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

О.І. Толочко

МОДЕЛЮВАННЯ СИСТЕМ АВТОМАТИЧНОГО КЕРУВАННЯ

Затверджено Вченою радою КПІ ім. Ігоря Сікорського
як підручник для студентів,
які навчаються за спеціальністю 141 «Електроенергетика,
електротехніка та електромеханіка»
за освітньою програмою «Електромеханічні системи автоматизації,
електропривод та електромобільність»

Київ
КПІ ім. Ігоря Сікорського

2024

Рецензенти: *Лозинський А.О.* доктор техн. наук, професор, директор інституту енергетики та систем керування національного університету «Львівська політехніка»
Шаповал І.А. доктор техн. наук, заступник директора інституту електродинаміки НАН України з наукової роботи

Відповідальний редактор *Ковбаса С.М.* доктор техн. наук, доцент кафедри автоматизації електромеханічних систем, електроприводу та електромобільности НТУУ «КПІ ім. Ігоря Сікорського»

Гриф видано Вченою радою Національного технічного університету України «Київський політехнічний університет України» імені Ігоря Сікорського (протокол №6 від 24 червня 2024 року)
Електронне мережне навчальне видання
Толочко Ольга Іванівна, д-р техн. наук, проф.

МОДЕЛЮВАННЯ СИСТЕМ АВТОМАТИЧНОГО КЕРУВАННЯ

Моделювання систем автоматичного керування [Електронний ресурс]: підручник для студентів спеціальності 141 «Електроенергетика, електротехніка та електромеханіка» за освітньою програмою «Електромеханічні системи автоматизації, електропривод та електромобільність» / О.І. Толочко; КПІ ім. Ігоря Сікорського. – Київ: КПІ ім. Ігоря Сікорського, 2023. – 328 с.

Розглянуто питання історії розвитку моделювання, класифікації моделей, методології цифрового математичного моделювання. Наведено основні відомості про моделювання в середовищі *Simulink,* про блоки бібліотек *Simulink* та рекомендації щодо їх застосування. Проаналізовано математичний опис лінійних та нелінійних неперервних та дискретних стаціонарних динамічних систем, представлення його у вигляді структурних схем, розробки *Simulink*-моделей, вибору методу і параметрів симуляції. Розглянуто методологію розробки математичних моделей розгалужених лінійних та нелінійних електричних кіл постійного та змінного струмів, двигунів постійного струму з незалежним збудженням, механічних систем з пружно-в'язкими кінематичними зв'язками та декількома ступенями вільності. Розглянуто особливості роботи цифрових інтеграторів та виконано порівняльний аналіз методів дискретної апроксимації неперервних динамічних об'єктів.

Викладено основні прийоми імітаційного фізичного моделювання електротехнічних та електромеханічних систем в середовищі *MATLAB-Simulink-SimPowerSystems*. Розглянуто методологію імітаційного фізичного моделювання електричних двигунів постійного та змінного струмів у різних режимах їх роботи та підходи до моделювання дводвигунних електроприводів, що працюють на спільний механічний вал.

Підручник рекомендовано для студентів вищих навчальних закладів спеціальності 141 «Електроенергетика, електротехніка та електромеханіка» освітньої програми «Електромеханічні системи автоматизації, електропривод и електромобільність».

ISBN УДК 62-82-52 (076.5)

ПЕРЕЛІК СКОРОЧЕНЬ

АД – асинхронний двигун

АОМ – аналогова обчислювальна машина

ДПС – двигун постійного струму

ДПФ – дискретна передавальна функція

ДР – диференціальні рівняння

ДМС – двомасова механічна система

ЕОМ – електронна обчислювальна машина

ОЗ – обмотка збудження

ОП – операційний підсилювач

ПК – персональний комп’ютер

ПФ – передавальна функція

РР – різницеві рівняння

ЦІ – цифровий інтегратор

ЦОМ – цифрова обчислювальна машина

ПЕРЕДМОВА

Математичне моделювання систем автоматичного керування є важливим етапом їх проектування. На цьому етапі виконується аналіз динамічних властивостей системи керування з точки зору відповідності технологічним вимогам, уточняється її структура і параметри. При моделюванні можна безбоязно досліджувати поведінку системи в аварійних ситуаціях, що неможливо на лабораторних і тим паче на діючих промислових установках. Іноді моделювання проводять для того, щоб оцінити коректність прийнятих при математичному опису системи спрощень**.**

Математичне моделювання на ЕОМ зводиться до розв'язання системи диференційних або різницевих рівнянь, що описують динамічні властивості досліджуваного об'єкта, в результаті чого отримують графіки перехідних процесів об'єкта в характерних режимах його роботи.

При дослідженні систем автоматичного регулювання і, зокрема, систем керування електроприводами вихідним матеріалом для моделювання дуже часто є структурна схема, що являє собою графічну інтерпретацію математичного опису системи. В структурну схему, крім неперервних або дискретних динамічних ланок можуть входити арифметичні ланки, «типові нелінійності» та нелінійності довільного вигляду, задані аналітично або графічно, різноманітні ключі, логічні елементи та т. і.

Перехідні процеси можна розраховувати як за допомогою програм, написаних будь-якою алгоритмічною мовою, що потребує від дослідника достатньо високої кваліфікації в галузі програмування та обчислювальної математики, так і за допомогою спеціалізованого програмного забезпечення, що дозволяє користувачу задавати моделі у вигляді математичних рівнянь або у вигляді структурних схем, обирати методи розв'язання диференційних рівнянь та їх параметри в діалоговому режимі та отримувати результати у зручній формі.

Одним із найзручніших програмних засобів структурного математичного моделювання на теперішній час є додаток *Simulink* пакета *MATLAB* фірми *Mathwork* [1-6].

Основою для розробки моделей в *Simulink* є бібліотеки блоків, з котрих складаються структурні схеми систем автоматичного керування, що мають бути дослідженими. Розрахунок перехідних процесів може бути виконаний за допомогою відповідних операцій *Simulink*-меню або в програмному режимі (з використанням функцій пакета *MATLAB*).

Наразі все більшу актуальність при дослідженні складних електромеханічних об'єктів набуває імітаційне фізичне моделювання, яке не потребують від користувача знання математичного опису досліджуваного процесу, потребують менше часу для налаштування. Такі моделі можна створювати і в *Simulink* шляхом використання віртуальних фізичних блоків бібліотеки *SimPowerSystems* . Блоки бібліотеки *SPS*, призначені для моделювання електричних та електромагнітних кіл, електронних пристроїв, електродвигунів і ліній електропередач, подано у вигляді позначень відповідних елементів на принципових електричних схемах [7-9]. Математичний опис окремих елементів цих бібліотек приховано від користувача, завдяки чому створюється ілюзія віртуального фізичного моделювання.

Метою курсу є ознайомлення з різним формами математичного опису фізичних процесів, використання їх при дослідженні електротехнічних пристроїв, механічних та електромеханічних систем методом математичного моделювання, набуття навичок структурного математичного та імітаційного фізичного моделювання лінійних та нелінійних, неперервних та дискретних динамічних систем.

При викладенні теоретичного матеріалу вважається, що студенти вже обізнані з основами *Windows*-технології, мають навички роботи і програмування в середовищі пакета *MATLAB* [2] та знають основи теорії автоматично-

го керування [4, 10] і основи електроприводу [11]. У якості об’єктів дослідження у прикладах взято розгалужені електричні кола постійного і змінного струмів, електричні машини, механічні системи поступального і обертального рухів з пружними кінематичними зв’язками та з декількома ступенями свободи, з якими студенти знайомляться при вивченні дисциплін «Теоретичні основи електротехніки», «Основи електроприводу», «Теоретична механіка», «Теорія електроприводу».

Отже, даний підручник розрахований на здобувачів ступеня бакалавр за освітньою програмою «Електромеханічні системи автоматизації, електропривод та електромобільність» спеціальності «Електроенергетика, електротехніка та електромеханіка». Він стане їм у пригоді при вивченні багатьох профілюючих дисциплін, виконанні розрахунково-графічних та курсових робіт і проектів. Він також створює базу для оволодіння знаннями і навичками з дисциплін, пов’язаних з моделюванням складних електротехнічних, електромеханічних, електронних і мехатронних систем, які вивчатимуться на старших курсах.

Підручник створено з орієнтацію на пакет *MATLAB* версії *R2013b*. Але ним можна користуватися і при роботі з більш новими версіями, що відрізняються одна від одної в основному інтерфейсом та наявністю додаткових інструментів та демонстрацій.

1. ЗАГАЛЬНІ ПОНЯТТЯ ПРО МОДЕЛЮВАННЯ

1.1. Історія розвитку моделювання

Під ***моделюванням***, в загальному випадку, розуміють заміну одного об'єкта, що зветься ***оригіналом***, іншим об'єктом, що зветься ***моделлю***, який має властивості оригіналу.

Моделювання як один із методів пізнання об'єктивної дійсності є філософською категорією.

Під моделлю в філософії розуміється така уявна або матеріально реалізована система, яка відображаючи або відтворюючи об'єкт дослідження, здатна заміщати його так, що її вивчення дає нам нову інформацію про досліджуваний об'єкт.

Слово модель походить від латинського «*modus*», «*modulus*», що означає образ, спосіб і т. п. Його первісне значення було пов'язано з будівельним мистецтвом, і майже у всіх європейських зиках воно вживалося у значенні способу, прообразу, або речі, схожої в якомусь відношенні з іншою річчю.

Один із способів класифікації моделей представлений на рис. 1.1.

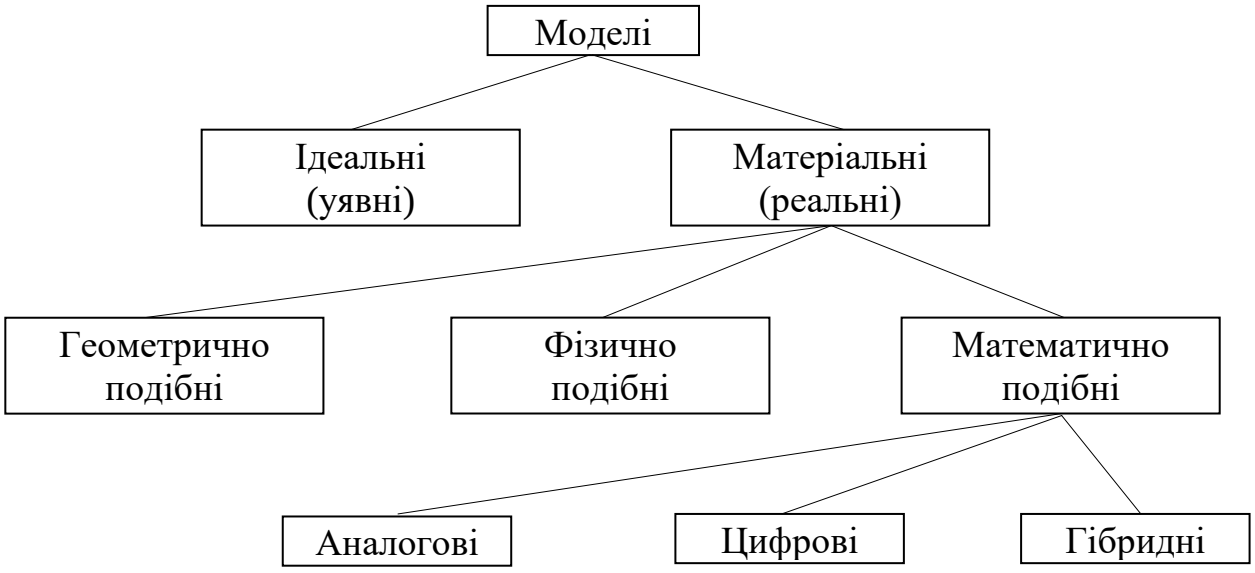


Рис. 1.1. Класифікація моделей

Усі моделі можна розділити на дві групи:

- ідеальні, або уявні;
- матеріальні, або реальні.

Важливою особливістю ***ідеальних моделей*** є те, що вони не завжди і не обов'язково втілюються у дійсності, хоча це і не виключено. Мало того, більшість ідеальних моделей і не претендують на матеріальне втілення. Найчастіше ідеальні моделі застосовують для того, щоб зобразити деяку область явищ за допомогою іншої, більш добре вивченої і більш звичної, тобто для того, щоб звести незрозуміле до зрозумілого.

Як приклад достатньо згадати гео- і геліоцентричні моделі Всесвіту, запропоновані відповідно Птоломеєм і Коперником, або уявлення Демокрита і Епікура про атоми, як про круглі частинки, «зчеплені між собою на зразок гілок сплетених». Фізики XVIII ст. намагалися зобразити оптичні, електричні, магнітні та інші мало вивчені явища за допомогою механічних: електричний струм порівнювали з течією рідини по трубках, рух молекул в газі – з рухом більярдних куль і т. п.

Матеріальні моделі діляться на три основні групи:

- просторово подібні;
- фізично подібні;
- математично подібні.

Просторові моделі представляють собою установки або споруди, що застосовуються для відображення просторових властивостей об'єкта. До цієї групи належать макети споруд, просторові моделі молекул і кристалів в хімії і т. п.

Фізичні моделі характеризуються схожістю фізичної природи і тотожністю законів руху з досліджуваним об'єктом. Фізичні моделі можуть відрізнятися від об'єкта не більше ніж зміною просторової і часової шкали. Прик-

ладами моделей, заснованих на зміні просторової шкали є моделі літаків, кораблів, гребель і т. п. Як приклад перетворення часової шкали у моделі в порівнянні з оригіналом можна навести використання в генетиці мушки дрозофіли у вигляді моделі для дослідження проблем спадковості через вельми велику швидкість розмноження цієї комахи.

Істотним недоліком фізичного моделювання є те, що при моделюванні кожного нового об'єкта потрібно будувати дуже дорогі нові моделі. До цього доводиться вдаватися навіть тоді, коли в уже існуючій моделі потрібно тільки лише змінити будь-якої параметр. Перевага ж фізичного моделювання полягає в тому, що для створення моделі не потрібно знати її математичний опис. Особливо широко фізичне моделювання почали застосовувати приблизно з середини XIX ст. в зв'язку з бурхливим розвитком експериментальної фізики.

XX століття принесло моделюванню нові успіхи, пов'язані з величезними досягненнями в області обчислювальної техніки, теорії автоматичного керування і кібернетики. На основі цих досягнень бурхливо розвивається метод математичного моделювання.

Математичне моделювання – це процес дослідження одного фізичного явища за допомогою іншого, описуваного ідентичними математичними виразами. Явище збігу математичного опису різних фізичних об'єктів називається ***принципом математичного ізоморфізму*** і являється основою аналогового математичного моделювання.

Розглянемо приклади з різних областей фізики. Процес нагрівання тіла в теплотехніці, процес руху тіла, скинутого з деякої висоти в аеродинаміці, і процес протікання струму в RL-колі з джерелом постійної ЕРС в електротехніці описуються відповідно диференціальними рівняннями першого порядку:

$$Q(t) = A\tau(t) + C\frac{d\tau(t)}{dt}; \qquad (1.1)$$

$$P(t) = \gamma v(t) + M\frac{dv(t)}{dt}; \tag{1.2}$$

$$E(t) = Ri(t) + L\frac{di(t)}{dt}. \tag{1.3}$$

У всіх цих рівняннях різними є тільки позначення, фізичний сенс величин і природа фізичних явищ, математичний же опис процесів є однаковим. У загальному вигляді рівняння описаних вище процесів можна представити у такий спосіб:

$$u(t) = k_1 x(t) + k_2\frac{dx(t)}{dt}, \tag{1.4}$$

де

t – незалежна змінна;

u – керуючий вплив (відома функція незалежної змінної);

x – вихідний сигнал (невідома функція незалежної змінної);

k_1, k_2 – постійні коефіцієнти.

Отже, математичне моделювання дозволяє вивчати одні фізичні явища за допомогою інших. Це робиться з метою спрощення і здешевлення виготовлення моделі, підвищення точності вимірювань і т.д. Найчастіше використовують електричні та електронні моделі, тому що вони компактні, позбавлені рухомих частин, дозволяють легко вимірювати і змінювати параметри моделі

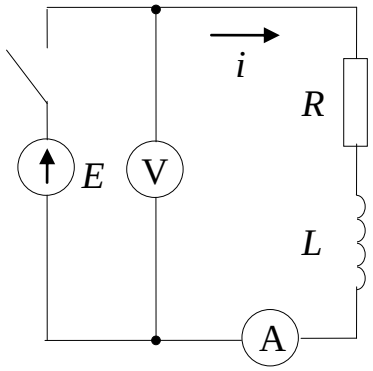


Рис. 1.2. Приклад електричної аналогової моделі

Наприклад, експеримент скидання парашутиста з літака, що описується рівнянням (1.2), можна досліджувати на електричної моделі з тотожним математичним описом (1.3), представленої на рис. 1.2.

Завданням математичного моделювання є розрахунок перехідних процесів, тобто графіків зміни залежних змінних в функції часу, що для

лінійних динамічних об’єктів еквівалентно рішенню диференційних рівнянь або їх систем.

Математичні моделі діляться на аналогові (неперервні) і цифрові (дискретні).

Аналогові математичні моделі відрізняються неперервністю перехідних процесів (інформація про стан моделі може бути отримана в будь-який момент часу) і паралельним принципом роботи (результати моделювання з’являються і змінюються в усіх точках моделі одночасно).

Інформація про стан ***цифрових математичних моделей*** може бути отримана тільки в деякі дискретні моменти часу, які визначаються тактовою частотою роботи дискретних пристроїв (апаратна реалізація) або кроком зміни незалежної змінної (програмна реалізація). Сигнали цифрових моделей розраховуються послідовно (по черзі).

Згідно з наведеними визначеннями модель рис. 1.1 відноситься до класу аналогових математичних моделей.

Аналогові математичні моделі при виборі параметрів потребують масштабування, завданням якого є забезпечення критерію подібності. Процеси називаються подібними, якщо в будь-який момент часу забезпечується пропорційність їх подібних величин.

