v)] = \\ = \lim_{x\to\infty} \frac{1}{N} \sum_{i=1}^{N} (v(i) - \mu_v)(v(i+\lambda) - \mu_v) = R_v(\lambda) - \mu_v^2, \tag{2.11}$$

де автоковаріаційна функція визначається як:

$$R_v(\lambda) = \mathrm{E}[v(k)v(k+\lambda))] = \lim_{x\to\infty} \frac{1}{N} \sum_{i=1}^{N} v(i)v(i+\lambda) = \sigma_v^2 + \psi_v^2, \tag{2.12}$$

а середній квадрат реалізації процесу $v(k)$ визначається як:

$$\psi_v^2 = \mathrm{E}[v^2] = \sigma_v^2 + \mu_v^2 = \lim_{x\to\infty} \frac{1}{N} \sum_{i=1}^{N} v^2(i). \tag{2.13}$$

Коваріаційна функція процесу вказує, в якій мірі знання минулого процесу дозволяє передбачити його майбутнє.

Взаємна коваріаційна функція вхідного $v(k)$ та вихідного $y(k)$ процесів системи пов'язана з автокореляційною функцією вхідного процесу $v(k)$ рівнянням Вінера-Хопфа [2, 9]:

$$R_{vy}(\lambda) = \lim_{x\to\infty} \sum_{i=1}^{N} R_v(i+\lambda) g(i), \tag{2.14}$$

де $g(i)$ – імпульсний відгук системи на відповідному кроці.

Нормовану кореляційну функцію можна визначити так:

$$\rho_{vy}(\lambda) = \frac{C_{vy}(\lambda)}{\sqrt{C_v(0)C_y(0)}} = \frac{R_{vy}(\lambda) - \mu_v\mu_y}{\sqrt{(R_v(0) - \mu_v^2)(R_y(0) - \mu_y^2)}}, \tag{2.15}$$

причому $\left|\rho_{vy}(\lambda)\right| \le 1$ для всіх λ.

Спектральні щільності реалізацій стаціонарних ергодичних процесів $v(k)$ та $y(k)$ визначаються як перетворення Фур'є їх коваріаційних функцій. В

літературі спектральну щільність часто також називають спектральною щільністю потужності або просто спектром. В залежності від виду коваріаційної функції розрізняють взаємні та автоспектри.

Наприклад автоспектр процесу $v(k)$ можна визначити як:

$$\Phi_v(\omega) = \sum_{\lambda=-\infty}^{\infty} R_v(\lambda) \cdot e^{-j\omega\lambda}. \qquad (2.16)$$

Він вказує швидкість зміни середнього квадрату реалізації процесу в залежності від частоти.

Взаємні коваріаційні функції та взаємні спектри інтерпретуються в схожий спосіб, але останні (як функції від частоти) дають змогу прямого оцінювання властивостей систем по спостереженням за входами та виходами.

2.2.3. Фільтри випадкових сигналів

Відомо [2], що будь-який стаціонарний випадковий процес $v(k)$ може бути з достатньою точністю описаний вихідним сигналом лінійного фільтру $H(q)$, на вхід якого подається білий шум $e(k)$ (див. (2.1) та (2.3)):

$$v(k) = H(q)e(k), \qquad (2.17)$$

Фільтрацію можна представити простою зваженою сумою минулих значень вхідного сигналу:

$$v(k) = e(k) + c_1 e(k-1) + ... + c_{Nc} e(k-Nc), \qquad (2.18)$$

або в інщій формі:

$$v(k) = C(q)e(k), \qquad (2.19)$$

де $C(q) = 1 + c_1 q^{-1} + ... + c_{Nc} q^{-Nc}$.

Модель такого фільтру називається моделлю з ковзним середнім (Moving Average, надалі MA).

Вихідний сигнал можна також записати у вигляді зваженої суми минулих значень вихідного сигналу $v(k)$:

$$v(k) + a_1 v(k-1) + ... + a_{Na} v(k-Na) = e(k), \qquad (2.20)$$

або в інщій формі:

$$v(k) = \frac{e(k)}{A(q)}, \tag{2.21}$$

де $A(q) = 1 + a_1 q^{-1} + ... + a_{Na} q^{-Na}$.

Модель такого фільтру називається моделлю авторегресії (AutoRegressive, надалі AR).

Очевидно, що по кінцевій кількості спостережень можна оцінити тільки кінцеву кількість параметрів, в даному випадку *Nc* (порядок поліному $C(q)$) та *Na* (порядок поліному $A(q)$).

Часто буває корисно враховуати в моделі компоненти MA та AR одночасно, що дає модель авторегресії з ковзним середнім (ARMA):

$$v(k) = \frac{C(q)}{A(q)} e(k). \tag{2.22}$$

Спектральну щільність збурення у вигляді стаціонарного випадкового процесу $v(k)$ можна описати через передатну функцію фільтру рівнянням:

$$\Phi_v(\omega) = \sigma_v^2 \left| H(e^{j\omega}) \right|^2. \tag{2.23}$$

2.2.4. Поліноміальна форма моделей систем

Загальну дискретну модель (2.3) системи з одним зовнішнім входом та одним виходом, що зазнає впливу збурення у вигляді стохастичного процесу (див. п. 2.2.3. та (1.20)) можна представити в поліноміальній формі як:

$$A(q)y(k) = \frac{B(q)}{F(q)} u(k - Nk) + \frac{C(q)}{D(q)} e(k); \tag{2.24}$$

де $B(q) = b_1 + b_2 q^{-1} + ... + b_{Nb} q^{-Nb+1}$;

$F(q) = 1 + f_1 q^{-1} + ... + f_{Nf} q^{-Nf}$;

$D(q) = 1 + d_1 q^{-1} + ... + d_{Nd} q^{-Nd}$,

тут $Nk \geq 1$ – запізнювання в системі; *Na* – кількість полюсів (див. (2.21)); $Nb - 1$ – кількість нулів, якщо вважати, що дискретна система без запізнювання має $Nk = 1$ (див. (1.12)); *Nf*, *Nd* – степені відповідних поліномів.

Розглянемо часткові випадки цієї моделі.

ARX (AutoRegressive with eXternal input model) – модель авторегресії із зовнішнім входом отримують, прийнявши $Nc = Nd = Nf = 0$ (див. також (2.21)):

$$A(q)y(k) = B(q)u(k - Nk) + e(k); \tag{2.25}$$

ARMAX (AutoRegressive with Moving Average and eXternal input model) – модель авторегресії з ковзним середнім та зовнішнім входом отримують, прийнявши $Nd = Nf = 0$ (див. також (2.22)):

$$A(q)y(k) = B(q)u(k - Nk) + C(q)e(k); \tag{2.26}$$

ARARX (generalized least-square model): $Nf = Nc = 0$.

$$A(q)y(k) = B(q)u(k - Nk) + \frac{e(k)}{D(q)}; \tag{2.27}$$

ARARMAX (extended matrix model): $Nf = 0$.

$$A(q)y(k) = B(q)u(k - Nk) + \frac{C(q)}{D(q)}e(k); \tag{2.28}$$

OE (Output-Error model): $Na = Nc = Nd = 0$.

$$y(k) = \frac{B(q)}{F(q)}u(k - Nk) + e(k); \tag{2.29}$$

BJ (Box-Jenkins model): $Na = 0$.

$$y(k) = \frac{B(q)}{F(q)}u(k - Nk) + \frac{C(q)}{D(q)}e(k); \tag{2.30}$$

MIMO модель (з довільною кількістю входів Nu) можна описати так:

$$A(q)y(k) = \sum_{i=1}^{Nu} \frac{B_i(q)}{F_i(q)}u_i(k - Nk_i) + \frac{C(q)}{D(q)}e(k). \tag{2.31}$$

2.2.5. Зображення стохастичних систем в просторі станів

Лінійна система в просторі станів описується рівняннями (1.27). З врахуванням випадковоих збурень (вектор $\mathbf{v}(k)$) їх можна записати так:

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A_d} \cdot \mathbf{x}(k) + \mathbf{B_d} \cdot \mathbf{u}(k); \\ \mathbf{y}(k) &= \mathbf{C} \cdot \mathbf{x}(k) + \mathbf{D} \cdot \mathbf{u}(k) + \mathbf{v}(k). \end{aligned} \tag{2.32}$$

Тут співвідношення між входом та виходом визначені через *Nx*-вимірний вектор стану $\mathbf{x}(k)$. Зв'язок між описом в просторі станів (2.32) та описом у вигляді передатної функції (2.1) можна встановити виразом:

$$G(q) = \mathbf{C}(q \cdot \mathbf{I_{Nx}} - \mathbf{A_d})^{-1}\mathbf{B_d} + \mathbf{D}, \tag{2.33}$$

де $\mathbf{I_{Nx}}$ – одинична матриця виміром $Nx \times Nx$.

В більш гнучкий відносно збурень формі (innovations form) (2.32) можна записати так:

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A_d} \cdot \mathbf{x}(k) + \mathbf{B_d} \cdot \mathbf{u}(k) + \mathbf{L_d} \cdot \mathbf{e}(k); \\ \mathbf{y}(k) &= \mathbf{C} \cdot \mathbf{x}(k) + \mathbf{D} \cdot \mathbf{u}(k) + \mathbf{e}(k), \end{aligned} \tag{2.34}$$

де $\mathbf{e}(k)$ – вектор збурень у вигляді білого шуму.

Така форма еквівалентна (2.3) де:

$$H(q) = \mathbf{C}(q\mathbf{I_{Nx}} - \mathbf{A_d})^{-1}\mathbf{L_d} + \mathbf{I_{Ny}} \tag{2.35}$$

тут *Ny* – розмірність векторів $\mathbf{y}(k)$ та $\mathbf{e}(k)$.

Часто можливо описати систему безпосередньо в формі (2.34). В інших випадках має сенс спочатку описати характер збурень, що діють на систему. Це призводить до стохастичної моделі в такій формі:

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A_d} \cdot \mathbf{x}(k) + \mathbf{B_d} \cdot \mathbf{u}(k) + \mathbf{w}(k); \\ \mathbf{y}(k) &= \mathbf{C} \cdot \mathbf{x}(k) + \mathbf{D} \cdot \mathbf{u}(k) + \mathbf{e}(k), \end{aligned} \tag{2.36}$$

де $\mathbf{w}(k)$ – стохастичні процеси з деякою коваріацією. Для стаціонарних систем (2.36) є еквівалентом (2.34), якщо матриця $\mathbf{L_d}$ вибрана як усталений фільтр Калмана (див. параграф 3.3.).

Часто виявляється зручним спочатку описати систему в безперервному часі. Зв'язок між системами в дискретному та безперервному часі виду (2.32), (2.34), (2.36) можна встановити за допомогою виразів (1.28). Зв'язок між дискретним ($\mathbf{L_d}$) та безперервним ($\mathbf{L}$) фільтром Калмана визначається як:

$$\mathbf{L_d} = \int_0^T e^{\mathbf{A} \cdot i}\mathbf{L}\, di, \tag{2.37}$$

де $\mathbf{A}$ – перехідна матриця стану безперервної системи; T – період дискретизації.

2.3. Алгоритм ідентифікації систем

Ідентифікація системи будується на попередній інформації, що включає: 1) масиви виміpених значень входів та виходів системи (дані); 2) набір моделей - кандидатів (структури моделей при параметричній ідентифікації); 3) способи оцінювання (методи ідентифікації); 4) критерії вибору конкретної моделі з набору отриманих та методи перевірки адекватності.

Треба добре розуміти, що не існує стандартних процедур, які гарантують отримання цілком правдоподібної моделі. Параметрична ідентифікація це циклічний процес, що полягає у багатократному виборі структури моделі, розрахунку найкращої моделі обраної структури, та оцінці властивостей отриманої моделі.

Приблизний алгоритм параметричної ідентифікації може бути таким:

1) Планування експерименту та формування масивів даних шляхом вимірювань вхідних та вихідних сигналів системи, що підлягає ідентифікації.

2) Можлива попередня обробка отриманих даних (наприклад, фільтрація чи видалення тренду з даних).

3) Визначання структури моделі (з набору моделей-кандидатів) в межах якої буде визначена модель.

4) Розрахунок найкращої моделі обраної структури відповідно до виміpених даних та заданого критерію.

5) Дослідження властивостей отриманої моделі та перевірка її адекватності.

Якщо модель задовольняє поставленим умовам, то процес ідентифікації припиняється; інакше – повторюється 3 етап для випробування іншої структури моделі. Можливо також треба змінити методи оцінювання (етап 4) або повторно сформувати масиви даних (етапи 1 і 2).

Непараметрична ідентифікація має схожий алгоритм, де етапи 3 та 4 замінені безпосередньо на процедуру оцінювання (див. параграф 2.5.).

Бібліотека SIT містить функції для кожного з етапів (див. додаток 1.1.).

2.4. Попередня обробка даних

До початку ідентифікації необхідно візуально оцінити характер вимірених даних. За допомогою функції idplot можна по експерементальним даним побудувати графік залежності виходів та входів системи від часу. Такий графік дозволяє відповісти на запитання:

– Чи впливають зміни вхідного сигналу на вихідний сигнал?

– Чи спостерігаються нелінійні ефекти (такі як різні відгуки на однакові рівні вхідного сигналу)?

– Чи існують частини даних, що виглядають невірогідними або не несуть ніякої інформації? І т.і.

Приклад 2.1.

Побудувати часові графіки вимірених даних D_1 (див. додаток 4.):

```
» idplot(D1)                  % Або idplot([yd ud])
```

Результат побудови зображено на рис. 2.1.

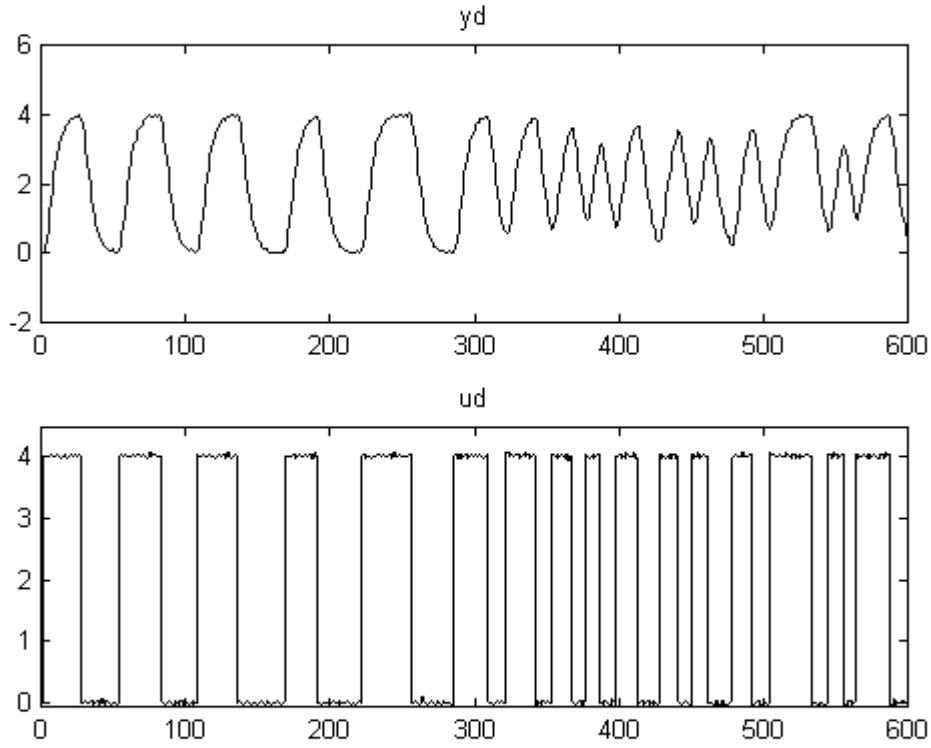

Рис. 2.1.

Кінець приклада.

Функція dtrend дозволяє видаляти лінійний тренд або видаляти середні значення процесу (тренд нульового порядку).

Приклад 2.2.

З рис. 2.1. видно, що дані D_1 не мають лінійного тренду. Для коректного застосування методів ідентифікації видалимо середні значення з вимірювань):

```
» D1d=dtrend(D1);
» idplot(D1d,100:300)     % Графік для відрахунків 100 - 300
```

Результат операції для частини даних зображено на рис. 2.2.

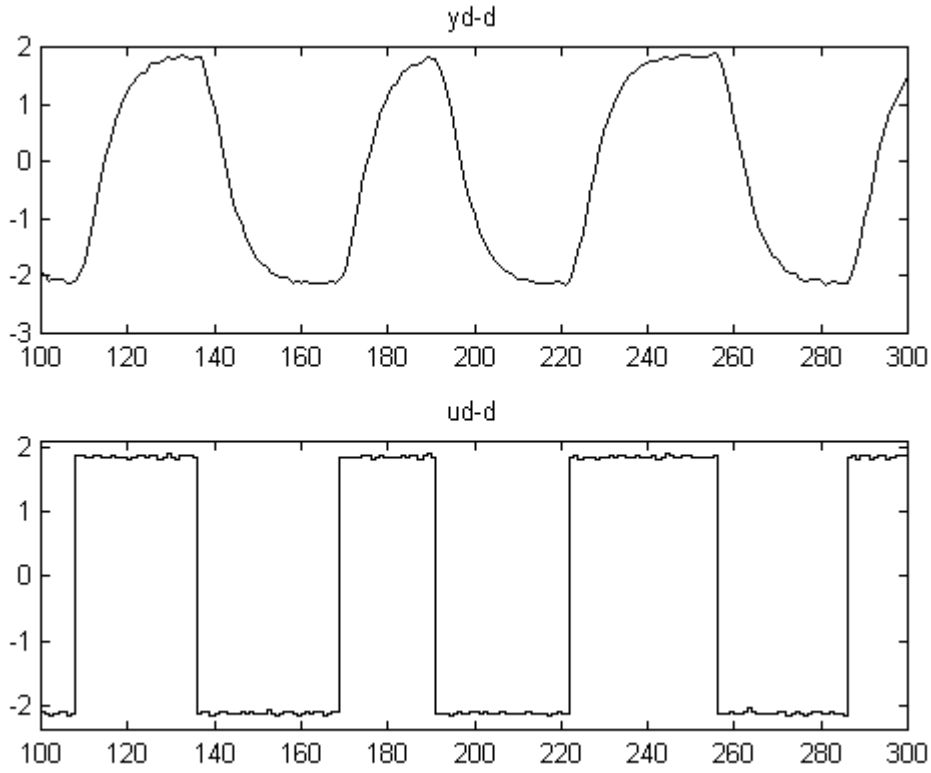

Рис. 2.2.

Кінець приклада.

При наявності високочастотного шуму в системі, його можна спробувати видалити з даних за допомогою функції idfilt.

Змінити період дискретизації даних можна за допомогою функції idresamp. Функція збільшує період дискретизації шляхом видалення зайвих вимірювань, або зменшує період дискретизації шляхом інтерполяції даних.

Далі з масивів даних, треба виділити дві частини (характер яких задовольняє дослідника). Ці підмасиви будуть використовуватись надалі для ідентифікації та перевірки адекватності моделі системи.

Приклад 2.3.

Дані на рис. 2.2. виглядають цілком вірогідними. Розділимо їх на дві приблизно рівні частини, одну з яких (D_{1e}) будемо використовувати для оцінювання моделей, а іншу (D_{1v}) – для перевірки їх адекватності:

```
» D1e=[D1d(1:300,1) D1d(1:300,2)];      % Дані для оцінювання
» D1v=[D1d(301:600,1) D1d(301:600,2)];  % Дані для перевірки
                                        % адекватності
```

Кінець приклада.

2.5. Непараметрична ідентифікація

2.5.1. Пряме оцінювання імпульсного відгуку системи

Таке оцінювання можна зробити за допомогою методу кореляційного аналізу [8, 26]. Припустимо, що входом $u(k)$ системи виду (2.1) є білий шум з коваріаційною функцією (див. (2.12)):

$$R_u(\lambda) = \mathrm{E}[u(k+\lambda)u(k)] = \begin{cases} \sigma_u^2 & \text{ï ðè } \lambda = 0 \\ 0 & \text{ï ðè } \lambda \neq 0 \end{cases}. \qquad (2.38)$$

Тоді взаємна коваріаційна функція між входом $u(k)$ та виходом $y(k)$ визначається рівнянням Вінера-Хопфа (див. (2.14)):

$$R_{yu}(\lambda) = \mathrm{E}[y(k+\lambda)u(k)] = \sigma_u^2 g(\lambda), \qquad (2.39)$$

де $g(\lambda)$ – імпульсний відгук системи на λ-му кроці дискретизації, який можна оцінити як:

$$\hat{g}(\lambda) = \frac{1}{\sigma_u^2 N}\sum_{i=1}^{N} y(i+\lambda)u(i), \qquad (2.40)$$

тут N – кількість вимірювань.

Якщо вхід $u(k)$ не є білим шумом, то його можна "відбілити" за допомогою фільтра:

$$u_e(k) = L(q)u(k). \qquad (2.41)$$

Отримати оцінки $g(k)$ методом кореляційного аналізу можна за допомогою функції cra.

Приклад 2.4.

Отримаємо пряму оцінку імпульсного відгуку системи по вимірюванням D_{1d} (див. приклад 2.2.) для 25 лагів:

```
» ir=cra(D1d,25);
```

Оцінку імпульсної перехідної характеристики системи зображено на рис. 2.3.

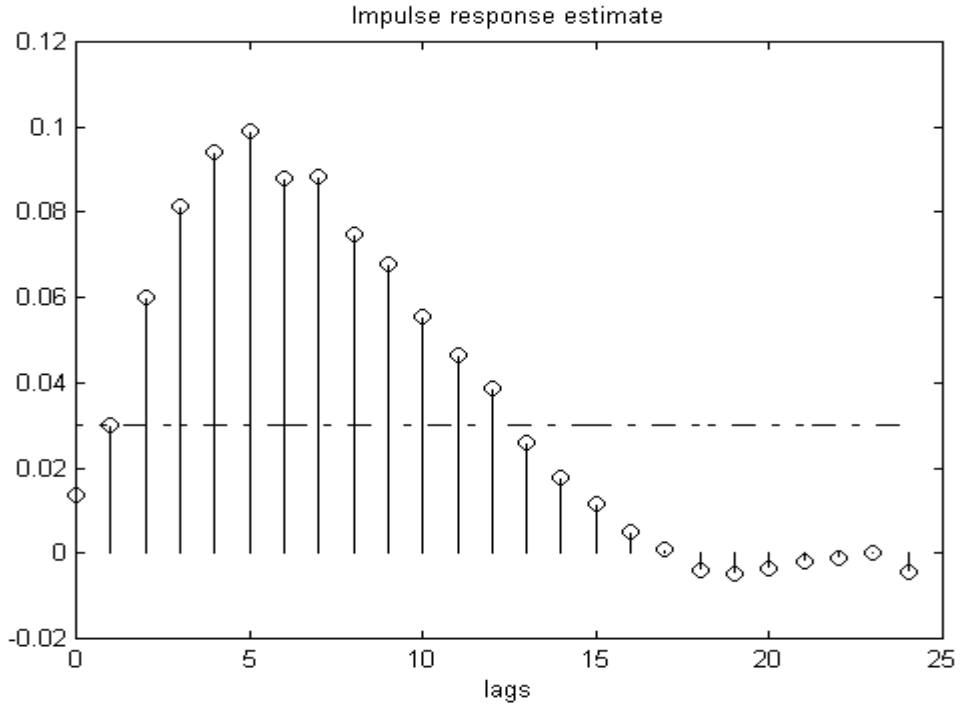

Рис. 2.3.

Побудуємо тепер оцінку перехідної характеристики системи. Результат зображено на рис. 2.4.

```
» srcra=cumsum(ir); stem(srcra)
» title('Перехідна х-ка D1d')
» xlabel('T')
```

Кінець приклада.

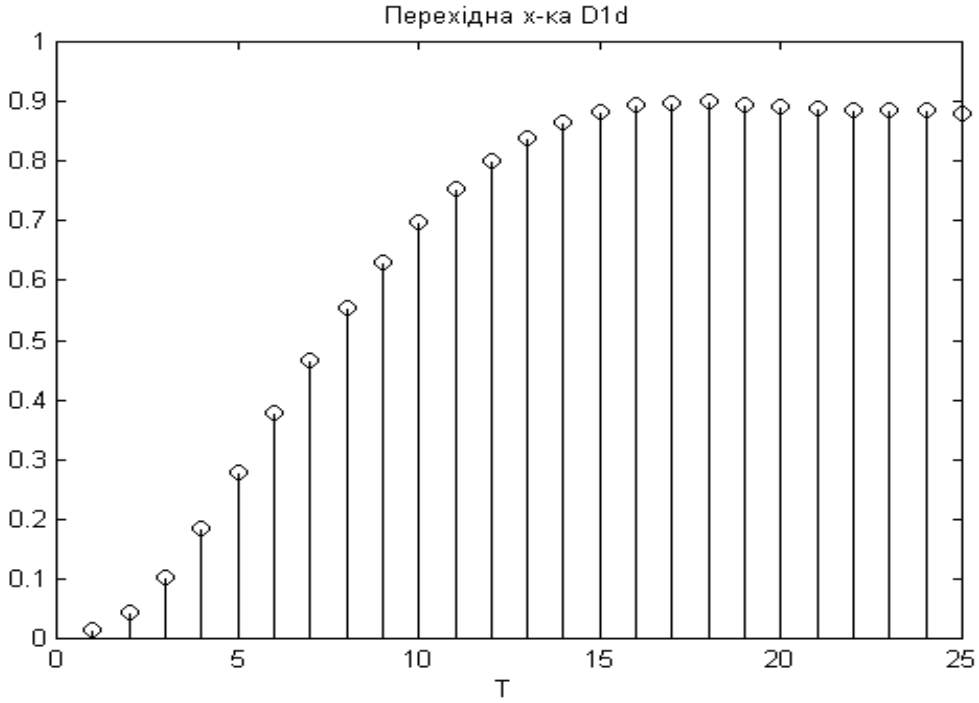

Рис. 2.4.

2.5.2. Пряме оцінювання частотних характеристик системи

За допомогою метода спектрального аналізу можна безпосередньо оцінювати частотну характеристику система (1.15) та спектральну щільність збурення в системі (2.23).

Взаємна спектральна щільність вхідного $u(k)$ та вихідного $y(k)$ процесів системи визначається, як перетворення Фур'є їх взаємної коваріаційної функції (2.39) аналогічно (2.16). Якщо в загальній моделі (2.1) вважати, що процеси $u(k)$ та $v(k)$ незалежні, то можна записати:

$$\Phi_y(\omega) = \left|G(e^{j\omega})\right|^2 \Phi_u(\omega) + \Phi_v(\omega); \tag{2.42}$$

$$\Phi_{yu}(\omega) = G(e^{j\omega})\Phi_u(\omega). \tag{2.43}$$

Процедура оцінювання має таку послідовність [26]: Спочатку треба отримати оцінки коваріаційних функцій $\hat{R}_y(\lambda)$, $\hat{R}_{yu}(\lambda)$ та $\hat{R}_u(\lambda)$ відповідно до (2.10) та (2.12). Далі треба отримати оцінки відповідних спектрів:

$$\hat{\Phi}_y(\omega) = \sum_{\lambda=-m}^{m} \hat{R}_y(\lambda) W_m(\lambda) \cdot e^{-j\omega\cdot\lambda}, \tag{2.44}$$

де $W_m(\lambda)$ – це, так зване, вікно лагу з шириною m.

Оцінки $\hat{\Phi}_u(\omega)$ та $\hat{\Phi}_{yu}(\omega)$ отримують аналогічно (2.44).

Тоді оцінку частотної характеристики системи визначають, як:

$$\hat{G}_N(e^{j\omega}) = \frac{\hat{\Phi}_{yv}(\omega)}{\hat{\Phi}_u(\omega)}, \tag{2.45}$$

а оцінку спектру збурення визначають, як:

$$\hat{\Phi}_v(\omega) = \hat{\Phi}_y(\omega) - \frac{\left|\hat{\Phi}_{yv}(\omega)\right|^2}{\hat{\Phi}_u(\omega)}. \tag{2.46}$$

Провести ідентифікацію методом спектрального аналізу можна за допомогою функції spa. Результат оцінювання системи представляється як модель в ff-формі (див. п. 2.6.2.).

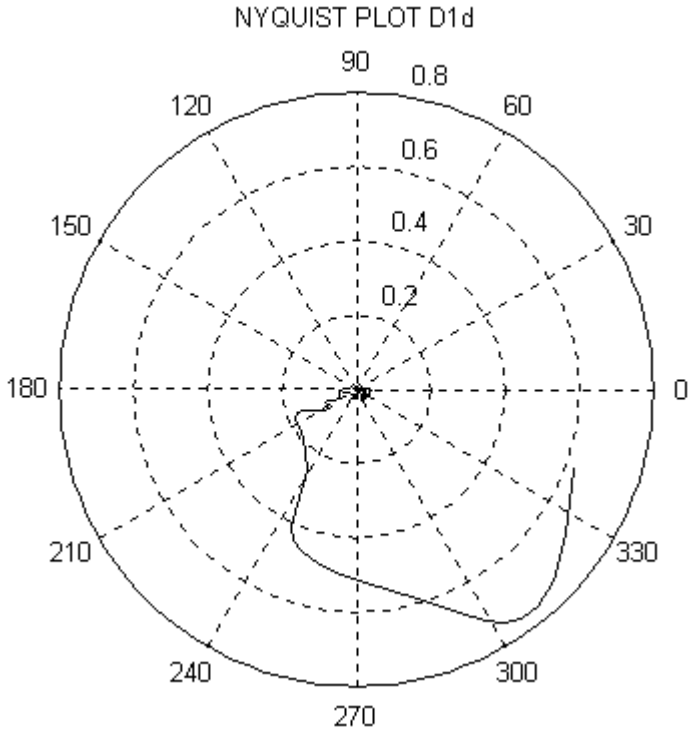

Рис. 2.4.

Приклад 2.5.

Отримаємо пряму оцінку частотних характеристик системи по вимірюванням D_{1d} (див. попередній приклад), та представимо їх у вигляді годографа Найквіста (див. п. 2.8.1.):

```
» Dff=spa(D1d);
» nyqplot(Dff)
```

Результат побудови зображено на рис. 2.4.

Кінець приклада.

2.6. Форми та структури моделей

2.6.1. Theta-форма моделей

Основною формою зображення моделей в SIT є, так звана, theta-форма (надалі th). Вона включає інформацію про структуру та значення параметрів в термінах моделей (2.24) або (2.34). Крім того вона включає інформацію про оцінку стандартних відхилень параметрів. Отримати інформацію про модель в th-формі можна за допомогою функції present.

Моделі в th-формі є результатом параметричної ідентифікації. Крім того для аналізу моделі засобами SIT th-форму можна створити штучно за допомогою функції poly2th.

Зворотне перетворення, тобто отримання з th-форми поліномів моделі (2.24), можна зробити за допомогою функції th2poly.

Для представлення моделі в zpk та tf формах (див. п. 1.2.1.) використовуються функції th2zp та th2tf. Для представлення моделі в ss-формs (див. п. 1.2.4.) використовується функція th2ss.

Приклад 2.6.

Представимо дискретну модель з періодом дискретизації $T = 1$

$$G_1(q) = q^2 \frac{-0.11q + 0.14q^2}{1 - 1.66q + 0.687q^2} = \frac{-0.11q^3 + 0.14q^4}{1 - 1.66q + 0.687q^2} = \frac{-0.11z + 0.14}{z^4 - 1.66z^3 + 0.687z^2}$$

(див. приклад 1.12.) в th-формі. Вектори-рядки параметрів поліномів (2.24) мають вид: [a_0 a_1 a_2] = [1 –1.66 0.687]; [b_0 b_1 b_2 b_3 b_4] = [0 0 0 –0.11 0.14]; $C = 1$; $D = 1$; $F = 1$, а дисперсію білого шуму вважаємо $\sigma_e^2 = 0$:

```
» Gth=poly2th([1 -1.66 0.687],[0 0 0 -0.11 0.14],1,1,1,0,1);
» present(Gth)
Loss fcn: 1   0 Sampling interval 1
```

```
The polynomial coefficients and their standard deviations are
B =
         0         0         0   -0.1100    0.1400
         0         0         0         0         0
A =
    1.0000   -1.6600    0.6870
         0         0         0
```

Отримаємо з моделі в th-формі (SIT) передатну функцію в tf-формі (CST) (див. п. 1.2.2.):

```
» [num,den]=th2tf(Gth)
num =
         0         0         0   -0.1100    0.1400
den =
    1.0000   -1.6600    0.6870         0         0
» tf(num,den,1)
Transfer function:
      -0.11 z + 0.14
--------------------------
z^4 - 1.66 z^3 + 0.687 z^2
Sampling time: 1
```

Або через оператор зсуву назад:

```
» [a,b,no,no,no,no,T]=th2poly(Gth);
» a,b,T
a =
    1.0000   -1.6600    0.6870
b =
         0         0         0   -0.1100    0.1400
T = 1
» tf(b,a,T,'var','q')
Transfer function:
 -0.11 q^3 + 0.14 q^4
----------------------
1 - 1.66 q + 0.687 q^2
Sampling time: 1
```

Отримаємо тепер з моделі в th-формі (SIT) модель в просторі станів (ss-форма CST) (див. п. 1.2.4.) і для перевірки перетворимо її tf-форму:

```
» [a,b,c,d]=th2ss(Gth)
a =
    1.6600    1.0000         0         0
   -0.6870         0    1.0000         0
         0         0         0    1.0000
         0         0         0         0
b =
         0
         0
   -0.1100
    0.1400
c =
     1     0     0     0
d =  0
» tf(ss(a,b,c,d,1))
Transfer function:
      -0.11 z + 0.14
-------------------------
z^4 - 1.66 z^3 + 0.687 z^2
Sampling time: 1
```

Кінець приклада.