Розглянемо процес масштабування і розрахунку параметрів на прикладі моделі рис. 1.1. Введемо масштаби за залежною (v) і незалежною (t) змінними та по вхідному сигналу (P) як відношення відповідних величин:

$$m_v = \frac{i}{v}; \qquad m_P = \frac{E}{P}; \qquad m_t = \frac{t_m}{t}. \tag{1.5}$$

В результаті підстановки виразів для величин процесу-оригіналу, отриманих із співвідношень (1.5), у рівняння (1.2), отримаємо:

$$E(t_m) = \gamma \frac{m_P}{m_v} i(t_m) + M \frac{m_P m_t}{m_v} \cdot \frac{di(t_m)}{dt_m}. \tag{1.6}$$

При порівнянні рівнянь (1.6) і (1.2) видно, що для забезпечення критерію подібності активний опір та індуктивність необхідно обирати з умов:

$$R = \gamma \frac{m_P}{m_v}; \qquad L = M \frac{m_P m_t}{m_v}. \tag{1.7}$$

Вхідний сигнал моделі E розраховується за формулою $E = P \cdot m_P$, а досліджувані величини процесу-оригіналу визначаються після вимірювання подібних сигналів моделі із співвідношень $v = i / m_v$, $t = t_m / m_t$.

Наведена електрична аналогова модель є ***моделлю прямої аналогії***, тому шо кожному фізичному параметру процесу-оригіналу відповідає подібний фізичний параметр процесу-моделі.

Інший вид математичного моделювання, заснований на непрямій аналогії, полягає в тому, що модель будується на підставі математичного опису, перетвореного в систему диференціального рівнянь (ДУ) 1-го порядку в нормальній формі Коші або відповідної цьому математичному опису структурній схемі, що складається з суматорів, інтеграторів, пропорційних ланок і нелінійних елементів. Наприклад, для процесу, що описується рівнянням (1.2),

$$\frac{dv}{dt} = \frac{1}{M} P - \frac{\gamma}{M} v. \tag{1.8}$$

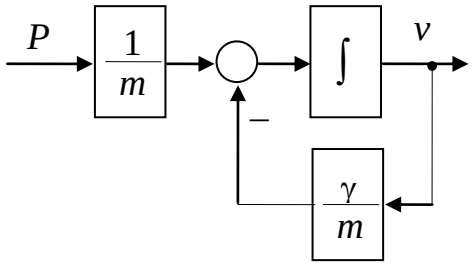


Рис. 1.3. Структурна схема досліджуваного механічного процесу

Побудована на підставі (1.8) структурна схема моделі матиме вигляд рис 1.3:

На базі такої структурної схеми можна створити як аналогову, так і цифрову математичну моделі об’єкта дослідження.

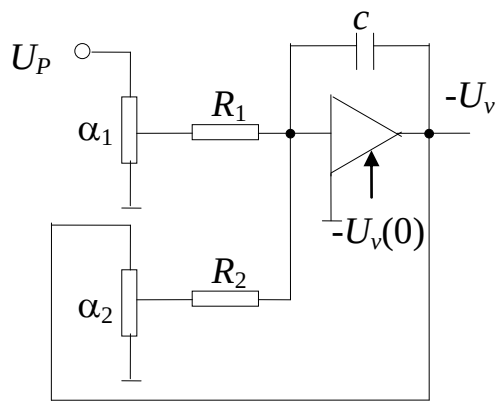
Аналогові моделі непрямої аналогії будують, як правило, на інтегруючих, масштабних і пропорційних операційних підсилювачах (ОП), що входять до складу ***аналогових обчислювальних машин (АОМ)***, які аж до 80-х рр. успішно конкурували як пристрої для ма-

тематичного моделювання з ***цифровими обчислювальними машинами (ЦОМ)***.

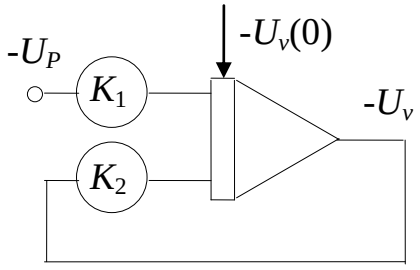
Аналогові моделі, побудовані за структурною схемою рис. 1.2 на основі ОП з регульованими коефіцієнтами передачі, представлені на рис. 1.4а,б. Модель рис. 1.4б є схематичним зображенням моделі рис. 1.4а. Знак «-» перед вихідною напругою моделі враховує інверсні властивості ОП.

Для розрахунку коефіцієнтів передачі ОП необхідно, як і в попередньому прикладі, виконати масштабування:

$$m_v = \frac{U_v}{v}; \qquad m_P = \frac{U_P}{P}; \qquad m_t = \frac{t_m}{t}. \tag{1.9}$$



а)



б)

Рис. 1.4. Аналогові математичні моделі на операційних підсилювачах

Підстановкою в (1.8) виразів для масштабів (1.9), отримаємо таке рівняння:

$$\frac{dU_v}{dt_m} = \frac{m_v}{Mm_P m_t} U_P - \frac{\gamma}{Mm_t} U_v, \tag{1.10}$$

звідкіля

$$K_1 = \alpha_1 R_1 C = \frac{m_v}{Mm_P m_t}; \; K_2 = \alpha_2 R_1 C = \frac{\gamma}{Mm_t}.$$

До складу АОМ, крім операційних підсилювачів, входять функціональні перетворювачі, блоки множення-ділення, набори резисторів, конденсаторів, потенціометрів, діодів, джерел напруги, апаратури для вимірювання та реєстрації сигналів та наборні поля. У якості найбільш популярних вітчизняних АОМ можна назвати МН-7, МН-17М, ЭМУ-10, АВК-31.

Зовнішній вигляд машин МН-7 та АВК-31 показано на рис. 1.5.

Основними перевагами АОМ в період піку їх популярності (60-70 рр.)

були доступність для індивідуального користування, наочність і неперервність інформації, висока швидкість моделювання, можливість неперервної зміни параметрів (за допомогою потенціометрів) і паралельний принцип роботи (рішення з'являється у всіх точках моделі одночасно). Однак після появи персональних ЕОМ гідності аналогових машин в значній мірі потьмяніли і на перший план вийшли їх недоліки.

Рис. 1.5. Зовнішній вигляд аналогових обчислювальних машин МН-7 та АВК-31

Основний недолік АОМ полягає в тому, що аналогові моделі збираються з окремих вирішальних блоків і елементів, що входять в комплект машини. Якісний і кількісний набір блоків АОМ накладає обмеження на складність розв'язуваних на ній задач. Зокрема, порядок диференційних рівнянь обмежується кількістю інтегруючих ОП, число нелінійних елементів – числом функціональних перетворювачів і т. д.). Ці ж обставини унеможливлюють одночасне зберігання великої кількості моделей, транспортування, обміну ними і т. п. Крім того, вирішальні блоки і апаратура АОМ, що реєструє вихідні сигнали, володіють невисокою точністю і недостатньою завадозахищеністю; вони менш точно і більш складним чином, ніж ЦОМ відтворюють різні нелінійні залежності, логічні функції, мало придатні для моделювання дискретних систем і процесів. До недоліків АОМ можна також віднести та-

кож необхідність масштабування і відносну складність зміни параметрів (один і той же параметр може входити у вирази для декількох коефіцієнтів, змінювати ці параметри доводиться вручну).

Цифрове моделювання полягає в розрахунку перехідних процесів в досліджуваному об'єкті на підставі його математичного опису з використанням чисельних методів на ***цифрових обчислювальних машинах (ЦОМ)***, які частіше називають ***ЕОМ (електронні обчислювальні машини)***, тому що АОМ, які також відносяться до ЕОМ, в останній час стали екзотикою.

У ***створенні та розвитку ЦОМ*** слід перелічити такі етапи: механічна обчислювальні машини з паровим приводом Чарльза Беббіджа (1822-1871 рр.); розрахункова перфораційна машина Холеріта (1896 р.), перші релейні комп'ютери з програмним керуванням (К. Цузе – 1938-1945 рр., Д. Штибитц – 1940-1944 рр.); перша електронна лампова обчислювальна машина ENIAC (Д. Маучлі – 1946 р. з такими характеристиками: 18000 електронних ламп, площа 90 × 15 м^2, вага 30 т, потужність 150 кВт. тактова частота 100 кГц, швидкість додавання 0,2 мс, швидкість множення – 2,8 мс, що на три порядки скоріше за релейні машини); ЕОМ з архітектурою Неймана – 1948-1950 рр. На заміну ламповим ЕОМ, які називають машинами першого покоління у 50-х рр. прийшли транзисторні (друге покоління) та у 60-х рр. – комп'ютери, побудовані на інтегральних мікросхемах (третє покоління), найбільш відомими серед яких були IBM-360, IBM-370 та їх східноєвропейські аналоги ЕС-1022, ЕС-1032. Четверте покоління ЕОМ (70-ті рр.), до якого належать і персональні комп'ютери, створено на базі мікропроцесорів.

Цифрове моделювання з використанням мов програмування високого рівня вимагає від програміста досить високої кваліфікації і знання методів обчислювальної математики. У зв'язку з цим після переходу від «великих ЕОМ» до персональних комп'ютерів (ПК) розроблено багато пакетів, в яких користувачеві досить ввести або рівняння, що описують об'єкт, або структу-

рну схему, вибрати метод і параметри чисельного інтегрування і вказати сигнали, інформацію про які він хотів би отримати в результаті моделювання. До таких пакетів належать Electronics Workbench, OrCad, VisSim, LabVew, Modelica та багато інших.

Найбільш популярним з таких пакетів є *MATLAB*, що містить в своєму складі в якості додатка програми структурного моделювання *Simulink*, а також масу «інструментів» (*Toolboxes*) для аналізу, синтезу та оптимізації систем автоматичного регулювання як в числовому так і в аналітичному вигляді.

Основною перевагою сучасних ЦОМ є її універсальність і можливість тривалого зберігання практично необмеженої кількості моделей. Універсальність ЦОМ полягає в тому, що на ній можна моделювати як неперервні, так і дискретні процеси, досліджувати роботу логічних схем, легко і з високою точністю відтворювати різні нелінійності, автоматизувати процес зміни параметрів і вибір їх оптимальних значень. ЦОМ, на відміну від АОМ, надає можливість зберігання, і надійного тиражування практично необмеженої кількості моделей, в тому числі і дуже складних. Перевагою цифрового моделювання є також і те, що немає необхідності масштабування, тому що діапазон чисел досить великий (до 1.798×10^{308}) і забезпечується гарна візуалізація результатів як в чисельної, так і в графічній формі.

До ***недоліків цифрового моделювання*** можна віднести те, що ЦОМ є машиною послідовної дії (сигнали моделі розраховуються в певній черговості), а також дискретність інформації та необхідність використання для вирішення диференціальних рівнянь ітераційних чисельних методів.

Сучасні системи електроприводу являють собою цифроаналогові динамічні системи, в яких електромеханічна частина описується диференційними рівняннями, а система керування і силовий підсилювач (перетворювач напруги або частоти) – різницевими та логічними рівняннями. Для моделювання таких систем можна застосовувати ***гібридне (цифроаналогове) мате-***

матичне моделювання, яке реалізується на гібридних обчислювальних системах (ГОС). Такий підхід дозволяє максимально використовувати переваги обох видів математичного моделювання.

Об'єктом дослідження при математичному моделюванні є ***перехідні процеси***, тобто графіки зміни координат системи у функції часу.

Процеси і системи, в яких вони протікають, можуть мати саму різну фізичну природу: вони можуть бути тепловими, гідро- і аеродинамічними, електричними і електромагнітними, механічними, ядерними, оптичними. До електромеханічних систем відносяться системи електроприводу різних механізмів. Основними елементами систем електроприводу є електродвигуни та робочі органи виконавчих механізмів. Крім цього, вони можуть містити перетворювачі електричної енергії та елементи системи керування (регулятори, датчики, коригувальні пристрої і т. п.).

1.2. Класифікація моделей за їх математичним описом

Математичний апарат, що застосовується при математичному опису системи, визначається належністю системи або перехідних процесів, що у ній відбуваються до того чи іншого класу.

Поняття «процес» у загальному сенсі цього слова означає зміну у часу матерії, енергії або інформації у деякій системі. У вузькому сенсі словом «процес» позначають ту саму динамічну систему, величини якої змінюються певним чином під впливом зовнішніх впливів і (або) параметричних збурень та ненульових початкових умов. Застосування того чи іншого математичного опису при створенні математичних моделей визначає також і тип моделей. Отож, процеси і системи, у яких вони протікають, та математичні моделі, що використовуються при їх дослідженні, можна розподілити на такі класи:

- лінійні (*linear*) та нелінійні (*nonlinear*);
- неперервні (*continuous*), дискретні (*discrete*) та гібридні;
- з зосередженими параметрами та з розподіленими параметрами;

- стаціонарні системи (*time invariant systems*) та системи зі змінними параметрами або структурою (*time variant systems*);
- стохастичні і детерміновані.

Неперервні (аналогові) лінійні стаціонарні динамічні системи (Continuous Linear Time Invariant systems – LTI-systems) описуються ***звичайними лінійними диференційними рівняннями (ДР)*** з постійними коефіцієнтами, а дискретні ***(Discrete LTI-systems)*** – ***різницевими рівняннями (РР)***. Незалежною змінною у цих рівняннях є час. Схематично ці форми математичного опису можна подати у вигляді ***неперервних та дискретних структурних схем***.

У неперервних системах кожна координата може змінити своє значення в будь-який момент часу і на будь-яку величину, а в дискретних системах - тільки в деякі дискретні моменти часу (***квантування за часом – Sample Time***) і на величину, кратну деякій константі (***квантування за рівнем***). Існують також гібридні системи, у яких одні величини змінюються неперервно, а інші – дискретно.

Крім диференційних та різницевих рівнянь, що описують динаміку систем, взаємозв'язок між окремими величинами системи може бути відображений за допомогою алгебраїчних (лінійних або нелінійних) та трансцендентних рівнянь. Такі рівняння називають ***рівняннями зв'язку або рівняннями виходу.***

У лінійних (linear) системах всі зв'язки між вхідними та вихідними величинами системи є лінійними. У структурних схемах таких систем крім лінійних динамічних ланок можуть бути присутніми тільки пропорційні блоки (операції множення координати на постійний коефіцієнт) та алгебраїчні суматори. Перехідні процеси в лінійних системах не змінюють свого характеру при зміні амплітуди вхідних сигналів.

В нелінійних (nonlinear) системах можуть бути присутніми як лінійні, так і нелінійні зв'язки між окремими координатами. Наявність хоча б одного нелінійного зв'язку робить систему нелінійною. До нелінійних операцій також відносяться операції множення та ділення декількох сигналів. Нелінійні системи в залежності від амплітуди вхідних сигналів можуть змінювати свої динамічні і статичні властивості.

Якщо аналітичний вираз нелінійного зв'язку між деякими сигналами системи залишається невідомим, то його завдають у вигляді ***таблиці «вхід-вихід»***. Для математичного опису табличних функцій використовують їх ***апроксимацію або інтерполювання***.

Стаціонарні (Time Invariant) системи не змінюють своїх властивостей протягом часу спостереження перехідних процесів, а ***нестаціонарні (Time Variant) системи*** – це системи зі змінними параметрами. Отже, перехідні процеси нестаціонарних систем змінюються навіть при однакових вхідних сигналах, але прикладених в різні моменти часу.

Нестаціонарні процеси і системи можна розділити на детерміновані і стохастичні.

Детерміновані процеси – це такі, які можна описати за допомогою явних математичних формул. ***Стохастичні процеси*** є випадковими. Для їх математичного опису застосовують поняття теорії вірогідності та методи математичної статистики.

В окрему групу динамічних об'єктів слід виділити системи з розподіленими параметрами, які описуються диференційними рівняннями у часткових похідних. При аналізі і особливо при синтезі систем керування такими об'єктами досить часто намагаються замінити диференціальні рівняння у часткових похідних звичайними ДР.

Контрольні питання та завдання

1. Що таке моделювання? Навіщо його використовують?

2. Охарактеризуйте види моделей і їх характерні ознаки.
3. Порівняйте між собою геометричне і фізичне моделювання.
4. Назвіть недоліки та переваги фізичного та математичного моделювання.
5. Порівняйте між собою аналогове та цифрове математичне моделювання.
6. Як поділяються системи, процеси, що в них протікають, та математичні моделі для їх дослідження за математичним описом?

2. ОСНОВНІ ЕТАПИ МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ

Процес моделювання можна розбити на такі етапи:

- розробка математичного опису об’єкта моделювання;
- перетворення його до вигляду, зручного для моделювання;
- алгоритмізація;
- програмування;
- введення програми та її налагодження;
- експлуатація програми;
- аналіз отриманих результатів.

2.1. Розробка математичного опису досліджуваних процесів

Розробка математичного опису досліджуваного об’єкта є дуже важким і відповідальним етапом математичного моделювання. В теоретичному плані на цьому етапі математичну модель можна вже вважати створеною.

Для розробки математичного опису дослідник має бути добре обізнаним з ***фізикою процесів***, які він вивчає, та добре володіти різноманітним ***математичним апаратом***.

Крім того, перед розробкою математичної моделі завжди роблять деякі ***припущення***, які дозволять спростити модель. Припущення мають бути такими, щоб суттєві властивості процесу-оригіналу модель відображала, а несуттєвими його особливостями нехтувала. Розподіл особливостей процесу-оригіналу на суттєві і несуттєві не є однозначним. Він залежить від того, яку саме задачу розв’язує дослідник і носить суб’єктивний характер. Зазвичай на першому етапі досліджень приймають якомога більше припущень. Наприклад, при малому такті квантування за часом нехтують дискретністю об’єкта за часом, при великій розрядності обчислювального пристрою і роботі його в арифметиці з плаваючою точкою нехтують дискретністю за рівнем, при роботі систем у штатних режимах нехтують нелінійностями типу «обмеження

координат» та насиченням потоку у сталі магнітопроводів, при повільній зміні параметрів у часі вважають об’єкти стаціонарними. Досить часто також нехтують такими явищами як тертя, пружність кінематичних передач, наявність у статичних характеристиках гістерезису. При моделюванні електричних машин та багатьох інших пристроїв не враховують неточності їх виготовлення. Наприклад, повітряний зазор в електричних машинах вважають постійним, розподілення магнітного потоку в повітряному зазорі рівномірним,обмотки двигунів постійного струму симетричними і т.п. Різноманітні припущення є однією з причин наявності декількох математичних моделей одного об’єкта.

Наприклад, щоб отримати більш-менш доступні для дослідження математичні моделі асинхронного двигуна (АД), при їх розробці на першому етапі зазвичай роблять такі допущення:

- намагнічувальні сили обмоток розподілені синусоїдально вздовж кола рівномірного повітряного зазору, тобто відсутні вищі гармоніки магнітного потоку;
- втрати та насичення магнітних кіл у сталі статора та ротора відсутні;
- обмотки статора та ротора симетричні, тобто фазні обмотки мають однакову кількість витків;
- комплексні опори обмоток не мають ємних складових;
- параметри обмоток ротора приведені до статора;
- відсутнє явище витиснення струму з пазів;
- джерело живлення є ідеальним джерелом ЕРС або струму.

Далі моделі можна ускладнювати залежно від того, яке саме із знехтуваних явищ повинно бути досліджено.

При використанні моделей АД у трифазних природних координатах досить просто можна врахувати асиметрію електромагнітних параметрів фаз статора і ротора, несинусоїдальність і асиметрію сигналів живлення, наяв-

ність в них вищих гармонійних складових, тощо.

Задача математичного моделювання складних фізичних об'єктів спрощується завдяки тому, що у фахових літературних джерелах вже існує велика кількість варіантів математичного опису важливих для науки і практики явищ та пристроїв.

2.2. Перетворення математичного опису до вигляду, зручного для математичного моделювання

2.2.1. Відображення математичного опису у вигляді структурних схем

Форма математичного опису досліджуваних систем залежить від обраного математичного забезпечення. Оскільки ***Simulink є програмою структурного моделювання***, то на основі математичного опису треба створити структурну схему системи. Для складання структурної схеми на базі ДР або РР останні спочатку записують в операторній формі, а потім з них визначають аналогові або дискретні передавальні функції, які з'єднують між собою у відповідності з алгебраїчними рівняннями зв'язку та рівняннями виходу, як це буде показано в наступних лекціях. По суті ***структурні схеми*** є графічним відображенням математичного опису.

2.2.2. Деталізація структурних схем

Іноді структурні схеми та відповідні їм структурні моделі деталізують. ***Деталізованою зветься така структурна схема***, у якій єдиним типом неперервної динамічної ланки є інтегратор (***Integrator***) $1/s$, а єдиним типом дискретної динамічної ланки – блок запізнення на період дискретності (***Unit Delay***) $1/z = \exp(-Ts)$. Деталізовані структурні моделі лінійних динамічних систем можуть отримувати у своєму складі ще тільки пропорційні ланки (блоки ***Gain***) та алгебраїчні суматори (блоки ***Sum, Subtract, Add, Sum of Elements***). В теорії автоматичного керування деталізовані структурні схеми на-

зивають ***структурними схемами у просторі станів***, а вихідні сигнали інтеграторів називають ***змінними стану***.

Деталізація структурних схем доцільна в наступних випадках:

- при розрахунку перехідних процесів з ненульовими початковими умовами;
- для запобігання операції явного диференціювання для отримання інформації про похідну від вихідного сигналу;
- при необхідності виділення у явному вигляді неперервних та дискретних змінних стану та конкретизації їх розташування (математичний опис у просторі станів – State Space);
- при необхідності переводу інтеграторів, що входять до складу динамічних ланок з обмеженням вихідного сигналу з режиму інтегрування у режим запам'ятання вихідної інформації (обмеження ПІ-регулятора);
- при створенні матричних структурних моделей.

Слід відзначити, що з неперервних динамічних ланок, що входять до складу основних *Simulink*-бібліотек тільки на інтеграторі можна встановити початкові умови, і тільки інтегратор може обробляти векторні сигнали.

Деталізовані структурні схеми легко утворюються з математичного опису системи у нормальній формі Коші, або відповідним перетворенням передавальної функції (ПФ).

Наприклад, з ПФ загального вигляду без нулів

$$W(s)=\frac{y(s)}{u(s)}=\frac{H(s)}{G(s)}=\frac{1}{s^n+\alpha_{n-1}s^{n-1}+...+\alpha_1 s+\alpha_0} \qquad (2.1)$$

можна отримати ДР n-го порядку в операторній формі, яке розв'язане відносно старшої похідної:

$$s^n y(s)=u(s)-\alpha_{n-1}s^{n-1}y(s)-...-\alpha_1 py(s)+\alpha_0 y(s). \qquad (2.2)$$

Після введення позначень

$$x_n(s)=y(s),\quad x_{n-1}(s)=sy(s),\quad ...,\quad x_2(s)=s^{n-2}y(s),\quad x_1(s)=s^{n-1}y(s) \qquad (2.3)$$

рівняння (2.2) можна перетворити до системи диференційних рівнянь першого порядку в нормальній формі Коші

$$\begin{cases} sx_1(p) = u(s) - \alpha_{n-1}x_1(s) - \ldots - \alpha_1 x_{n-1}(s) - \alpha_0 x_n(s), \\ sx_2(s) = x_1(s), \\ \ldots \\ sx_n(s) = x_{n-1}(s), \end{cases} \tag{2.4}$$

якій відповідає деталізована структурна схема, зображена на рис. 2.1.

При наявності у ПФ загального вигляду нулів

$$W_1(s) = \frac{y_1(s)}{u(s)} = \frac{H(s)}{G(s)} = \frac{\beta_m s^m + \beta_{m-1}s^{m-1} + \ldots + \beta_1 s + \beta_0}{s^n + \alpha_{n-1}s^{n-1} + \ldots + \alpha_1 s + \alpha_0} \tag{2.5}$$

($m \le n$) вихідний сигнал $y_1(s)$ можна отримати з вихідного сигналу $y(s)$ системи рис. 2.1 за формулою

$$y_1(s) = \beta_0 y(s) + \beta_1 s y(s) + \ldots + \beta_{m-1}s^{m-1}y(s) + \beta_m s^m y(s),$$

або з урахуванням позначень (2.3)

$$y_1(s) = \beta_0 x_n(s) + \beta_1 x_{n-1}(s) + \ldots + \beta_{m-1}x_{n-m+1}(s) + \beta_m x_{n-m}(s). \tag{2.6}$$

Деталізована структурна схема системи з ПФ (2.5) при $m = n$ зображена на рис. 2.2. Слід відзначити, що кожна структурна схема може мати декілька варіантів її деталізації, пов’язаних з можливістю виконання еквівалентних структурних перетворень.

Для неперервних динамічних блоків першого-другого порядків можна запропонувати деталізовані структурні схеми, подані в табл. 2.1.

Приклад 2.1. *Отримати реакції аперіодичної ланки з передавальною функцією* $W(p) = \dfrac{5}{2p+1}$ *на стрибкоподібний сигнал, що змінюється в момент часу 1 с. з 1 на 0 при ненульових початкових умовах.*

Моделі до цього прикладу показано на рис. 2.3.

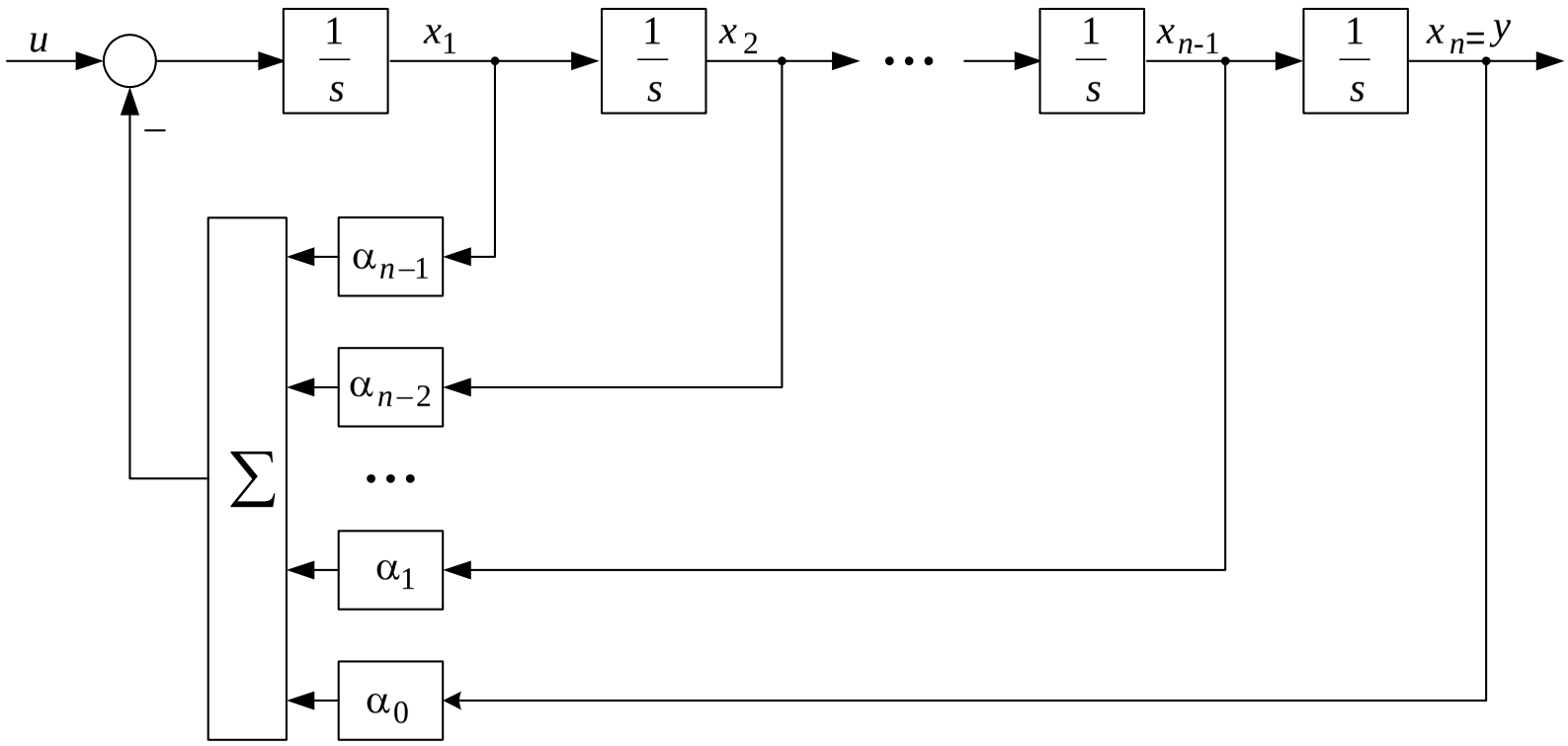


Рис. 2.1. Скалярна структурна схема у просторі станів
системи з передавальною функцією (2.1)

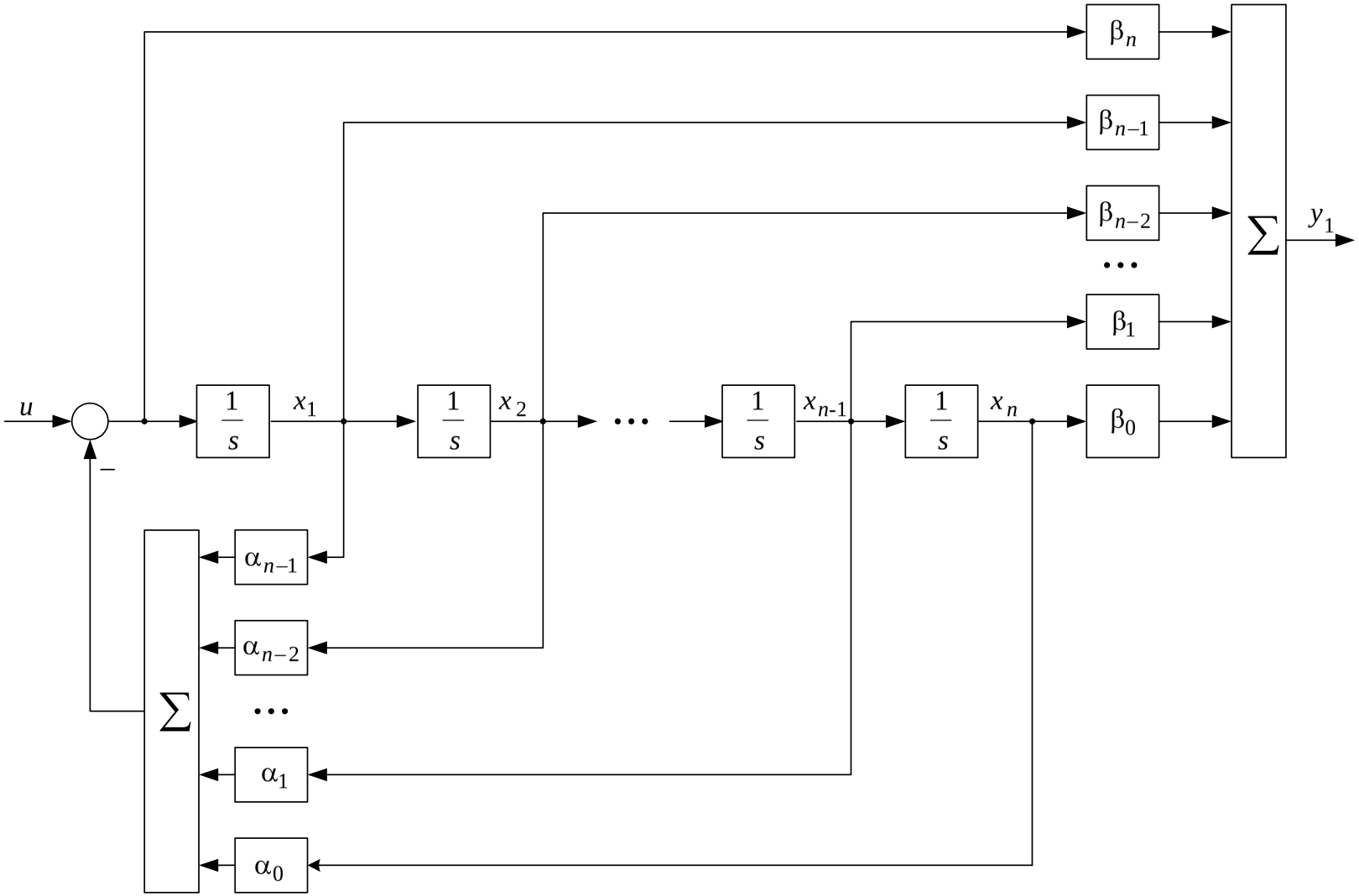
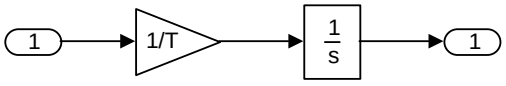
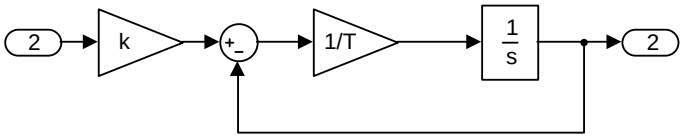
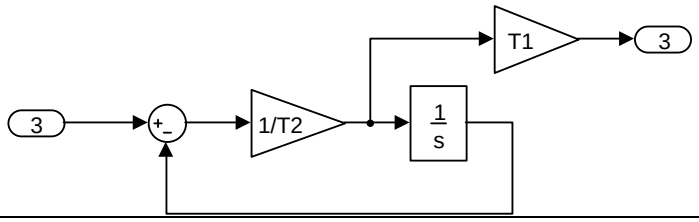
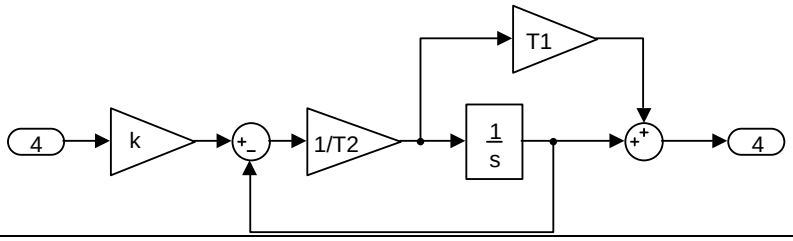
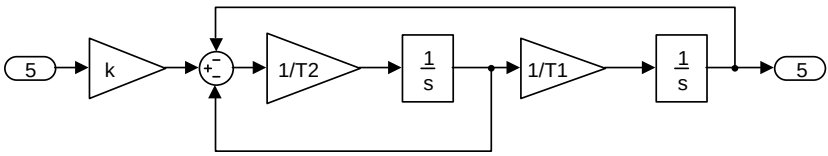
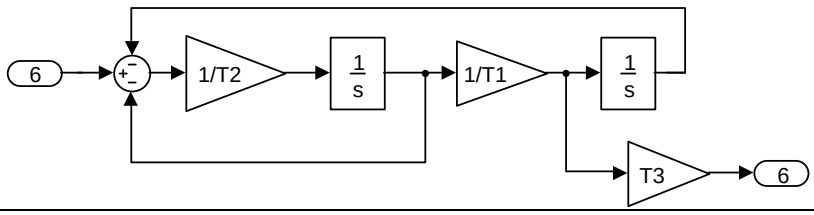
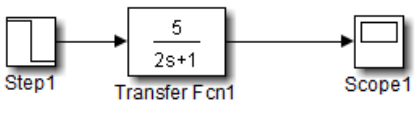
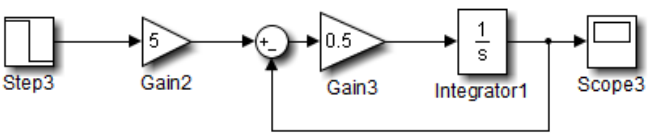


Рис. 2.2. Скалярна структурна схема у просторі станів системи з передавальною функцією (2.5)

Таблиця 2.1

Вихідна передавальна функція	Перетворена передавальна функція	Деталізована структурна схема
$\frac{1}{Ts}$	$\frac{1}{T}\cdot\frac{1}{s}$	
$\frac{k}{Ts+1}$	$k\cdot\frac{\frac{1}{Ts}}{\frac{1}{Ts}+1}$	
$\frac{T_1 s}{T_2 s+1}$	$\frac{\frac{1}{T_2 s}}{\frac{1}{T_2 s}+1}\cdot T_1 s$	
$k\frac{T_1 s+1}{T_2 s+1}$	$k\left(\frac{1}{T_2 s+1}+\frac{T_1 s}{T_2 s+1}\right)$	
$\frac{1}{T^2 s^2+2\xi Ts+1}=$ $=\frac{1}{T_1 T_2 s^2+T_1 s+1}$	$\frac{\frac{1}{T_1 s(T_2 s+1)}}{\frac{1}{T_1 s(T_2 s+1)}+1}$	
$\frac{T_3 s}{T_1 T_2 s^2+T_1 s+1}$	$\frac{T_3 s}{T_1 T_2 s^2+T_1 s+1}$	