2.6.2. Моделі у вигляді частотних функцій

Ідентифікація методом спектрального аналізу дає модель у вигляді масивів даних для побудови частотних характеристик (1.15) та спектру (2.23) системи для 128 значень частоти, що рівномірно розподілені в діапазоні $0 \le \omega \le \omega_N$, де $\omega_N = \pi/T$ – частота Найквіста (див. приклад 2.5.). Крім того модель включає оцінки стандартних відхилень даних. Така форма представлення моделей в SIT називається ff-формою (frequency function).

Для перетворення моделі з th-форми в ff-форму використовується функція th2ff. Таке перетворення роблять для дослідження моделі частотними методами (див. п. 2.8.1.).

2.6.3. Перетворення типу безперервний / дискретний час

Для SIT природним є робота з дискретними моделями. Такі моделі є результатом:

– Оцінювання за допомогою функцій ar, armax, arx, bj, ivar, iv4, n4sid та oe (див. параграф 2.7.).

– Оцінювання за допомогою функції pem, коли структура моделі задана в просторі станів за допомогою функції ms2th (або mf2th) з відповідним параметром ‘d’ (див. п. 2.6.4.) або в поліноміальній формі (2.24).

– Коли модель сформована за допомогою функції poly2th з додатним періодом дискретизації (див. п. 2.6.1.).

Модель в th-формі може бути представлена також в безперервному часі. Така ситуація виникє, коли:

– Структура моделі задана в просторі станів за допомогою функції ms2th (або mf2th) з відповідним параметром ‘c’ (див. п. 2.6.4.) та модель оцінюється за допомогою функції pem (навіть при використанні дискретних даних).

– Модель сформована за допомогою функції poly2th з від’ємним періодом дискретизації.

– Модель примусово перетворено в безперервний час за допомогою функції thd2thc.

Всі перетворення моделей в безперервному часі (див. пп. 2.6.1. та 2.6.2.) дають також безперервні моделі. Абсолютне значення періоду дискретизації моделі в безперервному часі є періодом дискретизації даних, що використовувались для її оцінки.

Для дослідження моделі в безперервному часі засобами SIT, її необхідно перетворити в дискретну за допомогою функції thc2thd.

2.6.4. Визначення структури моделі

Структура поліноміальної моделі при оцінюванні засобами SIT задається безпосередньо при виклику відповідної функції (див. параграф 2.7.) у вигляді вектору порядків поліномів та початкових умов оцінювання.

Крім того, структуру поліноміальної моделі можна задати у вигляді моделі в th-формі за допомогою функції poly2th (див. п. 2.6.1. та приклад 2.6.). В цьому випадку значення параметрів моделі розглядаються як початкові умови оцінювання.

Структура моделі в просторі станів виду (2.34) при оцінюванні методом підпростору (див. п. 2.7.5.) задається її порядком при виклику функції n4sid.

Задати довільну структуру моделі в просторі станів для оцінювання методом прогнозованої помилки (див. п. 2.7.2.) можна в th-формі за допомогою функцій: modstruc, яка формує матрицю структури, та функції ms2th, яка формує з матриці модель в th-формі. При визначенні структури в матрицях параметри, що підлягають оцінюванню, позначаються як *NaN* (стандартне позначення невизначеності в MatLab).

Приклад 2.7.

Припустимо, що структура дискретної моделі 'd' з періодом дискретизації $T = 1$ в просторі станів, (див. (2.34) та попередній приклад), має шість невідомих параметрів:

$$\mathbf{x}(k+1) = \begin{bmatrix} \theta_1 & 1 & 0 & 0 \\ \theta_2 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \mathbf{x}(k) + \begin{bmatrix} 0 \\ 0 \\ \theta_3 \\ \theta_4 \end{bmatrix} \cdot \mathbf{u}(k) + \begin{bmatrix} \theta_5 \\ \theta_6 \\ 0 \\ 0 \end{bmatrix} \cdot \mathbf{e}(k);$$

$$\mathbf{y}(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \cdot \mathbf{x}(k) + \mathbf{e}(k); \qquad \mathbf{x}(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

```
» A=[NaN 1 0 0;NaN 0 1 0;0 0 0 1;0 0 0 0];
» B=[0;0;NaN;NaN];
```

```
» C=[1 0 0 0];
» L=[NaN;NaN;0;0];
» x0=[0;0;0;0];
» Gss=ms2th(ms,'d',[],[],1);
» present(Gss)
Loss fcn: 1 Discrete time model with sampling interval 1
The state-space matrices with standard deviations given as
imaginary parts are
a =
        0 + 0.0000i   1.0000                  0                   0
        0 + 0.0000i        0             1.0000                   0
        0                  0                  0             1.0000
        0                  0                  0                   0
b =
  1.0e-013 *
        0
        0
        0 + 0.2220i
        0 + 0.2220i
c =
     1     0     0     0
d =  0
k =
  1.0e-013 *
        0 + 0.2220i
        0 + 0.2220i
        0
        0
x0 =
     0
     0
     0
     0
lambda = 1
```

Отримано фіксовану структуру з початковими значеннями параметрів, що підлягають оцінюванню методом підпростору або методом прогнозованої помилки.

Кінець приклада.

Для формування матриці структури в канонічній формі треба замість modstruc використовувати функцію canform.

2.7. Методи параметричної ідентифікації систем

2.7.1. Метод максимальної правдоподібності

Якщо вважати вихід системи $y(k)$ випадковим стаціонарним процесом, то швидкість зміни його функції розподілення визначається функцією щільності ймовірності виду:

$$p(y,\theta) = \lim_{\Delta y \to 0} \frac{\text{prob}\,[y(k) \in \Delta y]}{\Delta y}, \qquad (2.47)$$

яка залежить від невідомих параметрів моделі системи θ.

Суть метода максимальної правдоподібності полягає в підборі таких параметрів θ, щоб максимізувати функцію правдоподібності (2.47).

Для того, щоб записати вираз щільності розподілення імовірності (2.47) необхідно мати аналітичне представлення закону розподілення процесу [9].

2.7.2. Метод прогнозованої помилки

У випадках, коли збурення в лінійній системі є нормально-розподіленим (гауссовим) процесом, метод максимальної правдоподібності аналогічен методу прогнозованої помилки (prediction error method).

Якщо вважати, що система описується загальною моделлю виду (2.3), то при наявності спостережуваних процесів $u(k)$ та $y(k)$ прогнозовані помилки системи $e(k)$ можна розрахувати за формулою:

$$e(k) = H^{1}(q)(y(k) - G(q)u(k)) \qquad (2.48)$$

Суть метода полягає у визначенні оцінок $\hat{G}$ та $\hat{H}$ шляхом мінімізації функції втрат виду:

$$J_N(G, H) = \sum_{i=1}^{N} e^2(i) \tag{2.49}$$

що означає:

$$[\hat{G}_N, \hat{H}_N] = \text{argmin} \sum_{i=1}^{N} e^2(i) \tag{2.50}$$

Функція втрат (2.49) в загальному випадку нелінійна, тому для вирішення (2.50) застосовують один з методів нелінійного безумовного пошуку.

Базуючись на методі (2.50) можна оцінювати моделі будь-якої структури. Для моделі в загальній формі (2.24) оцінку методом прогнозованої помилки можна здійснити за допомогою функції pem, яка використовує метод нелінійного пошуку Гаусса-Ньютона [7].

Приклад 2.8.

Отримаємо по результатам вимірювань D_{1e} (див. приклад 2.3.) оцінку моделі ОЕ (2.29) зі структурою: *Nb* = 2; *Nf* = 3; *Nk* = 1, методом прогнозованої помилки. Вектор структури параметрів загальної моделі (2.24) в цьому випадку має вид: [*Na Nb Nc Nd Nf Nk*] = [0 2 0 0 3 1].

```
» G1oe=pem(D1e,[0 2 0 0 3 1]);
» present(G1oe)
Loss fcn: 0.04213   Akaike`s FPE: 0.043558 Sampling interval 1
The polynomial coefficients and their standard deviations are
B =
         0    0.0388   -0.0032
         0    0.0129    0.0931
F =
    1.0000   -1.8695    1.0858   -0.1799
         0    1.7582    2.8281    1.1534
```

Кінець приклада.

Крім того для деяких часткових випадків загальної моделі більш ефективним може виявитись використання спеціальних функцій:

– для моделі ARMAX (авторегресії з ковзним середнім (2.26)) застосовують функцію armax;

– для моделі OE (Output-Error (2.29)) застосовують функцію oe;

– для моделі BJ (Box-Jenkins (2.30)) застосовують функцію bj.

Всі ці функції також оцінюють моделі систем з кількома входами (див. (2.31)).

Приклад 2.9.

Отримаємо оцінку інших типів моделей (див. п. 2.2.4.).

Для моделі OE тієї ж структури (див. попередній приклад) вектор структури параметрів має вид: [*Nb Nf Nk*] = [2 3 1].

```
» G2oe=oe(D1e,[2 3 1]);
» present(G2oe)
Loss fcn: 0.0042008   Akaike`s FPE: 0.0043431 Sampling interval 1
The polynomial coefficients and their standard deviations are
B =
         0    0.0385    0.0099
         0    0.0021    0.0023
F =
    1.0000   -1.6311    0.7169   -0.0369
         0    0.0260    0.0470    0.0218
```

Порівняйте з результатом в попередньому прикладі.

Отримаємо оцінку моделі ARMAX (2.26) зі структурою: *Na* = 3; *Nb* = 3; *Nc* = 2; *Nk* = 1. Вектор структури параметрів в цьому випадку має вид:

[*Na Nb Nc Nk*] = [3 3 2 1].

```
» Garmax=armax(D1e,[3 3 2 1]);
» present(G1armax)
Loss fcn: 0.0021553   Akaike`s FPE: 0.0022734 Sampling interval 1
The polynomial coefficients and their standard deviations are
B =
         0    0.0374   -0.0288   -0.0091
         0    0.0036    0.0082    0.0049
A =
    1.0000   -2.6204    2.2843   -0.6645
         0    0.0184    0.0350    0.0169
```

```
C =
    1.0000   -1.6970    0.7444
         0    0.0363    0.0342
```

Отримаємо тепер оцінку моделі BJ (2.30) зі структурою: *Nb* = 3; *Nc* = 3; *Nd* = 2; *Nf* = 2; *Nk* = 1. Вектор структури параметрів в цьому випадку має вид: [*Nb Nc Nd Nf Nk*] = [3 3 2 2 1].

```
» Gbj=bj(D1e,[3 3 2 2 1]);
» present(Gbj)
Loss fcn: 0.0012429   Akaike`s FPE: 0.0013286 Sampling interval 1
The polynomial coefficients and their standard deviations are
B =
         0    0.0447   -0.0055    0.0130
         0    0.0029    0.0053    0.0044
F =
    1.0000   -1.5703    0.6234
         0    0.0181    0.0159
C =
    1.0000   -0.5950   -0.1094    0.4033
         0    0.0565    0.0642    0.0536
D =
    1.0000   -1.2643    0.3649
         0    0.0212    0.0192
```

Кінець приклада.

2.7.3. Метод найменших квадратів

Для ARX-моделі (2.25), що є частковим випадком поліноміальної моделі (2.24), задача (2.50) має аналітичне рішення [9].

Якщо (2.1) розглядати як рівняння авторегресії, то його можна записати в такій формі:

$$y(k) = \boldsymbol{\phi}^{\mathrm{T}}(k)\boldsymbol{\theta} + e(k), \qquad (2.51)$$

де вектор вимірювань визначається як:

$$\boldsymbol{\phi}^{\mathrm{T}}(k) = [\ -y(k-1)\ \ldots\ -y(k-Na)\ u(k-Nk)\ u(k-Nk-1)\ \ldots\ u(k-Nk-Nb+1)\ ], \qquad (2.52)$$

а вектор параметрів, що підлягають оцінюванню, визначається як:

$$\boldsymbol{\theta}^{\mathrm{T}} = [\, a_1\, a_2\, \ldots\, a_{Na}\, b_1\, b_2\, \ldots\, b_{Nb}\,]. \qquad (2.53)$$

При наявності серії з N вимірювань (2.51) дає систему N лінійних рівнянь, які можна записати у векторно-матричній формі:

$$\mathbf{y} = \boldsymbol{\Omega}\cdot\boldsymbol{\theta} + \mathbf{e}, \qquad (2.54)$$

де вектор вимірених входів має вид:

$$\mathbf{y}^{\mathrm{T}} = [\, y(1)\, y(2)\, \ldots\, y(N)\,]; \qquad (2.55)$$

матриця вимірювань має вид:

$$\boldsymbol{\Omega} = [\boldsymbol{\phi}^{\mathrm{T}}(i)], \quad \text{де } i = \overline{1,\, N}, \qquad (2.56)$$

а **e** – це вектор збурень, що не вимірюються.

Якщо вектор збурень **e** розглядати, як залишкові похибки системи:

$$\mathbf{e} = \mathbf{y} - \boldsymbol{\Omega}\cdot\hat{\boldsymbol{\theta}}, \qquad (2.56)$$

то функцію втрат (2.49) можна записати у вигляді:

$$J_N = \mathbf{e}^{\mathrm{T}}\mathbf{e}, \qquad (2.58)$$

і задача (2.50) приймає вид:

$$\frac{\partial J_N}{\partial \boldsymbol{\theta}} = -\boldsymbol{\Omega}^{\mathrm{T}}[y - \boldsymbol{\Omega}\cdot\boldsymbol{\theta}] = 0\,. \qquad (2.59)$$

Задача (2.59) має аналітичне рішення:

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{\Omega}^{\mathrm{T}}\boldsymbol{\Omega})^{-1}\boldsymbol{\Omega}^{\mathrm{T}}\mathbf{y}\,. \qquad (2.60)$$

Такий спосіб оцінювання називається методом найменших квадратів (least-squares method). В SIT він реалізован у вигляді функції arx.

Приклад 2.10.

Отримаємо за результатами вимірювань D_{1e} (див. приклад 2.3.) оцінку моделі ARX (2.25) зі структурою: Na = 3; Nb = 2; Nk = 1, методом найменших квадратів. Вектор структури параметрів моделі в цьому випадку має вигляд:

[*Na Nb Nk*] = [3 2 1].

```
» Garx=arx(D1e,[3 2 1]);
» present(Garx)
Loss fcn: 0.0021937   Akaike`s FPE: 0.002268 Sampling interval 1
The polynomial coefficients and their standard deviations are
```

```
B =
         0    0.0384    0.0296
         0    0.0039    0.0050
A =
    1.0000   -1.1958    0.0293    0.2353
         0    0.0529    0.0860    0.0399
```

Кінець приклада.

2.7.4. Метод допоміжної змінної

Якщо вимірювані сигнали та збурення (шум) в системі (див. (2.54)) корельовані:

$$\lim_{N\to\infty}\frac{1}{N}\mathbf{\Omega}^{\mathrm{T}}\mathbf{e}\neq 0, \tag{2.61}$$

то для оцінювання методом найменших квадратів необхідно знати розподілення шуму.

Дещо інший підхід можна застосовувати для ARX-моделей (2.25), якщо розглядати вхід системи як сигнал, що пройшов через формуючий фільтр:

$$N(q)s(k) = M(q)u(k) \tag{2.62}$$

і став некорельованим з шумом системи але залишився корельованим з корисними сигналами системи. Такий підхід називається методом допоміжної змінної $s(k)$ (instrumental variable method) [9].

Суть метода полягає в припущенні, що можна підібрати таку матрицю $\mathbf{\Psi}$, що:

$$\lim_{N\to\infty}\frac{1}{N}\mathbf{\Psi}^{\mathrm{T}}\mathbf{e}=0. \tag{2.63}$$

Тоді (2.54) можна переписати у вигляді:

$$\mathbf{\Psi}^{\mathrm{T}}\mathbf{y} = \mathbf{\Psi}^{\mathrm{T}}\cdot\mathbf{\Omega}\cdot\boldsymbol{\theta} + \mathbf{\Psi}^{\mathrm{T}}\mathbf{e}, \tag{2.64}$$

і оскільки $\mathbf{\Psi}^{\mathrm{T}}\mathbf{e}\to 0$, рішенням задачі (2.59) буде:

$$\hat{\boldsymbol{\theta}} = (\mathbf{\Psi}^{\mathrm{T}}\mathbf{\Omega})^{-1}\mathbf{\Psi}^{\mathrm{T}}\mathbf{y}, \tag{2.65}$$

де матрицю допоміжної змінної $\mathbf{\Psi}$ можна представити, наприклад, так:

$$\mathbf{\Psi} = [\boldsymbol{\psi}^{\mathrm{T}}(i)],\ \text{ де } i=\overline{1,\,N}, \tag{2.66}$$

де вектор вимірювань формується як:

$$\boldsymbol{\psi}^{\mathrm{T}}(k) = [\, -s(k-1) \ldots -s(k-Na)\; u(k-Nk)\; u(k-Nk-1) \ldots u(k-Nk-Nb+1)\,]. \quad (2.67)$$

В SIT метод допоміжної змінної реалізован у вигляді функції iv4.

Приклад 2.11.

Вирішемо задачу з попереднього приклада методом допоміжної змінної:

```
» Giv=iv4(D1e,[3 2 1]);
» present(Giv)
Loss fcn: 0.0027653   Akaike`s FPE: 0.002859 Sampling interval 1
The polynomial coefficients and their standard deviations are
B =
         0    0.0420   -0.0150
         0    0.0041    0.0183
A =
    1.0000   -2.0571    1.3888   -0.3044
         0    0.3779    0.6279    0.2663
```

Порівняйте з результатом в попередньому прикладі.

Кінець приклада.

2.7.5. Метод підпростору

Для оцінювання моделей в просторі станів використовують групу методів підпростору (subspace methods) [26].

Матриці моделі в просторі станів $\mathbf{A_d}$, $\mathbf{B_d}$, **C**, **D**, та $\mathbf{L_d}$ з (2.34) можуть бути ефективно оцінені безпосередньо, без попереднього визначення будь-яких специфічних параметрів методом підпростору, ідею якого можна пояснити так: Якщо послідовність векторів стану $\mathbf{x}(k)$ відома, то разом з векторами $\mathbf{y}(k)$ та $\mathbf{u}(k)$, друге рівняння (2.34) є рівнянням лінійної регресії, та **C** і **D** можуть бути оцінені методом найменших квадратів. Тоді $\mathbf{e}(k)$ можна визначити та розглядати як відомий сигнал у першому рівнянні (2.34), яке також стає рівнянням лінійної регресії для $\mathbf{A_d}$, $\mathbf{B_d}$ та $\mathbf{L_d}$.

Можна також вважати (2.32) рівнянням лінійної регресії для $\mathbf{A_d}$, $\mathbf{B_d}$, **C**, і **D** з двома виходами $\mathbf{y}(k)$ і $\mathbf{x}(k+1)$, та знайти сумісний процес і шуми вимірювань

як остаткові похибки від цієї регресії. Фільтр Калмана $\mathbf{L_d}$ можна тоді визначити з рівняння Ріккаті (див. параграф 3.3.).

Таким чином, як тільки визначені стани, оцінювання матриць простору станів стає легкою задачею. Відомо [26], що всі стани $\mathbf{x}(k)$ в моделі, подібній до (2.34) можуть бути сформовані як лінійні комбінації прогнозованих на i кроків вперед виходів системи ($i = \overline{1, N}$). Таким чином, задача зводиться до визначення прогнозованих значень та вибору базису серед них.

Методи підпростору дають ефективний та надійний шлях визначення прогнозів прямою проекцією на послідовність спостерігаємих даних. В SIT оцінити модель в просторі станів методом підпростору можна за допомогою функції n4sid.

Приклад 2.12.

Отримаємо оцінку моделі третього порядку з одним виходом в просторі станів при нульових початкових умовах по результатам вимірювань D_{1e} методом підпростору:

```
» Gss=n4sid(D1e,3,[],[],[0 1 0]);
» present(Gss)
Loss fcn: 0.001262   Akaike's FPE: 0.0014231 Discrete time model
with sampling interval 1            The state-space matrices are
a =
    0.9152    0.5177    0.1333
   -0.0427    0.6722   -0.8007
    0.0030    0.0356   -0.1993
b =
    0.1882
    0.0749
   -0.0073
c =
    0.4757   -0.6935   -0.4940
d = 0
```

```
k =
    1.3357
    0.1386
   -0.2398
x0 =
    0
    0
    0
lambda = 0.0201
```

Кінець приклада.

Крім того, модель в просторі станів можна також оцінити методом прогнозованої помилки: функція pem (див. пп. 2.6.4. та 2.7.2.).

2.8. Дослідження моделей

2.8.1. Побудова характеристик моделі

Отримати стратегію подальшої ідентифікації, або відібрати кращу оцінку з ряду отриманих дозволяє порівняння характеристик моделей.

Побудувати діаграму нулів та полюсів моделі можна за допомогою функції zpplot. Для цього модель спочатку треба представити в zpk-формі (див. п. 2.6.1.).

Розташування нулів та полюсів в безпосередній близькості на діаграмі (в межах відповідних довірчих інтервалів) говорить про можливість їх взаємного скорочення, що дає модель нижчого порядку. Довірчі інтервали методом підпростору та методом допоміжної змінної не розраховуються.

Приклад 2.13.

Побудувати діаграму нулів та полюсів моделі $G_{1\text{armax}}$ із структурою $Na = 3$; $Nb = 3$; $Nc = 2$; $Nk = 1$ (див. приклад 2.9.):

```
» zpplot(th2zp(G1armax))
» grid on
```

Результати побудови зображені на рис. 2.5.

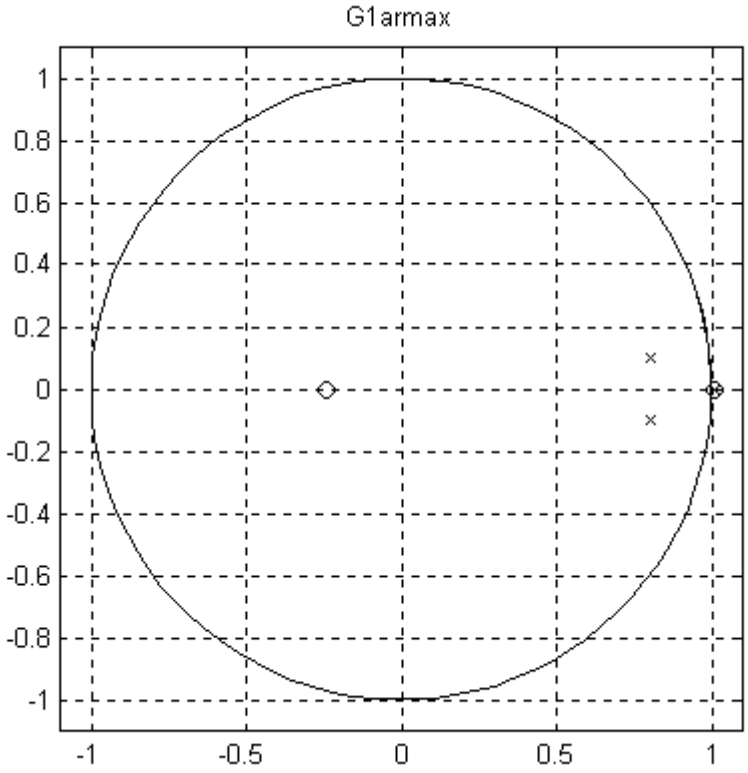

Рис. 2.5.

З діаграми видно, що об’єкт з такою моделлю перебуває на межі стійкості. Крім того, один полюс співпадає з нулем об’єкта, що припускає їх взаємне скорочення. Спробуємо отримати оцінку ARMAX моделі зниженого порядку (зі структурою $Na = 2$; $Nb = 2$; $Nc = 2$; $Nk = 1$) та побудуємо діаграму нулів та полюсів цієї моделі:

```
» G2armax=armax(D1e,[2 2 2 1]);
» present(G2armax)
Loss fcn: 0.0014995   Akaike`s FPE: 0.0015607 Sampling interval 1
The polynomial coefficients and their standard deviations are
B =
         0    0.0396    0.0048
         0    0.0024    0.0031
A =
    1.0000   -1.6247    0.6707
         0    0.0093    0.0083
```

```
C =
    1.0000   -0.9933    0.2569
         0    0.0584    0.0576
```

Для побудови діаграми разом з довірчими інтервалами другим параметром функції zpplot треба вказати їх розмір (у відсотках).

```
» zpplot(th2zp(G2armax),99)
```

Результат побудови зображено на рис. 2.6.

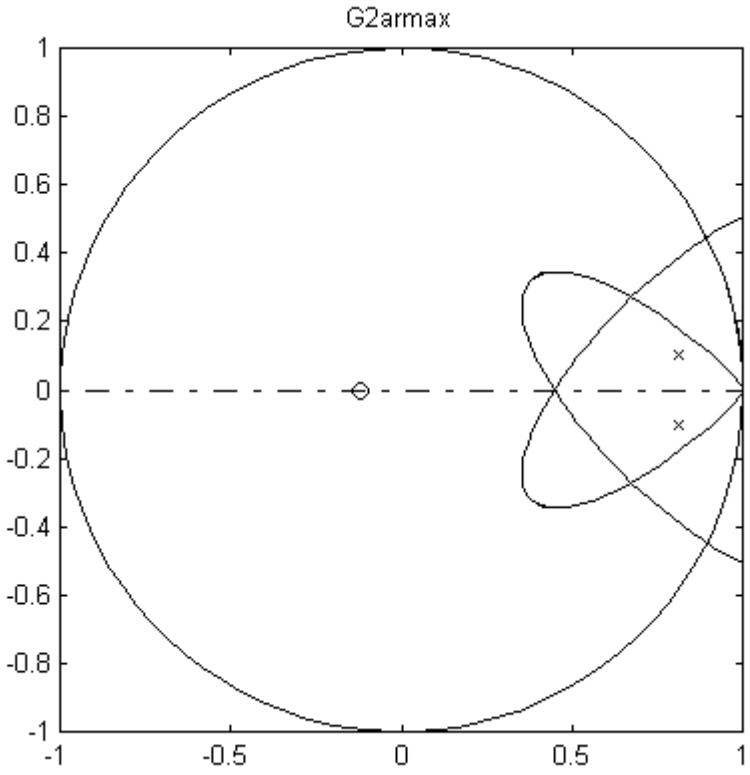

Рис. 2.6.

Кінець приклада.

Розрахувати відгук моделі в часовій області на довільний вхід можна за допомогою функції idsim. В якості входу моделі можна використовувати стандартні сигнали (для побудови перехідних характеристик) або послідовність вимірених даних (для подальшого порівняння з реальним виходом системи). Крім того функція idsim дозволяє враховувати адитивний шум (для імітаційного моделювання).

Приклад 2.14.

Порівняємо вихід моделі ARX (див. приклади 2.10.) з реальним виходом системи (масиви даних D_{1v} див. приклад 2.3.) для перших ста відрахунків:

```
» rarx=idsim(D1v(1:100,2),Garx);
» plot([rarx D1v(1:100,1)])
» xlabel('T')
```

За результатами порівняння (рис. 2.7.) можна сказати, що ARX модель обраної структури загалом непогано описує систему.

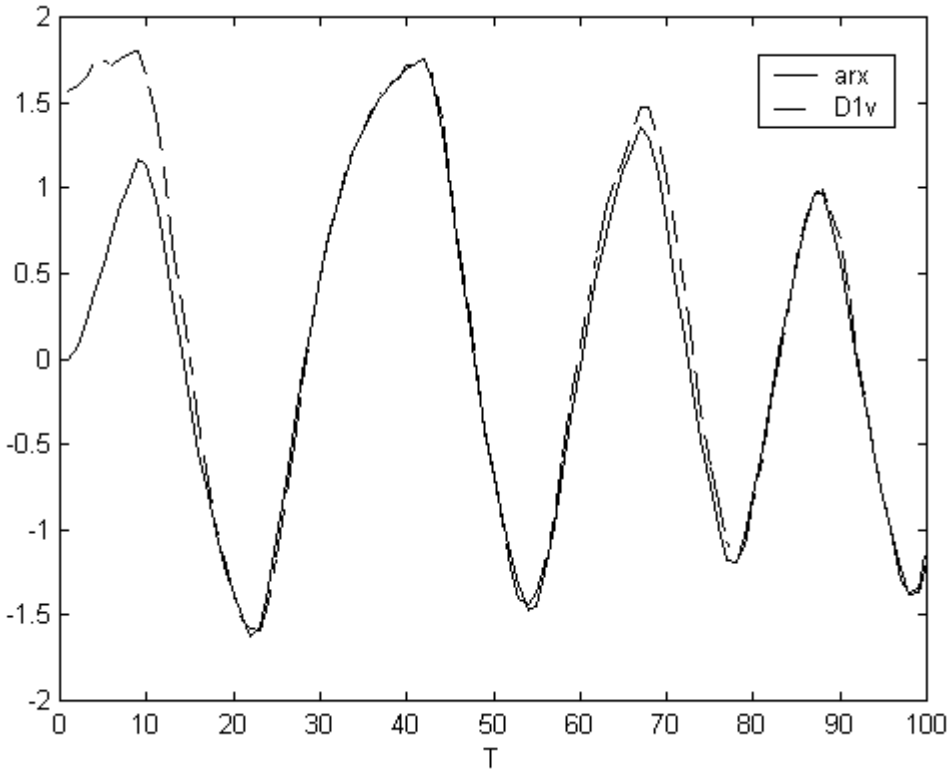

Рис. 2.7.

Побудуємо перехідну характеристику системи за оціненою ARX моделлю та порівняємо її з оцінкою, отриманою методом кореляційного аналізу (див. приклад 2.4.):

```
» st=ones([25 1]);
» srarx=idsim(st,Garx);
» plot([srarx srcra])    % Або stairs([srarx srcra])
» xlabel('T')
» grid on
```

Результат порівняння зображено на рис. 2.8.

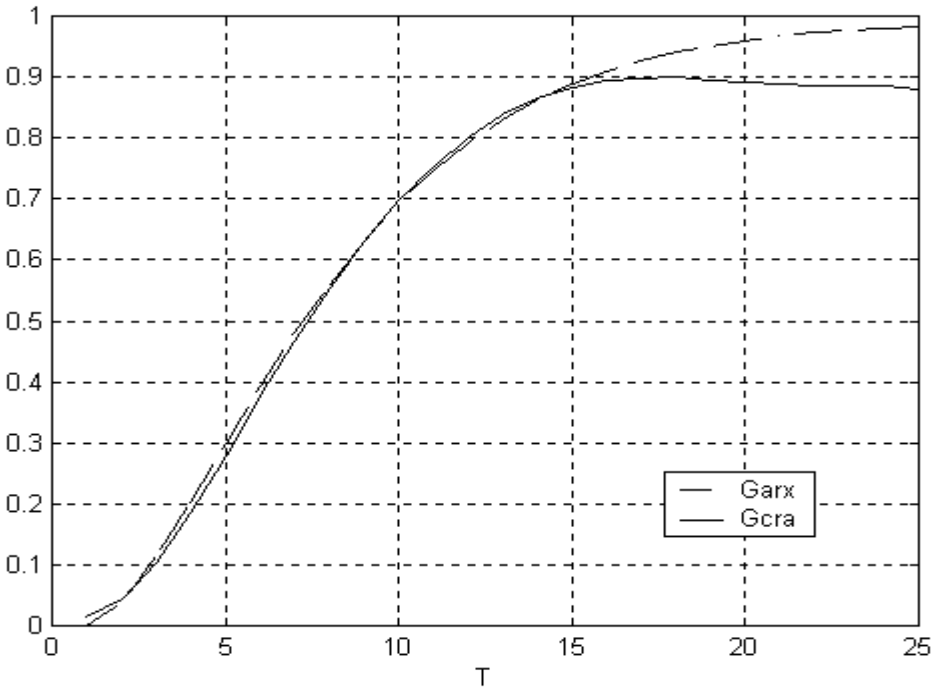

Рис. 2.8.

Кінець приклада.

Різні типи вхідних сигналів для функції idsim можна сформувати за допомогою функції idinput.

Для дослідження моделей в частотній області в SIT існує кілька засобів. Побудувати частотні функції та спектри можна за допомогою функції ffplot, або функції bodeplot (логарифмічні частотні характеристики). Побудувати амплітудно-фазову частотну характеристику можна за допомогою функції nyqplot. Для побудови частотних характеристик модель треба представити в ff-формі (див. п. 2.6.2.).

Приклад 2.15.

Побудуємо спектр та діаграми Боде для моделі ARX обраної структури (див. попередній приклад):

```
» [fr sp]=th2ff(Garx);
» bodeplot([fr sp])
```

Діаграми Боде зображені на рис. 2.9. а автоспектр збурення моделі на рис. 2.10.

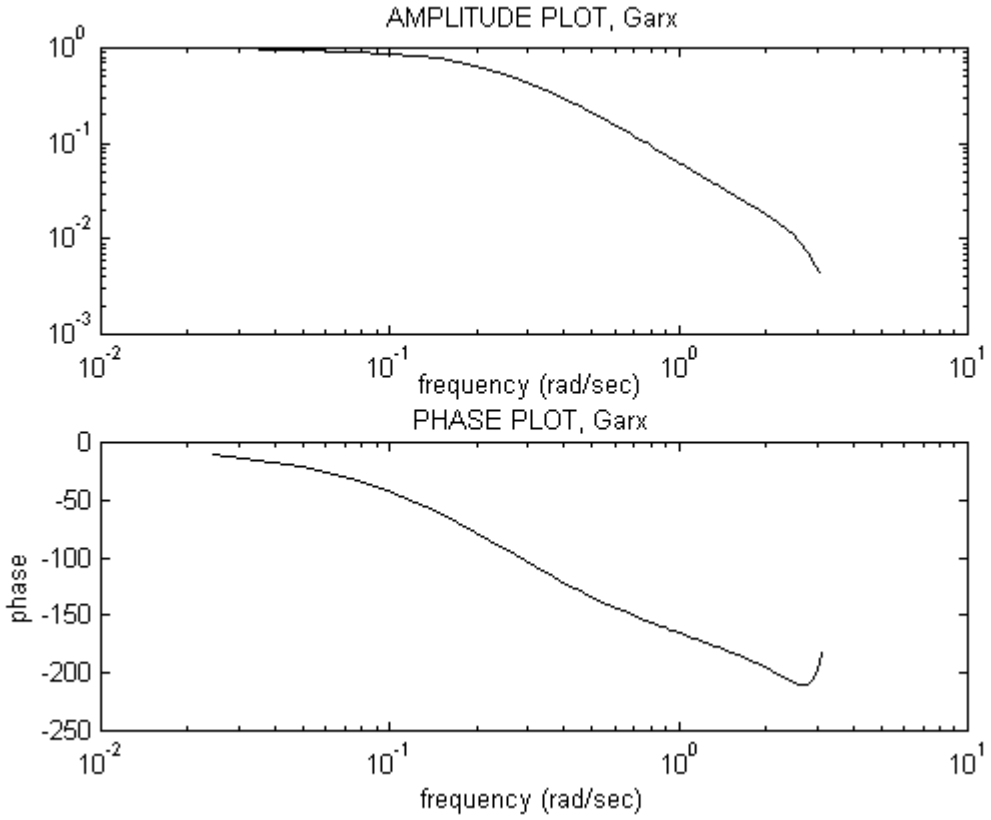

Рис. 2.9.

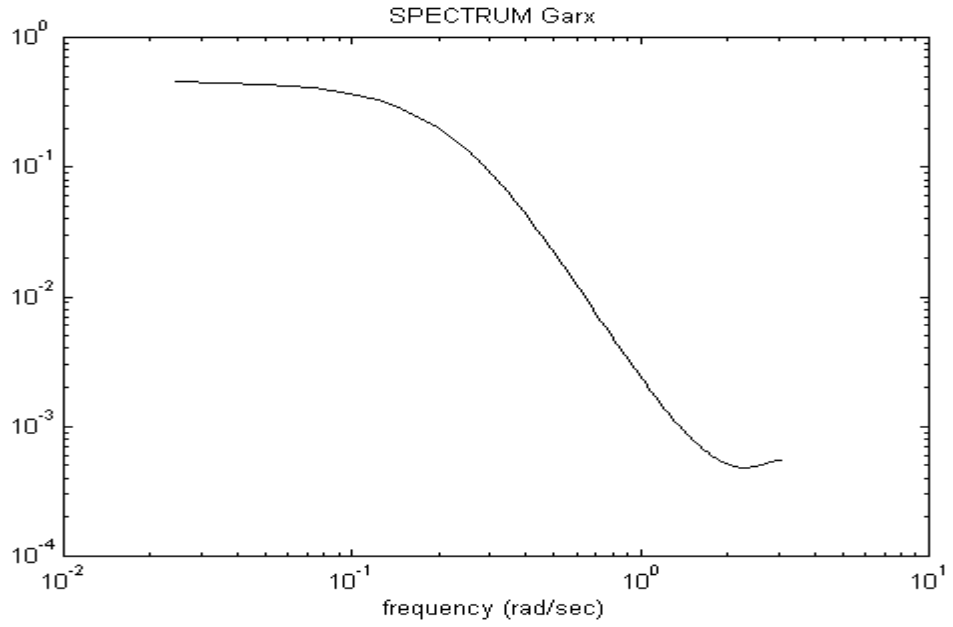

Рис. 2.10.

Кінець приклада.

В прикладі 2.5. представлено спосіб побудови годографа Найквіста.

Якщо кілька різних моделей дають в робочому діапазоні частот досить близькі (в межах довірчого інтервалу) частотні характеристики, то треба вибрати найпростішу модель.

2.8.2. Перевірка адекватності моделі

Найпростіший спосіб визначення якості отриманої моделі – це порівняння відгуків моделі та системи на вхідну послідовність даних, що відрізняється від послідовності, яка використовувалась для оцінювання (див. параграф 2.4.). Таке порівняння можна зробити за допомогою функції idsim (див. п. 2.8.1. та приклад 2.14.) або функції compare.

Остання функція також розраховує збіг (fit δ) між виміреним виходом системи та предбаченим виходом моделі у вигляді:

$$\delta = \frac{1}{N}\sum_{i=1}^{N}(y(i)-\hat{y}(i))^2 . \qquad (2.68)$$

Приклад 2.16.

Порівняємо вихід моделі ARX (2.25) зі структурою: $Na = 3$; $Nb = 2$; $Nk = 1$, що оцінена методом найменших квадратів, та реальний вихід системи (див. приклад 2.14.) з кількісним визначенням збігу δ:

```
» compare(D1v,Garx);
```

Результат порівняння зображено на рис. 2.11.

Визначимо також збіг для моделей OE зі структурою: $Nb = 2$; $Nf = 3$; $Nk = 1$ (див. приклади 2.8. та 2.9.), що оцінені різними способами, моделі ARX зі структурою: $Na = 3$; $Nb = 2$; $Nk = 1$, що оцінена методом допоміжної змінної (див. приклад 2.11.), моделей ARMAX (2.26) зі структурою: $Na = 3$; $Nb = 3$; $Nc = 2$; $Nk = 1$; та $Na = 2$; $Nb = 2$; $Nc = 2$; $Nk = 1$; (див. приклади 2.9. та 2.13.), моделі BJ (2.30) зі структурою: $Nb = 3$; $Nc = 3$; $Nd = 2$; $Nf = 2$; $Nk = 1$ (див. приклад 2.9.), та моделі третього порядку з нульовими початковими умовами в просторі станів (див. приклад 2.12.):

```
» [no f1oe]=compare(D1v,G1oe); f1oe
f1oe =   0.2293
» [no f2oe]=compare(D1v,G2oe); f2oe
```

```
f2oe =   0.2322
» [no fiv]=compare(D1v,Giv); fiv
fiv =    0.2310
» [no f1armax]=compare(D1v,G1armax); f1armax
f1armax = 0.2298
» [no f2armax]=compare(D1v,G2armax); f2armax
f2armax = 0.2293
» [no fbj]=compare(D1v,Gbj); fbj
fbj =    0.2299
» [no fss]=compare(D1v,Gss); fss
fss =    0.2303
```

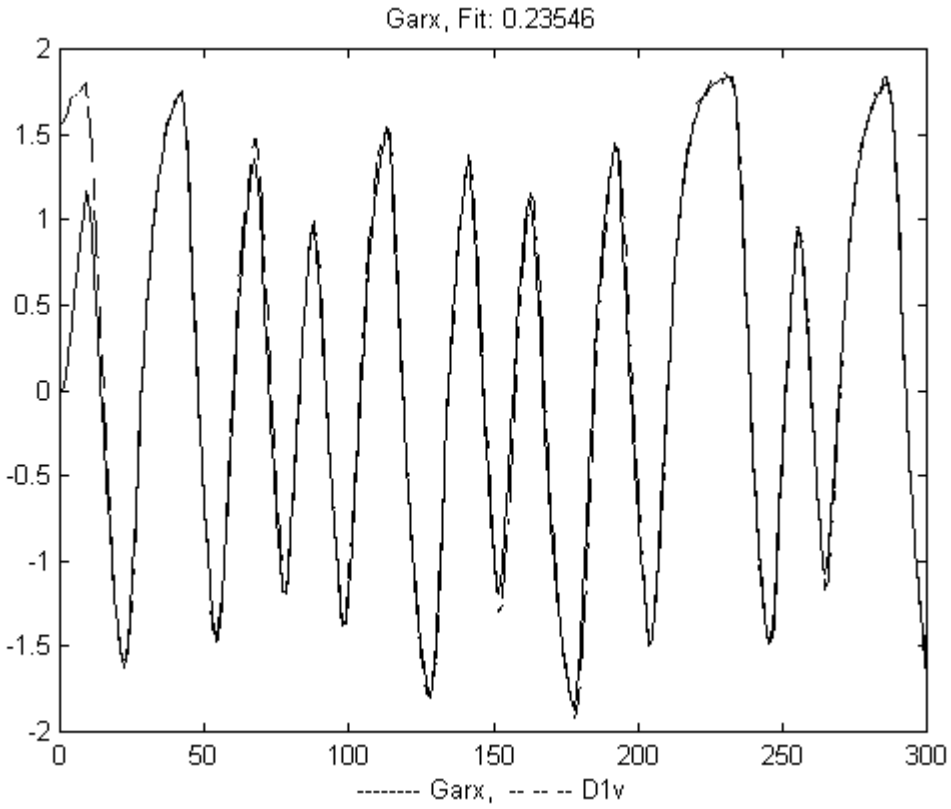

Рис. 2.11.

З порівняння збігу видно, що всі отримані моделі достатньо добре описують систему, але кращими слід визнати модель OE, що оцінена у загальній поліноміальній формі (G_{1oe}) та модель ARMAX (G_{2armax}) зниженого порядку.

Кінець приклада.

Іншим методом перевірки якості моделі є аналіз її остаткової похибки (2.48). Якщо модель цілком адекватно описує систему, то остаткова похибка $e(k)$ повністю не залежить від входу системи $u(k)$ (тобто є ідеальним білим шумом). Автоковаріаційна функція такого сигналу $R_e(\lambda)$ повинна задовольняти умовам (2.38).

Таким чином про адекватність моделі можна судити по характеру автоковаріаційної функції похибки $e(k)$ та взаємної коваріаційної функції $R_{ue}(\lambda)$ між входом моделі $u(k)$ та похибкою для заданої кількості лагів (див. п. 2.2.2.).

Остаткову похибку у вигляді прогнозованої помилки можна розрахувати за допомогою функції pe.

Функція resid розраховує остаткову похибку та будує автоковаріаційну і взаємну коваріаційну функцію для 25 лагів разом з 99 %-ми довірчими інтервалами.

Приклад 2.17.

Перевіремо кореляційні властивості остаткових похибок моделей, що визначені як кращі в попередньому прикладі.

```
» e1oe=resid(D1v,G1oe);
» figure
» e2armax=resid(D1v,G2armax);
```

Графіки кореляційних функцій для моделі G_{1oe} зображені на рис. 2.12., а для моделі G_{2armax} – на рис. 2.13.

Побудуємо графіки остаткової похибки (прогнозованої помилки) моделей, що розглядаються:

```
» plot([e1oe e2armax])
» xlabel('T')
» title('Остаткові похибки моделей G1oe та G2armax')
```

Результат побудови зображено на рис. 2.14.

Кінець приклада.

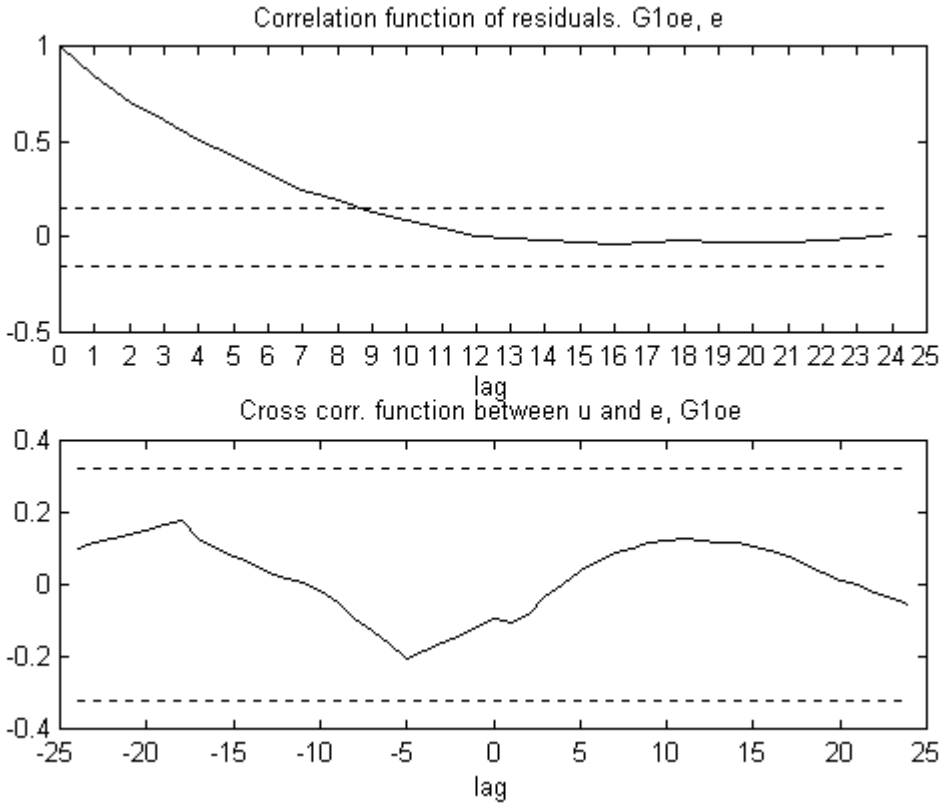

Рис. 2.12.

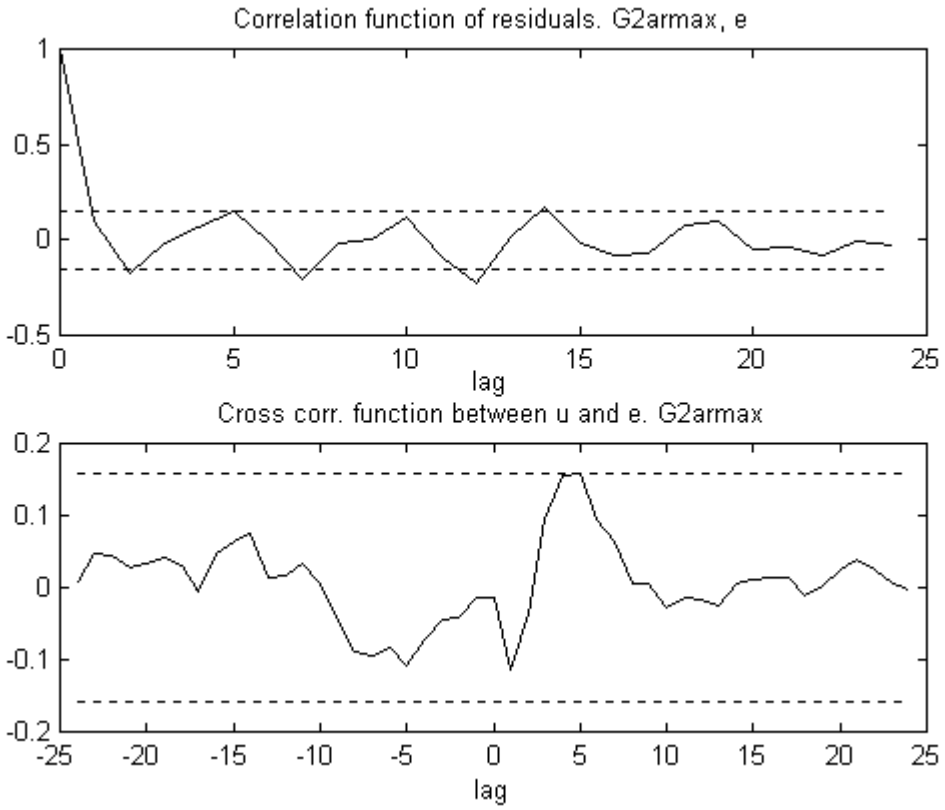

Рис. 2.13.

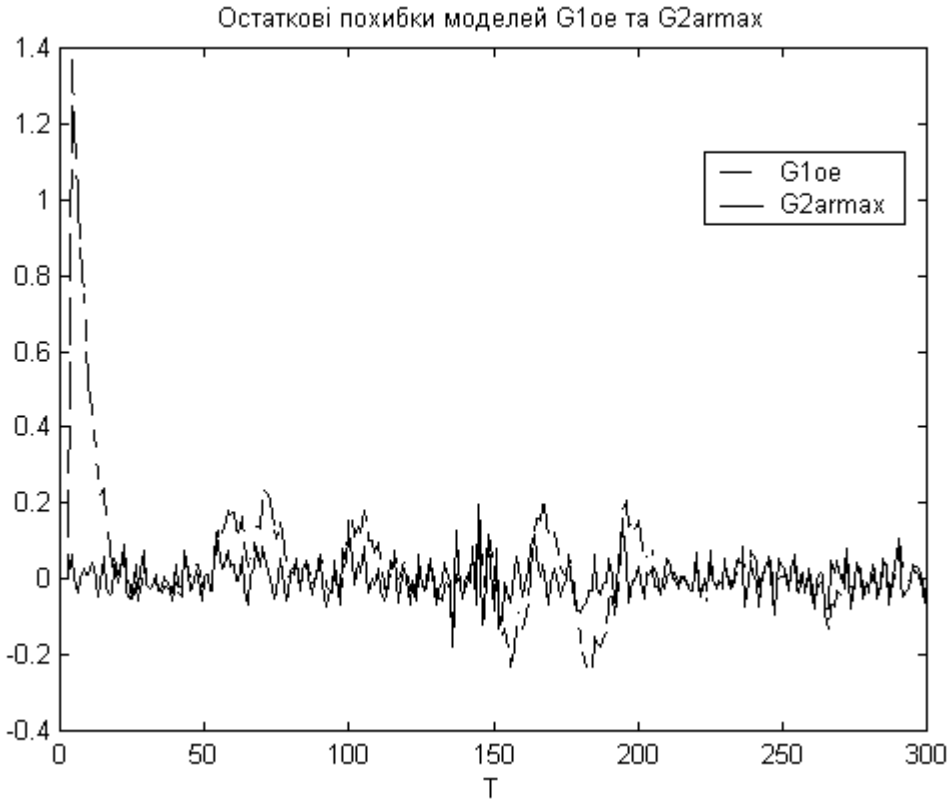

Рис. 2.14.

2.8.3. Деякі особливі випадки ідентифікації

При оцінюванні ARX-моделі (2.25) методом допоміжної змінної (див. п. 2.7.4.), або при оцінюванні OE-моделі (2.29) головна увага приділяється визначенню динаміки самої системи G, а не фільтру випадкового сигналу H (див. (2.3)). Тому перевірка адекватності таких моделей зводиться до аналізу тільки взаємної коваріаційної функції (остаткова похибка моделі може істотно відрізнятись від білого шуму).

Приклад 2.18.

З врахуванням сказаного в якості остаточної моделі системи, що досліджується, слід визнати модель G_{1oe} (див. приклади 2.8., 2.16. та 2.17.) тому, що вона дає найкращий збіг з реальним виходом системи, має найпростішу структуру, та показує задовільну взаємну кореляційну характеристику між входом та остатковою похибкою.

Перетворимо цю модель в tf-форму (CST):

```
» [no B no no F dsp T]=th2poly(G1oe);
» dsp                    % Дисперсія збурення (білого шуму)
dsp = 0.0421
» Go=tf(B,F,T,'var','q') % Дискретна передатна функція
Transfer function:
     0.03881 q - 0.003241 q^2
-----------------------------------
1 - 1.87 q + 1.086 q^2 - 0.1799 q^3
Sampling time: 1
» Wo=d2c(Go)             % Відповідна передатна функція
                         % в безперервному часі
Transfer function:
0.01976 s^2 + 0.07454 s + 0.07825
---------------------------------
s^3 + 1.715 s^2 + 0.6317 s + 0.08
```

Кінець приклада.

Часто необхідно вимірювати вхідні та вихідні сигнали в замкненій системі. Про зворотний зв'язок в системі говорить наявність кореляції між $u(k)$ та $e(k)$ на від'ємних лагах (на графіку взаємної коваріаційної функції), або істотні значення імпульсного відгуку на від'ємних лагах при оцінюванні методом кореляційного аналізу. Загалом метод прогнозованої помилки добре оцінює моделі за такими даними, хоча для OE-моделі (2.29) та BJ-моделі (2.30) оцінка фільтру випадкового сигналу H може виявитись невірною.

Методи спектрального аналізу та допоміжної змінної дають неадекватні оцінки систем зі зворотним зв'язком.

2.9. Інструмент ідентифікації Ident

Всі описані можливості SIT доступні в більш зручній формі через інтерактивний інструмент, що реалізовано на базі графічного інтерфейсу користувача (GUI), який можна викликати командою ident.

Серед додаткових можливостей Ident можна відзначити макрокоманди швидкого виконання.

Команда Preprocess/Quickstart виконує таку послідовність дій:

– будуються графіки часової залежності входів та виходів системи;

– з даних видаляється тренд (нульового порядку);

– дані розділяються на дві, однакові за обсягом, частини;

– одна з частин передається для подальшого оцінювання;

– інша частина передається для подальшої перевірки адекватності моделей.

При виборі команди Estimate/Quickstart оцінюються такі моделі:

– імпульсний відгук системи методом кореляційного аналізу;

– частотний відгук системи методом спектрального аналізу;

– ARX-модель четвертого порядку з запізнюванням, що визначено при прямому оцінюванні імпульсного відгуку, методом найменших квадратів;

– модель в просторі станів методом підпростору.

Крім того команда будує три графіки:

– імпульсних відгуків ARX-моделі, моделі в просторі станів та прямої оцінки імпульсного відгуку;

– частотних відгуків ARX-моделі, моделі в просторі станів та прямої оцінки частотного відгуку;

– виміреного виходу системи та виходів ARX-моделі та моделі в просторі станів з відображенням збігу (2.68) для кожної з моделей.

Команда **To LTI Viewer** перетворює модель (див. п. 2.6.1.) в ss-форму та завантажує її в LTI-Viewer (див. п. 1.7.5.).

Команда **To Workspace** експортує модель в робочу область MatLab в th або ff-формі.

Інші команди ident практично повністю аналогічні відповідним функціям режиму інтерпретації (командного рядка).

Розділ 3. Синтез систем автоматичного керування

Під проектуванням систем керування надалі ми будемо розуміти процес вибору параметрів коригуючого пристрою в замкненій системі керування. Методи проектування, що розглядаються в цьому розділі, є інтерактивними і суміщають добір параметрів з аналізом, розрахунком та моделюванням процесів в динамічних системи. Для синтезу систем використовуються функції бібліотеки CST [23, 27] та базові можливості MatLab [20 – 22, 35, 37].

3.1. Синтез систем методом кореневого годографа

Кореневий годограф зображує траєкторії полюсів (див. п. 1.7.1.) замкненої системи при варіюванні одного з параметрів в системі, коли інші параметри системи є сталими. Найчастіше метод кореневого годографа використовується для настроювання коефіцієнта зворотного зв'язку з метою бажаного розташування полюсів замкненої SISO системи керування.

На рис. 3.1. зображено замкнену систему регулювання по каналу збурення, а на рис. 3.2. – за каналом завдання,

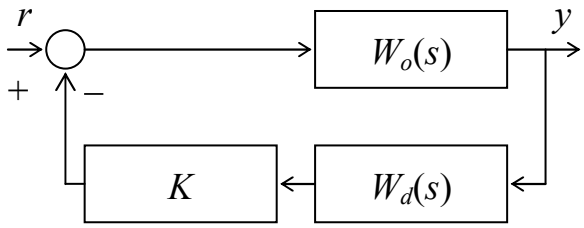

Рис. 3.1.

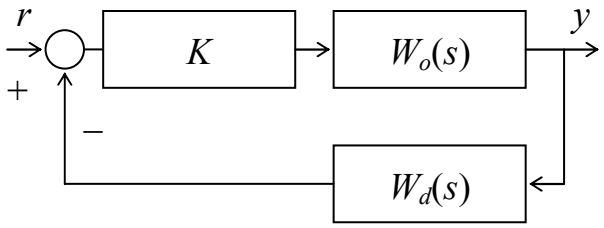

Рис. 3.2.

де $W_o(s)$ – передатна функція об'єкта керування; $W_d(s)$ – передатна функція вимірювача (датчика); K – скалярний коефіцієнт передачі компенсуючого пристрою, який підлягає настроюванню; r – вхідний сигнал (збурення або завдання). Полюси таких замкнених системи – це корені характеристичного рівняння:

$$1 + K \cdot W_o(s) \cdot W_d(s) = 0. \qquad (3.1)$$

Метод передбачає побудову траєкторій полюсів на комплексній площині в залежності від значення *K*.

При аналізі кореневого годографа слід пам'ятати деякі його властивості: Число гілок кореневого годографа дорівнює числу *Na* полюсів замкненої системи. Гілки кореневого годографа починаються при $K = 0$ в полюсах передатної функції розімкненої системи. При $K \to \infty$ *Nb* гілок кореневого годографа прямують до *Nb* нулів передатної функції розімкненої системи, а інші *Na* – *Nb* гілок прямують до нескінченності та мають асимптоти, що виходять з однієї точки на дійсній від'ємній півосі. Гілки, що розташовані ближче до уявної осі та початку координат, мають домінуючий вплив на характер перехідних процесів в системі [11].

Побудувати кореневий годограф SISO системи, як функцію коефіцієнта передачі зворотного зв'язку можна за допомогою команди rlocus. Визначити значення *K* для будь-якої точки такого годографа можна в інтерактивному режимі за допомогою команди rlocfind.

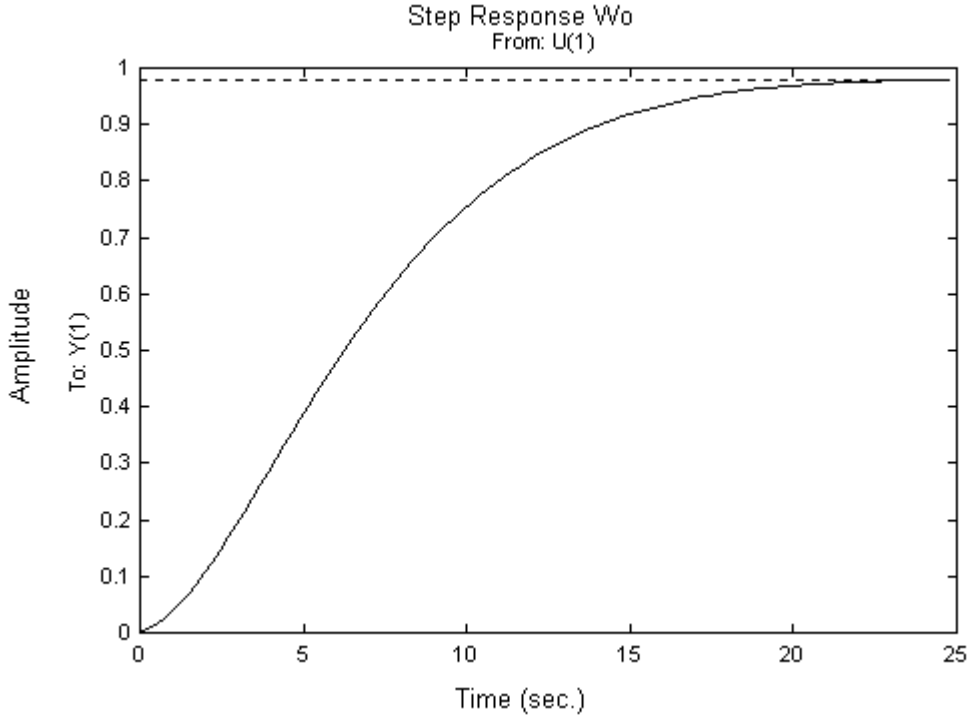

Рис. 3.3.

Приклад 3.1.

Об’єкт керування в безперервному часі описується передатною функцією: $W_o(s) = \dfrac{0.01976s^2 + 0.07454s + 0.07825}{s^3 + 1.715s^2 + 0.6317s + 0.08}$ (SISO система див. приклад 2.18.). Побудуємо перехідну характеристику об’єкта (рис. 3.3.):

```
» step(Wo)
```

Треба спроектувати компенсуючий пристрій, який би забезпечив час перехідного процесу в замкненій системі не більше 15 с при нанесенні одиничного збурення (рис. 3.1.). При цьому статична помилка регулювання не повинна перевищувати 3% від збурення.

Побудуємо кореневий годограф (рис. 3.4.):

```
» rlocus(Wo)
```

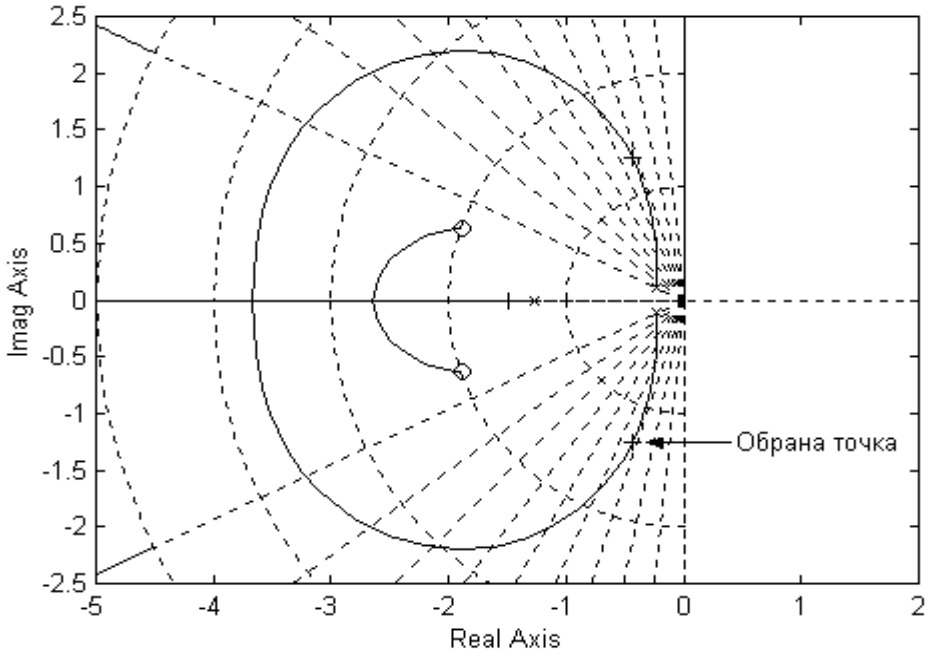

Рис. 3.4.

З рис. 3.4. видно, що замкнена система з об’єктом W_o є стійкою при всіх значеннях K. Оберемо точку, що відповідає невеликому значенню коефіцієнта згасання (на рисунку позначена "+"):

```
» sgrid
» K=rlocfind(Wo)
```

```
Select a point in the graphics window
selected_point = -0.4466 - 1.2584i
K = 33.0133
```

Побудуємо перехідну характеристику замкненої системи з отриманим коефіцієнтом зворотного зв'язку (рис. 3.5.):

```
» Wzs=feedback(Wo,K,-1);
» step(Wzs)
```

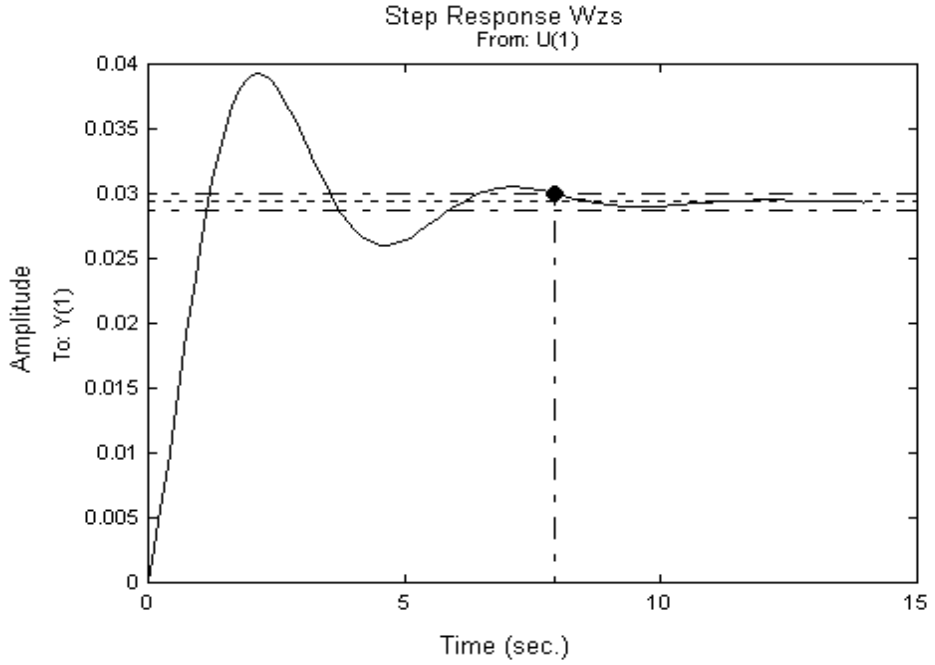

Рис. 3.5.

З рис. 3.5. видно, що такий компенсатор відповідає поставленим умовам. Для порівняння побудуємо імпульсні перехідні характеристики об'єкта та замкненої системи по каналу збурення (рис. 3.6.):

```
» impulse(Wo,Wzs)
```

Кінець приклада.

Проектування можна також виконувати за допомогою плоттера кореневих годографів rltool, який дозволяє:

– аналізувати розташування коренів лінійної стаціонарної замкненої SISO системи;

– визначати параметри компенсатора (полюси, нулі та коефіцієнт підсилення);

– досліджувати, як параметри компенсатора впливають на характер кореневого годографа та інших характеристик розімкненої та замкненої системи (наприклад перехідної характеристики) за допомогою LTI-Viewer (див. п. 1.7.5.).