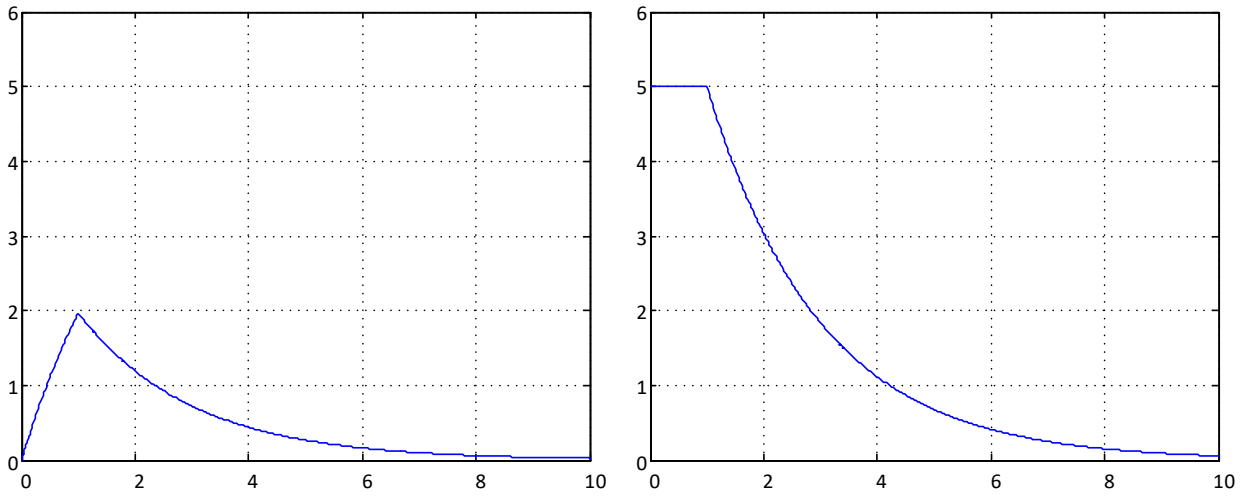




а) *б)*

Рис. 2.3. Моделі до прикладу 2.1

На рис. 2.3а аперіодична ланка промодельована одним блоком Transfer Function, у якому не передбачено встановлення початкових умов, а на рис. 2.3б – у деталізованому вигляді, обов'язковою ознакою якого є наявність інтегратора, в якому можна встановити початкові умови завдяки наявності параметру Initial Condition. Графіки перехідних процесів для обох випадків показані на рис. 2.4. У другому випадку початкова умова на інтеграторі відповідає усталеному значенню вихідного сигналу аперіодичної ланки при початковому значенні сигналу блока Step.



а) *б)*

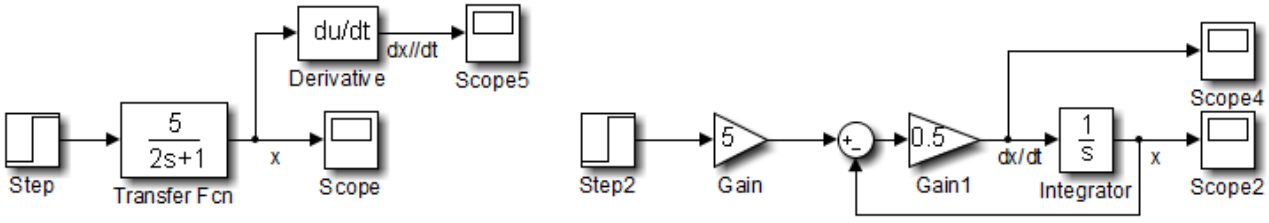
Рис. 2.4. Графіки до прикладу 2.1

Приклад 2.2. *Для аперіодичної ланки з передавальною функцією* $W(p)=\frac{5}{2p+1}$ *побудувати перехідну та вагову функції методом структурного математичного моделювання.*

Як відомо, перехідна функція h(t) *є реакцією на одиничну функцію Хевісайда 1(t), а вагова* w(t) *– реакцією на δ-функцію Дірака* δ(t)=d1(t)/dt. *Вагова та перехідна функції пов'язані між собою співвідношенням* w(t)=dh(t)/dt. *Одинична функція в Simulink формується блоком Step, а δ-функція, яка має форму прямокутного імпульсу з безкінечно великою амплітудою та безкінечно малою тривалістю, можна сформувати тільки дуже приблизно. Тому*

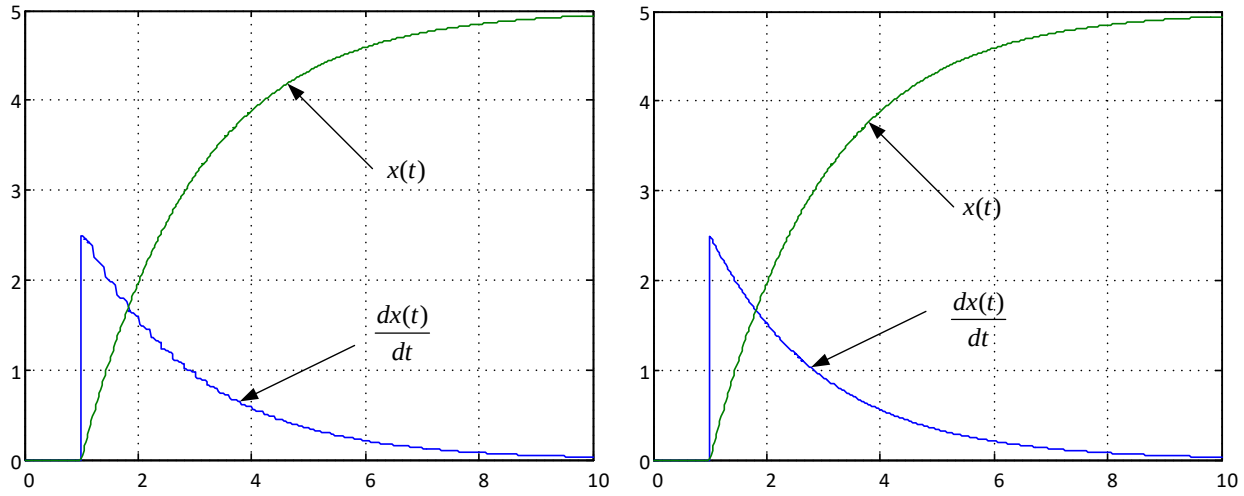
для того, щоб отримати вагову функцію, необхідно отримати похідну від перехідної функції, тобто на виході аперіодичної ланки установити диференційну ланку $W_d(p) = p$. *Але лінійна динамічна ланка Transfer Fn в Simulink не може реалізувати передавальну функцію, у якої порядок полінома в чисельнику перевищує порядок полінома у знаменнику.*

Для розрахунку похідних в Simulink передбачено блок Derivative (бібліотека Continues), що виконує диференціювання чисельними методами. Структурна модель із застосуванням блока Derivative показана на рис. 2.5а, а результати її роботи – на рис. 2.6а.



а) *б)*

Рис. 2.5. Моделі до прикладу 2.2



а) *б)*

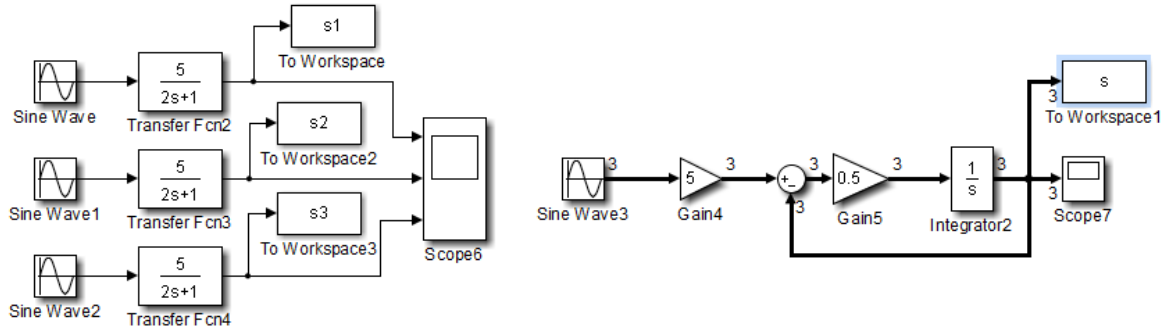
Рис. 2.6. Графіки до прикладу 2.1

Як бачимо, при використанні такої моделі вагова функція має дискретний характер, що зумовлено низькою точністю операції чисельного диференціювання.

Щоб позбавитися від цього недоліку, деталізуємо аперіодичну ланку згідно зі схемою, наведеною у табл. 2.1. Тоді потрібну похідну отримаємо на вході інтегратора деталізованої ланки, як це показано на рис. 2.5б. У цьому разі сигнал вагової функції стає гладким (див. рис. 2.6б).

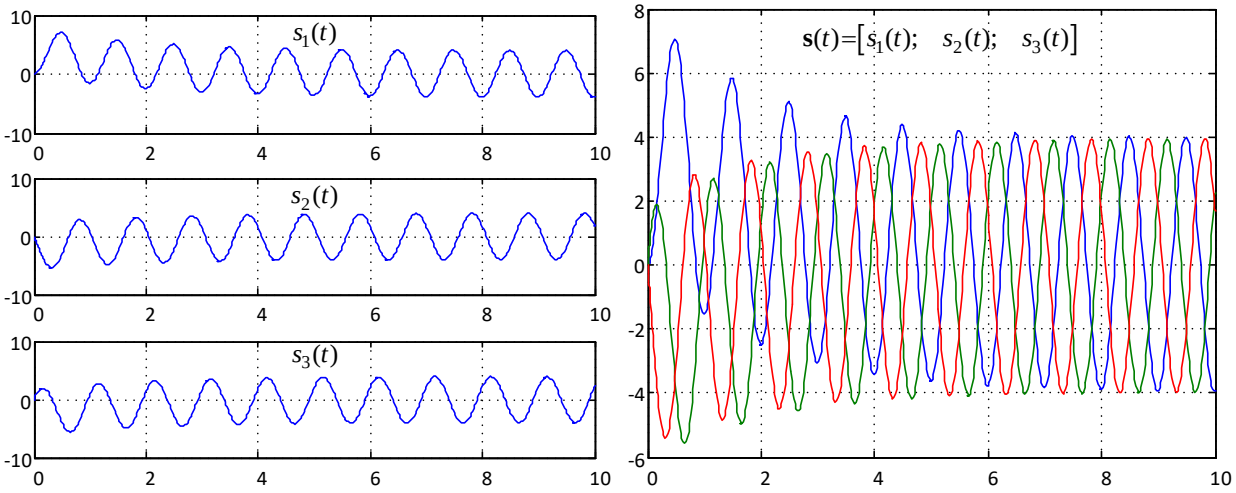
Приклад 2.3. *Скласти скалярну та векторну моделі, які формують 3 синусоїдальних сигнали однакової амплітуди, зсунуті один від одного на 120°, і пропускають кожен із них через аперіодичну ланку з передавальною функцією* $W(p)=\dfrac{5}{2p+1}$.

Скалярна модель до цього прикладу подана на рис. 2.7а, а матрична – на рис. 2.7б. Результати моделювання відображені на рис. 2.8а,б.



а) *б)*

Рис. 2.7. Скалярна (*а*) та векторна (*б*) моделі до прикладу 2.3



а) *б)*

Рис. 2.8. Графіки до прикладу 2.3

2.2.3. Нормування структурних схем

Дуже часто моделювання виконують у ***відносних одиницях (в.о.).***

Відносною величиною $\bar{y}$ називають відношення відповідної ***абсолютної величини*** y до її ***базового значення*** y_b:

$$\bar{y} = y / y_b. \tag{2.7}$$

Структурна схема у в.о. зветься унормованою.

Абсолютні фізичні величини, як правило, мають якусь розмірність, наприклад, струм в Амперах, напруга у Вольтах, потік у Веберах тощо, а ***відносні величини завжди безрозмірні***. При вдалому виборі базових величин нормування структурної схеми виключає з неї параметри, що не впливають на динаміку системи. Результати моделювання у відносних одиницях ***в.о.*** (англ. ***p.u. – per unit***) мають більш загальний характер, ніж у абсолютних одиницях ***а.о.*** (англ. ***SI Units***), і більш зручні для аналізу та для зображення декількох графіків перехідних процесів в одній системі координат.

Якщо відома передавальна функція (ПФ) ланки в а.о.

$$W(p) = y(p) / u(p), \tag{2.8}$$

то ПФ у в.о. можна розрахувати за формулою:

$$\bar{W}(p) = \bar{y}(p) / \bar{u}(p) = \frac{y(p) \cdot u_b}{y_b \cdot u(p)} = W(p) \cdot \frac{u_b}{y_b}. \tag{2.9}$$

2.3. Алгоритмізація і програмування

На стадії алгоритмізації розроблюється план розв’язання задачі дослідження, який включає у себе такі основні етапи:

- вибір режимів роботи, тобто форми та часу подачі вхідних сигналів, умов, за якими відбувається зміна структури моделі; вибір параметрів, вплив яких на якість перехідних процесів треба оцінити, тощо;
- ініціалізацію моделі, тобто введення вхідних даних, розрахунок незмін-

них параметрів та початкових умов, орієнтовний вибір методу та параметрів чисельного розв'язання системи диференційних рівнянь;

- організацію розрахунку перехідних процесів з фіксацією необхідних результатів;
- організацію обробки результатів моделювання у вигляді графіків, таблиць, анімації тощо.

Для орієнтовного ***вибору методу та параметрів чисельного інтегрування*** (ЧІ) ДР можна користуватися такими відомостями та міркуваннями:

1. Методи Рунге-Кутта можуть працювати з постійним кроком h_i та за алгоритмами, що організують автоматичний вибір кроку з умов забезпечення бажаної точності розрахунку ε.
2. Методи прогнозу та корекцій забезпечують бажану точність не за рахунок автоматичного вибору кроку, а за рахунок використання ітераційних формул.
3. Диференціальні рівняння, що описують поведінку систем зі сталими часу, що значно відрізняються одна від одної, тобто $(T_{\max}/T_{\min})>>1$, називаються жорсткими (*stiff*). Для зменшення часу їх розв'язання існують спеціальні методи ЧІ, наприклад, метод Гіра.
4. Постійний крок ЧІ слід обирати при наявності в моделі блоків, що отримують ділянки статичних характеристик з нескінченими коефіцієнтами підсилення у замкнених структурах, а також при моделюванні цифро-аналогових систем з постійними періодами дискретності.
5. Значення постійного кроку ЧІ та бажаної точності (максимально-припустимої похибки) повинно забезпечити компроміс між точністю та часом розрахунку перехідних процесів.
6. Вибір максимально припустимої похибки ЧІ. Чим вище порядок p методу ЧІ, тим вище точність розв'язання диференційних рівнянь при однаковому крокові ЧІ та тим менший крок забезпечує однакову точність:

$$\varepsilon = Ah^p \qquad (2.4)$$

(A – коефіцієнт, значення якого залежить від методу ЧІ, виду та параметрів системи ДР, що розв'язується).

7. В добре спроектованих, тобто не дуже коливальних, неперервних системах крок ЧІ визначається величиною мінімальної сталої часу досліджуваної моделі. При моделюванні таких систем задовільна точність досягається орієнтовно для методів Рунге-Кутта (4-5)-го порядків при $h_i = T_{\min}/2$, для методів Рунге-Кутта (2-3)-го порядків при $h_i = T_{\min}/5$, і для методу Рунге-Кутта 1-го порядку (метод Ейлера) $h_i = T_{\min}/10$.
8. При моделюванні періодичних процесів на кожному періоді треба розраховувати близько 100 точок.
9. При моделюванні цифро-аналогових систем необхідно слідкувати за тим, щоб фіксовані у часі переключення та всі періоди переривання були кратними кроку ЧІ.

Програмування полягає у розробці структурної *Simulink*-моделі або декількох моделей, програм та функцій, що забезпечують їх роботу згідно з планом модельного експерименту. Моделі створюють за допомогою стандартних бібліотечних блоків. Якщо їх не вистачає, користувач може створювати нові власні блоки, що виконують потрібні операції, та власні бібліотеки. Модель має бути наочною і досить компактною. Для цього застосовують об'єднання деяких частин моделі у підсистеми, маскування підсистем, змінюють назви блоків у відповідності з їх сенсом у конкретній моделі, назви деяких блоків приховують, підписують лінії зв'язку, користуються можливістю вибору кольорів та т. ін.

2.4. Налагодження та експлуатація програми. Аналіз результатів

При налагодженні моделі треба уточнити метод та параметри чисельного інтегрування (ЧІ), виявити випадкові та логічні помилки, оцінити адек-

ватність моделі, тобто відповідність її досліджуваному об’єкта і лояльність прийнятих при математичному опису припущень.

Для того, щоб упевнитись у ***правильності обраного кроку ЧІ***, можна порівняти між собою результати трьох розрахунків перехідних процесів, один із яких виконано з кроком, обраним з урахуванням перелічених в попередньому пункті міркувань, другий – з удвічі меншим, а третій – з удвічі більшим кроком. Якщо всі результати збігаються із задовільною точністю, то обраний крок можна збільшити, інакше його треба зменшити.

При ***уточненні часу перехідних процесів*** та часів зміни вхідних сигналів домагаються, щоб у час закінчення розрахунку всі сигнали або їхні амплітуди (при наявності гармонічних складових) досягли своїх усталених значень та щоб ділянки з усталеними значеннями сигналів мали невелику тривалість. Це дасть змогу спостерігати в гарному масштабі саме перехідні процеси, а не усталені режими.

Якщо кількість розрахованих точок виявляється недостатньою ***для якісної візуалізації*** можна виконати такі дії з параметрами вікна ***Model Configuration Parameters (^E)***:

- при використанні методу *ode45* встановити значення параметру *Refine factor* вкладки *Data Import/Export* більшим за 1 (максимум 10);
- замінити метод *ode45* методами *ode23, ode23s* або *ode15s*;
- зменшити на 1-2 порядки значення параметру *Relative tolerance*;
- проаналізувати та встановити потрібне значення параметру *Max step size*.