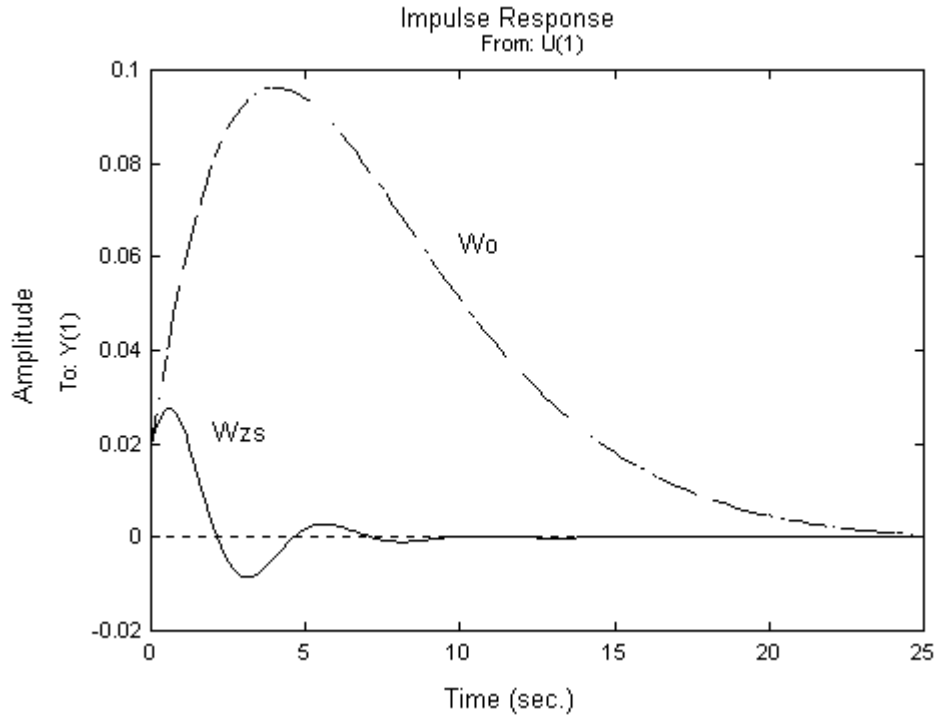

Рис. 3.6.

3.2. Розрахунок системи на заданий показник коливності

Цей метод застосовується для розрахунку параметрів типових лінійних регуляторів безперервної дії в SISO системах.

3.2.1. Типові закони регулювання

Ступінь наближення точки перетинання годографа АФХ розімкненої системи і від'ємної півосі до точки n $(-1, j\cdot 0)$ визначає запас стійкості (по Найквісту) замкненої системи автоматичного регулювання [11, 17]. При наближенні АФХ до точки n збільшується коливність в замкненій системі, а при перетинанні цієї точки в замкненій системі встановлюються незатухаючі коливання.

Оскільки модуль вектора АФХ системи визначається коефіцієнтом передачі розімкненої системи на даній частоті, то запас стійкості можна регулювати зміною параметра k_p:

$$W(s) = W_o(s) \cdot W_r(s) = k_p \cdot W_o(s), \tag{3.2}$$

де $W(s)$ та $W_r(s)$ – передатні функції розімкненої системи та регулятора відповідно. Регулятор такого типу називають пропорційним (П-регулятором) і в динамічному відношенні він є підсилювальною ланкою W_p:

$$W_r(s) = W_p(s) = k_p. \tag{3.3}$$

Надмірне збільшення запасу стійкості погіршує якість регулювання, тому що збільшуються час перехідного процесу в системі, динамічна та статична помилки регулювання по каналу як завдання, так і збурення.

Шляхом підбору оптимального значення k_r можна істотно зменшити статичну помилку регулювання, але її повна ліквідація в системі з П-регулятором неможлива навіть теоретично. Для цієї цілі використовується інтегральний закон регулявання. Передатна функція І-регулятора має вид:

$$W_r(s) = W_i(s) = \frac{1}{T_{in}s}, \tag{3.4}$$

де T_{in} – стала часу інтегрування, що визначає час, за який з моменту надходження на вхід І-регулятора постійного сигналу сигнал на виході регулятора досягає значення, рівного значенню вхідного сигналу.

П- та І-регулятори реагують тільки на вже існуючі порушення технологічного процесу. Якщо параметр, що регулюється, починає швидко змінювати своє значення, то це говорить про те, що в системі виникло велике збурення, яке призведе до значного відхилення технологічного параметра. В цьому випадку в системі треба мати регулятор, який при великій швидкості зміни параметра, що регулюється, (в початковий момент П-регулятор виробляє слабке керування, а І-регулятор тільки починає виробляти керуючий вплив) виробляв би істотне керування, яке ліквідує очікуване відхилення. Такий регулятор називається диференційним. Д-регулятор виробляє регулюючу дію,

пропорційну швидкості зміни відхилення параметра, що регулюється, і має передатну функцію:

$$W_r(s) = W_i(s) = T_{df} \cdot s, \tag{3.5}$$

де T_{df} – стала часу диференцювання, яка визначає величину регулюючої дії по швидкості.

Для врахування переваг різних законів регулювання використовують паралельне з'єднання розглянутих регуляторів.

Передатна функція ідеального ПІД-регулятора має вид:

$$W_r(s) = W_{pid}(s) = k_p + \frac{1}{T_{in}s} + T_{df}s = k_p \frac{T_i T_v s^2 + T_i s + 1}{T_i s}, \tag{3.6}$$

де $T_i = T_{in} \cdot k_p$ – час ізодрому;

$T_v = \dfrac{T_{df}}{k_p}$ – час випередження.

Інші типи регуляторів можна вважати частковими випадками ПІД-регулятора. Так при $T_v = 0$ та $T_i = \infty$ або ($T_{df} = 0$ та $T_{in} = \infty$) отримуємо П-регулятор (3.3).

3.2.2. Визначення параметрів типових регуляторів

Амплітудна частотна характеристика (див. (1.48)) замкненої системи $G(j\omega)$ з регулятором, що містить інтегральну складову, подібна до характеристики системи другого порядку [4]. Відношення максимуму АЧХ (на резонансній частоті, див п. 1.7.4.) до її значення на нульовій частоті називають показником коливності:

$$M = \frac{\left|G(j\omega_r)\right|}{\left|G(0)\right|}. \tag{3.7}$$

Із зменшенням коефіцієнта згасання ζ показник коливності збільшується, тобто збільшується резонансний пік АЧХ. Тому система регулювання має тим більший запас стійкості, чим менше значення M її характеризує. Відомо [13], що задовільну якість перехідних процесів в замкнених системах забезпечує $M = 1.3 \ldots 1.6$.

Обмеження на коливність власних процесів в замкненій системі можна сформулювати як вимогу, щоб годограф амплітудно-фазової частотної характеристики (див. п. 1.7.3.) розімкненої системи $W(j\omega)$ з регулятором не перетинав області, обмеженої окружністю з координатами $(-x_M, 0)$ та радіусом r_M (М-колом), де

$$x_M = \frac{M^2}{M^2 - 1}; \quad r_M = \frac{M}{M^2 - 1}. \tag{3.8}$$

Таким чином, розрахунок системи на обраний показник коливності полягає у визначенні параметрів регулятора, які забезпечують торкання АФХ розімкненої системи з обраним М-колом [3, 14].

При визначенні параметрів регулятора достатньо розглядати тільки третій квадрант годографа АФХ системи.

В загальному випадку умові максимальної фільтрації збурень відповідає максимально можливе значення k_r для П-регулятора та відношення k_r / T_i для ПІ- та ПІД-регуляторів. Для визначення параметрів ПІД-регулятора співвідношення T_v / T_i повинно бути в межах $0.3 \ldots 0.5$ для об'єктів з самовирівнюванням та 0.15 для об'єктів без самовирівнювання [16].

Побудувати годограф АФХ розімкненої системи з бажаним М-колом можна за допомогою функції rotach (див. додаток 5.).

Приклад 3.2.

Визначити параметри ПІ-регулятора, що забезпечують показник коливності $M = 1.5$ для системи з об'єктом W_o (див. попередній приклад).

Побудуємо годограф розімкненої системи разом з заданим М-колом:

```
» help rotach
  Побудова М-кола та годографа АФХ розімкненої системи
  Wrg=rotach(Wo,k,Ti,Tv,M)
  Wo  - передатна функція SISO об'єкта;
  k   - коефіцієнт пропорційності регулятора;
  Ti  - час ізодрому;
  Tv  - час випередження;
  M   - показник коливальності системи (M=1.3...1.6).
```

```
Реалізовані системи з регуляторами:
П   - при k>0 Ti=inf Tv=0
I   - при k=0 Ti>0   Tv=0
ПІ  - при k>0 Ti>0   Tv=0
ПД  - при k>0 Ti=inf Tv>0
ПІД - при k>0 Ti>0   Tv>0
Wrg - повертається передатна функція регулятора
```

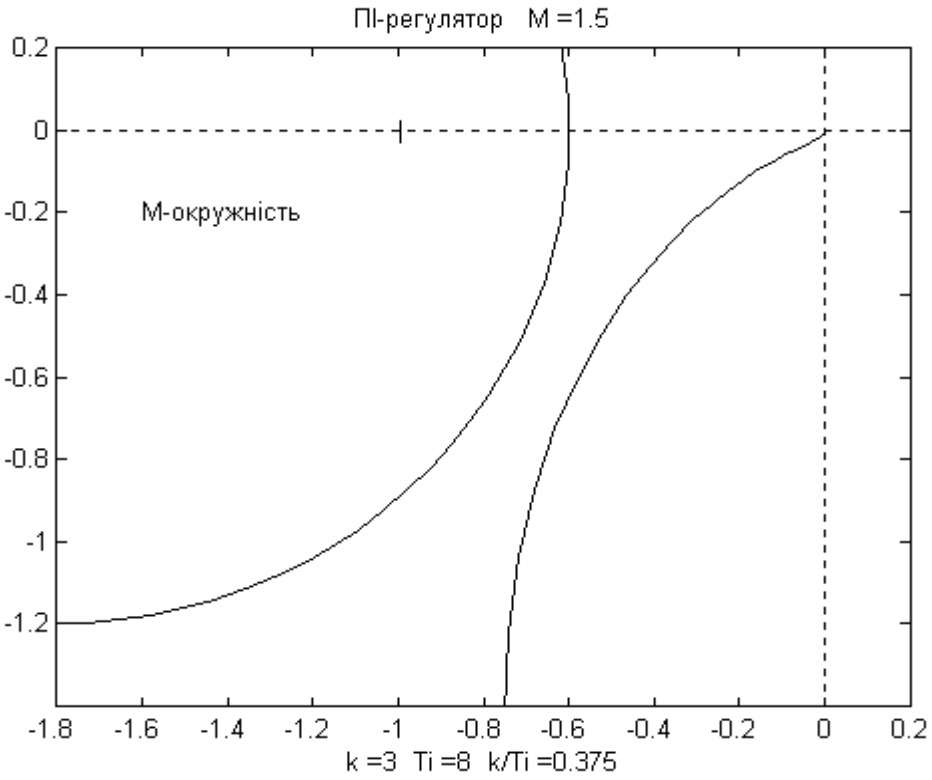

Рис. 3.7.

```
» rotach(Wo,3,8,0,1.5);
```

Результат побудови зображено на рис. 3.7. Як видно з рисунка, обрані параметри не є задовільними. Змінимо параметри та повторимо побудову (рис. 3.8.):

```
» rotach(Wo,4,6.07,0,1.5);
```

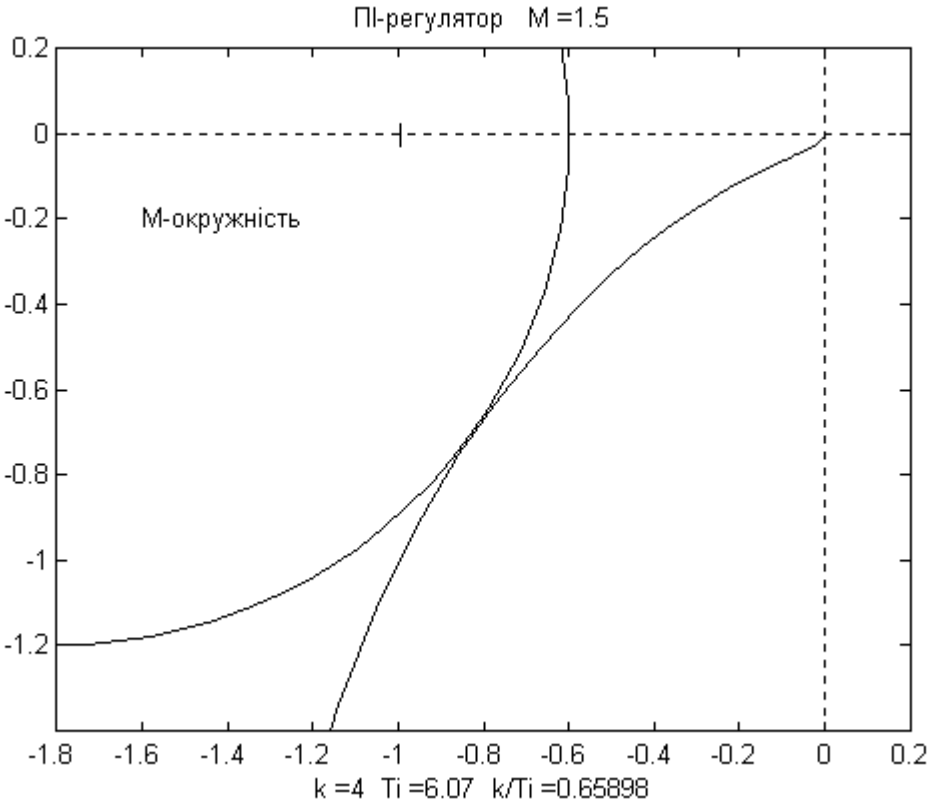

Рис. 3.8.

Така настройка регулятора відповідає поставленим умовам. Шляхом подібних операцій можна знайти і інші комбінації коефіцієнта пропорційності k та часу ізодрому T_i, що забезпечують показник коливності $M = 1.5$:

k	2	3	4	5	6	7	8
T_i	3.97	4.97	6.07	7.23	8.76	10.5	13
$k_0 = k / T_i$	0.504	0.603	0.659	0.692	0.685	0.667	0.615

Побудуємо лінію однакових значень показника коливності в площині настройок регулятора $\{k,\ k_0 = k/T_i\}$ (рис. 3.9.):

```
» x =2:8;
» y =[0.504 0.603 0.659 0.692 0.685 0.667 0.615];
» xn=2:0.1:8;
» yn=interp1(x,y,xn,'cubic'); % Кубічна інтерполяція
» plot(xn,yn), xlabel('k'), ylabel('k0 = k/Ti')
```

Остаточно виберемо настройку регулятора, що відповідає найбільшому значеню k / T_i (на рисунку позначена "+"). Слід відзначити, що деякі методики

рекомендують обирати настройку, що відповідає точці, яка знаходиться поблизу екстремальної точки справа від неї на лінії однакових значень показника коливності [17].

```
» [k k0]=ginput(1)
k  = 5.2074
k0 = 0.6930
» hold on, plot(k,k0,'+'), hold off
```

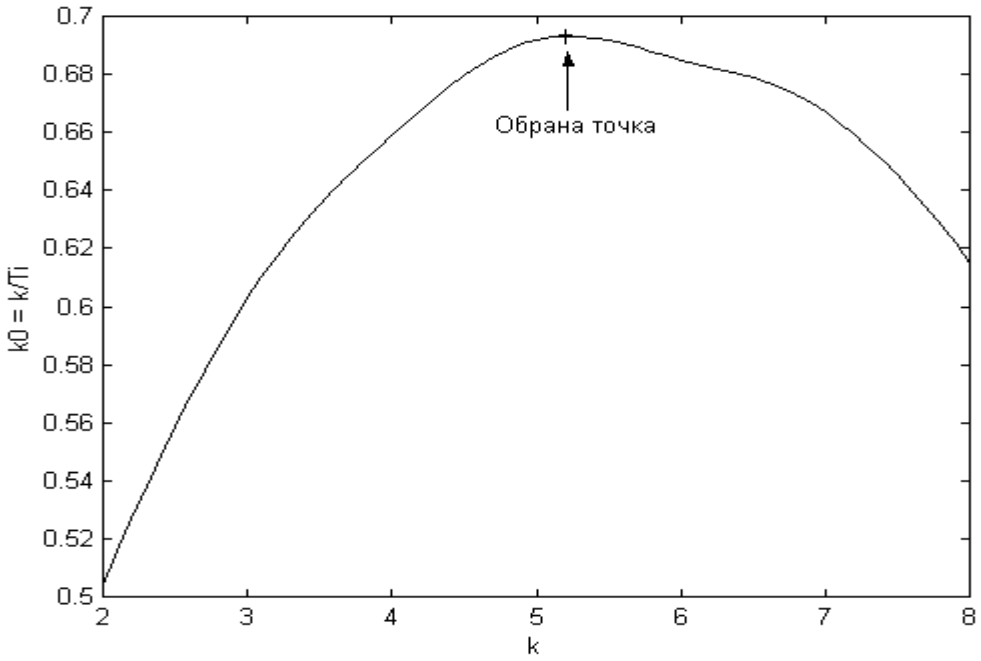

Рис. 3.9.

Побудуємо перехідні характеристики замкненої системи з таким регулятором за каналами збурення та завдання (рис. 3.10). Для цього спочатку треба сформувати MISO систему W_{sys} з двома входами (збурення та завдання) та одним виходом (див. п. 1.5.4.):

```
» Wrg=rotach(Wo,k,k/k0,0,1.5)   % Передатна функція ПІ-регулятора
Transfer function:
39.13 s + 5.207
---------------
    7.514 s
» S=append(Wo,Wrg);
» Q=[1 2;2 -1];
» Wsys=tf(connect(S,Q,[1 2],1));% Передатна функція системи
```

```
» set(Wsys,'InputName',...
        {'Збурення' 'Завдання'},'OutputName','Вихід');
» grid on, step(Wsys);
```

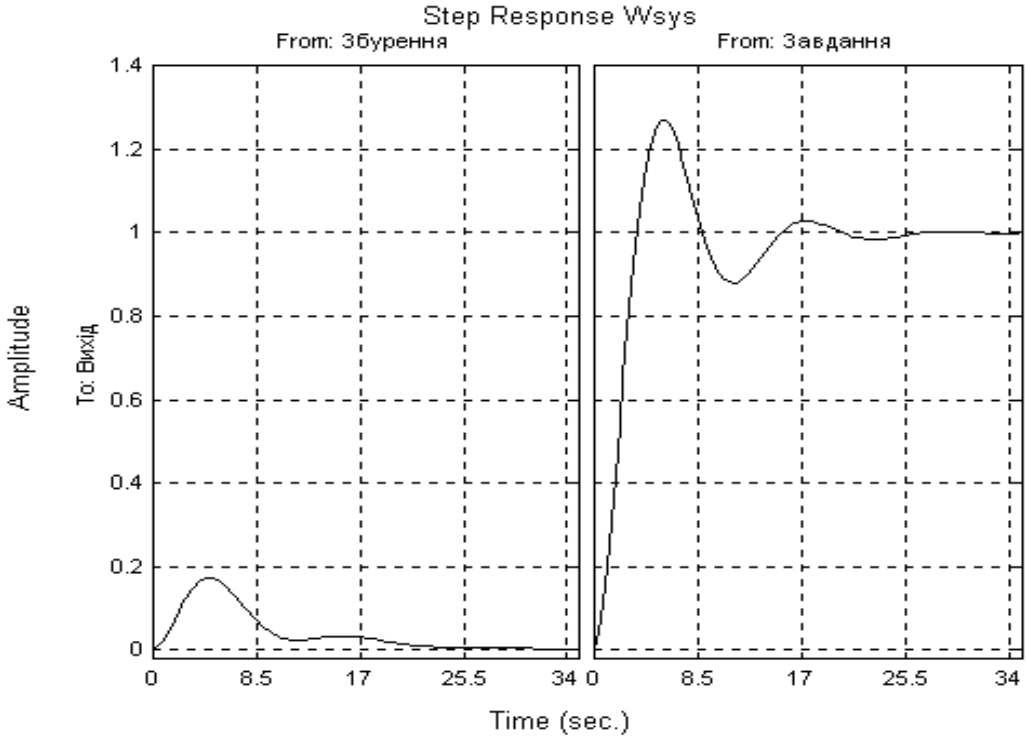

Рис. 3.10.

Кінець приклада.

3.3. Проектування систем з ЛКГ-регулятором

Якщо об'єкт є стохастичним і його модель можна описати в просторі станів рівняннями виду (2.34) або (2.36), то для стабілізації виходів **y** системи в околі нульового значення можна застосувати лінійне квадратичне гауссове (ЛКГ) керування. В безперервному часі модель стохастичного об'єкта в просторі станів можна представити як:

$$\begin{aligned} \frac{d\mathbf{x}(t)}{dt} &= \mathbf{A}\cdot\mathbf{x}(t)+\mathbf{B}\cdot\mathbf{u}(t)+\mathbf{G}\cdot\mathbf{w}(t); \\ \mathbf{y}_{\mathbf{v}}(t) &= \mathbf{C}\cdot\mathbf{x}(t)+\mathbf{D}\cdot\mathbf{u}(t)+\mathbf{H}\cdot\mathbf{w}(t)+\mathbf{v}(t). \end{aligned} \tag{3.9}$$

Вважається, що на об'єкт діє випадкове збурення **w** та керування **u**, яке формує регулятор по результатам вимірювань $\mathbf{y}_{\mathbf{v}}$, що зашумлені випадковим сигналом **v** (процеси **w** та **v** розглядаються як білий шум) [6, 12, 15].

Схема системи з ЛКГ-регулятором зображена на рис. 3.11.

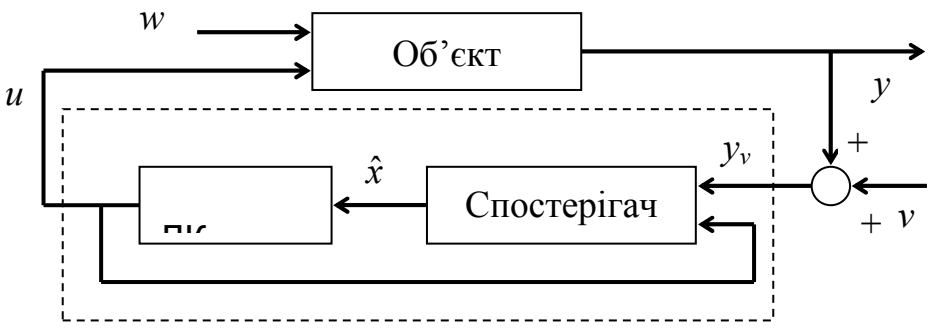

Рис.3.11

ЛКГ-регулятор має два компонента: оптимальний регулятор, що реалізує керування по квадратичному критерію якості, та спостерігач Калмана. Кожен з компонентів проектується окремо. Система з ЛКГ-регулятором може бути дискретною або безперервною.

3.3.1. Оптимальний регулятор стану

В ЛКГ-керуванні регулююча дія обумовлена квадратичним критерієм якості виду:

$$J(u) = \int_0^\infty (\mathbf{x}^{\mathrm{T}}(t) \cdot \mathbf{Q} \cdot \mathbf{x}(t) + \mathbf{2x}(t) \cdot \mathbf{N} \cdot \mathbf{u}(t) + \mathbf{u}(t) \cdot \mathbf{R} \cdot \mathbf{u}(t))\, dt, \qquad (3.10)$$

де вагові матриці **Q**, **N** та **R** задаються проектувальником і характеризують співвідношення між якістю регулювання (як швидко **x**(*t*) наближується до нуля) та витратами на керуюче зусилля.

На цьому етапі проектування відшукується закон керування:

$$\mathbf{u}(t) = -\mathbf{K} \cdot \mathbf{x}(t), \qquad (3.11)$$

що мінімізує критерій (3.10). Матриця зворотного зв’язку **K** для систем в безперервному часі визначається як:

$$\mathbf{K} = \mathbf{R}^{-1}(\mathbf{B}^{\mathrm{T}}\mathbf{S} + \mathbf{N}^{\mathrm{T}}), \qquad (3.12)$$

де матриця S є розв’язком алгебраїчного рівняння Ріккаті:

$$\mathbf{A}^{\mathrm{T}}\mathbf{S} + \mathbf{S} \cdot \mathbf{A} - (\mathbf{S} \cdot \mathbf{B} + \mathbf{N}) \cdot \mathbf{R}^{-1}(\mathbf{B}^{\mathrm{T}}\mathbf{S} + \mathbf{N}^{\mathrm{T}}) + \mathbf{Q} = \mathbf{0}. \qquad (3.13)$$

Такий оптимальний регулятор стану називають ЛК-регулятором. Синтезувати його можна за допомогою функції lqr.

Для дискретних систем квадратичний критерій (3.10) матиме вид:

$$J(u) = \sum_{i=1}^{\infty} (\mathbf{x}^{\mathrm{T}}(i) \cdot \mathbf{Q} \cdot \mathbf{x}(i) + \mathbf{2x}(i) \cdot \mathbf{N} \cdot \mathbf{u}(i) + \mathbf{u}(i) \cdot \mathbf{R} \cdot \mathbf{u}(i)), \tag{3.14}$$

і ЛК-регулятор формується як:

$$\mathbf{K} = (\mathbf{B}_\mathbf{d}^{\mathrm{T}} \mathbf{S} \cdot \mathbf{B}_\mathbf{d} + \mathbf{R})^{-1} (\mathbf{B}_\mathbf{d}^{\mathrm{T}} \mathbf{S} \cdot \mathbf{B}_\mathbf{d} + \mathbf{N}^{\mathrm{T}}), \tag{3.15}$$

де матриця S є вирішення дискретного рівняння Ріккаті:

$$\begin{aligned} -(\mathbf{A}_\mathbf{d}^{\mathrm{T}} \mathbf{S} \cdot \mathbf{B}_\mathbf{d} + \mathbf{N})(\mathbf{B}_\mathbf{d}^{\mathrm{T}} \mathbf{S} \cdot \mathbf{B}_\mathbf{d} + \mathbf{N})(\mathbf{B}_\mathbf{d}^{\mathrm{T}} \mathbf{S} \cdot \mathbf{B}_\mathbf{d} + \mathbf{R})^{-1} (\mathbf{B}_\mathbf{d}^{\mathrm{T}} \mathbf{S} \cdot \mathbf{B}_\mathbf{d} + \mathbf{N}^{\mathrm{T}}) + \\ + \mathbf{A}_\mathbf{d}^{\mathrm{T}} \mathbf{S} \cdot \mathbf{A}_\mathbf{d} - \mathbf{S} + \mathbf{Q} = \mathbf{0}. \end{aligned} \tag{3.16}$$

Синтезувати дискретний ЛК-регулятор можна за допомогою функції dlqr.

За допомогою функції lqry можна розрахувати оптимальний регулятор для дискретних та безперервних систем.

3.3.2. Фільтр Калмана

ЛК-закон регулювання неможна реалізувати без вимірювання всіх станів системи. Замість вимірювання ми можемо синтезувати спостерігач, який продукує вектор оцінок $\hat{\mathbf{x}}(t)$, такий, що закон керування :

$$\mathbf{u}(t) = -\mathbf{K} \cdot \hat{\mathbf{x}}(t) \tag{3.17}$$

залишається оптимальним у вищезгаданому сенсі. Такий спостерігач об'єднує в собі фільтр Калмана та модель об'єкта. В безперервному часі він має вид:

$$\begin{aligned} &\frac{d\hat{\mathbf{x}}(t)}{dt} = \mathbf{A} \cdot \hat{\mathbf{x}}(t) + \mathbf{B} \cdot \mathbf{u}(t) + \mathbf{L}(\mathbf{y}_\mathrm{v}(t) - \mathbf{C} \cdot \hat{\mathbf{x}}(t) - \mathbf{D} \cdot \mathbf{u}(t)); \\ &\begin{bmatrix} \hat{\mathbf{y}}(t) \\ \hat{\mathbf{x}}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{C} \\ \mathbf{I} \end{bmatrix} \cdot \hat{\mathbf{x}}(t) + \begin{bmatrix} \mathbf{D} \\ \mathbf{0} \end{bmatrix} \cdot \mathbf{u}(t), \end{aligned} \tag{3.18}$$

де матриця коефіцієнтів зворотного зв'язку фільтра Калмана **L** визначається через рівняння Ріккаті з урахуванням коваріаційних матриць шумів (див. п. 2.2.2.).

Коваріаційна матриця шуму вимірювання визначається як:

$$\mathbf{R}_{\mathbf{v}}(\lambda)=\mathrm{E}[\mathbf{v}(t)\mathbf{v}^{\mathrm{T}}(t+\lambda)]=\begin{bmatrix} R_{v_1}(\lambda) & R_{v_1v_2}(\lambda) & \cdots & R_{v_1v_{Ny}}(\lambda) \\ R_{v_2v_1}(\lambda) & R_{v_2}(\lambda) & \cdots & R_{v_1v_{Ny}}(\lambda) \\ \vdots & \vdots & \ddots & \vdots \\ R_{v_{Ny}v_1}(\lambda) & R_{v_{Ny}v_2}(\lambda) & \cdots & R_{v_{Ny}}(\lambda) \end{bmatrix}, \qquad (3.19)$$

де *Ny* – кількість елементі вектора $\mathbf{v}(t)$; $R(\lambda)$ – коваріаційні функції між відповідними елементами вектора (кожен з яких є стохастичним процесом).

Для сигналу $\mathbf{v}(t)$ у вигляді білого шуму:

$$\mathbf{R}_{\mathbf{v}}(0)=\begin{bmatrix} \sigma^2_{v_1} & 0 & \cdots & 0 \\ 0 & \sigma^2_{v_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma^2_{v_{Ny}} \end{bmatrix}, \quad \mathbf{R}_{\mathbf{v}}(\lambda\neq 0)=\mathbf{0}. \qquad (3.20)$$

Коваріаційні матриці $\mathbf{R_w}$ та $\mathbf{R_{wv}}$ визначаються аналогічно.

Фільтр Калмана є оптимальним спостерігачем, коли він працює з гауссовим білим шумом. Тоді він мінімізує усталену коваріаційну матрицю похибки:

$$\mathbf{P}=\lim_{x\to\infty}\mathrm{E}[(\mathbf{x}(t)-\hat{\mathbf{x}}(t))(\mathbf{x}(t)-\hat{\mathbf{x}}(t))^{\mathrm{T}}]. \qquad (3.21)$$

Дискретний фільтр калмана визначається аналогічно (3.18).

Синтезувати фільтр Калмана можна за допомогою функції kalman.

3.3.3. ЛКГ-регулятор

Для формування ЛКГ-регулятора треба з'єднати ЛК-регулятор і фільтр Калмана як зображено на рис. 3.11. Такий регулятор описується рівняннями:

$$\begin{aligned} &\frac{d\hat{\mathbf{x}}(t)}{dt}=[\mathbf{A}-\mathbf{L}\cdot\mathbf{C}-(\mathbf{B}-\mathbf{L}\cdot\mathbf{D})\cdot\mathbf{K}]\cdot\hat{\mathbf{x}}(t)+\mathbf{L}\cdot\mathbf{y}_{\mathbf{v}}(t); \\ &\mathbf{u}(t)=-\mathbf{K}\cdot\hat{\mathbf{x}}(t). \end{aligned} \qquad (3.22)$$

Здійснити таке з'єднання в CST можна за допомогою функції lqgreg.

Приклад 3.3.

Спроектувати ЛКГ-регулятор для системи в безперервному часі з об'єктом, що описується ОЕ моделлю (2.29), схема якої зображена на рис. 3.12.

Динаміка об’єкта визначається передатною функцією W_o. Збурення в об’єкті існує у вигляді білого шуму e з дисперсією $\sigma_e^2 = 0.0421$ (див. приклад 2.18.).

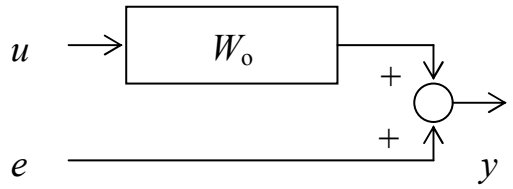

Рис. 3.12.

Сформуємо модель об’єкта в просторі станів:

```
» dis=0.0421;
» Po= ss(minreal([Wo 1]));
» set(Po,'inputname',['u';'e'])
```

Квадратичний критерій (3.10) враховує як низькочастотні, так і високочастотні складові процесу. Видалимо високочастотні складові шляхом введення в систему фільтра у вигляді аперіодичної ланки першого порядку $W_f = \dfrac{1}{1+T_f s}$, яка має сталу часу $T_f = 0.1$ с і практично не вносить змін в динаміку об’єкта. Остаточно модель об’єкта матиме вигляд:

```
» Pf=tf(10,[1 10]);      % Фільтр високочастотної складової
» Px=Po*Pf;               % Загальна модель
» set(Px,'outputn','y'); Px
a =
                    x1           x2           x3           x4
          x1       -10      0.15808      0.59632        1.252
          x2         0       -1.715      -0.6317       -0.16
          x3         0            1            0            0
          x4         0            0          0.5            0
```

```
b =
                    u            e
           x1       0            4
           x2     0.5            0
           x3       0            0
           x4       0            0
c =
                    x1           x2           x3           x4
            y      2.5            0            0            0

d =
                    u            e
            y       0            0
Continuous-time model.
```

Розрахуємо ЛК-регулятор для канала керування u – вихід системи y. Задамо вагові коефіцієнти критерія (3.10): **Q** = 1; **N** ≈ 0; **R** = 0:

```
» kx = lqry(Px(1,1),1,1e-8,0);
```

Розрахуємо спостерігач Калмана та сформуємо ЛКГ-регулятор, вважаючи, що дисперсія шуму вимірювання близька до нуля:

```
» estx = kalman(Px,dis,1e-8); % спостерігач
» Regx = lqgreg(estx,kx);
```

Порівняємо реакцію об’єкта та системи з ЛКГ-регулятором на шум з дисперсією σ_e^2. Результат імітації зображено на рис. 3.13.:

```
» Wsys = feedback(Px,Regx,1,1,+1);      % Канал u (перший вхід) -
                                        % y (перший вихід)
» t = 0:0.2:50;                         % Час моделювання
» e = sqrt(dis)*randn(length(t),1);     % Білий шум
» lsim(Px(1,2),':',Wsys(1,2),'-',e,t)  % Канал e (другий вхід) -
                                        % y (перший вихід)
```

З рисунка видно, що спроектований ЛКГ-регулятор зменшує шум в системі майже вчетверо.

Кінець приклада.

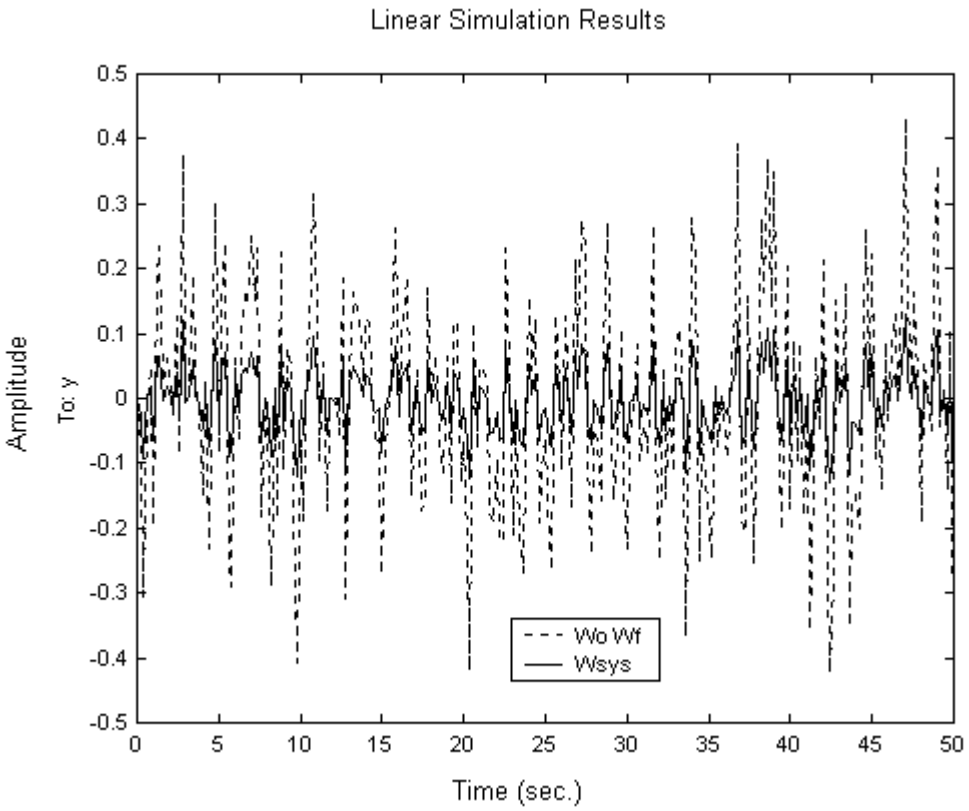

Рис. 3.13.

3.4. Проектування систем по заданому розташуванню полюсів

Розташування полюсів замкненої системи має прямий вплив на її часові характеристики, такі як час зростання, час встановлювання та перехідні коливання.

Метод передбачає завдання потрібних часових характеристик системи (вибір бажаного розташування полюсів замкненої системи) та розрахунок параметрів компенсатора, що відповідають цьому розташуванню [6, 12].

Вважається, що система має детерміновані початкові умови і її модель описується в просторі станів рівняннями виду (1.24) або (1.27). Схема системи з таким компенсатором зображена на рис. 3.14. Проектування динаміки компенсатора має два етапи: вибір регулятора стану та проектування спостерігача стану.

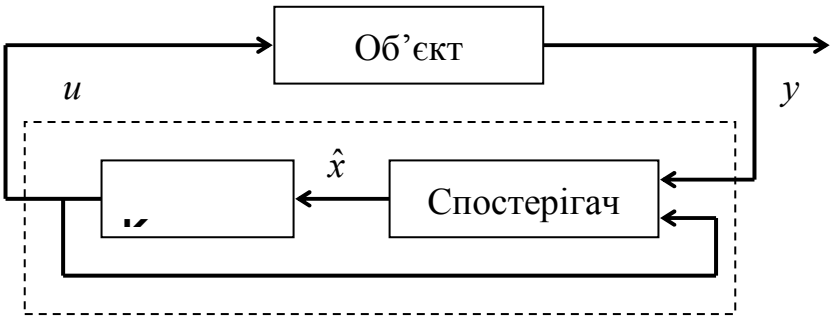

Рис.3.14

3.4.1. Вибір регулятора стану

При зворотному зв’язку (3.11) динаміка замкненої системи в безперервному часі описується рівнянням:

$$\frac{d\mathbf{x}(t)}{dt} = (\mathbf{A} - \mathbf{B} \cdot \mathbf{K}) \cdot \mathbf{x}(t) \tag{3.23}$$

і полюси замкненої системи є власними значеннями мтариці **A** – **B**·**K**. В методі, що розглядається, ми можемо вибрати матрицю зворотного зв’язку **K** так, що ці полюси розташуються в будь-якому бажаному місці комплексної площини (за умови, що **A** та **B** є керованими).

Сформувати регулятор стану по заданому розташуванню полюсів можна за допомогою функції place, використовуючи матриці {**A**, **B**} першого рівняння моделі в просторі станів (3.9). Ця функція працює з SISO та MIMO системами.

Сформувати регулятор стану для одновимірної системи можна також за допомогою функції acker, але функції place використовує робастний алгоритм і завжди гарантує високу якість розрахунків.

3.4.2. Проектування спостерігача стану

Неможливо реалізувати закон керування (3.11) без інформації про поточний стан системи, тобто без вимірювання вектора **x**(*t*) (див. п. 3.3.2.). Необхідно спроектувати пристрій, що продукує вектор $\hat{\mathbf{x}}(t)$, за умови, що закон (3.17) забезпече бажане розташування полюсів. Такий пристрій називається спостерігачем і його динаміка описується рівнянням:

$$\frac{d\hat{\mathbf{x}}(t)}{dt} = \mathbf{A}\cdot\hat{\mathbf{x}}(t) + \mathbf{B}\cdot\mathbf{u}(t) + \mathbf{L}\cdot(\mathbf{y}(t) - \mathbf{C}\cdot\hat{\mathbf{x}}(t) - \mathbf{D}\cdot\mathbf{u}(t)). \qquad (3.24)$$

Полюси спостерігача, це власні значення матриці **A** – **L**·**C**, які можуть бути призначені довільно належним вибором матриці коефіцієнтів **L**. Як правило, динаміку спостерігача стану роблять більш швидкою, ніж динаміка системи регулювання в цілому.

Розрахувати матрицю **L** можна за допомогою функції acker або place, використовуючи спряжену систему $\{\mathbf{A}^{\mathrm{T}}, \mathbf{C}^{\mathrm{T}}\}$, або функцію kalman (див. п. 3.3.2.)

Сформувати спостерігач можна за допомогою функції estim.

Заміна вектора $\mathbf{x}(t)$ його оцінкою $\hat{\mathbf{x}}(t)$ дає динаміку компенсатора:

$$\begin{aligned} &\frac{d\hat{\mathbf{x}}(t)}{dt} = [\mathbf{A} - \mathbf{L}\cdot\mathbf{C} - (\mathbf{B} - \mathbf{L}\cdot\mathbf{D})\cdot\mathbf{K}]\cdot\hat{\mathbf{x}}(t) + \mathbf{L}\cdot\mathbf{y}(t); \\ &\mathbf{u}(t) = -\mathbf{K}\cdot\hat{\mathbf{x}}(t). \end{aligned} \qquad (3.25)$$

Остаточно отримана замкнена динамічна система може бути описана так:

$$\begin{bmatrix} \frac{d\mathbf{x}(t)}{dt} \\ \frac{d\varepsilon(t)}{dt} \end{bmatrix} = \begin{bmatrix} \mathbf{A} - \mathbf{B}\cdot\mathbf{K} & \mathbf{B}\cdot\mathbf{K} \\ \mathbf{0} & \mathbf{A} - \mathbf{L}\cdot\mathbf{C} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{x}(t) \\ \varepsilon(t) \end{bmatrix}, \qquad (3.26)$$

де $\boldsymbol{\varepsilon}(t) = \mathbf{x}(t) - \hat{\mathbf{x}}(t)$.

Отже, ми по суті призначили всі полюси замкненої системи незалежним розташуванням в бажаних місцях комплексної площини власних значень матриць **A** – **B**·**K** та **A** – **L**·**C**.

Сформувати компенсатор з регулятора та спостерігача можна за допомогою функції reg.

Приклад 3.4.

Спроектувати регулятор стану для об'єкта з передатною функцією W_o (див. попередній приклад). Представимо модель в просторі станів:

```
» Px=ss(Wo);
» [A,B,C]=ssdata(Px);
» size(A)
ans = 3     3
```

Модель має третій порядок. Розрахуємо матрицю **K** коефіцієнтів зворотного зв'язку, задавши полюси замкненої системи: –2 та –1.5±0.5·*j*.

```
» p1=[-2 -1.5+j*0.5 -1.5-j*0.5];
» k=place(A,B,p1);
place: ndigits= 15
```

Розрахуємо матрицю **L** коефіцієнтів зворотного зв'язку спостерігача, задавши полюси спостерігача: –1 та –1.5±0.2·*j*.

```
» p2=[-1 -1.5+j*0.2 -1.5-j*0.2];
» l=place(A',C',p2);
place: ndigits= 15
```

Сформуємо компенсатор:

```
» Regx=reg(ss(Wo),k,l');
```

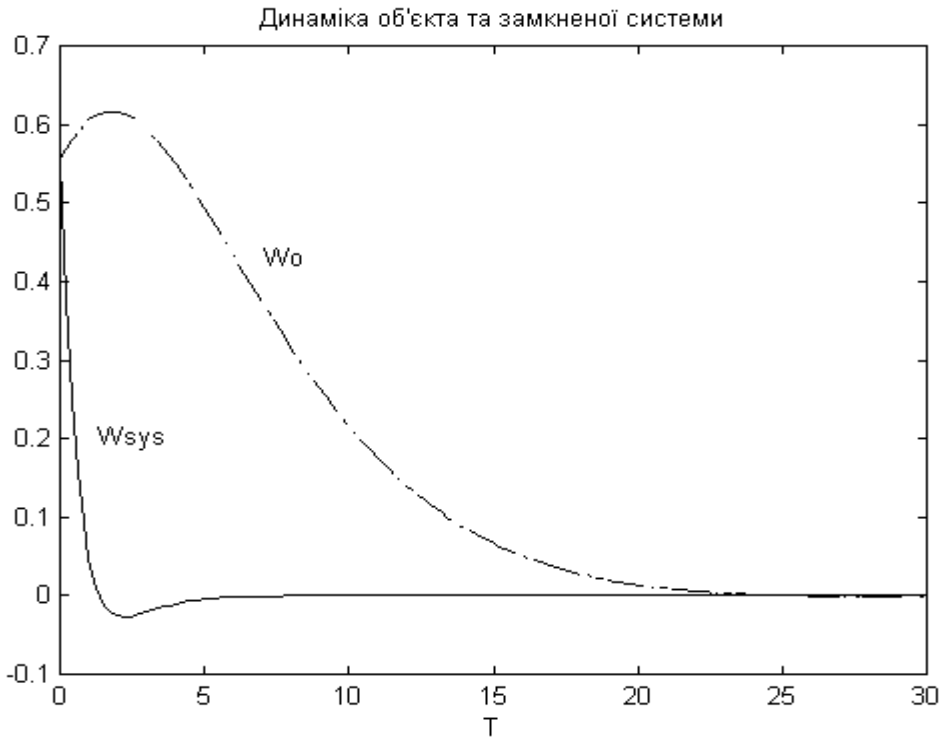

Рис. 3.15.

Порівняємо перехідні процеси в об'єкті та замкненій системі з компенсатором при довільному початковому стані (рис. 3.15.):

```
» Wsys = feedback(Px,Regx,+1);  % Модель замкненої системи
» t = 0:0.2:30;                 % Час моделювання
» z = zeros(length(t),1);       % Нульове збурення
» Xo1=2*rand(3,1);              % Початкові умови для об'єкта
» Xo2=[Xo1;2*rand(3,1)];        % Початкові умови для системи
» [Y1,T1,X1]=lsim(Px,z,t,Xo1);  % Динаміка об'єкта
» [Y2,T2,X2]=lsim(clx,z,t,Xo2); % Динаміка замкненої системи
» plot(T1,Y1,'--',T2,Y2,'-')
» title('Динаміка об''єкта та замкненої системи')
» xlabel('T')
```

Порівняємо динаміку першої змінної стану x_1 та її оцінки $\hat{x}_1$ (рис. 3.16.):

```
» diap=1:30;                          % 6с (30 відрахунків по 0.2с)
» plot(T2(diap),X2(diap,1),T2(diap),X2(diap,4));
» title('Змінна стану x1 та її оцінка')
» xlabel('T')
```

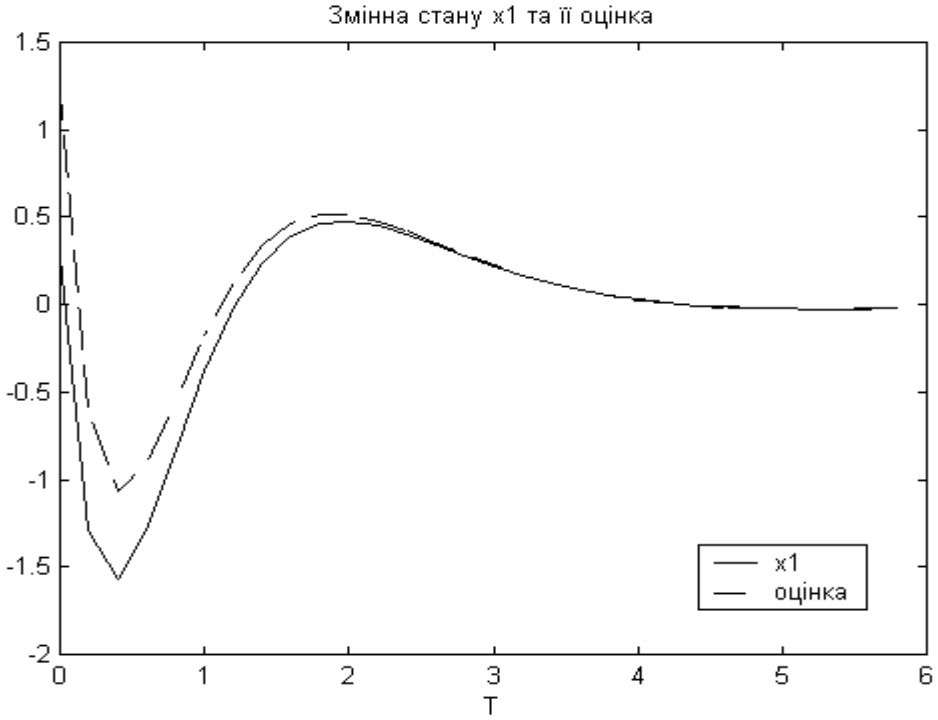

Рис. 3.16.

Для порівняння побудуємо імпульсні перехідні характеристики об'єкта та замкненої системи з компенсатором (рис. 3.17.):

```
» impulse(Wo,Wsys)
```

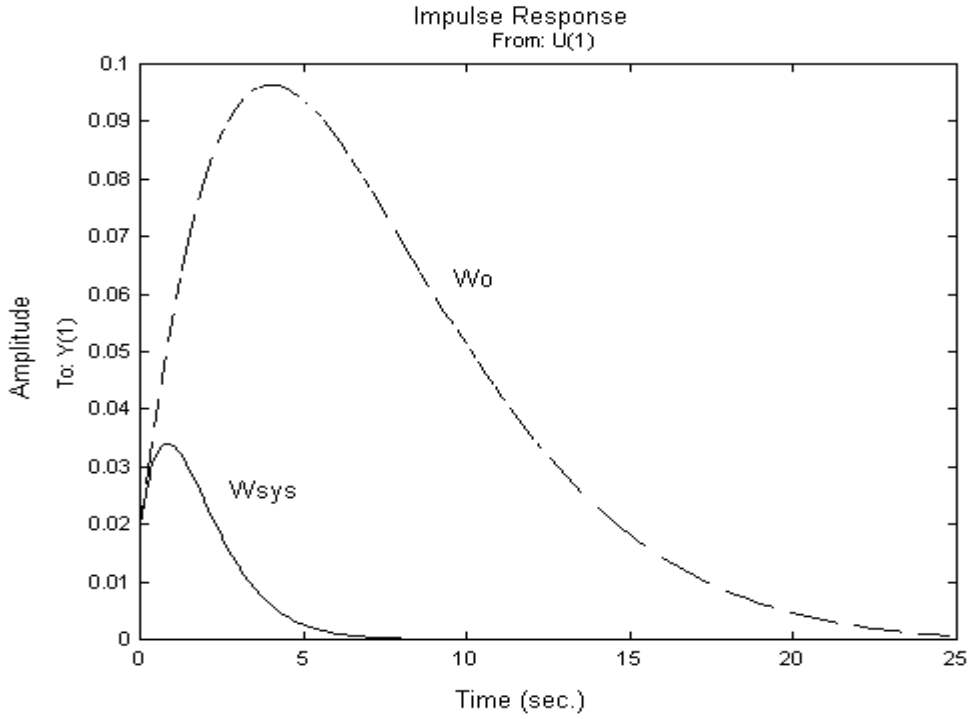

Рис. 3.17.

Кінець приклада.

3.4.3. Загальні зауваження

Система може виявитись погано обумовленою, якщо вибрати нереалістичне розташування полюсів. На практиці треба уникати:

– розташування кількох полюсів в одному місці;

– переміщення полюсів, які є слабо контрольованими або такими, що слабо спостерігаються. Це, здебільшого, потребує великих коефіцієнтів зворотного зв’язку, що робить замкнену систему чутливою до збурень.

Розділ 4. Імітаційне моделювання

Імітаційним моделюванням називають розробку та спостереження за поведінкою моделі системи, що зазнає вплив вхідних сигналів, які можуть мати випадковий характер [24].

В цьому розділі розглянуто способи побудови моделей динамічних систем у вигляді структурних схем та дослідження їх поведінки в умовах, близьких до реальних за допомогою інструментального засобу візуального моделювання Simulink [24; 25; 28].

4.1. Побудова моделей динамічних об'єктів

4.1.1. Зображення об'єктів в безперервному часі

Будь-який динамічний об'єкт (або систему об'єктів), що описується лінійними диференційними рівняннями, в безперервному часі можна задати структурною схемою (блок-діаграмою) за допомогою блоків Integrator, Derivative та Zero-Pole або Transfer Fcn, що містяться в підрозділі Continuous бібліотеки Simulink (див. додаток 3.1.).

Крім того динамічні об'єкти зручно задавати засобами командного режиму MatLab, як lti-об'єкти (див. розділ 1.), за допомогою блоку LTI System, що міститься в бібліотеці Control System Toolbox (див. додаток 3.4.).

Приклад 4.1.

Зобразимо об'єкт в безперервному час, що описується передатною функцією $W_2(s)$ (див. приклад 1.4.) та побудуємо його перехідну характеристику. Для цього в якості джерела вхідного сигналу треба використовувати блок Step, що міститься в підрозділі Sources бібліотеки Simulink.

Динаміку об'єкта задамо кількома способами (див. рис. 4.1.). Вибір варіанта об'єкта для побудови перехідної характеристики на діаграмі

здійснюється за допомогою блоку Switch (підрозділ Nonlinear бібліотеки Simulink).

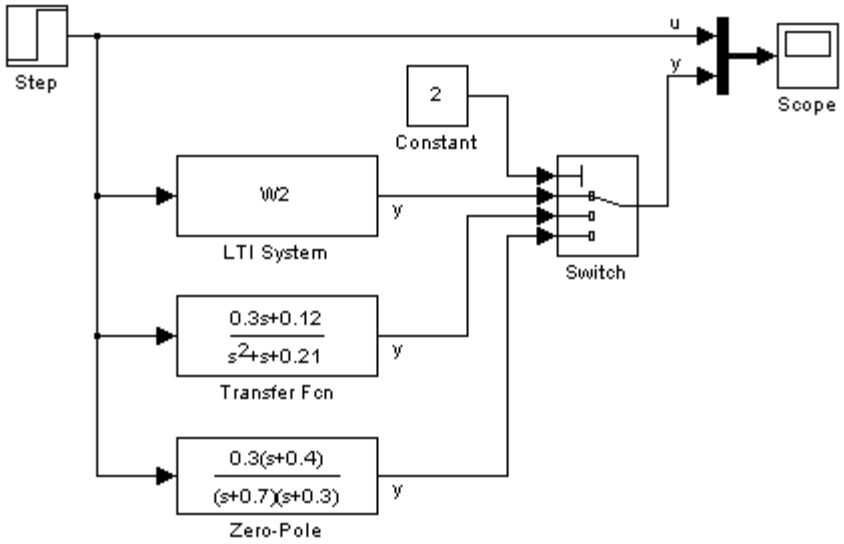

Рис. 4.1.

```
» W2=zpk(-0.4,[-0.7 -0.3],0.3)
Zero/pole/gain:
  0.3 (s+0.4)
---------------
(s+0.7) (s+0.3)
» W2=tf(W2)                          % В поліноміальній формі
Transfer function:
 0.3 s + 0.12
--------------
s^2 + s + 0.21
```

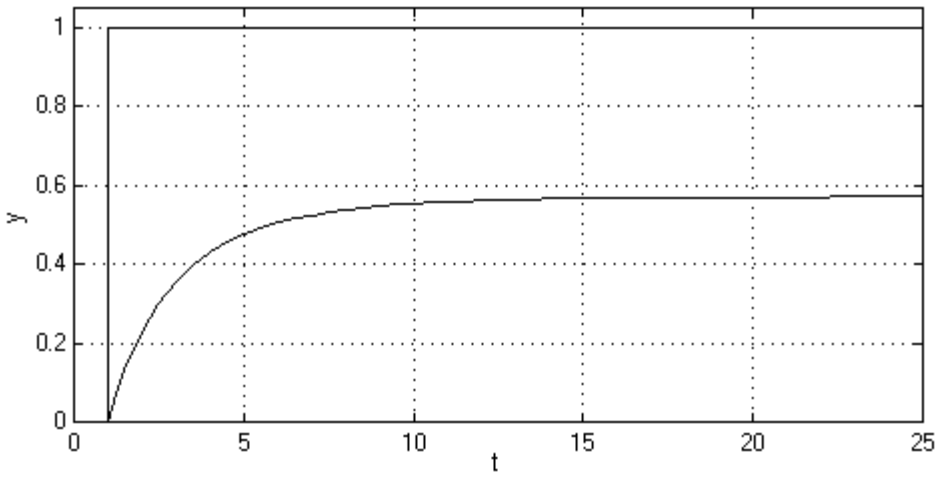

Рис. 4.2.

Параметри блоків Zero-Pole та Transfer Fcn задаються подібно відповідним параметрам функцій zpk та tf (див. п. 1.2.1.). В якості параметра блоку LTI System вказано ім'я змінної *W*2.

Для побудови перехідної характеристики об'єкта використовується блок Scope (підрозділ Sinks бібліотеки Simulink), вікно якого зображено на рис. 4.2.

Кінець приклада.

4.1.2. Зображення дискретних об'єктів

Блок-діаграми дискретних об'єктів будуються подібно діаграмам об'єктів в безперервному часі. Блоки Discrete Time Integrator, Discrete Zero-Pole та Discrete Transfer Fcn містяться в підрозділі Discrete бібліотеки Simulink. Блок Discrete Transfer Fcn дозволяє задавати дискретну передатну функцію від z. Дискретну передатну функцію від z^{-1} (див. п. 1.2.2.) можна задати за допомогою блоку Discrete Filter.

Для побудови діаграм дискретних об'єктів можна також використовувати блок LTI System.

Приклад 4.2.

Зобразимо дискретний аналог об'єкта з попереднього приклада для періода дискретизації $T = 1$ та побудуємо його перехідну характеристику. Динаміку об'єкта задамо за допомогою блоку LTI System (рис. 4.3.).

```
» G=c2d(W2,1)
Zero/pole/gain:
  0.22661 (z-0.671)
---------------------
(z-0.4966) (z-0.7408)
Sampling time: 1
```

Слід відзначити, що початок нанесення одиничного ступінчастого збурення в моделі, як і в попередньому прикладі, – 1 с (параметр *Step time* блоку Step). Тобто одиничне збурення виникає в системі після першого крока дискретизації. Перехідна характеристика об'єкта зображена на рис. 4.4.

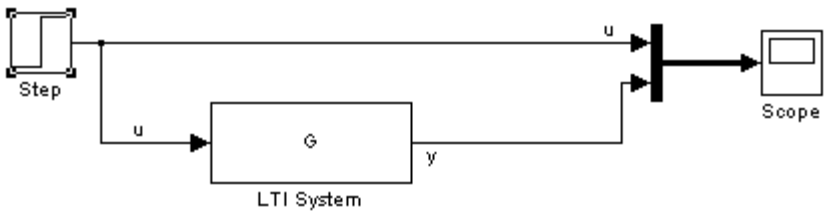

Рис. 4.3.

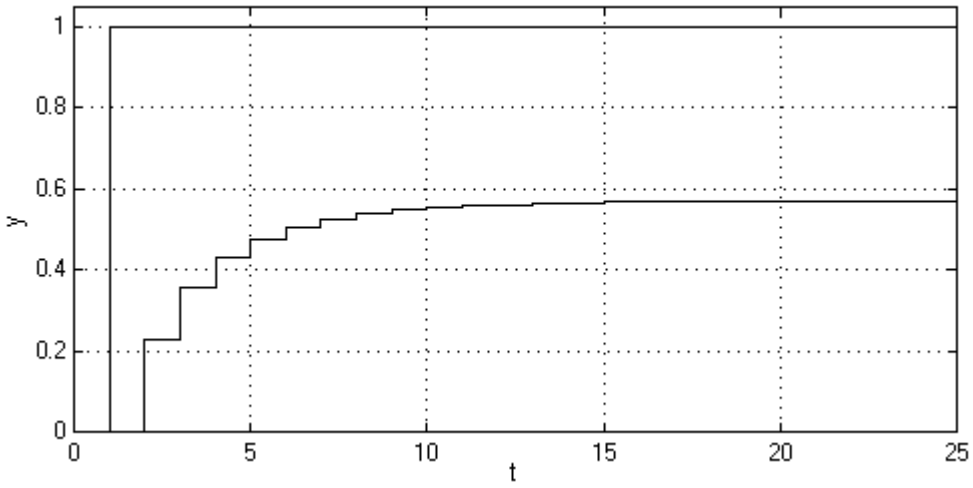

Рис. 4.4.

Кінець приклада.

4.1.3. Зображення об’єктів з запізнюванням

Запізнювання в об’єктах в безперервному часі задають за допомогою блоків Transport Delay та Variable Transport Delay (підрозділ Continuous бібліотеки Simulink).

Для дискретних об’єктів затримка на один період дискретизації може бути реалізована блоком Unit Delay (підрозділ Discrete бібліотеки Simulink).

При зображенні об’єктів з запізнюванням за допомогою блоку LTI System слід пам’ятати, що властивість lti-об’єкта *ioDelayMatrix* системою Simulink ігнорується. Тому запізнювання слід задавати властивістю *InputDelay* або *OutputDelay* (див. параграф 1.4.).

Приклад 4.3.

Побудуємо модель об’єкта з запізнюванням, що описується передатною функцією $W_4(s)$ (див. приклад 1.11). Динаміку об’єкта задамо кількома

способами (див. рис. 4.7.). Вибір варіанта об’єкта для побудови перехідної характеристики на діаграмі здійснюється за допомогою блоку Manual Switch (підрозділ Nonlinear бібліотеки Simulink).

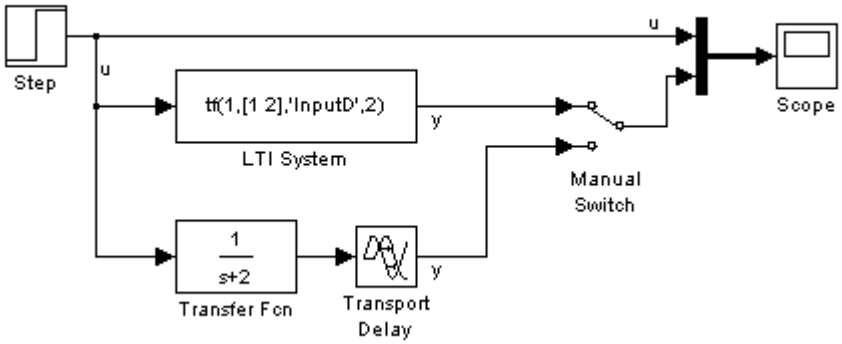

Рис. 4.5.

Перехідна характеристика об’єкта з запізнюванням зображена на рис. 4.6.

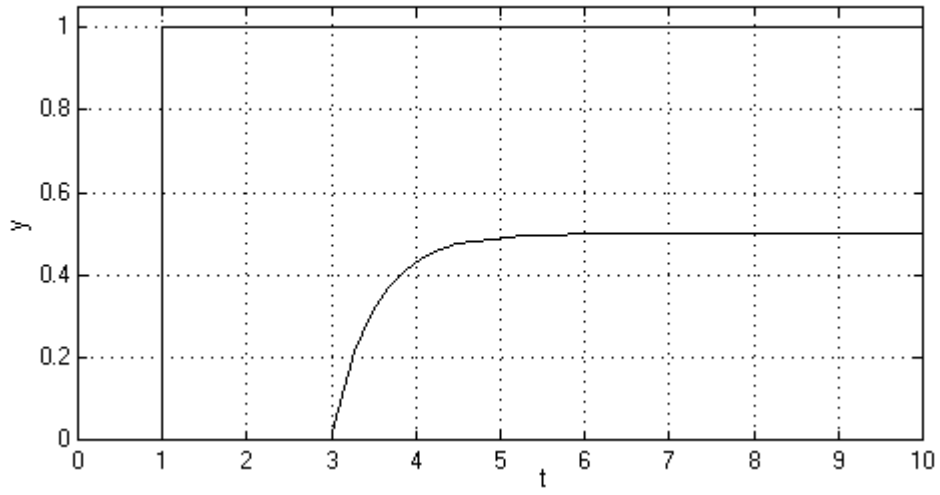

Рис. 4.6.

Кінець приклада.

4.1.4. Зображення об’єктів в просторі станів

Об’єкти в безперервному часі та дискретні об’єкти в просторі станів, що описуються системами рівнянь виду (1.24) та (1.27) можна побудувати, як блок-схеми матричних виразів за допомогою блоків інтегрування (див. п.п. 4.1.1. та 4.1.2.) та блоків Gain і Matrix Gain, що містяться в підрозділі Math бібліотеки Simulink (див. додаток 3.1.).

Більш зручним є зображення об’єктів в просторі станів за допомогою спеціалізованих блоків State-Spase (підрозділ Continuous бібліотеки Simulink) та Discrete State-Spase (підрозділ Discrete бібліотеки Simulink), або за допомогою блоку LTI System.

Приклад 4.4.

Зобразимо об’єкт $W_2(s)$ (див. приклад 4.1.) в просторі станів різними способами.

```
» W2=ss(W2)
a =
                      x1           x2
           x1         -1        -0.21
           x2          1            0
b =
                      u1
           x1      0.54772
           x2            0
c =
                      x1           x2
           y1      0.54772      0.21909
 d =
                      u1
           y1            0
Continuous-time model.

» [A B C D]=ssdata(W2)
A =
   -1.0000   -0.2100
    1.0000         0
B =
    0.5477
         0
C =
    0.5477    0.2191
D = 0
```

Параметри блоку State-Spase задамо відповідними матрицями рівнянь стану. Блок-діаграма такого об’єкта зоблажена на рис. 4.7.

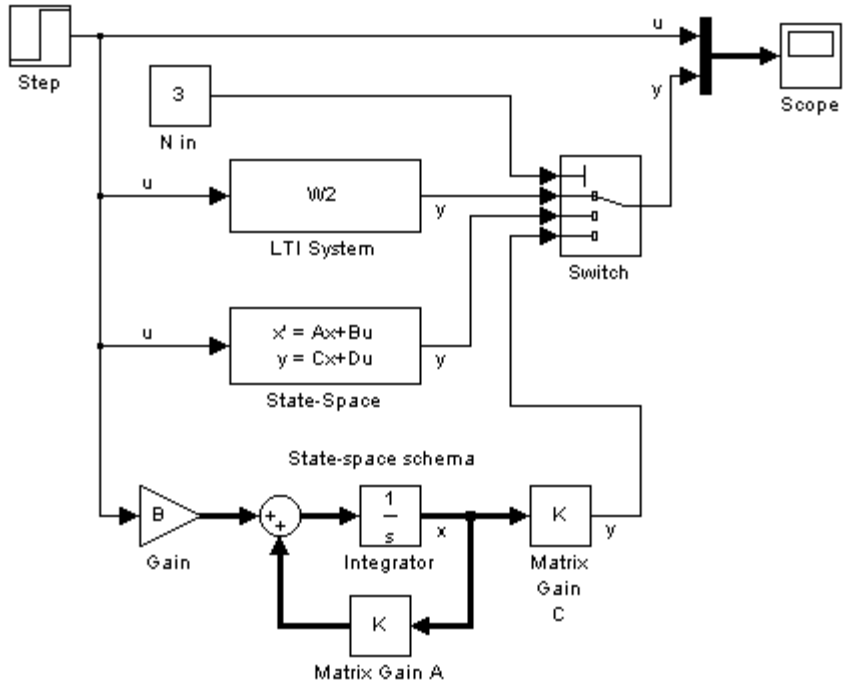

Рис. 4.7.