Якщо кількість розрахованих точок при заданих параметрах перевищує заданий діапазон, що визначається параметром *Limit data point to last*, який за замовчанням має значення 1000, то у пам’яті залишиться інформація тільки про останні 1000 точок. Що позбутися цього недоліку, треба або прибрати прапорець з поля цього параметру, або збільшити його значення (максимум

дорівнює *Inf*).

Одним з перших кроків ***оцінки адекватності моделі*** є орієнтовна оцінка статичних та динамічних властивостей системи в характерних режимах її роботи на основі власних аналітичних досліджень або аналізу літературних джерел.

Експлуатація програми полягає в розв'язанні методом математичного моделювання конкретних науково-дослідних задач.

Останнім етапом має бути ***аналіз одержаних результатів***, без чого усі попередні етапи не мають сенсу. При аналізі результатів дослідник отримує нові знання про об'єкт або переконується у вірогідності висновків, одержаних іншим шляхом, наприклад, аналітичними методами, використовуючи інформацію, свої знання, вміння та навички, одержані при вивченні дисциплін за фахом.

Контрольні питання та завдання

1. В якому порядку виконується дослідження динамічних систем методом математичного моделювання?
2. Чому один і той же процес може мати декілька різних математичних моделей?
3. Наведіть приклади припущень, до яких вдаються при розробці математичних моделей електромеханічних об'єктів.
4. Що таке деталізована структурна схема? З яких блоків може складатися деталізована структурна модель лінійної динамічної неперервної системи?
5. Наведіть приклад явного та неявного диференціювання вихідного сигналу неперервної динамічної ланки. Порівняйте перехідні процеси. Чи можна застосувати метод неявного диференціювання для динамічних ланок, передавальні функції яких мають однаковий порядок поліномів у чисельнику і знаменнику (однакову кількість ну-

лів і полюсів)?

6. Наведіть приклад моделювання будь-якої динамічної ланки з ненульовими початковими умовами.
7. Порівняйте скалярну і матричну структурні моделі системи, що описується системою диференційних рівнянь ($f = 50 Hz$; $\omega = 2\pi f$, rad / s; $U_m = 380V$; $R_s = 0,04\ \Omega$; $L_s = 2\ mHn$)

$$\begin{cases} U_A(t) = U_m \sin(\omega t) = R_s I_A(t) + L_s \dfrac{dI_A(t)}{dt}, \\ U_B(t) = U_m \sin(\omega t - 2\pi / 3) = R_s I_B(t) + L_s \dfrac{dI_B(t)}{dt}, \\ U_C(t) = U_m \sin(\omega t - 4\pi / 3) = R_s I_C(t) + L_s \dfrac{dI_C(t)}{dt}. \end{cases}$$

8. Що таке абсолютні та відносні одиниці, базові величини? Що таке нормування математичного опису? Як унормувати структурну схему?
9. Виконати нормування структурної схеми двигуна постійного струму з незалежним збудженням, структурна схема якого показана на рис. 2.9, при застосуванні таких базових величин: $U_{яb} = E_{дb} = U_{яn}$ – номінальна напруга якоря, $\omega_b = U_{яb} / c = \omega_0$ – швидкість ідеального холостого ходу, $I_{яb} = U_{яb} / R_я = I_{кз}$ – струм короткого замикання якоря, $M_b = cI_{яb} = M_{кз}$ – момент короткого замикання. Врахуйте, що $\dfrac{J\omega_0}{M_{кз}} = T_м = \dfrac{JR_я}{c^2}$ – електромеханічна стала часу приводу.

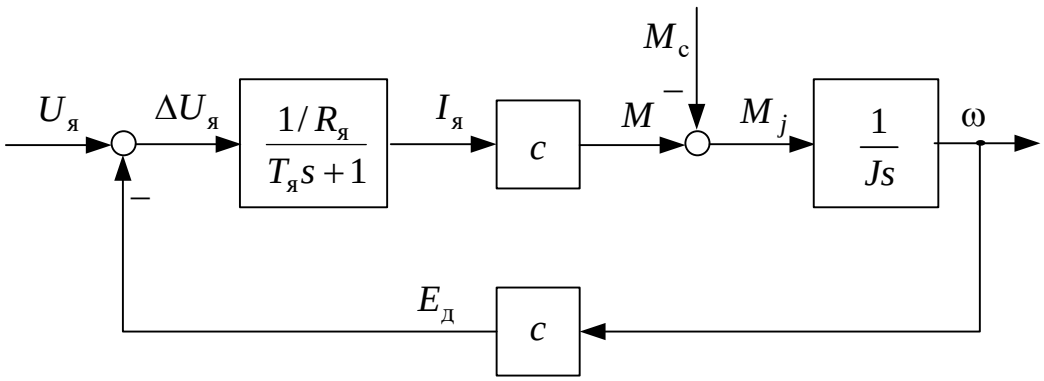


Рис. 2.9. Структурна схема двигуна постійного струму
в абсолютних одиницях

10. У чому полягає налаштування моделі?
11. Які методи чисельного рішення диференційних рівнянь Ви знаєте? Охарактеризуйте їх.
12. Як обирати час перехідного процесу? Як покращити якість візуалізації перехідних процесів? Проілюструйте відповідь за допомогою моделі, що складається з джерела синусоїдального сигналу з частотою 50 Гц та аперіодичної ланки зі сталою часу 0,04 с.
13. Як підібрати крок інтегрування при застосуванні методів чисельного інтегрування з постійним кроком?
14. Що таке жорсткі диференціальні рівняння. Які методи треба застосовувати при їх розв’язанні?
15. Як оцінити адекватність розробленої математичної моделі?

3. СЕРЕДОВИЩЕ ПРОГРАМИ СТРУКТУРНОГО МОДЕЛЮВАННЯ *Simulink* СИСТЕМИ ПРОГРАМУВАННЯ *MATLAB*

3.1. Запуск *Simulink*. Огляд бібліотек

Simulink можна запустити з командної строки пакета *MATLAB* одноменною командою

>>*simulink*

де >> – запрошення *MATLAB* до вводу команди, або спеціальною кнопкою *Simulink Library* на панелі інструментів, що показана на рис. 3.1.

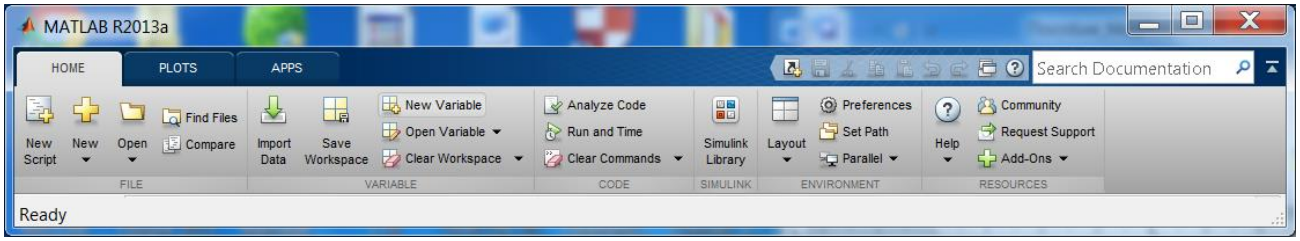


Рис. 3.1. Фрагмент командного вікна системи *MATLAB*

При цьому на екрані відкривається вікно з бібліотеками блоків, представлене на рис. 3.2, яке має назву *Simulink Library Browser (Навігатор Бібліотек).*

Кожну бібліотеку можна розкрити подвійним натиском миші на піктограмі бібліотеки в правій частині вікна або вибором назви бібліотеки в лівій частині вікна. При цьому піктограми блоків обраної бібліотеки заміщають піктограми бібліотек.

Натиском правою кнопкою миші на імені бібліотеки через меню, що випадає, можна відкрити будь-яку бібліотеку у вигляді, звичному для користувачів більш старих версій (*MATLAB 4.0 - MATLAB 5.1, Simulink2 - Simulink3*).

Simulink зі старим інтерфейсом [3] можна також відкрити з командної строки командою

>>*simulink3*

При цьому відкривається вікно, зображене на рис. 3.3.

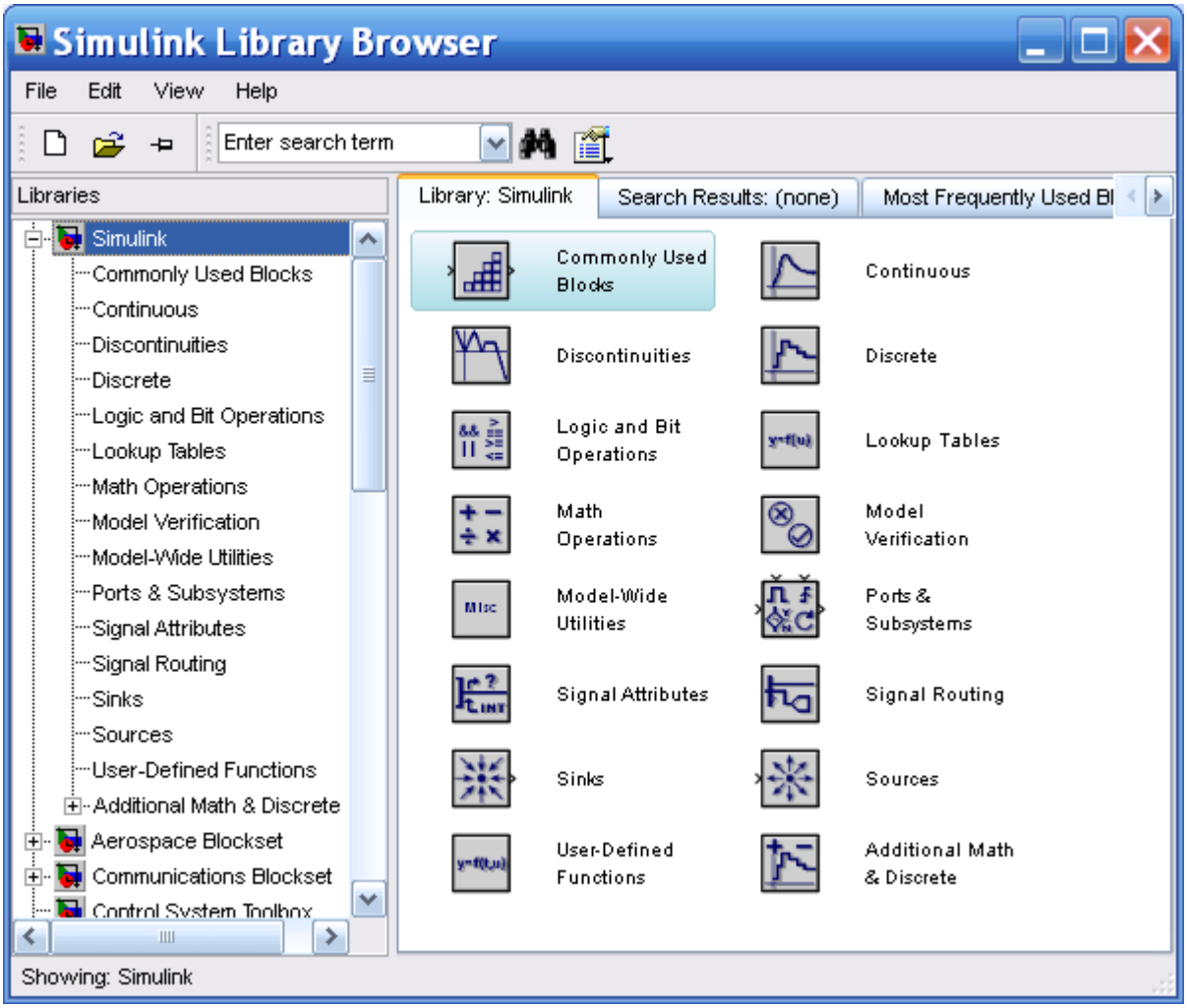


Рис. 3.2. Вікно *Simulink Library Browser*

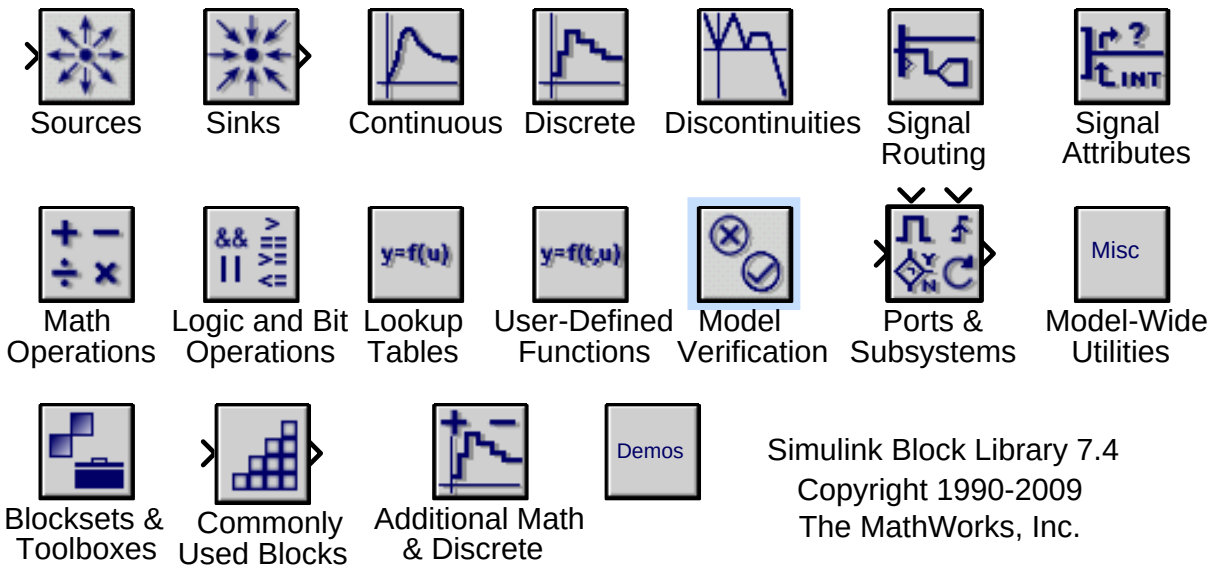


Рис. 3.3. Вікно бібліотек блоків *Simulink*

З цього вікна кожну бібліотеку можна розкрити подвійним натиском миші по її піктограмі. Вікна бібліотек будуть показані нижче (при розгляді їхніх блоків).

До основних *Simulink-бібліотек* відносяться (верхній ряд вікна рис. 3.3)

- *Sources* – джерела вхідних сигналів;
- *Sinks* – блоки реєстрації та візуалізації;
- *Continuous* – неперервні лінійні динамічні ланки;
- *Discrete* – дискретні лінійні динамічні ланки;
- *Discontinuities* – типові нелінійності;
- *Math Operations* – математичні операції;
- *Logic & Bit Operations* – логічні та бітові операції;
- *Lookup Tables* – кусково-лінійна апроксимація табличних функцій;
- *User Defined Functions* – функції користувача;
- *Ports & Subsystems* – порти та підсистеми;
- *Commonly Used Blocks* – блоки, що використовуються найчастіше;
- *Blocksets & Toolboxes* – комплектні блоки та комплекти інструментів.

Елементи розділу *Blocksets & Toolboxes* показано на рис. 3.4. Перелічимо деякі з них:

- *Signal Processing Toolbox* – створення, обробка та аналіз сигналів;
- *Robust Control Toolbox* – інструменти робастного керування;
- *Fuzzy Logic Toolbox* – інструменти нечіткої логіки;
- *System Identification* – ідентифікація систем;
- *MPC Toolbox* – інструменти прогнозного керування (*MPC Model Predictive Control Toolbox*);
- *Neural Network Blockset* – елементи нейрональних мереж;
- *Simpower Systems* – елементи електричних схем і систем, силова

електроніка, електричні машини;

- *Report Generator* – генератор звітів;
- *Simulink design optimisation* – оптимізація синтезу;
- *Real-Time Workshop* – майстерня реального часу;
- *Simulink extras* – спеціальні блоки *Simulink*.

Ретельний опис кожного з цих розділів вимагає окремої книги.

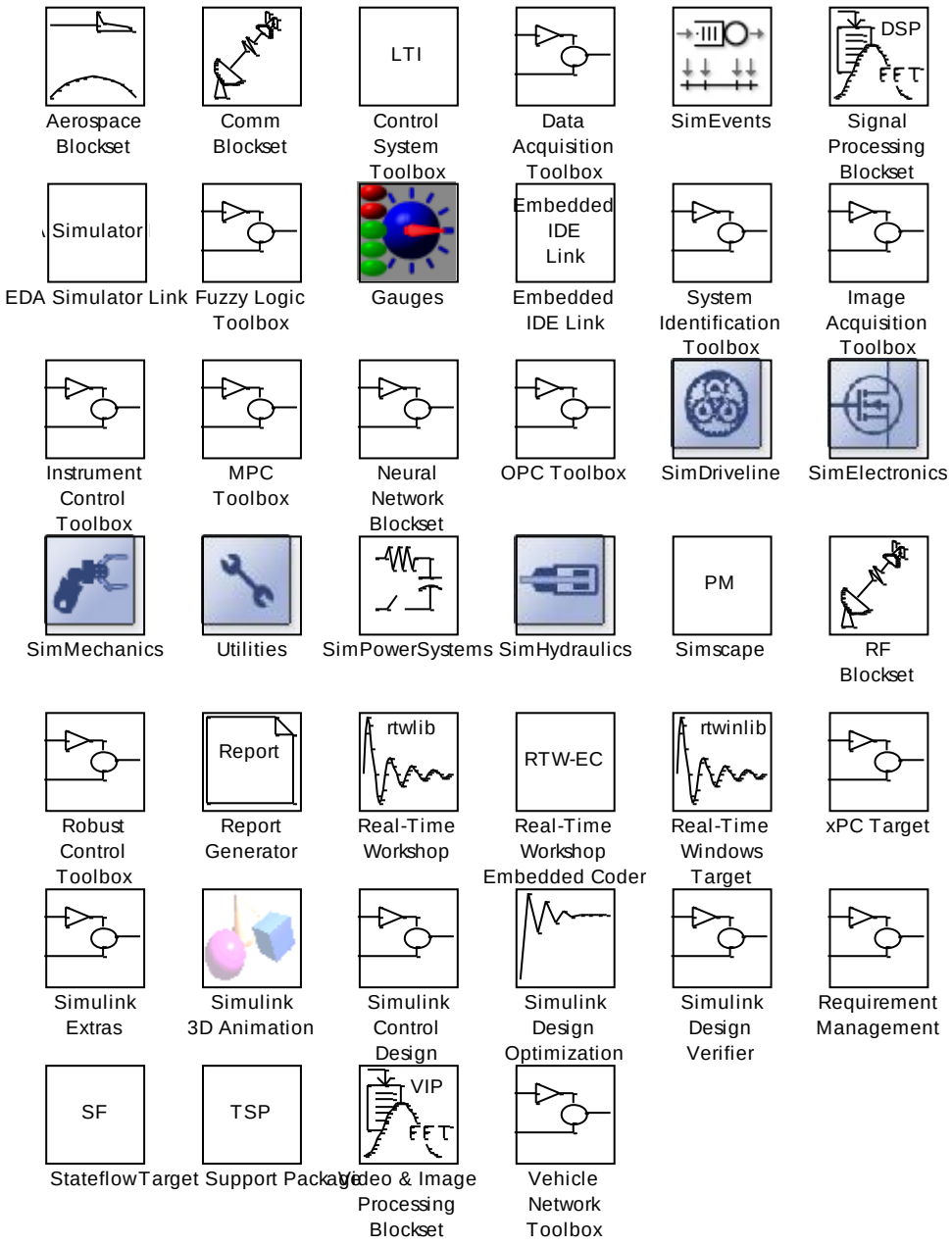


Рис. 3.4. Вікно бібліотек *Blocksets & Toolboxes*

Зі спеціальних бібліотек (*Simulink extras*), представлених на рис. 3.5,

для спеціалістів в галузі електропривода найбільшої уваги заслуговують [3]:

- *Additional Linear* – аналогові лінійні неперервні динамічні ланки з початковими умовами (п. у.);
- *Additional Discrete* – дискретні динамічні ланки з п. у.;
- *Additional Sinks* – пристрої, що реєструють, для спектрального аналізу;
- *Flip Flops* – тригери.

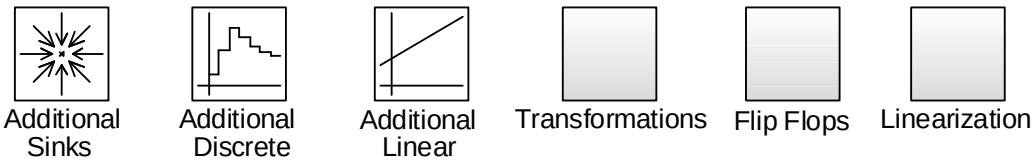


Рис. 3.5. Вікно бібліотек *Simulink extras*

Окремо зупинимося на додатку віртуального фізичного моделювання *SimScape*, до складу якого, як видно з рис. 3.6, поряд з фундаментальною бібліотекою *Foundation Library* та бібліотекою механічних блоків *SimMe?hanics* входять бібліотеки гідравлічних блоків *SimHidravlics* (насоси), електронних пристроїв *SimElectronics*, трансмісій *SimDriveline*.

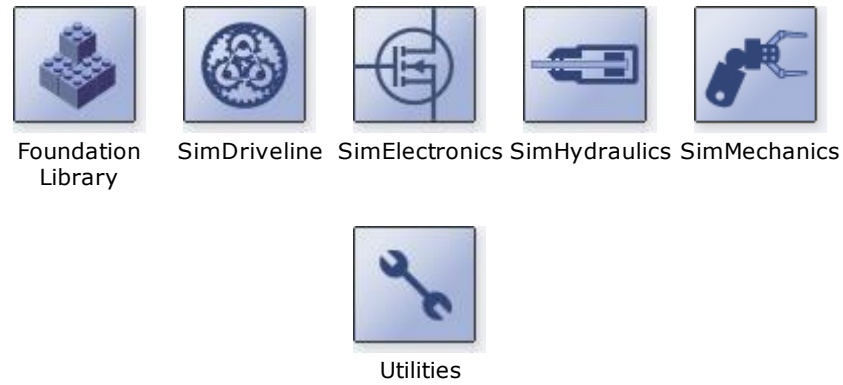


Рис. 3.6. Вікно бібліотек *SimScape*

При подвійному натиску мишею по піктограмі блока відкривається вікно введення його параметрів, у якому відображуються ім’я блока, його тип,

короткий опис і рядки введення параметрів із супроводжуючими їх запитами. Більш докладну інформацію про блок можна побачити, звернувшись до функції *Help*. Піктограма блока найчастіше відображує його динамічні або статичні властивості і реагує на зміну параметрів.

Як приклад, на рис. 3.7 наведено вікно введення параметрів блока *Transfer Function*.

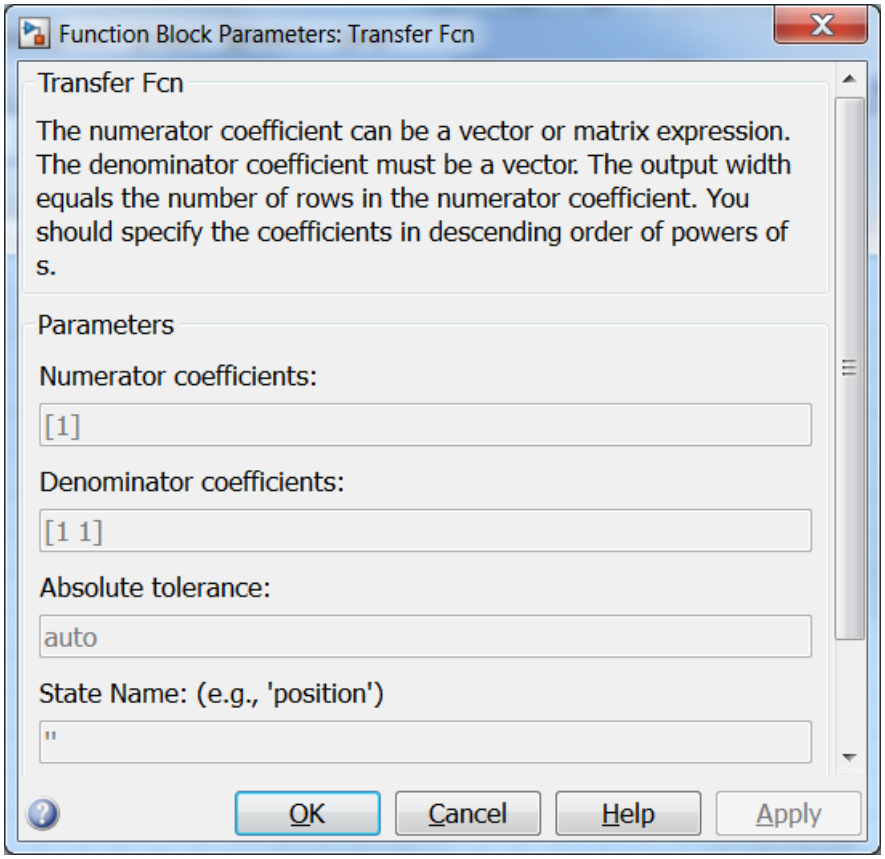


Рис. 3.7. Вікно введення параметрів блока *Transfer Fcn*

Параметри блоків можуть бути константами, змінними, функціями або виразами, припустимими в *MATLAB*. Будь-які змінні, від яких залежить параметр, мають бути визначені в робочому середовищі до початку процесу моделювання, а в іншому випадку *Simulink* сигналізує про помилку в цьому блоці.

Клавіша *Apply* дозволяє оцінити результат зміни параметрів, не закриваючи вікна їхнього введення, клавіша *OK* закриває вікно введення парамет-

рів із запам'ятовуванням виконаних змін, клавіша *Cancel* закриває вікно введення параметрів зі скасуванням внесених змін.

Для знайомства з можливостями додатка *Simulink* та особливостями роботи деяких блоків можна скористуватися демонстраціями, які зручно запускати з вікна *MATLAB Demo Window*. Воно з'являється на екрані при виконанні команди

```
demo
```

або

```
demo simulink.
```

Для збереження моделей і інших файлів користувача в *Windows* передбачена папка *C:\Users\...\Documents\Matlab*. У ній можна створити нову папку, наприклад, *C:\Users\...\Documents\Matlab\models*. Шлях до файлів-моделей необхідно додати у список доступних директорій за допомогою операції *Set Path→ Add Folder… / Add with Subfolders…* функції *File* основного меню командного вікна *MATLAB*, наведеного на рис. 3.8. ***Усі компоненти шляху до файлів MATLAB можуть складатися тільки з латинських букв, цифр і знаку підкреслення та починатися з латинської букви.*** Нову папку можна установити в початок списку (*Move to Top*), у кінець списку (*Move to Bottom*), перемістити на одну позицію вниз (*Move Down*) або нагору (*Move Up*). Якщо не виконати запис нового списку у файл, то установка діє тільки протягом поточного сеансу роботи. Для того, щоб використовувати нову установку в майбутніх сеансах, потрібно записати її у файл (*MATLAB \ toolbox\local\pathdef.m*) за допомогою клавіші *Save.*

Файли моделей і бібліотек додатка *Simulink* повинні мати маску **.mdl* або **.slx*. Вони є текстовими файлами особливої структури і містять інформацію про структуру та параметри моделей, достатню для їхнього графічного відображення і моделювання. З вікон *Simulink* вони відкриваються в графічному вигляді, а з будь-якого текстового редактора (наприклад, вбудованого текстового редактора системи *MATLAB edit.exe*) – у текстовому. Почи-

наючи з версії *MATLAB-2012a*, розширення *mdl* замінено на *slx*. ***Текстовий редактор*** відкривається кнопкою *New Script* з командного вікна *MATLAB.*

Відкрити вікно для створення нової моделі можна не тільки командою *File → New → Model (^N)* будь-якого *Simulink-вікна*, але й кнопкою *New → Simulink → Simulink Model* пакета *MATLAB.*

Вже існуючі текстові файли і моделі відкриваються командою *Open* з відповідного вікна, а також з командного рядка *MATLAB* введенням у ній імені файлу без розширення. ***Оскільки MATLAB не розрізняє імена змінних і різноманітних файлів, то всі вони повинні мати оригінальні імена***, які мають відрізнятися також від зарезервованих імен системи *MATLAB.*

Якщо при спробі відкрити модель або виконати яку-небудь програму з командного рядка на екран виводиться повідомлення «*Undefined function or variable 'name'»* (невизначена функція або змінна), то це означає, що користувач або помилився при наборі імені файлу, або цей файл знаходиться в недоступній для системи *MATLAB* директорії (у більш пізніх версіях *MATLAB* приєднання нових директорій до списку доступних виконується системою програмування автоматично при зверненні до будь-якого файлу цієї директорії).

В кожному з вікон середовища *MATLAB* можна виконувати деякі команди за допомогою функцій меню або віртуальних кнопок. Але перелік цих команд залежить від встановленої конфігурації та змінюється з появою нових версій. До того ж є команди, які потребують від користувача підвищеної кваліфікації. Тому нижче розглянуто найважливіші з них.

3.2. Основні команди *Simulink*

3.2.1. Команди, що виконуються з вікна *Simulink Library Browser*

Меню вікна *Simulink Library Browser* містить такі функції: *File* – робота з файлами, *Edit* – редагування, *View* – вигляд вікна; *Help* – допомога. Кожну функцію цього меню можна вибрати мишею або клавішами <→>,

<←> і <*Enter*>.

Функція File містить наступні операції: *New...* ▸ – створити вікно для введення нової моделі (*Model (^N)*) або бібліотеки (*Library*), *Open... (^O)* – відкрити модель, *Close* – закрити вікно *Simulink Library Browser*, *Preferences…* – переваги (настроювання середовища).

Команда Preferences, при виборі якої відкривається вікно, наведене на рис. 3.8, дозволяє виконувати настроювання середовища *MATLAB* і *Simulink*.

Функція Edit дозволяє знайти блок за його ім'ям – *Find block…(^F)* та скопіювати обраний блок або бібліотеку в активне вікно моделі або бібліотеки – *Add to the current model (^I).*

При використанні команд ***функції View*** слід звернути увагу на команди функції *Zoom*, що керують розміром моделі *Zoom in* (^+), *Zoom out* (^-), *Normal view 100%* (*Alt+1*). Змінювати розмір моделі можна також колесом миші.

3.2.2. Команди, що виконуються з вікон *Simulink*-моделей

Меню вікон *Simulink*-моделей відрізняється від меню вікна *Simulink Library Browser* і отримує функції та кнопки, кількість та перелік яких обирається у вкладці *Editor Defaults* вікна *Simulink Properties*. Одна з можливих конфігурацій вікна *Simulink*-моделі наведена на рис. 3.9.

Перші п'ять кнопок виконують стандартні операції *Windows:* 1 – *New Model* (відкрити нове *Simulink*-вікно), 2 – *Save* (зберегти), 3 – *Print* (надрукувати), 4,5 – Undo, Redo (скасувати, відновити результат останнього редагування).

Інші клавіши виконують специфічні операції *Simulink*: 6,7,8 – Model Browser (уперед, назад, активізувати батьківську модель); 9 – *Library Browser* (активізувати „навігатор бібліотек"); 10 – *Model Configuration Parameters* (параметри моделювання); 11 – *Model Explorer* (увімкнути „навігатор моделі"); 12 – Stepping Options (вибір параметрів покрокового режиму) 13 – *Run*

(почати моделювання); 14 – *Step Forward* (крок уперед); 15 – *Stop simulation* (зупинка моделювання); 17 – *Simulation Stop Time* (час моделювання); 18 – *Simulation Mode* (вибір режиму моделювання).

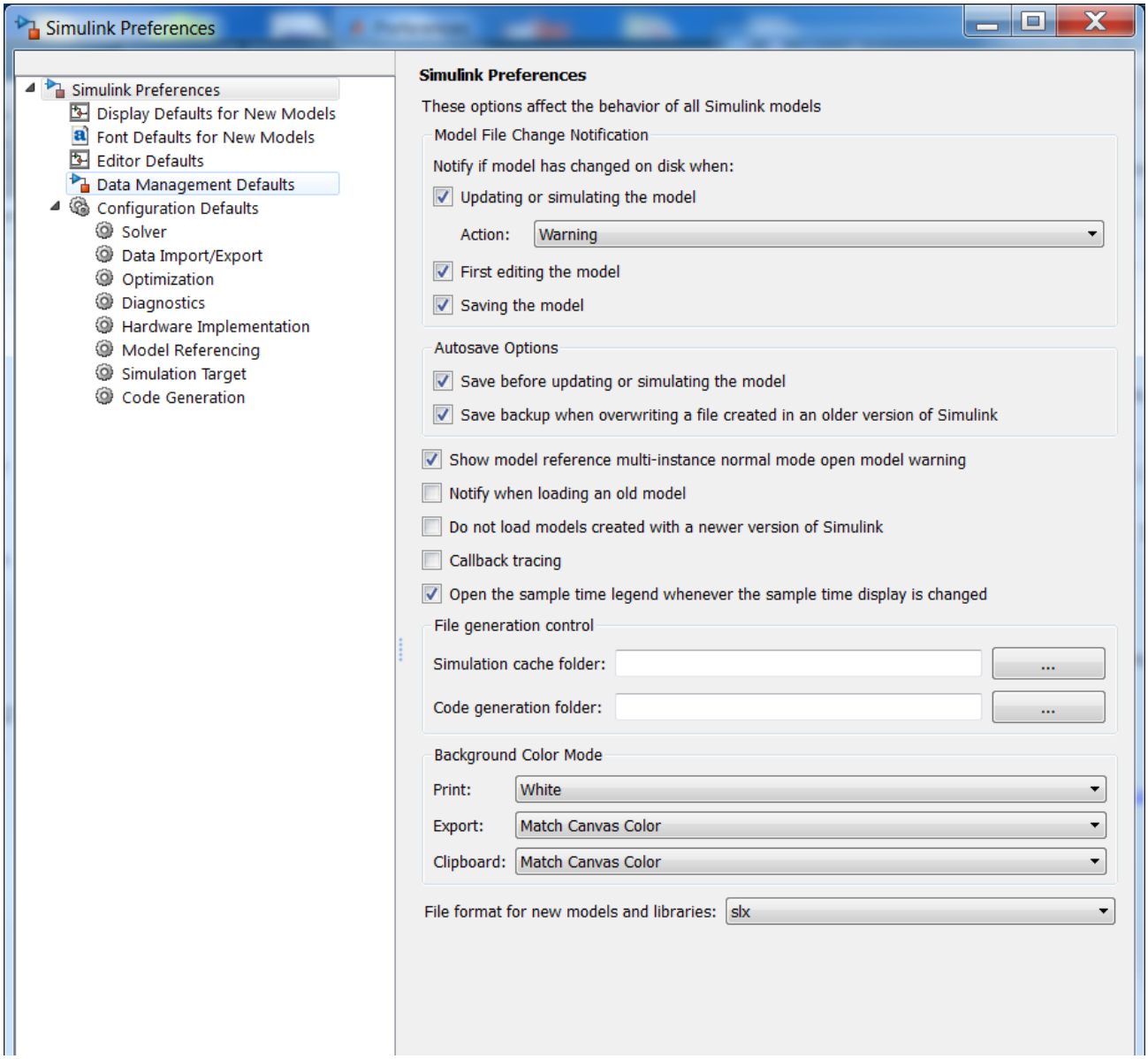


Рис. 3.8. Вікно настроювання середовища *MATLAB* і *Simulink*

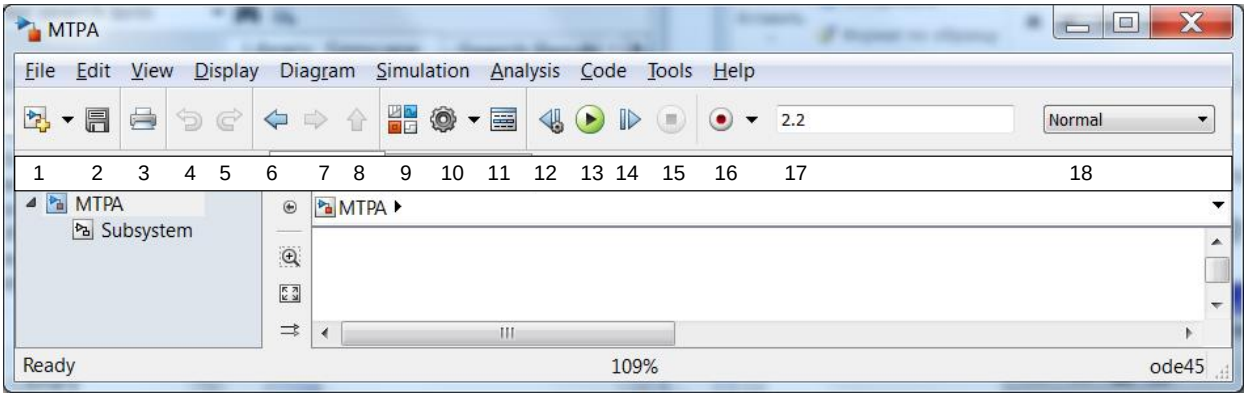


Рис. 3.9. Меню та кнопки вікна *Simulink*-моделі

Установки режиму моделювання 18 не слід змінювати починаючим користувачам.