Всі три варіанти дають однакові перехідні характеристики, аналогічні зображеній на рис. 4.2.

Кінець приклада.

4.2. Організація обміну даними

Можливість обміну інформацією між блок-схемою Simulink та системою MatLab значно розширює ефективність імітаційного моделювання. Ці засоби дають можливість, наприклад, вводити початкові умови моделювання та поточні дані з файлу або безпосередньо з робочого простору (WorkSpace) командного вікна MatLab, або виводити результати розрахунків для подальшого їх аналізу іншими інструментами MatLab.

4.2.1. Введення даних

Ввести в діаграму значення скалярної або векторної змінної з WorkSpace, а також розрахувати значення функції MatLab від константи можна за

допомогою блоку Constant (підрозділ Sources бібліотеки Simulink). Параметром цього блоку є константний вираз.

Ввести значення змінних з WorkSpace можна також за допомогою блоків Fcn та MATLAB Fcn (підрозділ Functions & Tables бібліотеки Simulink). Параметром блоку Fcn є вираз, записаний в стилі мови С, а параметром блоку MATLAB Fcn – вираз MatLab. До складу цих виразів можуть входити змінні з WorkSpace.

В тому ж підрозділі містяться блоки Look-Up Table та Look-Up Table (2-D), які дозволяють вводити в діаграму значення функції, заданої одно- та двовимірною таблицею. В якості такої таблиці можна використовувати змінні MatLab (вектори та матриці). При організації введення значень часової функції (наприклад, результатів вимірювання часових характеристик) на вхід блоку треба подати сигнал від таймера: блоки Clock та Digital Clock, що містяться в підрозділі Sources бібліотеки Simulink (див. приклад 4.7.).

Для введення даних, що залежать від часу, Simulink має спеціалізовані засоби: блоки From File та From Workspace, що містяться в підрозділі Sources бібліотеки Simulink, та блок In, що міститься в підрозділі Signals & Systems бібліотеки Simulink.

Блок From File отримує інформацію із заданого файлу даних MatLab (*.mat), який має містити матрицю зі структурою:

$$\mathbf{M}_{ff} = \begin{bmatrix} \mathbf{T} \\ \mathbf{U} \end{bmatrix}, \tag{4.1}$$

де вектор часу має вид:

$$\mathbf{T} = \begin{bmatrix} t_1 & t_2 & \dots & t_n \end{bmatrix}, \tag{4.2}$$

а матриця даних визначається як:

$$\mathbf{U} = \begin{bmatrix} u_1(t_1) & u_1(t_2) & \dots & u_1(t_n) \\ u_2(t_1) & u_2(t_2) & \dots & u_2(t_n) \\ \vdots & \vdots & \ddots & \vdots \\ u_m(t_1) & u_m(t_2) & \dots & u_m(t_n) \end{bmatrix}. \tag{4.3}$$

Тут m – кількість параметрів, що вводиться, а n – кількість відрахунків (див. приклад 4.9.).

Блок From Workspace отримує інформацію із заданого двовимірного масиву зі структурою:

$$\mathbf{M}_{fw} = [\mathbf{T}^{\mathrm{T}} \quad \mathbf{U}^{\mathrm{T}}]. \tag{4.4}$$

Аналогічну дію можуть виконувати блоки In. Всі блоки In верхнього рівня діаграми обслуговує одна матриця $\mathbf{M}_{\mathrm{in}}$ зі структурою (4.4), яку можна задати на сторінці Workspace I/O вікна Simulation Parameters (параметр *Input*), що викликається командою Parameters... з меню Simulation діаграми. При цьому сумарна кількість входів діаграми (які забезпечують блоки In) повинна дорівнювати m (див. приклад 4.8.).

4.2.2. Виведення даних

Для виведення результатів моделювання з блок-діаграми Simulink в MatLab передбачені блоки Scope, To File та To Workspace, які містяться в підрозділі Sinks бібліотеки Simulink, та блок Out, що міститься в підрозділі Signals & Systems бібліотеки Simulink.

Блок To File записує інформацію у заданий файл. Формат запису співпадає з форматом (4.1) (див. блок From File).

Блок To Workspace формує в робочому просторі MatLab матрицю із заданим ім’ям та структурою:

$$\mathbf{M}_{tw} = \mathbf{U}^{\mathrm{T}}. \tag{4.5}$$

Слід відзначити, що вектор часу при виведенні інформації у вигляді матриці цим блоком не формується.

Для організації виводу даних через блок Scope необхідно задати ім’я змінної MatLab із структурою (4.4) (див. блок From Workspace) на сторінці Data history вікна Properties блоку (див. приклад 4.16.).

При виведенні даних через блоки Out верхнього рівня діаграми треба задати окремо ім’я вектора часу:

$$\mathbf{T}_{out} = \mathbf{T}^{\mathrm{T}}, \tag{4.6}$$

та ім'я матриці даних $\mathbf{M}_{out}$ із структурою (4.5) (див. блок To Workspace) на сторінці Workspace I/O вікна Simulation Parameters, що викликається командою Parameters... з меню Simulation діаграми. Кількість стовпців $\mathbf{M}_{out}$ дорівнює m - сумарній розмірності всіх виходів діаграми (які забезпечують блоки Out).

Для блоків виведення даних можна задати додаткові параметри:

Maximum number of rows (для блоків Out та To Workspace) і *Limit rows to last* (для блоку Scope) – максимальна кількість відрахунків, що записується в матрицю. Якщо загальна кількість відрахунків перевищує значення параметра, то виводиться вказана кількість останніх відрахунків.

Decimation (для всіх блоків виведення даних) – проріджження, тобто виводиться тільки кожен k-ий відрахунок (де k – фактор проріджження). Для блоку Scope замість параметра *Decimation* можна задати параметр *Sample time*, який дозволяє виводити дані з бажаною дискретністю (див. також п. 4.5.1.).

4.2.3. Лінеаризація моделей

Системи Simulink дозволяє отримати лінійну (або лінеаризовану) модель в просторі станів з імітаційної моделі, заданої у вигляді блок-діаграми. Такий підхід дає змогу розробити складну динамічну модель засобами візуального програмування, а потім здійснити її аналіз за допомогою числених інструментів, що їх надають бібліотеки MatLab.

Для лінеаризації імітаційних моделей в безперервному часі використовується функція linmod. Дискретну імітаційну модель можна лінеаризувати за допомогою функції dlinmod. Перед початком лінеаризації на блок-діаграмі треба відімкнути блоки, що генерують вхідні сигнали, та задати входи та виходи системи за допомогою блоків In та Out.

Якщо блок-діаграма містить нелінійні блоки, то при виклику функцій linmod та dlinmod треба вказуват вхідний вектор та вектор стану, що визначають точку, відносно якої здійснюється лінеаризація.

Якщо блок-діаграма містить блоки Derivative або Transport Delay, то перед початком лінеаризації їх необхідно замінити на відповідні блоки Switched derivative та Switched transport delay, що міститься в підрозділі Linearization бібліотеки Simulink Extras (див. додаток 3.2.). Крім того функцію блоку Derivative можна спробувати врахувати в інших динамічних блоках. Слід пам'ятати, що запізнювання, задане блоком LTI System, лінеаризації також не підлягає.

Приклад 4.5.

Отримаємо формальну модель з блок-діаграми об'єкта з запізнюванням, побудованої в прикладі 4.3. На діаграмі (рис. 4.8.) для порівняння зображено два варіанти об'єкта з запізнюванням. Ім'я блок-діаграми – *delay*.

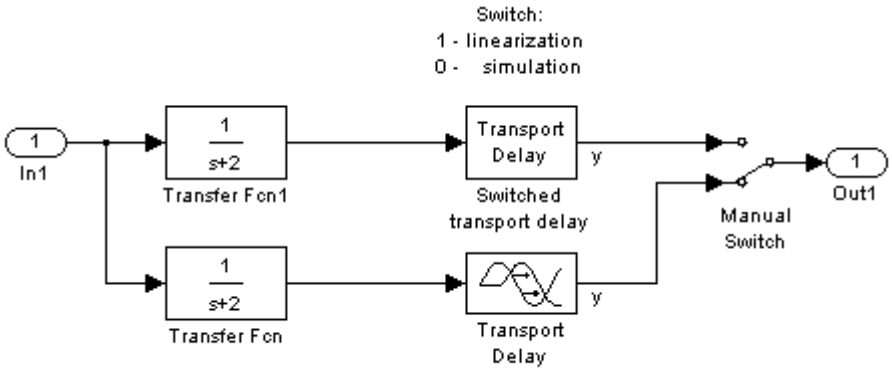

Рис. 4.8.

Встановимо перемикач на діаграмі у верхню позицію.

```
» [a1 b1 c1 d1]=linmod('delay');
» Wd1=ss(a1,b1,c1,d1);
```

Тепер встановимо перемикач у нижню позицію.

```
» [a2 b2 c2 d2]=linmod('delay');
» Wd2=ss(a2,b2,c2,d2);
» step(Wd1,Wd2,7)
```

Модель *Wd*1 є результатом лінеаризації першого варіанта побудови об'єкта, а *Wd*2 – другого. Перехідні характеристики, побудовані по отриманим моделям, зображені на рис. 4.9. (порівняйте з рис. 4.6.). Як видно з рисунка, лінеаризація імітаційної моделі, що містить блок Transport Delay є неадекватною.

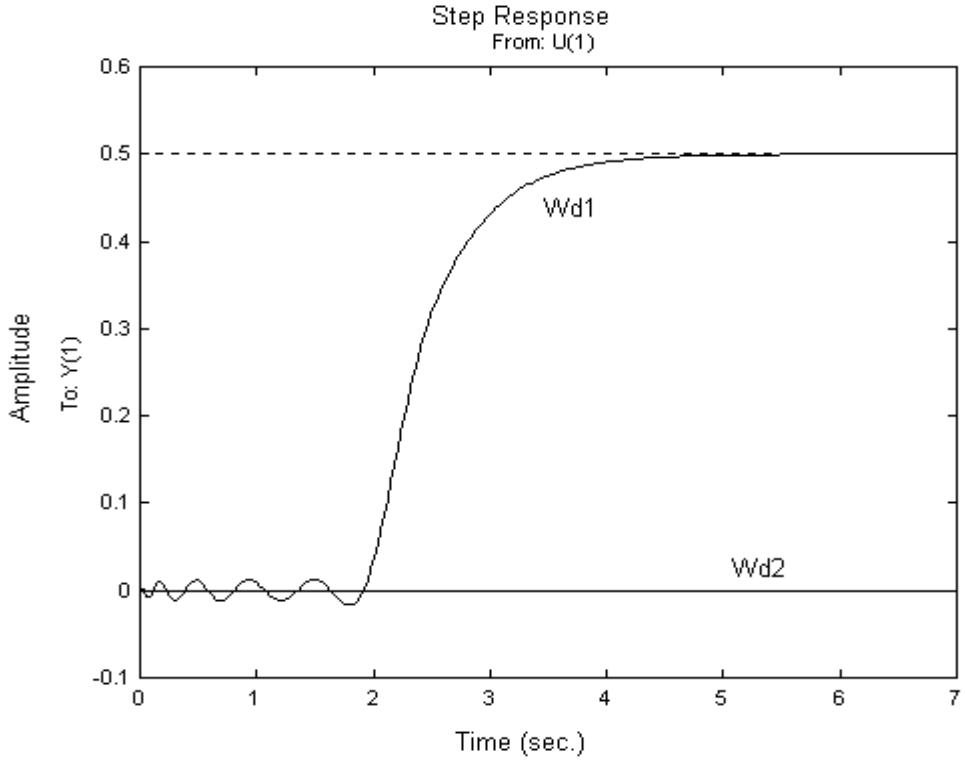

Рис. 4.9.

Кінець приклада.

В інший спосіб отримати формальну модель з блок-діаграми можна за допомогою Simulink LTI Viewer, який викликається командою Linear Analysis з меню Tools діаграми і відрізняється від LTI Viewer (див. п. 1.7.5.) додатковим меню – Simulink.

Отримати лінійну модель з длок-діаграми та завантажити її в Simulink LTI Viewer можна за допомогою команди Get Linearized Model з меню Simulink. Після цього модель можна передати в робочий простір MatLab у вигляді змінної (lti-об'єкта) в ss-формі (див. п. 1.2.4.) за допомогою команди Export... з меню File в'ювера.

Перед початком лінеаризації необхідно визначити на діаграмі входи та виходи моделі за допомогою блоків Input Point та Output Point з бібліотеки Control System Toolbox.

Якщо імітаційна модель є нелінійною, то параметри лінеаризації можна задати командою Set Operating Point... з меню Simulink в'ювера.

Приклад 4.6.

Отримаємо формальну модель з блок-діаграми, наведеної в прикладі 4.4. Для цього виділемо, наприклад, фрагмент діаграми з побудовою моделі об'єкта у вигляді матричного вираза та визначимо вхід та вихід моделі відповідними блоками. Змінена діаграма зображена на рис. 4.10. (ім'я *linss*) а перехідна характеристика, побудована в'ювером по отриманій моделі, на рис. 4.11.

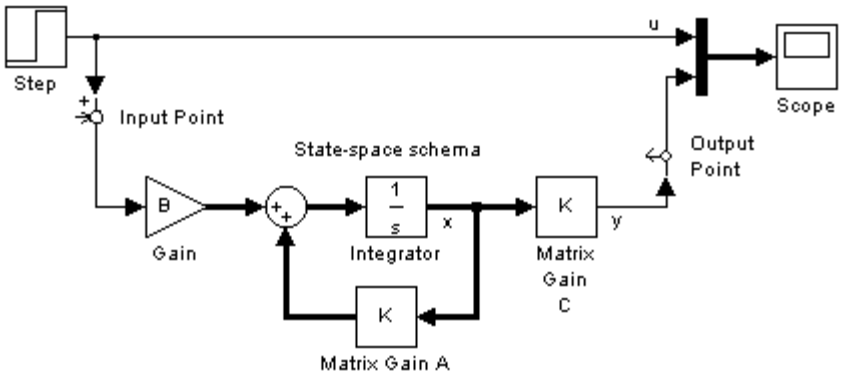

Рис. 4.10.

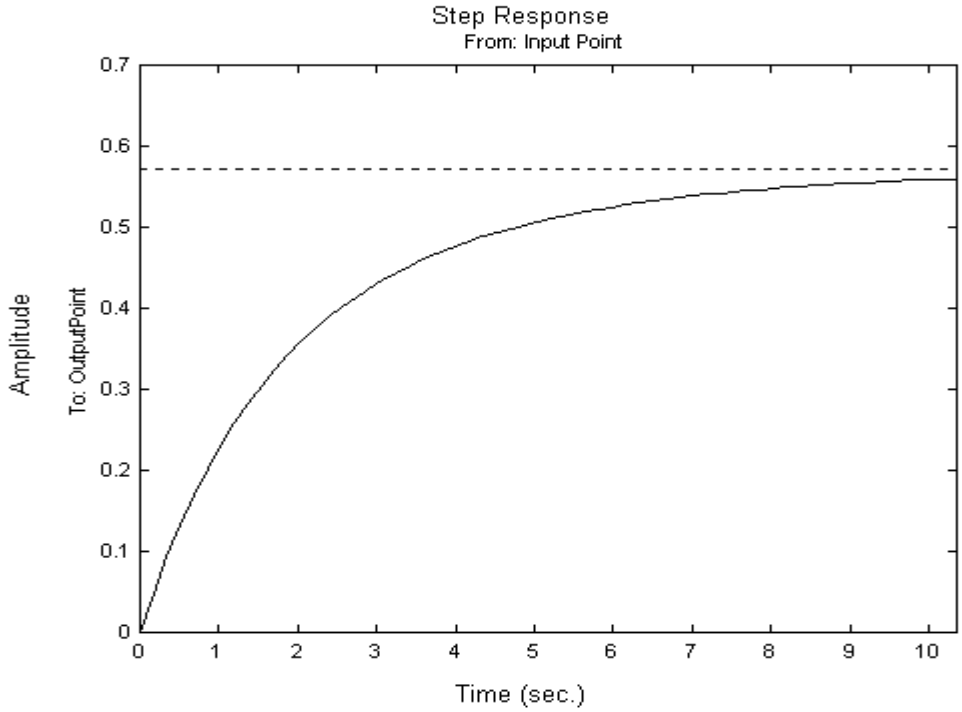

Рис. 4.11.

За допомогою команди Export... в'ювера отримаємо модель *linss*_1, яку представимо у вигляді передатної функції:

```
» tf(linss_1)
```

```
Transfer function from input "InputPoint" to output "OutputPoint":
 0.3 s + 0.12
--------------
s^2 + s + 0.21
```

Результат повністю збігається з вихідною передатною функцією $W_2(s)$ (див. приклад 4.1.).

Кінець приклада.

4.3. Ідентифікація лінійних динамічних об’єктів

Бібліотека System Identification Toolbox надає низку блоків для визначення моделі динамічного об’єкта безпосередньо в блок-діаграмі Simulink (див. додаток 3.5.). Ці блоки реалізовано для найбільш вживаних структур моделей. Блоки використовують рекурентні алгоритми ідентифікації.

Вхідними сигналами блоків можуть слугувати результати вимірювань входів та виходів реальних динамічних систем. Результатом ідентифікації є модель у вигляді дискретної передатної функції, яка виводиться в командне вікно MatLab через задану кількість кроків оцінювання (параметр *Calculate after how many points*). Під час ідентифікації в окремому графічному вікні блоки будують гафіки вихідних сигналів системи і моделі, та остаткової похибки моделі (див. п. 2.8.2.).

Приклад 4.7.

Ідентифікуємо об’єкт у вигляді OE-моделі (2.29) за результатами вимірювань його вхідних та вихідних сигналів. Для цього скористаємося матрицею даних *D*1*e* (розмір 300×2), отриманою в прикладі 2.3. Перший її стовпець є вектором вихідних сигналів об’єкта, а другий – вхідних сигналів.

Введення даних організуємо за допомогою блоків Look-Up Table (див. п. 4.2.1.). Тоді значення параметра *Vector of output values* блоку для введення вектора вхідних сигналів треба задати як *D*1*e*(1:300,2) а блоку для введення вектора вихідних сигналів – як *D*1*e*(1:300,1). Параметр *Vector of input values* обох блоків задамо як [1:300].

Для оцінювання оберемо універсальний блок ідентифікації General model estimator using Predictive Error Method. Його параметр *Orders of model* [*na nb nc nd nf nk*] задамо як [0 2 0 0 3 1] (див. приклад 2.8.).

Змінемо модельний час діаграми: привласнимо параметру *Stop time* діаграми значення 300 с (див. п. 4.5.1.). Діаграму зображено на рис. 4.12.

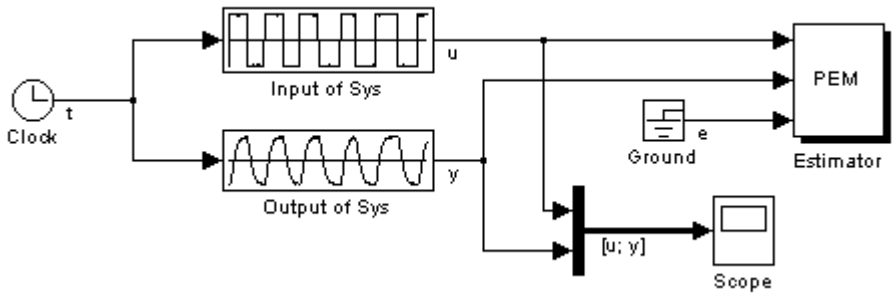

Рис. 4.12.

Часові залежності даних для ідентифікації можна бачити у вікні блоку Scope, яке зображено на рис. 4.13.

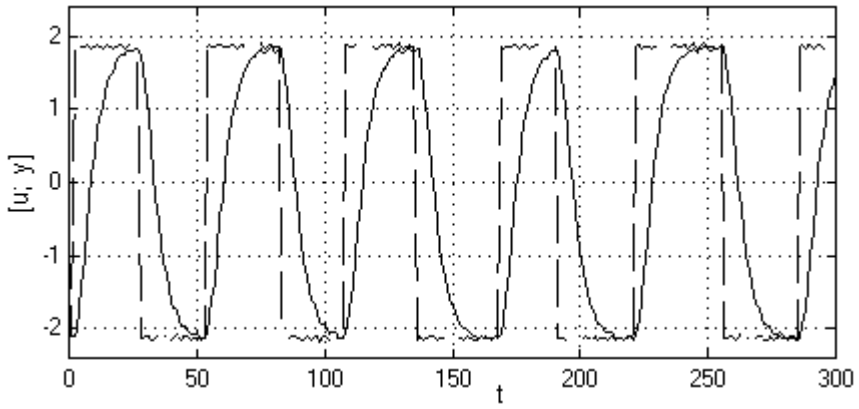

Рис. 4.13.

Гафіки вихідних сигналів системи і отриманої моделі, та остаткової похибки моделі зображені на рис. 4.14. Передатна функція OE-моделі виводиться в командне вікно MatLab тільки після закінчення оцінювання оскільки параметр *Calculate after how many points* задано рівним часу моделювання .

```
» Transfer function:
num/den =
          0.036902 z^2 + 0.013339 z
   -------------------------------------
   z^3 - 1.5493 z^2 + 0.57005 z + 0.030795
Noise model:
num/den =
   1
   -
   1
```

Кінець приклада.

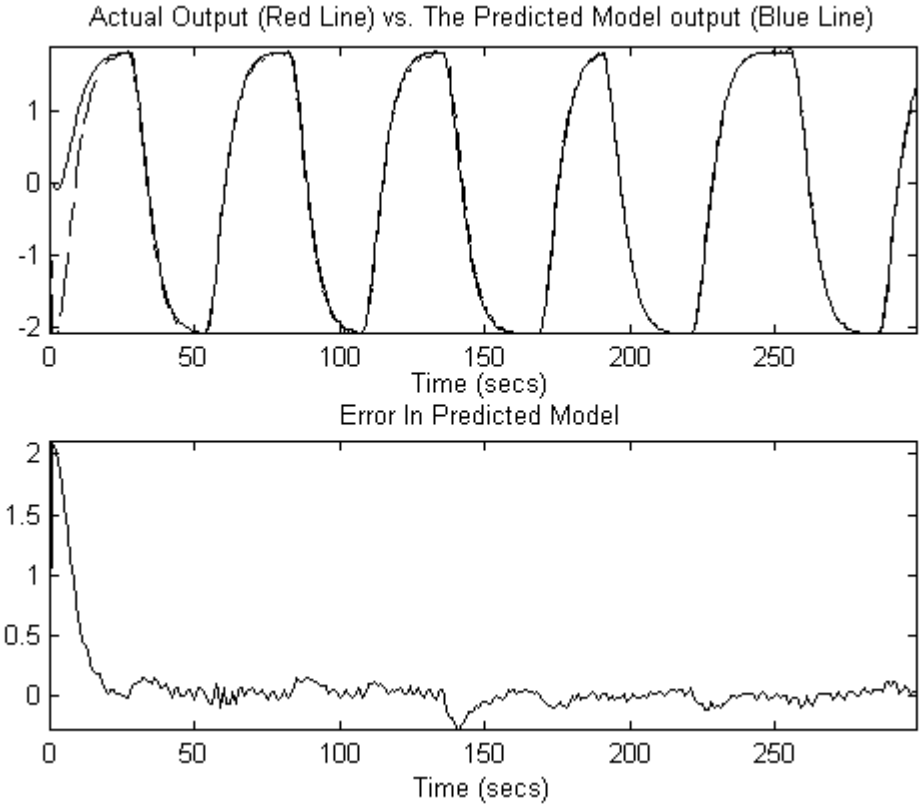

Рис. 4.14.

Приклад 4.8.

Повторимо ідентифікацію об’єкта (див. попередній приклад), використовуючи для оцінювання спеціалізований блок Output-error model estimator. Його параметр *Orders of model* [*nb nf nk*] задамо як [2 3 1] (див. приклад 2.9.). Діаграму зображено на рис. 4.15. Ім’я блок-діаграми – *id*2.

Для введення інформації з матриці даних застосуємо блок In. Для цього привласнимо параметру *Input* діаграми (див. п. 4.2.1.) матрицю в форматі (4.4): [[1:300]',*D*1*e*]. Зміну розташування сигналів у векторній лінії зв'язку (для узгодження із входами блоку оцінювання) виконаємо за допомогою блоку Selector, що міститься в підрозділі Signals & Systems бібліотеки Simulink.

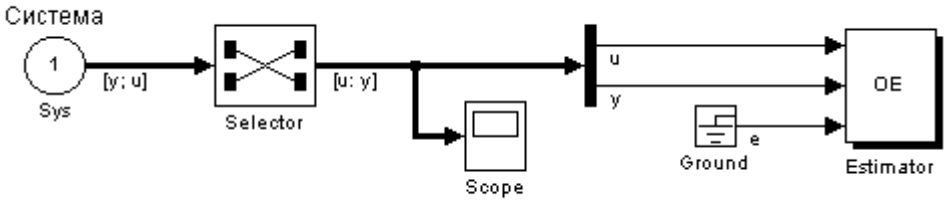

Рис. 4.15.

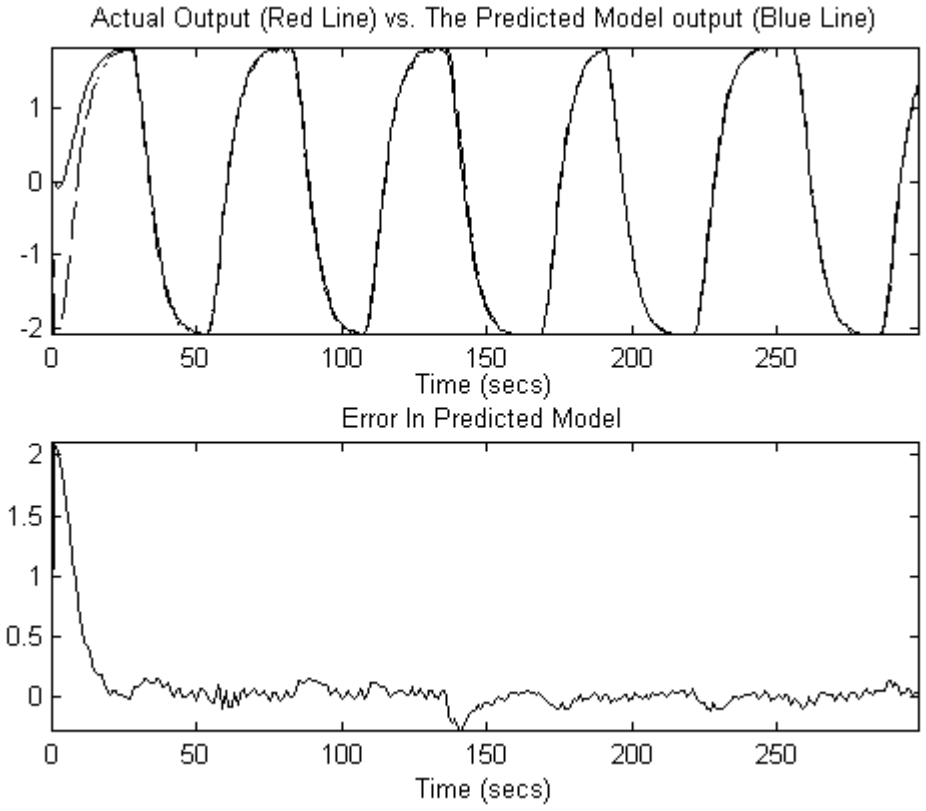

Рис. 4.16.

Часові залежності даних для ідентифікації (вікно блоку Scope) зображено на рис. 4.13. Графіки вихідних сигналів системи і оціненої моделі, та остаткової похибки моделі зображені на рис. 4.16, а сама модель має вигляд:

```
Transfer function:
num/den =
        0.041845 z^2 - 0.0059326 z
   ----------------------------------
   z^3 - 1.83 z^2 + 1.0125 z - 0.14567
Noise model:
num/den =
   1
   -
   1
```

Кінець приклада.

Порівняйте результати в прикладі 4.8. з результатами в прикладі 4.7. а також в прикладах 2.8. та 2.9.

Приклад 4.9.

Повторимо ідентифікацію об’єкта (див. попередній приклад), використовуючи інші засоби введення даних (див. п. 4.2.1.).

При введенні даних за допомогою блоку From Workspace в якості його параметра *Data* задамо матрицю в форматі (4.4): [[1:300]’,fliplr(*D*1*e*)]. Тут функція fliplr, що переставляє стовпці матриці симетрично відносно вертикальної осі, використана для узгодження із входами блоку оцінювання.

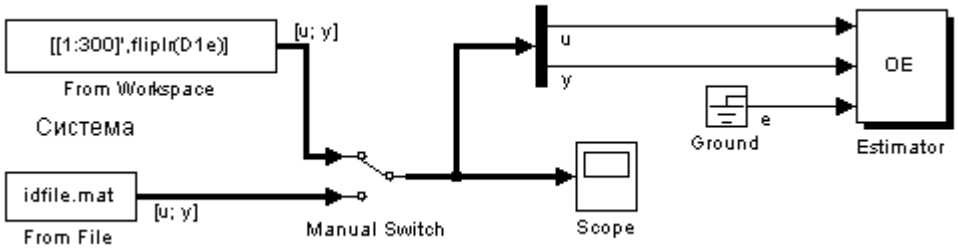

Рис. 4.17.

Для організації введення даних за допомогою блоку From File необхідно сформувати файл (idfile.mat) з матрицею в форматі (4.1), ім’я якого треба задати в якості параметра *File name* блоку:

```
» idf=[[1:300];fliplr(D1e)'];
» save idfile idf
```

Блок-діаграма зображена на рис. 4.17. Результати імітації за будь-яким варіантом повністю співпадають з результатами в попередньому прикладі.

Кінець приклада.

4.4. Імітація замкнених динамічних систем

4.4.1. Система з ПІ-регулятором

ПІ-регулятор можна сформувати в блок-діаграмі як паралельне з'єднання інтегратора та підсилюючої ланки, або одним з блоків, що реалізують передатну функцію динамічного об'єкта (див. (3.6)).

Крім того можна використати блок PID Controller або PID Controller (with Approximate Derivative), що міститься в підрозділі Additional Linear бібліотеки Simulink Extras (див. додаток 3.2.). Параметрами цих блоків є коефіцієнти:

$$\begin{aligned} P &= k_p; \\ I &= k_0 = \frac{1}{T_{in}} = \frac{k_p}{T_i}; \\ D &= T_{df} = T_v \cdot k_p = 0, \end{aligned} \tag{4.7}$$

що визначають пропорційну, інтегральну та диференційну складові відповідно (див. параграф 3.2.).

Приклад 4.10.

Зобразимо у вигляді блок-діаграми ПІ-регулятор (спроектований в прикладі 3.2.) різними способами та побудуємо його перехідну характеристику за допомогою блоку XY Graph, що міститься в підрозділі Sinks бібліотеки Simulink.

ПІ-регулятор описується передатною функцією:

$$W_{rg}(s) = \frac{39.13 \cdot s + 5.207}{7.514 \cdot s};$$

і має настройки (4.7): $P = 5.2074$; $I = 0.693$; $D = 0$.

Діаграма ПІ-регулятор зображена на рис. 4.18., а його перехідна характеристика (вікно блоку XY Graph) – на рис. 4.19.

Кінець приклада.

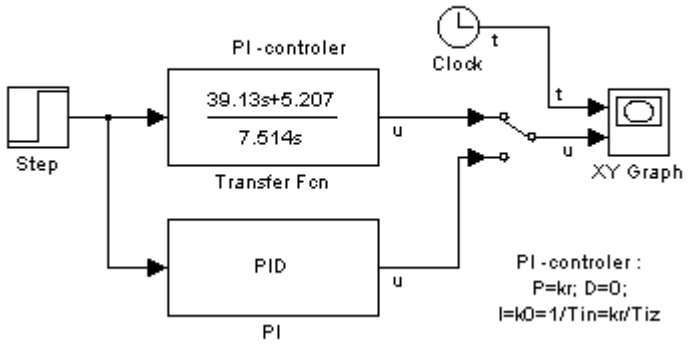

Рис. 4.18.

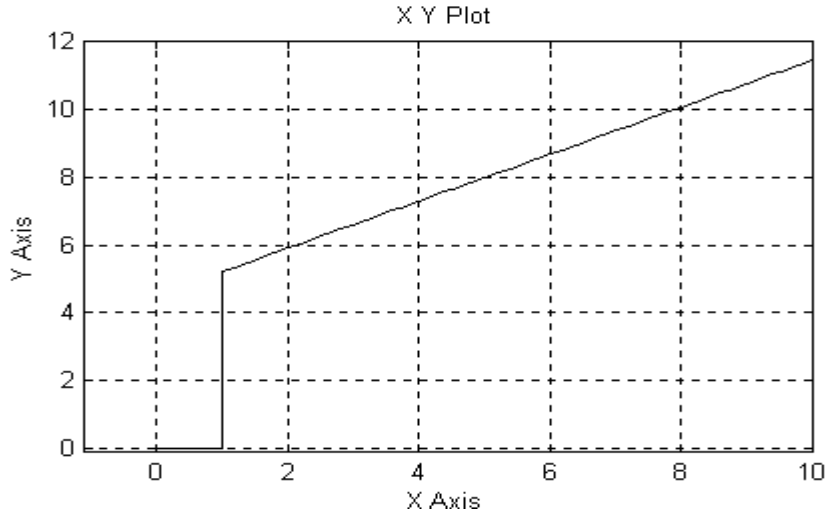

Рис. 4.19.

Приклад 4.11.

Побудуємо імітаційну модель замкненої системи з об'єктом W_o та ПІ-регулятором W_{rg} (див. приклади 3.2. та 4.10.).