У полі заголовка вікна рис. 3.9 відображено шлях до моделі у файловій системі. Символ “*” у кінці шляху означає, що після редагування моделі не виконано її запис у файл.

Функція File вікна моделі містить такі операції роботи з файлами: *New* ▸ – створити вікно для введення нової моделі (*Model ^N*) або бібліотеки (*Library*); *Open... (^O)* – відкрити модель; *Close (^W)* – закрити модель; *Save (^S)* – зберегти модель у колишньому файлі; *Save As..* – зберегти модель у новому файлі; *Model Properties* ▸ – властивості моделі; *Preferences…* – переваги (настроювання середовища); *Print...(^P)* – вивід на принтер; *Exit MATLAB* – вихід із системи *Matlab*.

Вікно Model Properties, яке відкривається при виборі відповідної операції, містить вкладки *Summary, Callbacks, Summary* і *History*. Вкладка *Summary* дозволяє увести автора (*Creator*), дату створення (*Created:*) і опис моделі (*Model description:*).

Вкладка Callbacks дозволяє визначити команди, що будуть автоматично виконані перед завантаженням моделі (*Model pre-load function:*), при її ініціалізації (*Model initialization function:*), перед стартом моделювання (*Simulation start function:*), після його закінчення (*Simulation stop function:*) і перед збереженням моделі у файлі (*Model pre-save function:*).

Вкладка ***History*** містить дані про модифікацію моделі.

Вікно друку моделі Print Model, наведене на рис. 3.10, дає можливість, крім звичайних для *Windows* параметрів, установлювати глибину роздруківки блоків моделі при наявності в ній підсистем: *Current system, Current system and above, Current system and bellow, All systems*. Зміст цих установок на-

очно продемонстровано у графічній формі.

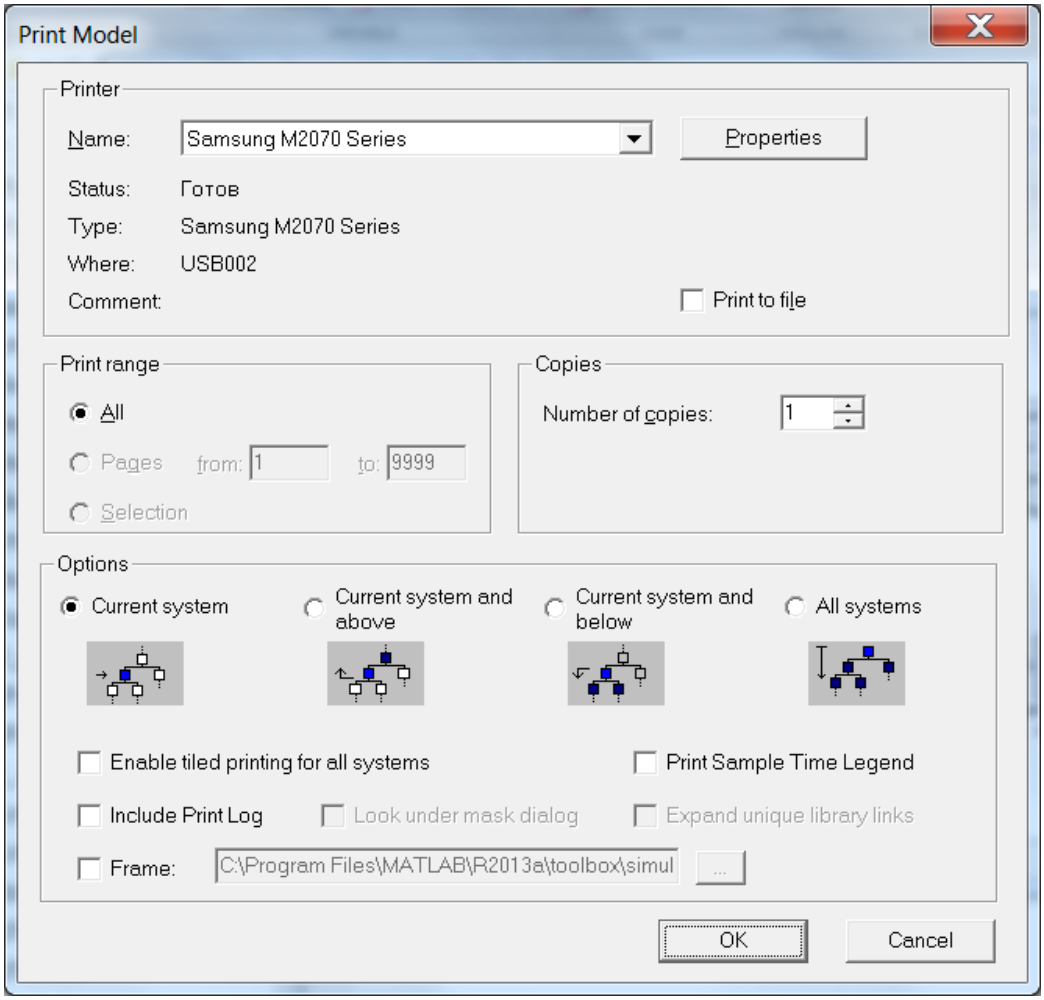


Рис. 3.10. Вікно друку *Simulink*-моделі

Функція Edit вікна моделі містить як стандартні *Windows*-операції редагування, так і деякі специфічні операції: *Undo (^Z)* – відмінити останню зміну; *Redo (^Y)* – відновити останню зміну; *Cut (^X)* – переміщення обраних об’єктів у буфер; *Copy (^C)* – копіювання обраних об’єктів у буфер; *Paste (^V)* – копіювання змісту буфера в обране місце графічного вікна; *Delete (Del)* – вилучення обраних об’єктів; *Select All (^A)* – вибір всіх об’єктів в активному вікні; *Copy Current View to Clipboard* – копіювання моделі в буфер; *Find…(^F)* – пошук блоків, сигналів, коментарів та інших об’єктів моделі.

Функція View вікна моделі, крім прапорців *Toolbar* і *Statusbar* та команд керування масштабом зображення *Zoom in*, *Zoom out*, *Fit system to view* і *Normal (100%)*, містить наступні операції: *Go to parent* – перехід до батьків-

ської моделі; *Model browser options* ▸ – властивості «навігатора моделі»; *Block data tips options* ▸ – властивості типів даних блоків; *Show Library Browser* – перегляд «навігатора бібліотек».

Найбільш споживаною операцією функції ***Display*** є *Signals & Ports*, за допомогою якої виконуються операції *Signal Dimensions, Wide Nonscalar Lines, Port Data Types* та деякі інші.

Важливими операціями функції ***Diagram*** є *Format* та *Rotate & Flip*. До команд форматування моделі належать такі команди [3]:

- *Font Style* – установлення типу та розміру шрифту для текстових написів (для цього треба попередньо виділити блок, а не його ім'я);
- *Foreground Colour* ▸ – установлення кольору піктограми та імені блока;
- *Background Colour* ▸ – установлення кольору фону блока;
- *Block shadow* – показати/сховати тінь від блока;
- *Show Block Name* – сховати/показати ім'я блока;
- *Canvas Colour* ▸ – колір екрана.

До команд повороту елементів моделі належать команди

- *Clockwise (^R)* – повернути блок на 90° за годинниковою стрілкою;
- *Counterblockwise (^↑R)* – повернути блок на 90° проти годинникової стрілки;
- *Flip Block (^I)* – повернути блок праворуч-ліворуч або зверху вниз;
- *Flip Block Name* – змінити положення імені блока (для горизонтально розташованих блоків воно може міститися вгорі або внизу, а для розташованих вертикально – праворуч або ліворуч).

Якщо фоновий колір блока не збігається з кольором екрана, то вихідні лінії зв'язку цього блока набувають кольору його фону (*Background*), інакше вони набувають кольору піктограми й імені блока (*Foreground*). Вищесказане відноситься і до контурних ліній блока.

Функція ***Simulation*** керує наступними режимами:

- *Update diagram…(^D)* – оновити блок-діаграму;
- *Model Configuration Parameters...(^E)* – установка і редагування параметрів моделювання;
- *Start (^T)* – старт моделювання.

Вікно параметрів моделювання має декілька вкладок (панелей), з яких найчастіше використовують вкладки *Solver* та *Data Import/Export.*

Панель *Solver* подана на рис. 3.11 та рис. 3.12 відповідно. Вони визначають метод і параметри симуляції (розв'язання рівнянь, що описують математичну модель).

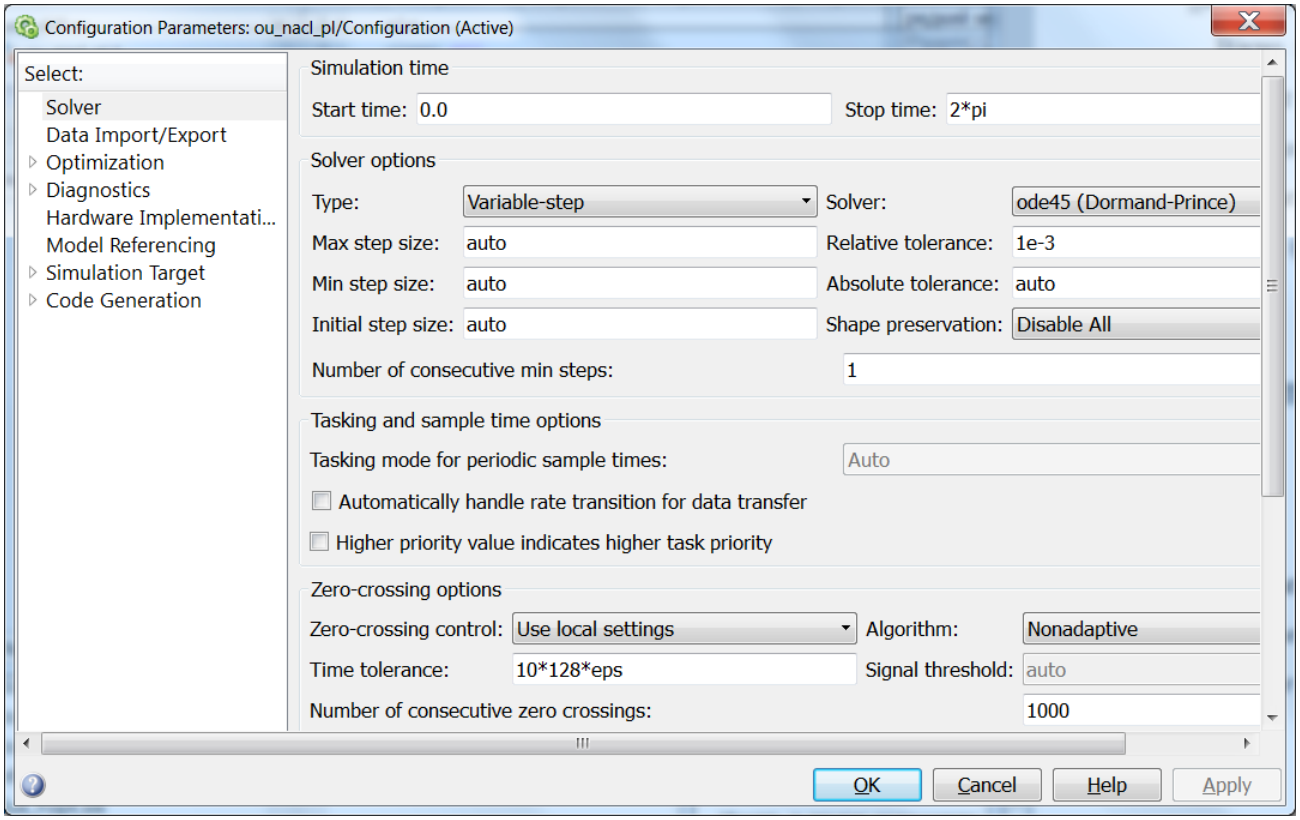


Рис. 3.11. Вкладка *Solver* вікна параметрів симуляції

Панель Solver передбачає введення таких основних параметрів:

- *Start time* – час початку моделювання;
- *Stop time* – час закінчення моделювання;
- *Type* – тип розв'язання: *Variable-Step* (зі змінним кроком), *Fixed-Step*

(з фіксованим кроком);

- *Solver* – метод розрахунку перехідних процесів (для неперервних систем метод розв'язання звичайних ДР з початковими умовами).

Серед ***методів розв'язання ДР з фіксованим кроком (Fixed Step)*** в *Simulink* пропонуються ***методи Рунге-Кутта першого-п'ятого порядків***, а саме:

- *ode*5 – метод Рунге-Кутта 5-го порядку (*Dormand-Prince formula*);
- *ode*4 – метод Рунге-Кутта 4-го порядку (*fourth-order Runge-Kutta formula*);
- *ode*3 – метод Рунге-Кутта 3-го порядку (*Bogacki-Shampine formula*);
- *ode*2 – модифікований метод Ейлера (*improved Euler's formula / Heun's method*);
- *ode*1 – метод Ейлера (*Euler's method*).

Серед ***методів розв'язання ДР зі змінним кроком (Variable Step)*** в *Simulink* пропонуються комбіновані методи Рунге-Кутта (4-5)-го (*ode*45) і (2-3)-го (*ode*23) порядків та деякі методи, більш пристосовані для розв'язання жорстких ДР:

- *ode*45 – комбінований метод Рунге-Кутта (4-5)-го порядку з автоматичним вибором кроку, призначений для розв'язання нежорстких ДР;
- *ode*23 – комбінований метод Рунге-Кутта (2-3)-го порядку з автоматичним вибором кроку, призначений для розв'язання нежорстких та слабко жорстких ДР;
- *ode*113 – багатокроковий метод Адамса-Моултона-Бешфорта (метод прогнозу та корекцій змінного порядку), призначений для розв'язання нежорстких ДР; може бути більш ефективним, ніж *ode*45, при високій точності;
- *ode*15*s* – модифікація методу Гіра: багатокроковий метод (1-5)-го порядку (за замовчанням 5-го), спрямований на розв'язання жорстких ДР;

- *ode*23*s* – однокроковий метод, що базується на модифікованій формулі Розенброка 2-го порядку;
- *ode*23*t* – метод трапецій з використанням „вільного” інтерполянта, рекомендований для розв’язання помірно жорстких ДР;
- *ode*23*tb* – комбінований метод, що використовує на першому етапі формулу трапецій, а на другому – зворотну диференційну формулу другого порядку; як і метод *ode*23*s*, може бути при невисокій точності більш ефективним, ніж *ode*15*s*, для розв’язання жорстких ДР.

Якщо модель має тільки дискретні змінні стану, або зовсім не має динамічних ланок, то для моделювання використовують метод *discrete* з фіксованим або змінним кроком. Змінний крок обирають, коли модель має у своєму складі дискретні динамічні ланки з різними періодами переривання, або коли треба фіксувати моменти часу перетину деякими сигналами нульового рівня.

При використанні методу *ode*15*s* рекомендовано для підвищення стабільності розв’язання ДР знизити максимальний порядок формули *NDF*, що задається параметром *Maximum order*, з 5 (за замовчанням) до 3.

Для методів з фіксованим кроком треба задати значення цього параметру в полі *Fixed step size*.

Для методів зі змінним кроком задаються такі параметри:

- *Max step size* – максимальний крок; в режимі *auto* він розраховується за формулою $h_{\max} = (t_{stop} - t_{start})/50$;
- *Min step size* – мінімальний крок;
- *Initial step size* – початковий крок;
- *Relative tolerance* – відносна точність (за замовченням 10^{-3});
- *Absolute tolerance* – абсолютна точність (за замовченням 10^{-6});
- *Refine factor* – коефіцієнт збільшення кількості точок моделювання (ціле число); для методу *ode*45 рекомендовано значення 4 (максимальне зна-

чення – 10), для інших методів – 1.

Останній параметр знаходиться у вкладці *Data Import/Export* (див. рис. 3.12).

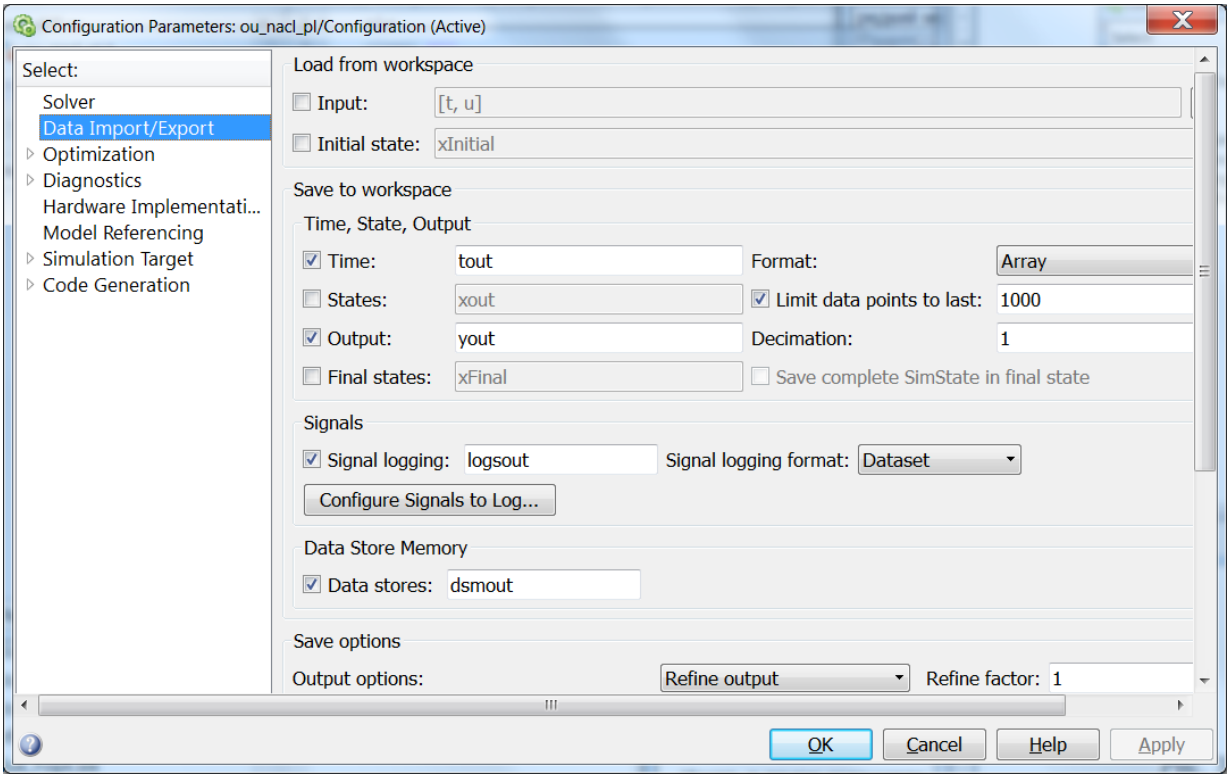


Рис. 3.12. Вкладка *Data Import/Export* вікна параметрів симуляції

На панелі *Data Import/Export* у першу чергу зверніть увагу на доцільність наявності «прапорців» перед параметрами *Time* і *Output* та усунення «прапорця» перед параметром *Limit Data Point to Last*. Сенс цих та деяких інших установок буде пояснений пізніше.