Збурення в систему введемо у вигляді послідовності прямокутних імпульсів з одиничною амплітудою та частотою 0.02 Гц. Для цього використаємо блок Signal Generator, що міститься в підрозділі Sources бібліотеки Simulink.

Задамо час моделювання – 150 с.

Діаграма системи зображена на рис. 4.20., а графік часової залежності виходa системи та об’єкта (вікно блоку Scope) – на рис. 4.21.

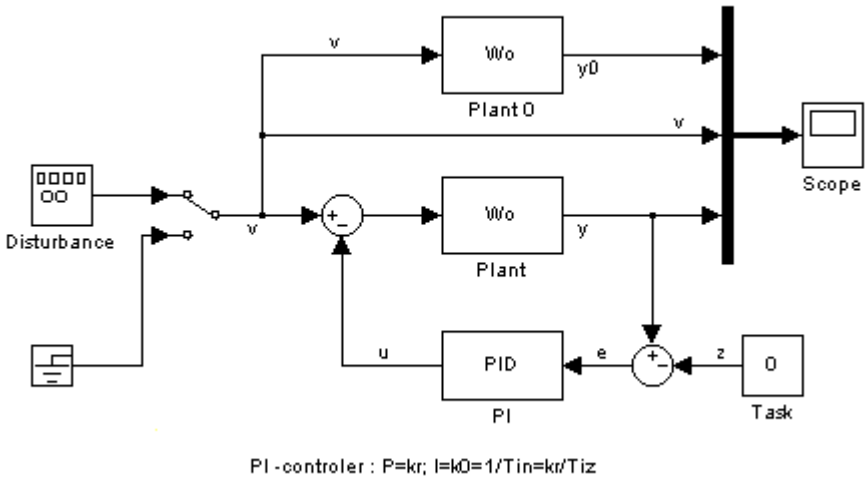

Рис. 4.20.

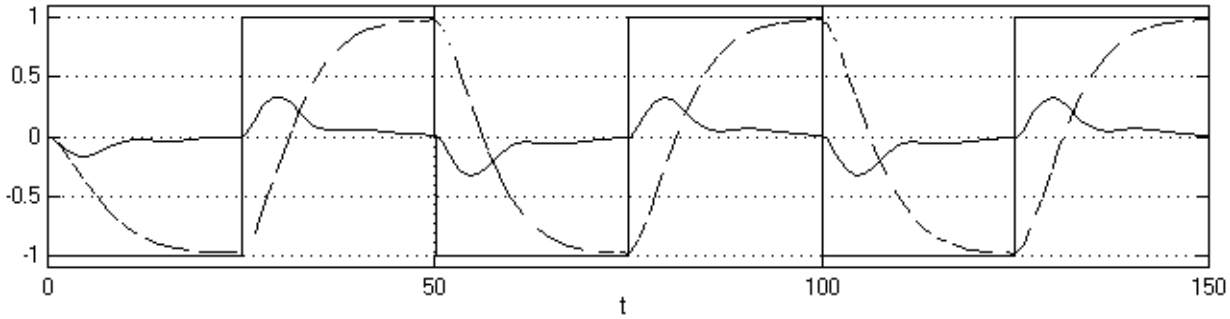

Рис. 4.21.

Кінець приклада.

4.4.2. Система з регулятором стану

Задати регулятор та спостерігач стану можна у вигляді блок-схеми матричного вираза (див. приклади 4.4. та 4.6.) або одним з блоків, що реалізують модель динамічного об’єкта у вигляді передатної функції або в просторі станів.

Приклад 4.12.

Побудуємо імітаційну модель замкненої системи з об’єктом W_o, регулятором стану **K** та спостерігачем, що має матрицю зворотного зв’язку **L** (див. приклад 3.4.).

Для того, щоб задати в системі довільний початковий стан (наприклад 2*rand(3,1), де rand – генератор випадкових, рівномірно розподілених чисел) необхідно представити модель об'єкта в просторі станів (P_x на діаграмі). Початковий стан для спостерігача задамо за допомогою блоку Initial Condition, що міститься в підрозділі Signals & Systems бібліотеки Simulink.

Діаграма системи зображена на рис. 4.22., а графік часової залежності виходу системи та об'єкта при довільному початковому стані (вікно блоку Scope) – на рис. 4.23.

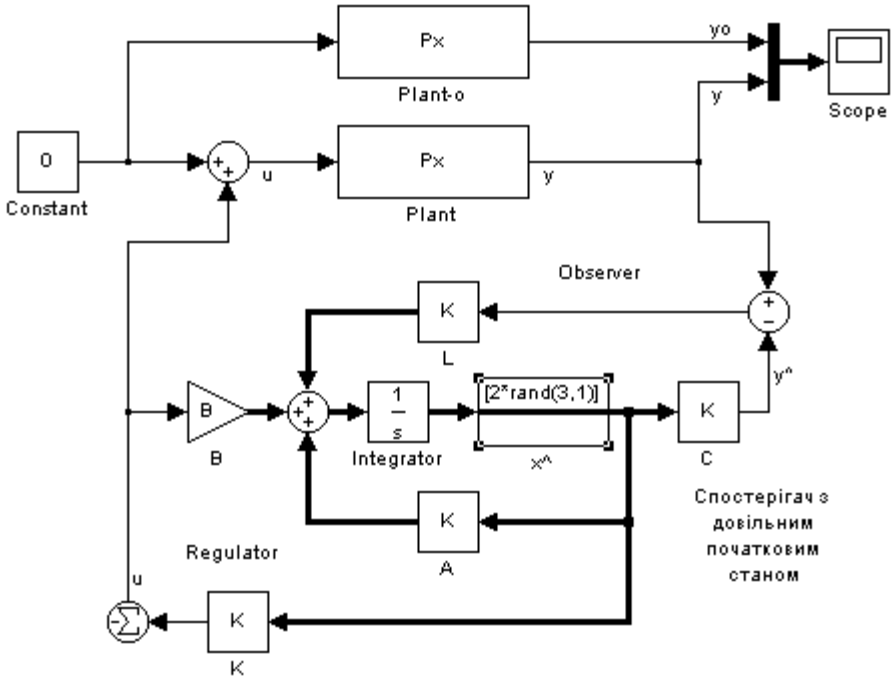

Рис. 4.22.

Кінець приклада.

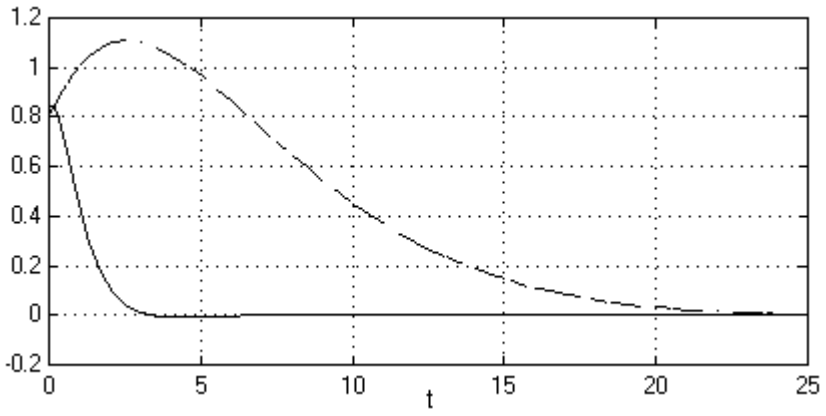

Рис. 4.23.

Приклад 4.13.

Побудуємо імітаційну модель замкненої системи з об'єктом P_x та компенсатором Reg_x у вигляді поліноміальної передатної функції (див. приклади 3.4. та 4.12.). Імітація дає результати, подібні до результатів в попередньому прикладі. Діаграма системи зображена на рис. 4.24.

```
» tf(Regx)
Transfer function:
  -179.7 s^2 - 332.3 s - 132.3
--------------------------------
s^3 + 7.285 s^2 + 17.11 s + 13.75
I/O groups:
    Group name      I/O    Channel(s)
    Measurement      I         1
     Controls        O         1
```

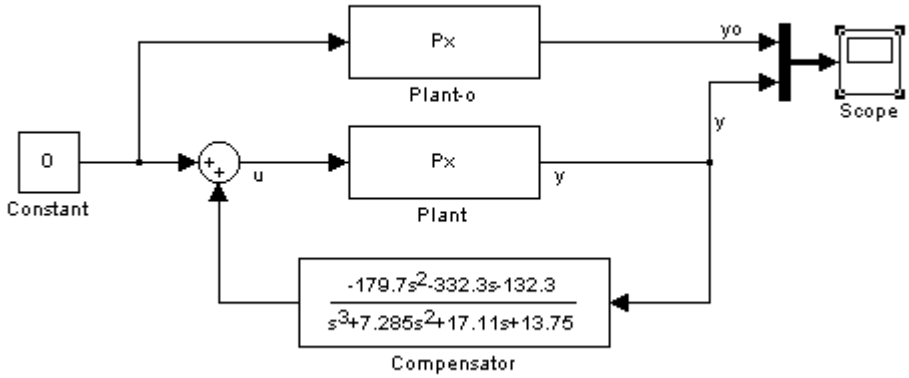

Рис. 4.24.

Кінець приклада.

4.4.3. Система з ЛКГ-регулятором

Блок-діаграма системи з ЛКГ-регулятором реалізується подібно будь-якій системі в просторі станів (див. п.п. 4.1.4. та 4.4.2.).

Збурення в системі моделюється за допомогою блоків Random Number або Band Limited White Noise, що містяться в підрозділі Sources бібліотеки Simulink. При цьому в першому випадку білий шум задається його

математичним сподіванням та дисперсією, а в другому – через його спектральну щільність.

Крім того збурення в системі можна задати за допомогою одного з генераторів випадкового сигналу, що їх надають різні бібліотеки MatLab. При цьому слід використовувати блоки введення даних (див. п. 4.2.1.).

Приклад 4.14.

Побудуємо імітаційну модель замкненої системи з об'єктом W_o, фільтром високих частот P_f, ЛК-регулятором $-K_x$ та фільтром Калмана Est_x (див. приклад 3.3.). Білий шум в системі промоделюємо за допомогою блоку Random Number, параметру *Varianse* якого привласнимо значення *dis* = 0.0421.

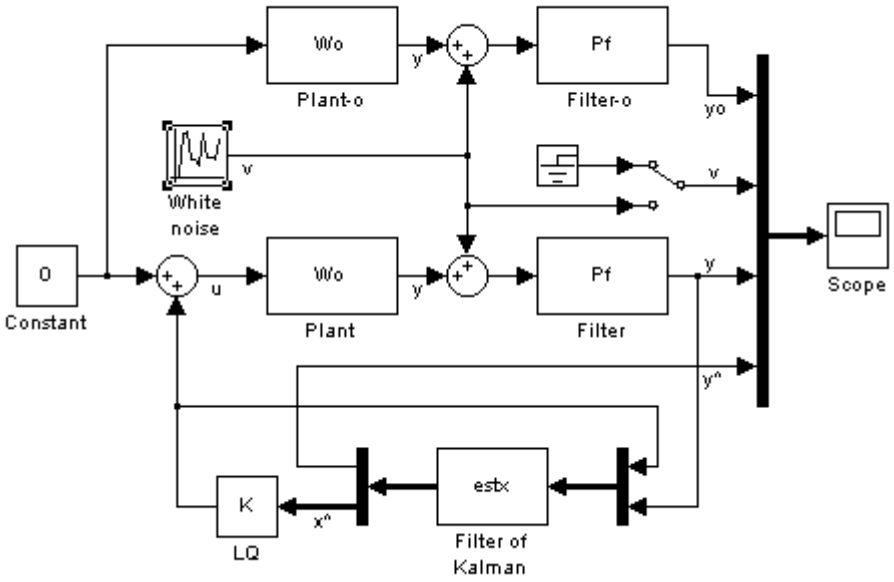

Рис. 4.25.

Діаграма системи зображена на рис. 4.25., а графік часової залежності виходу системи, спостерігача та об'єкта (вікно блоку Scope) – на рис. 4.26. Як видно з рисунка виходи системи та спостерігача практично співпадають.

Кінець приклада.

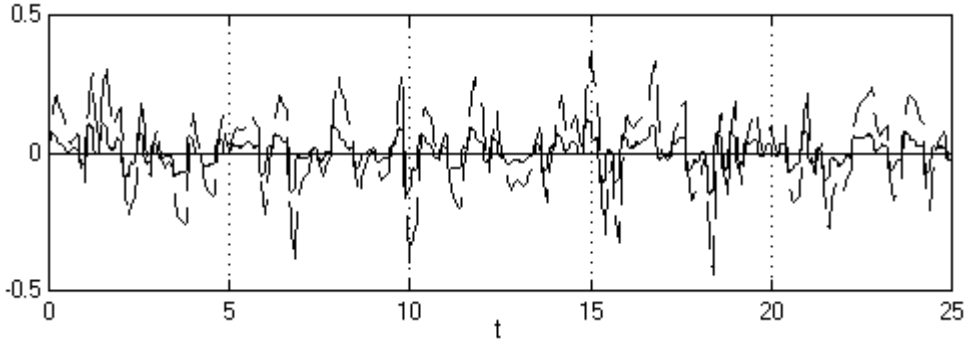

Рис. 4.26.

Приклад 4.15.

Побудуємо імітаційну модель замкненої системи з об'єктом W_o, фільтром P_f та ЛКГ-регулятором Reg_x (див. приклади 3.3. та 4.14.).

Діаграма системи зображена на рис. 4.27. Імітація дає результати, подібні до результатів в попередньому прикладі.

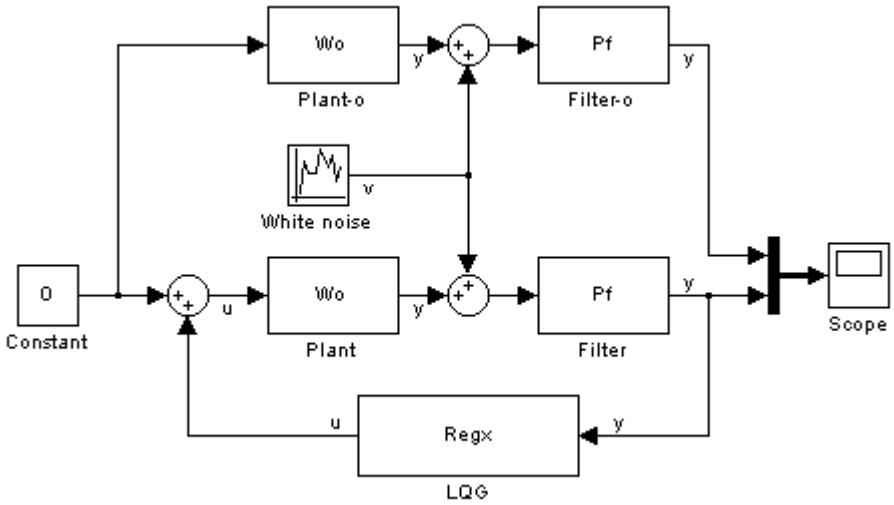

Рис. 4.27.

Кінець приклада.

4.4.4. Автоматична побудова блок-діаграм

При проектуванні системи регулювання методом кореневого годографа за допомогою плоттера кореневих годографів rltool (див. параграф 3.1.), існує можливість автоматичної побудови імітаційної моделі отриманої системи. Для цього необхідно вибрати команду Draw Simulink Diagram... з меню File плоттера. В якості входу побудованої моделі використовується періодичний

сигнал від блоку Signal Generator, що міститься в підрозділі Sources бібліотеки Simulink.

Приклад 4.16.

Побудуємо імітаційну модель замкненої системи з об'єктом W_o та компенсуючим пристроєм K, що розраховується методом кореневого годографа за допомогою плоттера rltool.

Після завантаження в плоттер моделі об'єкта та вибору структури моделі (див. рис. 3.1.) встановимо коефіцієнт зворотного зв'язку K = 33 (див. приклад 3.1.).

Блок-діаграма системи, синтезована плоттером, зображена на рис. 4.28.

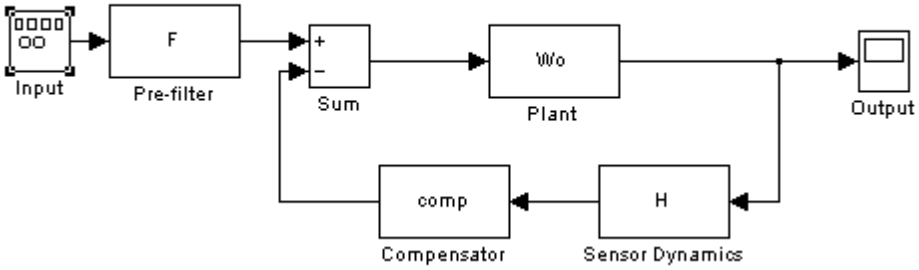

Рис. 4.28.

Перед побудоваю діаграми плоттер записує в робочий простір MatLab у вигляді змінних моделі структурних елементів системи:

```
» F
Zero/pole/gain:
1
» H
Zero/pole/gain:
1
» comp
Zero/pole/gain:
33
```

Входом системи є синусоїдальний сигнал від блоку Signal Generator. Задамо частоту сигналу 0.2 Гц та проведемо імітацію. Результат імітації (вікно блоку Scope) зображений на рис. 4.29.

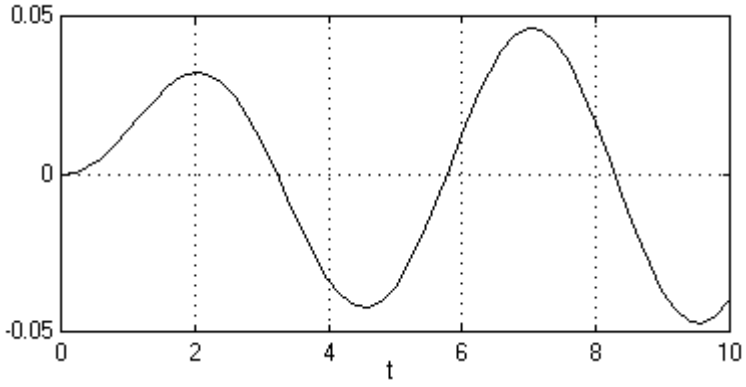

Рис. 4.29.

Організуємо виведення результатів імітації в робочий простір MatLab за допомогою блоку Scope. Для цього привласнимо параметру *Variable Name* ім'я матриці *ScopeData*, яка після завершення імітації буде містити виведені дані в форматі (4.4) (див. п. 4.2.2.).

Для порівняння побудуємо часову залежність виходу системи за даними з матриці (рис. 4.30.). Результати побудови повністю співпадають з результатами у вікні блоку Scope.

```
» plot(ScopeData(:,1),ScopeData(:,2))
» grid on; xlabel('t'); title('Результат імітації')
```

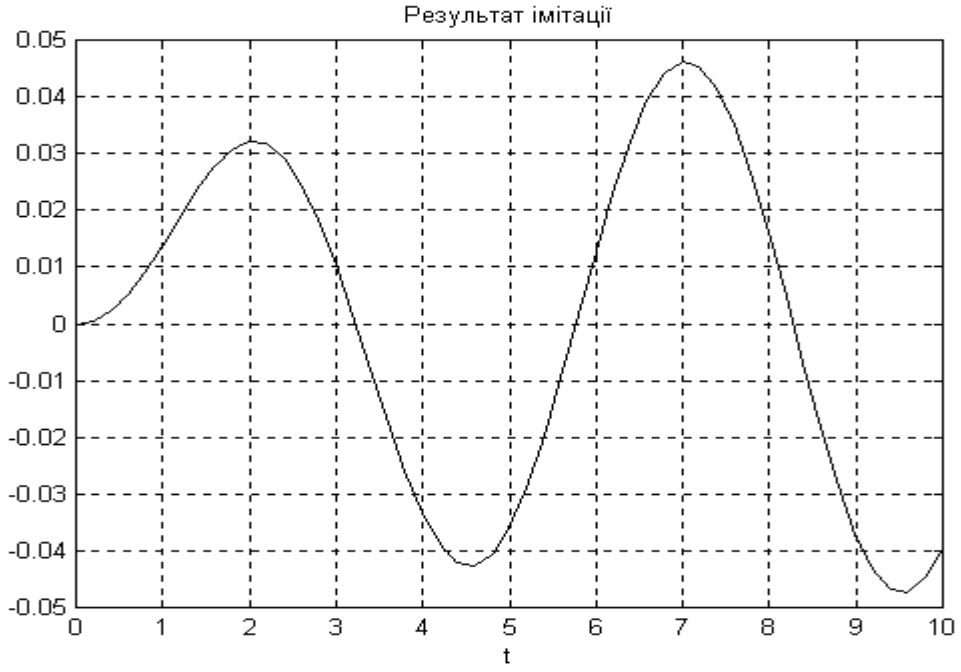

Рис. 4.30.

Кінець приклада.

4.5. Параметри імітаційної моделі

Блок-діаграма Simulink і кожен з блоків на діаграмі розглядається MatLab як графічний об'єк, і як будь-який об'єкт, має властивості, які надалі будемо називати параметрами.

4.5.1. Керування процесом імітації

Параметри, що керують роботаю імітаційної моделі можна задати на сторінці Solver вікна Simulation Parameters, яке викликається командою Parameters... з меню Simulation діаграми. Передусім це часовий інтервал моделювання (параметри *Start time* та *Stop time*) та спосіб розрахунку моделі (параметр *Type* та інші).

Будь-яка блок-діаграма динамічної системи розглядається Simulink як модель в просторі станів. Стани моделі визначаються шляхом чисельного інтегрування систем звичайних диференційних рівнянь (ode). В Simulink реалізовані різні модифікації методів Ейлера, Адамса та Рунге-Кутта (для фіксованого та змінного кроку інтегрування). Вибір конкретного метода інтегрування залежить від поставленої задачі та властивостей моделі. Докладніше ознайомитись з особливостями кожного з методів можна за допомогою довідкової системи MatLab (команда help *funfun*).

Крім того при моделюванні із змінним кроком можна задати параметри, що впливають на відображення та виведення даних блоками Scope, To File, To Workspace та Out (див. п. 4.2.2.):

Refine output – шляхом інтерполяції забезпечує виведення даних в додаткових *k* точках на кожному кроці моделювання (де *k* – значення параметра). Параметр не впливає на величину кроку.

Produce additional output – дозволяє додатково виводити інформацію в задані моменти часу.

Produce specified output only – дозволяє виводити інформацію тільки в задані моменти часу.

4.5.2. Настроювання параметрів блок-діаграми

Будь-який параметр діаграми (в тому числі і параметри керування процесом імітації) або блоку можна змінити з командного вікна MatLab за допомогою функцій get_param та set_param.

Отримати повний список параметрів діаграми або блоку можна командою:

get_param(‘*ім’я*’, ‘*ObjectParameters*’);

де *ім’я* - назва діаграми або блоку на діаграмі.

З’ясувати значення окремого параметра можна командою:

get_param(‘*ім’я*’, ‘*параметр*’);

де *параметр* - назва параметра.

Задати значення параметра можна командою:

set_param(‘*ім’я*’, ‘*параметр*’, ‘*значення*’);

де *значення* - значення параметра.

Приклад 4.17.

Визначимо значення модельного часу в блок-діаграмі *id*2 (приклад 4.8.).

```
» get_param('id2','StopTime')
ans = 300
```

Для автоматичного завантаження матриці даних *D*1*e* при виклику діаграми *id*2 задамо її параметр *PreLoadFcn*.

```
» save d1 D1e              % Формування файла d1.mat з матрицею D1e
» set_param('id2','PreLoadFcn','load d1')
» get_param('id2','PreLoadFcn')
ans = load d1
```

Кінець приклада.

Додаток 1.

Бібліотеки функцій MatLab

Для з'ясування синтаксису та особливостей вживання команд, функцій та інших засобів MatLab, що існують у вигляді М-файлів з ідентичними іменами і входять до складу бібліотек, можна скористатись командою help в такому форматі:

help *<ім'я файлу>*

Відшукати потрібний засіб MatLab по ключовому слову можна за допомогою команди lookfor в такому форматі:

lookfor *<ім'я файлу>* -all

Останній параметр не є обов'язковим і використовується для поглибленого пошуку.

1.1. Бібліотека System Identification Toolbox

Таблиця Д.1

Інструментальні засоби ідентифікації	
ident	Інтерактивний інструмент ідентифікації
midprefs	Встановлення каталогу для збереження стартової інформації
Імітаційне моделювання та прогнозування	
idinput	Генерація вхідних сигналів
idsim	Імітаційне моделювання загальної лінійної системи
idsimsd	Імітаційне моделювання виходів кількох моделей
pe	Розрахунок прогонозованої помилки
predict	Розрахунок прогнозованого виходу моделі
Маніпуляції з масивами даних	
dtrend	Видалення тренду з масиву вимірених даних
idfilt	Попередня фільтрація даних
idresamp	Зміна періоду дискретизації даних
Непараметричні методи оцінювання моделей	
covf	Оцінювання коваріаційної функції
cra	Оцінювання імпульсної характеристики та коваріаційної функції методом кореляційного аналізу
etfe	Оцінювання спектра та з використанням прямого перетворення Фур'є
spa	Оцінювання спектра та частотної характеристики методом спектрального аналізу

Продовження табл. Д.1

Параметричні методи оцінювання моделей	
ar	Оцінювання AR моделі
armax	Оцінювання ARMAX моделі
arx	Оцінювання ARX моделі методом найменших квадратів
bj	Оцінювання Box-Jenkins (BJ) моделі
canstart	Оцінювання багатовимірної моделі в просторі станів (канонічна форма)
iv4	Оцінювання ARX моделі чотириступеневим методом допоміжних змінних
ivar	Оцінювання AR моделі
ivx	Оцінювання ARX моделі методом допоміжної змінної з довільною кількістю змінних
n4sid	Оцінювання моделі в просторі станів методом підпростору
oe	Оцінювання Output-Error (OE) моделі
pem	Оцінювання загальної лінійної моделі методом прогнозованої помилки
Створення структур моделей	
arx2th	Визначення багатовимірної ARX структури
canform	Генерація канонічної форми
mf2th	Створення довільної структури лінійної моделі, що визначена у M-файлі
modstruc	Визначення моделі в просторі станів з відомими та невідомими параметрами
ms2th	Створення структури лінійної моделі в просторі станів з відомими та невідомими параметрами
poly2th	Створення структури моделі типу "вхід-вихід" задаванням поліномів чисельника та знаменник
Операції зі структурою моделі	
fixpar	Фіксація параметра в стуктурі моделі в просторі станів або ARX-моделі заданим значенням
sett	Встановлення інтервалу дискретизації
ss2th	Перетворення моделі в просторі станів у модель в канонічній параметричній формі
thinit	Вибір або рандомізація початкових значень параметрів
unfixpar	Вивільнення зафіксованого раніше параметра в структурі
Приведення форм моделей	
idmodred	Зниження порядку моделі
thc2thd	Приведення моделі до дискретної форми
thd2thc	Приведення дискретної моделі до моделі в безперервному часі
th2arx	Відновлення параметрів ARX моделі з th-форми моделі
th2ff	Відновлення частотної функції та спектра з th-форми моделі

th2par	Відновлення оцінених параметрів та їх дисперсій з th-форми моделі
th2poly	Відновлення поліноміальної передатної функції з th-форми моделі
th2ss	Відновлення матриць моделі в просторі станів з th-форми моделі
th2tf	Відновлення поліноміальної передатної функції з th-форми моделі
th2zp	Відновлення нулів, полюсів та коефіцієнтів передачі з th-форми моделі
Побудова характеристик моделей	
bodeplot	Побудова діаграм Боде
ffplot	Побудова частотних функцій та спектра
idplot	Побудова часових залежностей вимірених входів та виходів
nyqplot	Побудова годографа Найквіста
present	Виведення інформації про модель в th-формі у вікно MatLab
zpplot	Побудова діаграми нулів та полюсів
Отримання інформації про модель	
getff	Відновлення частотних функцій з частотної форми моделі
gett	Відновлення періоду дискретизації з th-форми моделі
getmfth	Відновлення імені M-файлу, в якому визначена структура моделі
getncap	Відновлення з th-форми моделі даних, по яким оцінено модель
getzp	Відновлення нулів та полюсів з моделі в zpk-форматі
th2par	див. Приведення форм моделей
Перевірка адекватності моделі	
compare	Порівняння виходу моделі з виміреним виходом системи
predict	Розрахунок передбачених виходів моделі
resid	Розрахунок та перевірка залишкової похибки моделі
idsim	див. Імітаційне моделювання та прогнозування
pe	
Вибір структури моделі	
arxstruc	Розрахунок функції втрат для ряду ARX структур моделі
ivstruc	Розрахунок функції втрат для ряду OE структур моделі
selstruc	Вибір структури
struc	Генерація ряду структур
Параметричні рекурсивні методи	
rarmax	Рекурсивне оцінювання ARMAX або ARMA моделей
rarx	Рекурсивне оцінювання ARX або AR моделей
rbj	Рекурсивне оцінювання BJ моделей
roe	Рекурсивне оцінювання OE моделей (IIR-фільтр)
rpem	Оцінювання загальної лінійної моделі типу "вхід-вихід" рекурсивним методом передбаченої помилки

Продовження табл. Д.1

rplr	Оцінювання загальної лінійної моделі типу "вхід-вихід" рекурсивним псевдолінійним регресивним методом
segment	Сегментація даних та оцінювання моделей для кожного сегмента

1.2. Бібліотека Control System Toolbox

Таблиця Д.2

Створення лінійних стаціонарних моделей та отримання з них інформації	
drmodel	Генерація випадкової дискретної моделі
drss	Генерація випадкової дискретної моделі в просторі станів
dss	Генерація дискретної моделі в просторі станів
dssdata	Отримання даних про модель в просторі станів в неявній формі Коші, або перетворення їх в масив комірок
filt	Генерація дискретної моделі у вигляді передатної функції в поліноміальній формі від $z^{-1}=q$
frd	Генерація моделі у вигляді частотних відгуків системи (frd-форма)
frdata	Отримання частотних відгуків моделі
get	Запит властивостей моделі
ltimodels	Отримання інформації про модель
ltiprops	Отримання інформації про всі властивості моделі (як змінної)
ord2	Генерація моделі другого порядку
rmodel	Генерація випадкової моделі в безперервному часі
rss	Генерація випадкової моделі в просторі станів для безперервного часу
set	Задання властивостей моделі (як змінної)
ss	Генерація моделі в просторі станів для безперервного часу
ssdata	Отримання даних про модель в просторі станів в явній формі Коші, або перетворення їх в масив комірок
tf	Генерація моделі у вигляді передатної функції в поліноміальній формі (tf-форма)
tfdata	Отримання даних про модель в tf-формі
totaldelay	Отримання загального запізнювання в системі для кожного каналу "вхід-вихід"
zpk	Генерація моделі у вигляді передатної функції, що виражена через нулі, полюса та коефіцієнти передачі (zpk-форма)
zpkdata	Отримання даних про модель в zpk-формі
Перетворення моделей	
c2d	Перетворення моделі в безперервному часі в дискретну форму
chgunits	Перетворення властивостей для моделі в frd-формі

Продовження табл. Д.2

d2c	Перетворення дискретної моделі у відповідну модель в просторі станів
d2d	Змінювання періоду дискретизації моделі
delay2z	Перетворення запізнювання для дискретних або frd-моделей
frd	Перетворення моделі до frd-форми
pade	Pade-апроксимація вхідного запізнювання
reshape	Зміна конфігурації масива моделей
ss	Перетворення моделі в простір станів
tf	Перетворення моделі у поліноміальну передатну функцію
zpk	Перетворення моделі у zpk-передатну функцію
Тестування моделей	
class	Визначення типу моделі
hasdelay	Тестування моделі на наявність запізнювання будь-якого типу
isa	Тестування моделі на відповідність вказаному типу
isct	Визначення, чи це модель в безперервному часі
isdt	Визначення, чи це дискретна модель
isempty	Визначення, чи це несформована модель
isproper	Визначення, чи це коректна модель
issiso	Визначення, чи це SISO-модель (модель з одним входом та одним виходом)
ndims	Визначення кількості вимірів моделі або масиву моделей
size	Визначення розмірностей масиву моделей (кількості входів, виходів та розмірність) або порядку моделі
Зниження порядку моделі	
balreal	Розрахунок збалансованої реалізації "вхід-вихід" моделі
minreal	Розрахунок мінімальної реалізації, або взаємна компенсація нулів та полюсів
modred	Видалення зайвих станів із збалансованої реалізації "вхід-вихід"
sminreal	Видалення станів моделі, що не впливають на відгук в каналах "вхід-вихід"
Робота з моделями в просторі станів	
canon	Отримання канонічної форми моделі
ctrb	Отримання матриці керованості
ctrbf	Розрахунок керованості у східчастій формі
gram	Розрахунок визначника керованості та спостережуваності
obsv	Отримання матриці
obsvf	Розрахунок спостережуваності у східчастій формі
ss2ss	Перетворення координат стану моделі
ssbal	Диагональне балансування моделі
Аналіз динаміки моделей	
covar	Розрахунок коваріації відгуку на білий шум

Продовження табл. Д.2

damp	Розрахунок власної частоти та згасання
dcgain	Розрахунок коефіцієнта передачі для малих частот
dsort	Відсортування полюсів за величиною для дискретних моделей
esort	Відсортування полюсів за величиною для моделей в безперервному часі
norm	Розрахунок норм моделей (H_2 та L_∞)
pole	Розрахунок полюсів моделі (див. eig)
pzmap	Побудова діаграми нулів та полюсів моделі
rlocfind	Визначення коефіцієнта зворотного зв'язку та полюсів, що відповідають вказаній точці на кореневому годографі
rlocus	Розрахунок та побудова кореневого годографа
sgrid	Накладання s-координатної сітки на кореневий годограф або діаграму нулів та полюсів для моделей в безперервному часі
zero	Розрахунок нулів моделі
zgrid	Накладання z-координатної сітки на кореневий годограф або діаграму нулів та полюсів для дискретних моделей
Конструювання моделей	
append	Формування діагональної матриці моделей
augstate	Збільшення виходів моделі шляхом додавання станів
connect	З'єднання моделей (з діагональної матриці) по обраній схемі
feedback	Здійснення зустрічно-паралельного з'єднання моделей (зворотний зв'язок)
lft	Здійснення з'єднання моделей в зірку
parallel	Здійснення паралельного з'єднання моделей
series	Здійснення послідовного з'єднання моделей
stack	Формування масиву моделей вздовж обраної розмірності з окремих моделей або масивів моделей
Аналіз часового вудгуку моделей	
gensig	Генерація вхідного тестового сигналу бажаного типу
impulse	Розрахунок імпульсної перехідної характеристики
initial	Розрахунок та побудова відгуку моделі в просторі станів при заданих початкових умовах
lsim	Розрахунок часового відгуку моделі на довільний вхідний сигнал
step	Розрахунок перехідної характеристики
Аналіз частотного відгуку моделей	
bode	Розрахунок та побудова логарифмічних частотних характеристик моделі (діаграм Боде)
evalfr	Розрахунок відгуку моделі для заданої комплексної частоти
freqresp	Розрахунок відгуку моделі для заданої послідовності дійсних частот
linspace	Створення вектору рівновіддалених частот

Продовження табл. Д.2

logspace	Створення вектору логарифмічно віддалених частот
margin	Розрахунок запасу по підсиленню та запасу по фазі
ngrid	Накладення координатної сітки на діаграму Ніколса
nichols	Розрахунок та побудова діаграми Ніколса
nyquist	Розрахунок та побудова годографа Найквіста
sigma	Розрахунок особливих (сингулярних) значень частотного вудгуку моделі
Синтез систем по заданому розташуванню полюсів	
acker	Розрахунок регулятора стану на задане розташування полюсів для SISO-об'єкта
estim	Розрахунок спостерігача стану
place	Розрахунок регулятора стану на задане розташування полюсів для об'єкта з довільною кількістю входів та виходів
reg	Формування компенсатора з регулятора стану та спостерігача
Синтез систем з ЛКГ-регулятором	
dlqr	Розрахунок оптимального ЛК-регулятора для дискретних моделей
kalman	Розрахунок спостерігача Калмана
kalmd	Розрахунок дискретного спостерігача Калмана для моделей в безперервному часі
lqgreg	Формування ЛКГ-регулятора з ЛК-регулятора та фільтра Калмана
lqr	Розрахунок оптимального ЛК-регулятора для моделей в безперервному часі
lqrd	Розрахунок дискретного ЛК-регулятора для моделей в безперервному часі
lqry	Розрахунок оптимального ЛК-регулятора для моделей із зваженим виходом
Розв'язок рівнянь	
care	Вирішення алгебраїчного рівняння Ріккаті в безперервному часі
dare	Вирішення дискретного алгебраїчного рівняння Ріккаті
dlyap	Вирішення дискретного рівняння Ляпунова
lyap	Вирішення рівняння Ляпунова в безперервному часі
Інструментальні засоби для аналізу моделей та синтезу систем	
ltiview	Інтерактивний інструмент для аналізування відгуків лінійних стаціонарних систем
rltool	Інтерактивний інструмент для побудови кореневих годографів замкнених SISO-систем

Додаток 2.

Властивості lti-моделей як об'єктів MatLab

Таблиця Д.3

Загальні властивості моделей		
IoDelayMatrix	Запізнювання моделі	матриця
InputDelay	Вхідні запізнювання моделі	вектор
InputGroup	Групи входів моделі	масив комірок
InputName	Назви входів моделі	вектор комірок з рядками
Notes	Опис моделі	текст
OutputDelay	Вихідні запізнювання моделі	вектор
OutputGroup	Групи виходів моделі	масив комірок
OutputName	Назви виходів моделі	вектор комірок з рядками
Ts	Період дискретизації моделі	скаляр
Userdata	Додаткові дані про модель	довільний тип
Властивості моделей, що задані у вигляді поліноміальних передатних функцій		
den	Знаменник	дійсний масив комірок з векторами-стовпцями
num	Чисельник	дійсний масив комірок з векторами-стовпцями
Variable	Змінна передатної функції	рядок 's', 'p', 'z', 'q' або 'z^–1'
Властивості моделей, що задані у вигляді zpk-передатних функцій		
k	Коефіцієнт передачі	дійсна матриця
p	Полюси	масив комірок з векторами-стовпцями
Variable	Змінна передатної функції	рядок 's', 'p', 'z', 'q' або 'z^–1'
z	Нулі	масив комірок з векторами-стовпцями
Властивості моделей в просторі станів		
a	Матриця стану A	дійсна матриця
b	Матриця керування B	дійсна матриця
c	Матриця вимірювання C	дійсна матриця
d	Матриця проникнення D	дійсна матриця
e	Дескриптор E матриці	дійсна матриця
StateName	Назви станів	вектор комірок з рядками
Властивості моделей, що задані через частотний відгук		
Frequency	Набір частот	дійсний вектор
ResponseData	Частотний відгук	комплексний масив
Units	Одиниці вимірювання частоти	рядок 'rad/s' або 'Hz'

Додаток 3.

Бібліотеки модулів Simulink

Отримати список функцій та команд для роботи з моделями Simulink в командному режимі MatLab можна за допомогою команди:

help simulink

3.1. Бібліотека Simulink

Таблиця Д.4

Підрозділ Continuous	
Integrator	Інтегратор (суматор безперервного часу)
Transfer Fcn	Поліноміальна передатна функція
State-Spase	Динамічна ланка, що задана в просторі станів
Zero-Pole	ZPK-передатна функція
Derivative	Диференціатор
Memory	Затримка на один інтервал інтегрування
Transport Delay	Транспортне запізнювання
Variable Transport Delay	Регульоване транспортне запізнювання
Підрозділ Discrete	
Unit Delay	Затримка на один період дискретизації
Discrete Time Integrator	Дискретний інтегратор (лічильник періодів дискретизації)
Zero-Order Hold	Екстраполятор нульового порядку
First-Order Hold	Екстраполятор першого порядку
Discrete State-Spase	Дискретна динамічна ланка в просторі станів
Discrete Filter	Дискретний фільтр
Discrete Transfer Fcn	Дискретна поліноміальна передатна функція
Discrete Zero-Pole	Дискретна zpk-передатна функція
Підрозділ Functions & Tables	
Look-Up Table	Таблично задана фінкція однієї змінної
Look-Up Table (2-D)	Таблично задана фінкція двох змінних
Fcn	Функція, що задана засобами мови C
MATLAB Fcn	Функція, що задана засобами мови MatLab
S-Function	S-функція (Simulink)
Підрозділ Math	
Gain	Пропорційна ланка (функція множення на коефіцієнт)
Sum	Суматор
Dot Product	Функція поелементного множення векторів
Matrix Gain	Функція множення матриці на вектор

Продовження табл. Д.4

Slider Gain	Регульована пропорційна ланка
Abs	Модуль (абсолютне значення)
Trigonometric Function	Тригонометрична функція (одна з набору)
Math Function	Математична функція (одна з набору)
Rounding Function	Функція округлення (одна з набору)
MinMax	Мінімум та максимум
Product	Множення та ділення сигналів
Combinatorial Logic	Комбінаторна логіка, що задана таблицею істинності
Logical Operator	Логічний оператор (один з набору)
Relational Operator	Оператор відношення (один з набору)
Sign	Визначення знаку вхідного сигналу
Algebraic Constraint	Алгебраїчне обмеження
Complex to Magnitude-Angle	Відновлення модуля та фази з комплексного числа
Magnitude-Angle to Complex	Формування комплексного числа з модуля та фази
Complex to Real-Imag	Формування дійсної та уявної частини комплексного числа
Real-Imag to Complex	Відновлення комплексного числа з дійсної та уявної частин
Підрозділ Nonlinear	
Rate Limiter	Обмежувач швидкості зміни сигналу
Saturation	Обмежувач сигналу
Quantizer	Квантування сигналу по рівню
Coulomb & Viscous Friction	Нелінійність типу кулоновське або в’язке тертя
Backlash	Мертвий хід
Dead Zone	Мертва зона
Switch	Перемикач з керуючим входом
Manual Switch	Керований перемикач
Multiport Switch	Багатоканальний перемикач з керуючим входом
Relay	Реле
Підрозділ Signals & Systems	
Hit Crossing	Сигналізатор нульового значення сигналу
In	Вхідний порт
Out	Вихідний порт
Mux	Мультіплексор (формування вектора сигналів)
Demux	Демультіплексор (формування окремих сигналів з вектора)
From	Приймач сигналів від блока Goto

Продовження табл. Д.4

Goto Tag Visibility	Ознака видимості сигналів передачи
Goto	Передавач сигналів
Data Store Read	Запис у спільну пам’ять
Data Store Memory	Спільна пам’ять підсистем однієї системи
Data Store Write	Читання зі спільної пам’яті
Enable	Вимикач підсистеми
Trigger	Синхронізатор роботи підсистеми
Ground	Заглушка для непід’єднаного входу блока
Terminator	Заглушка для непід’єднаного виходу блока
IC	Початкове значення вихідного сигналу
Subsystem	Підсистема
Selector	Перебудова вектора сигналів
Width	Блок визначення кількості елементів вектора сигналів
Merge	Об’єднання вхідних сигналів в один вихідний сигнал
Bus Selector	Перебудова та зміна розмірності вектора сигналів
Data Type Conversion	Перетворювач типів даних
Function-Call Generator	Блок виклику керованої підсистеми через задані проміжки часу
Configurable Subsystem	Об’єднання кількох блоків в підрозділ бібліотеки
Model Info	Блок виведення інформації про модель
Підрозділ Sinks	
Scope	Блок побудови часових залежностей сигналів
XY Graph	Блок побудови графіків
Display	Цифровий диплей
To File	Запис сигналів в файл матричної структури
To Workspace	Запис сигналів в матрицю MatLab
Stop Simulation	Умовне припинення моделювання
Підрозділ Sources	
Constant	Джерело постійного сигналу
Signal Generator	Генератор сигналів заданої форми
Step	Генератор ступінчастого сигналу
Ramp	Генератор лінійно зростаючого сигналу
Sine Wave	Генератор гармонічного сигналу
Repeating Sequence	Генератор послідовностей сигналів довільної форми, що повторюються
Discrete Pulse Generator	Дискретний генератор прямокутних імпульсів
Pulse Generator	Генератор прямокутних імпульсів

Продовження табл. Д.4

Chirp Signal	Генератор синусоїдального сигналу із зростаючою частотою
Clock	Годинник модельного часу
Digital Clock	Годинник дискретного модельного часу
From File	Читання сигналів з файлу матричної структури
From Workspace	Читання сигналів з матриць MatLab
Random Number	Генератор випадкових нормально розподілених сигналів
Uniform Random Number	Генератор випадкових рівномірно розподілених сигналів
Band Limited White Noise	Генератор білого шуму з обмеженою смугою частот

3.2. Бібліотека Simulink Extras

Таблиця Д.5

Підрозділ Additional Discrete	
Discrete Transfer Fcn (with initial outputs)	Дискретна поліноміальна передатна функція із заданим початковим виходом
Discrete Transfer Fcn (with initial states)	Дискретна поліноміальна передатна функція із заданими початковими умовами
Discrete Zero-Pole (with initial outputs)	Дискретна zpk-передатна функція із заданим початковим виходом
Discrete Zero-Pole (with initial states)	Дискретна zpk-передатна функція із заданими початковими умовами
Підрозділ Additional Linear	
PID Controller	ПІД-регулятор
PID Controller (with Approximate Derivative)	ПІД-регулятор з реальним диференціатором
State-Spase (with initial outputs)	Модель в просторі станів із заданим початковим виходом
Transfer Fcn (with initial outputs)	Поліноміальна передатна функція із заданим початковим виходом
Transfer Fcn (with initial states)	Поліноміальна передатна функція із заданими початковими умовами
Zero-Pole (with initial outputs)	ZPK-передатна функція із заданим початковим виходом
Zero-Pole (with initial states)	ZPK-передатна функція із заданими початковими умовами

Продовження табл. Д.5

Підрозділ Additional Sinks	
Auto Correlator	Блок визначення автокореляції даних
Averaging Power Spectral Density	Блок визначення усередненої спектральної щільності потужності вхідного сигналу
Averaging Spectral Analyzer	Блок спектрального аналізу усереднених даних
Cross Correlator	Блок визначення взаємної кореляції даних
Power Spectral Density	Блок визначення спектральної щільності потужності вхідного сигналу
Spectrum Analyzer	Блок спектрального аналізу даних
Підрозділ Flip Flops	
Clock	Цифровий таймер для логічних систем
D Flip-Flop	D тригер
D Latch	D фіксатор
J-K Flip-Flop	J-K тригер для від'ємного перепаду сигналів
S-R Flip-Flop	S-R тригер
Підрозділ Linearization	
Switched derivative for linearization	Блок визначення похідної в придатному для лінеаризації вигляді
Switched transport delay for linearization	Блок введення транспортного запізнювання в придатному для лінеаризації вигляді
Підрозділ Transformations	
Cartesian to Polar	Блок перетворення декартових координат в полярні
Cartesian to Spherical	Блок перетворення декартових координат в сферичні
Celsius to Fahrenheit	Блок перетворення градусів Цельсія в градуси Фаренгейта
Degrees to Radians	Блок перетворення кутових градусів в радіани
Fahrenheit to Celsius	Блок перетворення градусів Фаренгейта в градуси Цельсія
Polar to Cartesian	Блок перетворення полярних координат в декартові
Radians to Degrees	Блок перетворення радіан в кутові градуси
Spherical to Cartesian	Блок перетворення сферичних координат в декартові

3.3. Бібліотека Dials & Gauges Blockset

Таблиця Д.6

Підрозділ Dashboard - Based Instrumentation	
Dachboard	Створення приладової панелі з елементами ActiveX в окремому вікні
From Dashboard	Блок отримання інформації від приладової панелі
To Dachboard	Блок передачі інформації на приладову панель
Підрозділ Model-Based Instrumentation	
Містить блок ActiveX Control, що дозволяє розмістити орган керування типу ActiveX безпосередньо в моделі Simulink. Елементи ActiveX розташовані у власній бібліотеці Global Majic ActiveX Library, яка розділена на вісім підрозділів:	
Angular Gauges	Містить вимірювальні прилади з круглими шкалами
Buttons and Switches	Містить кнопки та вимикачі
Knobs and Selectors	Містить багатопозиційні перемикачі
Linear Gauges	Містить вимірювальні прилади з лінійними шкалами
Numeric Displays	Містить цифрові індикатори
Percent Indicators	Містить індикатори відсоткових часток
Sliders	Містить ручні регулятори типу "ковзний контакт"
Strip Chart	Дисплей для побудови стрічкових діаграм

3.4. Бібліотека Control System Toolbox (for Simulink)

Таблиця Д.7

LTI System	Дискретна лінійна стаціонарна динамічна система, або система в безперервному часі, що описується засобами MatLab, як передатна функція або модель в просторі станів
Input Point	Вхідна точка моделі для отримання динамічних характеристик по каналу "вхід-вихід"
Output Point	Вихідна точка моделі для отримання динамічних характеристик по каналу "вхід-вихід"

3.5. Бібліотека System Identification Toolbox (for Simulink)

Таблиця Д.8

AutoRegressive model estimator	Модуль ідентифікації AR-моделі
AutoRegressive with eXternal input model estimator	Модуль ідентифікації ARX-моделі
AutoRegressive Moving Average with eXternal input model estimator	Модуль ідентифікації ARMAX-моделі
Box-Jenkins model estimator	Модуль ідентифікації BJ-моделі
General model estimator using Predictive Error Method	Модуль ідентифікації загальної поліноміальної моделі методом прогнозованої помилки
Output-error model estimator	Модуль ідентифікації OE-моделі

Додаток 4.

Формування масивів даних для ідентифікації

Масиви інформації було отримано шляхом прямого вимірювання вхідних та вихідних сигналів на об’єкті. В якості об’єкта використовувався мікропроцесорний контролер Ломиконт з імітаційною керуючою програмою.

Програма імітує аперіодичний об’єкт третього порядку, як три послідовно-з’єднані фільтри (див. рис. Д1) з передатною функцією:

$$W_i = \frac{1}{T_i s + 1}.$$

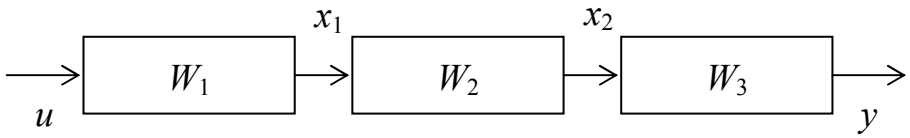

Рис. Д. 1.

Експерименти проводилися за допомогою вимірювального комплексу, що включає персональний комп’ютер класу PC AT, промисловий АЦП L-154 з інтерфейсом ISA ("L-card", Росія), та SCADA програму Oscil для DOS.

Програма імітації аперіодичної ланки третього порядку мовою Мікрол

```
Код ПрП 11

Заказ переменных  Заказ выходов ЦАП  Длина ПрП 235
КБ    0 - 0       0     000 - 007
ВА  000 - 007
АВ  000 - 007

Технологическая клавиатура
КТ1
 1  AB001 Y =
 2  AB000 U =
```

```
Текст ПрП
// секция 000  ( 51)

00  АЛГ011              ; ФЛТ фильтрация (апериодическое звено)
    1  ВХОД = +АВ000    ; u
    2 ВЫХОД = АВ002     ; x1
    3 Т ФЛТ = 00.00.02 ; T1
    4  АП Т = +0000     ;
    5 КАП Т = 1.0       ;
01  АЛГ011              ; ФЛТ фильтрация (апериодическое звено)
    1  ВХОД = +АВ002    ; x1
    2 ВЫХОД = АВ003     ; x2
    3 Т ФЛТ = 00.00.03 ; T2
    4  АП Т = +0000     ;
    5 КАП Т = 1.0       ;
02  АЛГ011              ; ФЛТ фильтрация (апериодическое звено)
    1  ВХОД = +АВ003    ; x2
    2 ВЫХОД = АВ001     ; y
    3 Т ФЛТ = 00.00.04 ; T3
    4  АП Т = +0000     ;
    5 КАП Т = 1.0       ;

; Под собственную память алгоритмов
; из 768 использовано 18 байт.
```

Після закінчення вимірювання за допомогою програми Oscil було створено текстовий файл pr.asc, в якому вимірені дані записано в три стовпчики: час, вхід АВ000 та вихід АВ001 об’єкта. Відліки виконувались через 50 мікросекунд (0.05 с). Час спостереження – 600с. Далі текстовим редактором з файлу pr.asc було видалено зайві символи.

Нижче наведено програму MatLab формування вектота *ud* вхідних сигналів та вектора *yd* вихідних сигналів об’єкта.

```
load pr.asc;           % Формується матриця pr 12000×3
                       % кожен рядок якої - одне вимірювання
                       % період дискретизацыъ вимірювань - 0.05 с.
for i=1:600            % Збільшення періоду дискретизації до 1 с
  pr(i+1:i+19,:)=[];   % шляхом видалення з матриці pr зайвих
end                    % вимірювань.
                       % Отримано матрицю pr 600×3.
ud=pr(:,2);            % Формування вектора вхідних
yd=pr(:,3);            % та вихідних сигналів об’єкта.
D1=[yd ud];            % Формування загального масиву даних.
```

В такий спосіб можна формувати набори масивів інформації з різним відношенням "сигнал – шум".

Додаток 5.

Програма визначення параметрів типових регуляторів

Програма організована як функція rotach системи MatLab і існує у вигляді файла rotach.m. Функція rotach має підфункцію circus для побудови окружностей, яка міститься в тому ж файлі. Нижче наведено повний текст цього файла.

```
function Wrg=rotach(Wo,k,Ti,Tv,M)

% Побудова М-кола та АФХ розімкненої системи з регулятором
% Wrg=rotach(Wo,k,Ti,Tv,M)
%
% Wo  - передатна функція SISO об'єкта;
% k   - коефіцієнт пропорційності регулятора;
% Ti  - час ізодрому;
% Tv  - час випередження;
% M   - показник коливності системи (M=1.3...1.6).
%
% Реалізовані системи з регуляторами:
% П   - при k>0 Ti=inf Tv=0
% I   - при k=0 Ti>0   Tv=0
% ПІ  - при k>0 Ti>0   Tv=0
% ПД  - при k>0 Ti=inf Tv>0
% ПІД - при k>0 Ti>0   Tv>0
%
% Wrg - повертається передатна функція регулятора

% М.В.Коржик, КПІ, 2000
% Version 1.1

if (M<1.2)|(M>1.7)
   error('Невірно задан показник коливальності')
end
```

```
% Формування регулятора
if k>0 & Ti==inf & Tv==0
    Wrg=tf(k,1);                          % П   -регулятор
    xsrg=strcat('k = ',num2str(k));
    lsrg=strcat('П-регулятор   M = ',num2str(M));
  elseif k==0 & Ti==inf & Tv>0
    Wrg=tf([Tv 0],1);                     % Д  -регулятор
    xsrg=strcat('Tv = ',num2str(Tv));
    lsrg=strcat('Д-регулятор   M = ',num2str(M));
  elseif k==0 & Ti>0 & Tv==0
    Wrg=tf(1,[Ti 0]);                     % I  -регулятор
    xsrg=strcat('Ti = ',num2str(Ti));
    lsrg=strcat('I-регулятор   M = ',num2str(M));
  elseif k>0 & Ti>0 & Tv==0
    Wrg=k*tf([Ti 1],[Ti 0]);              % ПI  -регулятор
    xsrg=strcat('k = ',num2str(k),'  Ti = ',num2str(Ti),...
                '  k/Ti = ',num2str(k/Ti));
    lsrg=strcat('ПI-регулятор   M = ',num2str(M));
  elseif k>0 & Ti==inf & Tv>0
    Wrg=k*tf([Tv 1],1);                   % ПД  -регулятор
    xsrg=strcat('k = ',num2str(k),'  Tv = ',num2str(Tv));
    lsrg=strcat('ПД-регулятор   M = ',num2str(M));
  elseif k>0 & Ti>0 & Tv>0
    Wrg=k*tf([Ti*Tv Ti 1],[Ti 0]);        % ПIД -регулятор
    xsrg=strcat(' k = ',num2str(k),'  Ti = ',num2str(Ti),...
                '  Tv = ',num2str(Tv),'  k/Ti = ',...
                num2str(k/Ti),'  Tv/Ti = ',num2str(Tv/Ti));
    lsrg=strcat('ПIД-регулятор   M = ',num2str(M));
  else
    error('Невірно задана структура регулятора');
end

% Формування розімкненої системи
W=minreal(Wo*Wrg);
```

```
% Визначення вектора частотного відгуку системи W
fr=logspace(-2,1,80);
resp=squeeze(frdata(frd(W,fr)));
% Або resp=squeeze(freqresp(W,fr));

% Побудова годографа
plot(resp); hold on

% Побудова М-кола
mx=M^2/(M^2-1);                          % Абсциса М-кола
mr=M/(M^2-1);                            % Радіус М-кола
circus(-mx,0,mr)

% Оформлення  графіка
ht=text(-mx+0.2,-0.2,'М-коло');
set(ht,'Color','red')
text(-1,0,'|')                           % Точка (-1,j*0)
title(lsrg)
xlabel(xsrg)
axis equal                               % Однаковий масштаб по осям
axis([-mx 0.2 -mr-0.2 0.2])

% Побудова координатних осей
line([-mx;0.2],[0;0],'LineStyle',':','Color','black');
line([0;0],[-mr-0.2;0.2],'LineStyle',':','Color','black');
hold off
% Виведення графічного вікна над іншими вікнами
set(gcf,'Visible','off','Visible','on')

function circus(x,y,r)
% Побудова окружності радіуса r з центром в точці x,y
% circus(x,y,r)

t=pi*(0:.02:2);
plot(x+r*cos(t),y+r*sin(t),'r')          % Червоний колір
```

Додаток 6.

Редагування графічного вікна Simulink

Найчастіше для відображення результатів моделювання в блок-діаграмах Simulink використовують блоки Scope та XY Graph, графічні вікна яких не мають власних засобів редагування. Крім того кольорова палітра вікна Scope не дає змоги використовувати його в текстових документах.

Нижче наведено текст скрипта smenu (файл smenu.m), що засобами дескрипторної графіки MatLab [22] встановлює нову кольорову палітру вікна Scope та вставляє смугу стандартних меню для подальшого його редагування.

В скрипті використовується функція gcbh, що повертає дескриптор виділеного на діаграмі блока.

```
% Встановлення властивостей вікна Scope.
%
% До виклику макроса smenu блок Scope
% на діаграмі треба виділити.

bl=[0 0 0];   % Чорний колір
wt=[1 1 1];   % Білий колір

% Отримання дескриптора вікна Scope
FigHndl=get_param(gcbh,'Figure');
% (властивість 'UserData' для вікна XY Graph)

% Отримання дескриптора об'єкта axes вікна Scope
AxsHndl=get(FigHndl,'CurrentAxes');

% Встановлення рядка меню та кольору вікна Scope
set(FigHndl,'MenuBar','figure','Color',wt);

% Встановлення кольору графіка та координатної сітки
set(AxsHndl,'Color',wt,'XColor',bl,'YColor',bl);
```

Список літератури

Література з ідентифікації та теорії керування

1. Острём К. Системы управления с ЭВМ / К. Острём, Б. Виттенмарк. – М. : Мир, 1987. – 480 с.

2. Эйкхофф П. Современные методы идентификации систем / П. Эйкхофф, А. Ванечек, Е. Савраги [и др.]. – М. : Мир, 1983. – 400 с.

3. Ротач В.Я. Автоматизация настройки систем управления / В.Я. Ротач, В.Ф. Кузищин, А.С. Клюев [и др.]. – М. : Энергоатомиздат, 1984. – 272 с.

4. Dorf R. Modern control systems / Richard C. Dorf. – Reading, MA : Addison-Wesley Publishing Company, 1990. – 604 p.

5. Изерман Р. Цифровые системы управления / Р. Изерман. – М. : Мир, 1984. – 541 с.

6. Куо Б. Теория и проектирование цифровых систем управления / Б. Куо. – М. : Машиностроение, 1986. – 448 с.

7. Дэннис Дж. Численные методы безусловной оптимизации и решения нелинейных уравнений / Дж. Дэннис мл., Р. Шнабель. – М. : Мир, 1988. – 440 с.

8. Бендат Дж. Применения корреляционного и спектрального анализа / Дж. Бендат, А. Пирсол. – М. : Мир, 1983. – 312 с.

9. Штейнберг Ш.Е. Идентификация в системах управления / Ш.Е. Штейнберг. – М. : Энергоатомиздат, 1987. – 80 с.

10. Остапенко Ю.О. Ідентифікація та моделювання технологічних об'єктів керування : Підручник / Ю.О. Остапенко. – К. : Задруга, 1999. – 424 с.

11. Воронов А.А. Теория автоматического управления. Ч. I. Теория линейных систем автоматического управления : Учеб. для вузов под ред. А.А. Воронова. – М. : Высш. шк., 1986. – 367 с.

12. Воронов А.А. Теория автоматического управления. Ч. II. Теория нелинейных и специальных систем автоматического управления : Учеб. для вузов под ред. А.А. Воронова. – М.: Высш. шк., 1986. – 504 с.

13. Кваско М.З. Расчет линейных систем регулирования с использованием вычислительной техники : Учеб. пособие / М.З. Кваско, В.В. Миленький. – К,: КПИ, 1988. – 92 с.

14. Кубрак А.И. Методы и программы для исследования систем автоматики : Учеб. пособие / А.И. Кубрак, А.И. Жученко, Л.Д. Ярощук. – К. : УМК ВО, 1989. – 228 с.

15. Жученко А.І. Математичні моделі цифрових систем керування: Навч. посібник / А.І. Жученко. – К.: ІЗМН, 1997. – 240 с.

16. Дубровный В.А. Справочник по наладке автоматических устройств контроля и регулирования / В.А. Дубровный, Е.И. Забокрицкий, В.Г. Тригуб, Б.А. Холодовский. – К. : Наукова думка, 1981. – 940 с.

17. Клюев А.С. Наладка средств автоматизации и автоматических систем регулирования : Справочное пособие под ред. А.С. Клюева. – М. : Энергоатомиздат, 1989. – 368 с.

18. Дорф Р. Современные системы управления / Р. Дорф, Р. Бишоп. – М. : Лаборатория Базовых Знаний, 2002. – 832 с.

19. Льюнг Л. Идентификация систем. Теория для пользователя / Леннарт Льюнг. – М. : Наука, 1991. – 432 с.

Література з системи MatLab

19. Потемкин В.Г. Введение в MatLab / В.Г. Потемкин. – М. : Диалог-МИФИ, 2000. – 247 с.

20. Потемкин В.Г. Система инженерных и научных расчетов MatLab 5.x: В 2-х т / В.Г. Потемкин. – М.: Диалог-МИФИ, 1999. Т1 – 366 с., Т2 – 304 с.

21. Потемкин В.Г. Система MatLab 5 для студентов / В.Г. Потемкин, П.И. Рудаков. – М. : Диалог-МИФИ, 1999. – 448 с.

22. Дьяконов В.П. MatLab 5.0/5.3. Система символьной математики / В.П. Дьяконов, И.В. Абраменкова. – М. : Нолидж, 1999. – 640 с.

23. Медведев В.С. Control System Toolbox. MatLab 5 для студентов / В.С. Медведев, В.Г. Потемкин. – М.: Диалг-МИФИ, 1999. – 287 с.

24. Гультяев А.К. MatLab 5.2. Имитационное моделирование в среде Windows : Практическое пособие / А.К. Гультяев. – СПб. : Корона принт, 1999. – 288 с.

25. Краснопрошина А.А. Сучасний аналіз систем управління із застосуванням MatLab, Simulink, Control System : Навчальний посібник / А.А. Краснопрошина, Н.Б. Репнікова, О.А. Ільченко. – К. : "Корнійчук", 1999. – 144 с.

26. System Identification Toolbox For Use with MatLab : User's guide [Electron resource]. – Natick, MA : The MathWorks, Inc, 1997. – 274 p.

27. Control System Toolbox For Use with MatLab : User's guide [Electron resource]. – Natick, MA: The MathWorks, Inc, 1999. – 649 p.

28. Simulink. Dynamic System Simulation for MatLab : Using Simulink [Electron resource]. – Natick, MA : The MathWorks, Inc, 1999. – 605 p.

29. Дэбни Дж. Simulink 4. Секркты мастерства / Дж. Б. Дэбни, Т.Л. Харман. – М. : Бином, 2003. – 403 с.

30. Мэтьюз Дж. Численные методы. Использование MatLab / Джон Г. Мэтьюз, Кертис Д. Финк. – М. : “Вильямс”, 2001. – 720 с.

31. Кетов Ю.Л. MatLab 7: программирование численных методов / Ю.Л. Кетов, А.Ю. Кетов, М.М. Шульц. – СПб. : БХВ-Петербург, 2005. – 752 с.

32. Иглин С.П. Математические расчеты на базе MatLab / С.П. Иглин. – СПб. : БХВ-Петербург, 2005. – 640 с.

33. Гультяев А.К. Визуальное моделирование в среде MatLab : Учебный курс / А.К. Гультяев. – СПб. : Питер, 2000. – 432 с.

34. Мартынов Н.Н. MatLab 5.x. Вычисления, визуализация, программирование / Н.Н. Мартынов, А.П. Иванов. – М. : Кудиц-образ, 2000. – 336 с.

35. Мартынов Н.Н. Введение в MatLab 6. / Н.Н. Мартынов. – М. : Кудиц-образ, 2002. – 352 с.

36. Ануфриев И.Е. Самоучитель MatLab 5.x/6.x / И.Е. Ануфриев. – СПб. : БХВ-Петербург, 2002. – 736 с.

37. Ануфриев И.Е. MatLab 7 / И.Е. Ануфриев, А.Б. Смирнов, Е.Н. Смирнова. – СПб. : БХВ-Петербург, 2005. – 1104 с.

38. Дьяконов В.П. Simulink 4. Специальный справочник / В.П. Дьяконов. – СПб. : Питер, 2002. – 528 с.

39. Дьяконов В.П. Математические пакеты расширения MatLab. Специальный справочник / В. Дьяконов, В. Круглов. – СПб. : Питер, 2001. – 480 с.

40. Дьяконов В.П. MatLab. Анализ, идентификация и моделирование систем. Специальный справочник / В. Дьяконов, В. Круглов. – СПб. : Питер, 2002. – 448 с.

Зміст

Навчальне видання

Коржик Михайло Володимирович, к.т.н.

МОДЕЛЮВАННЯ ОБ'ЄКТІВ ТА СИСТЕМ КЕРУВАННЯ ЗАСОБАМИ MATLAB

Навчальний посібник

Текст посібника подано у авторській редакції

Коржик М. В.

Моделювання об'єктів та систем керування засобами MatLab: навч. посіб. для студ. вищ. навч. закл. / М. В. Коржик. – Київ : НТУУ "КПІ", 2016. – 174 с. : іл.

Видання містить теоретичні відомості та опис основних засобів системи MatLab / Simulink для ідентифікації, аналізу, моделювання, імітації та синтезу систем керування з лінійними стаціонарними об'єктами. Всі теми проілюстровано великою кількістю прикладів, наведено необхідний довідковий матеріал.

Для студентів вищих навчальних закладів, які навчаються за спеціальностю "Автоматизація та комп'ютерно-інтегровані технології" та споріднених їй.