3.2.3. Команди, що виконуються з вікон *Simulink*-бібліотек

Меню вікон ***Simulink***-бібліотек має багато спільного з меню вікон *Simulink*-моделей.

Між ними існує така різниця:

- в меню бібліотек відсутні функції *Simulation* та *Tools*;
- у заблокованому стані у вікні бібліотек не можливо виконання операцій редагування, тобто усіх операцій функції *Format*, операцій *Clear*, *Cut*, *Paste*, *Create Subsystem*, *Mask Subsystem*, *Mask Parameters*, *Edit mask*, *Link*

options, *Update diagram*, *Undo*, *Redo* функції *Edit* та операцій редагування, які здійснюються за допомогою “миші” (пересування блоків, зміна їх розміру, назви, тощо);

- для розблокавання бібліотек функція *Edit* їхніх вікон має операцію *Unlock Library*.

Стан *Unlock Library* діє з моменту виконання відповідної операції до закриття вікна бібліотеки і поширюється на вкладені бібліотеки. При наступному відкритті бібліотеки вона буде знову заблокованою.

Користувач може редагувати існуючі бібліотеки (у розблокованому стані) та створювати власні [3].

Для створення нової бібліотеки треба відкрити її вікно через меню будь-якого вікна *Simulink*-моделі, *Simulink*-бібліотеки або *Simulink Library Browser: File* → *New* → *Library*, занести в нього необхідні блоки і закрити зі зберіганням у файлі.

Файли бібліотек, як і файли моделей, мають поширення *mdl* або (починаючи з версії 12a) *slx*.

3.3. Типи *Simulink*-блоків

Simulink-блоки можуть створюватися різними способами.

Основу стандартних *Simulink*-бібліотек складають ***вбудовані блоки (built-on)***, які не доступні для перегляду користувачем у вигляді текстового файлу або структурної схеми.

Деякі стандартні блоки створені за допомогою вже існуючих блоків і являють собою або модифікацію деякого блока з іншими параметрами і зміненою піктограмою, або ***комбінацію декількох блоків, об'єднаних в одну підсистему***. Решта ***блоків створені програмно за допомогою апарата MATLAB-функцій або S-функцій***. Після імені такого блока у вікні введення його параметрів стоять позначки (*mask*) і (*link*), що означають, що блок замаскований і зв’язаний з однією з бібліотек. У такий же спосіб може створюва-

ти нові блоки і користувач. ***Заглянувши під маску виділеного блока (Diagram → Mask → Look undo Mask – ^U)***, можна побачити в окремому вікні, з чого вони складаються (цю ж операцію можна виконати через контекстуальне меню, що відкривається щигликом правої кнопки миші по піктограмі блока). Для перегляду і редагування параметрів маскування (*Diagram → Mask → Edit mask… – ^M*) необхідно попередньо тимчасово вивести з ладу зв'язок з бібліотекою. Після перегляду і внесення змін в установки цього вікна зв'язок з бібліотекою можна або відновити, або перервати.

S-функції можуть бути написані за особливими правилами на мовах *MATLAB (*.m)*, або *C++ (*.cpp)*, або *C (*.c)*, або *Ada (*.adb)*, або *Fortran (*.f)*, а потім конвертовані в блоки *Simulink*.

3.4. Основні прийоми створення та редагування моделей

Перед початком формування нової моделі необхідно відкрити для неї нове вікно (*Simulink → File → New → Model (^N)* або *MATLAB → File → New → Model*). Спочатку воно має заголовок «*Untitled*». При запису в файл (*File → Save / Save as…*) у якості заголовку вікна буде фігурувати призначене користувачем ім'я файлу, яке за замовченням отримає розширення *slx*. При зберіганні моделі командою *Save as* файл можна записати і з розширенням *mdl*. Це зробить його доступним для виконанні у попередніх версіях *MATLAB-Simulink*.

Вже існуючу модель можна завантажити з меню *Simulink* або *MATLAB* (*File → Open*) та із командного рядка *MATLAB* введенням в ній імені файлу, в якому збережена модель, без поширення.

В основу створення та редагування моделей покладено, як і в більшості графічних *Windows*-додатків, ***принцип drag-and-drop (перетягни та покинь)***.

Копіювання блоків із бібліотек або з будь-якої вже існуючої моделі у вікно створеного файлу після їх відкриття виконується мишею за допомогою

операції *drag* (тягнути при натиснутій лівій клавіші миші). Аналогічно здійснюється переміщення блоків всередині вікна. Копіювання блоків всередині вікна виконується операцією *drag right* (тягнути при натиснутій правій клавіші миші). При цьому до імені нового блока додається цифра, що відображає порядковий номер копіювання, чим забезпечується унікальність імені кожного блока однієї моделі.

Імена блоків можна змінювати безпосереднім редагуванням. При цьому не можна їх дублювати або залишати блок без імені (ім'я – пустий рядок), але можна сховати ім'я (*Diagram → Format → Hide Block Name*). Зворотна операція здійснюється командою *Format → Show Block Name*. Для завершення редагування імені треба зробити щиглик мишею зовні поля введення тексту. Після цього ім'я аналізується системою та чи приймається, чи відкидається нею з виведенням повідомлення. Для зміни шрифту імені (*Format → Font…*) необхідно попередньо виділити сам блок, а не його ім'я.

В довільній точці активного *Simulink*-вікна, що виділяється щигликом лівої клавіші миші, можна вставити будь-які ***коментарі (Annotations)***. Введення та редагування коментарю виконуються точно так, як відповідні операції з ім'ям блока. В результаті утворюється новий специфічний блок *Note*, який відрізняється від інших блоків тим, що він не має ні піктограми, ні портів, ні вікна введення параметрів, а складається з одного імені, яке є текстом анотації. Рештою цей блок схожий на інші: його можна копіювати, пересувати, знищувати, виділяти і т. п.

Іменами можна відмічати і лінії зв'язку. Для цього необхідно клацнути двічі лівою кнопкою миші по обраній з'єднувальній лінії та ввести текст в утворене поле. Отриманий у такий спосіб надпис буде, на відміну від блока *Note*, прив'язаний до лінії зв'язку.

При пересуванні блока, до якого вже приєднані лінії зв'язку, останні витягуються або скорочуються; кінці ліній зв'язку, не приєднані до блока,

що пересовується, не змінюють своє положення. Перемістити блок в межах одного вікна без ліній зв'язку (висунути блок з моделі) можна операцією <*Shift*>+*drag*.

Для виконання деякої дії з будь-яким об'єктом моделі (блок, лінія зв'язку, сукупність елементів) його треба спочатку відмітити. Поодинокий об'єкт виділяється щигликом лівої клавіші миші (*click*). Фрагмент моделі можна виділити такими способами:

- помітити вибірково блоки або зв'язки, не відпускаючи клавішу *Shift* (<*Shift*> + *click*);
- помітити поспіль, тобто заключити за допомогою мишки у прямокутник;
- помітити все (*Edit* → *Select All* – ^*A*).

Про те, що виділення відбулося, свідчать маленькі незафарбовані квадратики, розташовані в кутах обраних блоків та поблизу кінців обраних ліній зв'язку.

З поміченим блоком можна виконувати такі операції:

- зміна розміру – *drag* за кут;
- знищення – <*Delete*>;
- усі доступні операції меню *Diagram: Format* (зміни кольорів та шрифтів, приховування імен), *Rotate & Flip* (повороти) та *Mask* (маскування).

Подвійний щиглик по піктограмі блока розкриває текстове вікно для введення його параметрів.

Параметри блока можуть бути подані як в чисельному вигляді, так і у вигляді імен змінних або математичних виразів. В останньому випадку до початку моделювання змінним мають бути присвоєні значення. Це можна зробити з командного рядка *MATLAB* або виконанням командного файлу.

Лінії зв'язку поєднують між собою вхідні та вихідні порти блоків. Стрілка на лінії зв'язку показує напрямок потоку даних. Утворюються вони

такими способами:

- довільно кусочно-неперервно (*drag* від порту з перериванням цієї операції в бажаних точках зламу);
- з автоматичним розташуванням точок зламу.

Автоматичне розташування точок зламу забезпечується при виконанні операції *drag* від порту до порту без переривання, а також при швидкому з'єднанні блоків: помічається початковий блок або блоки, натискається клавіша *Ctrl* і помічається кінцевий блок.

Для ***розгалуження ліній зв'язку*** використовують операцію *drag right* або ^ *drag left* або *drag* будь-якою клавішею але у зворотному напрямку, тобто від вхідного порту до точки розгалуження.

При проведенні ліній зв'язку не варто намагатися влучити точно в порт; лінія приєднається до порту, якщо відпустити клавішу миші при знаходженні графічного курсора всередині блока або поблизу порту (на відстані менше 5 пікселів).

З поміченими лініями зв'язку можна виконувати такі операції:

- паралельне пересування – *drag*, ухопившись за середину сегменту лінії зв'язку;
- зміна кута між сегментами лінії зв'язку – *drag*, ухопившись за вузол;
- додатковий злам сегмента лінії зв'язку – <*Shift*>+ *drag*, ухопившись за бажану точку зламу;
- ділення зв'язку – *drag*, ухопившись за першу ліву мітку зв'язку або *drag right* від будь-якої точки зв'язку;
- знищення – <*Delete*>.

З поміченими фрагментами моделі або з усією моделлю можна виконувати всі ті операції, що і з окремими блоками та/або зв'язками.

Для того, щоб зробити розташування об'єктів моделі більш зручним, вікна *Simulink* мають невидиму сітку 5x5 пікселів, до вузлів або до ліній якої

прив’язуються всі об’єкти. Дискретне пересування виділених об’єктів можна виконати клавішами <→>, <←>, <↑>, <↓>.

При редагуванні моделі ***графічний курсор змінює свою форму***, сигналізуючи користувачу про ту дію, до виконання якої підготовлена система:

 – готовий для наступних дій;

 – готовий для нанесення обмежувального прямокутника (виділення групи поруч розташованих об’єктів);

 – готовий для пересування блока, сегмента лінії, або виділеного фрагмента;

 – готовий до проведення лінії зв’язку;

 – готовий для перенесення або для створення нової точки зламу лінії;

 – готовий до зміни розміру блока.

Контрольні питання та завдання

(відповіді бажано доповнити ілюстраціями)

1. Як можна запустити *Simulink*?
2. Як створити вікно для введення нової моделі?
3. Перелічите і охарактеризуйте основні *Simulink*-бібліотеки.
4. Як подивитися демонстрації *Simulink*?
5. Яке розширення мають файли моделей *Simulink*?
6. Де рекомендовано зберігати файли користувача? Що треба зробити після створення нової папки?
7. Створіть папку для лабораторних робіт та забезпечте доступ до неї.
8. Які параметри середовища *MATLAB* і *Simulink* можна змінювати командою *Preferences*?
9. Як помітити блок та групу блоків?
10. Як можна змінювати розмір шрифту, розмір блока, розмір моделі?

11. Як відкрити вікно *Model Properties*? Яке призначення вкладки *Callbaccs* цього вікна?
12. Як скопіювати модель у буфер?
13. Як змінити ім’я блока в моделі, сховати/показати ім’я, змінити позицію імені, прив’язати ім’я змінної до лінії зв’язку? Наведіть приклади.
14. Як повернути блок на 90 градусів? Як відобразити блок (зліва направо, згори донизу або навпаки? Як розмножити блок?
15. Як змінити кольори елементів моделі та її фону? Наведіть приклади.
16. Як розгалузити лінію зв’язку, як додати до неї нову точку зламу, як змінити кут у точці зламу? Наведіть приклади.
17. Які методи розв’язання диференційних рівнянь пропонує *Simulink*? З яких міркувань обирають метод і параметри чисельного інтегрування при моделюванні?
18. Як змінити параметри моделювання? Як покращити якість візуалізації перехідних процесів?

4. ОСНОВНІ ЗАСОБИ РЕЄСТРАЦІЇ ТА ВІЗУАЛІЗАЦІЇ СИГНАЛІВ У СЕРЕДОВИЩІ *Simulink*

Для реєстрації та візуалізації сигналів у *Simulink* існує бібліотека *Sinks*, блоки якої подані на рис. 4.1.

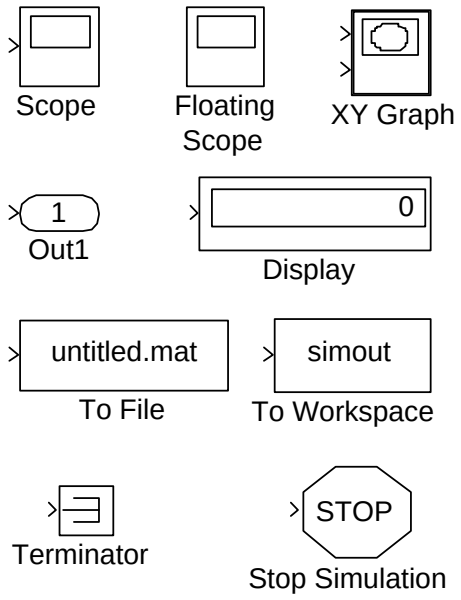


Рис. 4.1. Бібліотека вихідних блоків *Sinks*

Більшість з них призначені для запам'ятання результатів моделювання *(To File, To Workspace, Out)*, а також для їх графічного *(Scope, Floating Scope, XY Graph)* або чисельного *(Display)* відображення. Усі вони мають тільки входи і не мають виходів.

Блок ***Terminator* (*заглушка)*** приєднується до виходів блоків, не зв’язаних з іншими блоками для запобігання висновку попереджувального повідомлення про наявність у моделі не приєднаних виходів.

Розглянемо параметри, які є спільними для декількох блоків бібліотеки *Sinks*.

Параметр Decimation (проріджування) у блоках *Scope*, *To Workspace*, *Display* визначає дискретність запам'ятовування або відображення інформації і може приймати тільки додатні цілочислові значення *d*, що вказують через скількох кроків чисельного інтегрування варто фіксувати результати моделювання. При *d*=1 (значення за замовчуванням) блоком запам'ятовується інформація, отримана на кожнім кроці ЧІ, а при *d*=5 – тільки інформація, отримана на кожному п'ятому кроці. При інтегруванні з постійним кроком параметр *d* – це відношення кроку фіксації до кроку інтегрування: $h_{out} = dh_{int}$.

Якщо в процесі моделювання для рішення диференціальних рівнянь обрано метод ЧІ зі змінним кроком, а крок фіксації повинний бути постійним, то для дискретизації виведення замість параметра *Decimation* варто ви-

користовувати ***параметр Sample Time (період дискретності)***, обираючи його значення рівним бажаному кроку реєстрації: $T_s = h_{out}$. При $T_s = -1$ блок висновку успадковує період дискретності від блока, вихідні сигнали якого він реєструє.

Параметр Limit data point to last у блоках *Scope, To Workspace, Out* визначає максимальну кількість точок для збереження сигналу (відлік ведеться від останньої розрахованої точки). Для того, щоб не втратити інформацію, значення параметра *Limit data point to last* можна установити рівним ∞ (константа *Inf*) або прибрати прапорець, що робить цю опцію активною.

Параметр Save format у блоках *Scope, To Workspace, Out* визначає формат запам'ятовування даних.

Інформація може бути збережена в наступних форматах: *Structure with time* (*структура з часом моделювання*), *Structure* (*структура без часу моделювання*) і *Array* (*масив*).

Найпростішим типом даних є ***Array***. У цьому випадку кожен сигнал записується в окремий стовпець матриці. Один рядок матриці містить стан усіх сигналів у конкретний момент часу. Кількість рядків при *d*=1 дорівнює числу кроків моделювання. Якщо дані збережені в матриці *y*, то виділити з неї *j*-ий сигнал можна операцією y(:,j), а всі сигнали в *i*-й зареєстрований момент часу – операцією y(i,:).

Якщо дані збережені у змінній y, що має формат *Structure with time*, то вектор-стовпець часу визначається звертанням до поля *time* змінної y (y.time), звертання y.blockName формує шлях до блока (ім'я-моделі / ім'я підсистеми / ім'я-блока), звертання y.signals.dimensions – кількість вхідних сигналів, звертання y.signals.label – мітку лінії зв’язку, а звертання y.signals.values – чисельні значення матриці вхідних сигналів, організованої так само, як при використанні формату *Array*. Структура без часу (*Structure*),

відрізняється від структури з часом тим, що її поле *time* має значення порожньої матриці.

4.1. Блоки візуалізації

Блок ***Scope*** **(*осцилограф*)** у процесі моделювання відображає вихідні сигнали приєднаних до нього блоків. Для того, щоб побачити графіки перехідних процесів, необхідно відкрити його вікно (див. рис. 4.2), що поряд з полем виведення графіків містить такі «кнопки» (рис. 4.3):

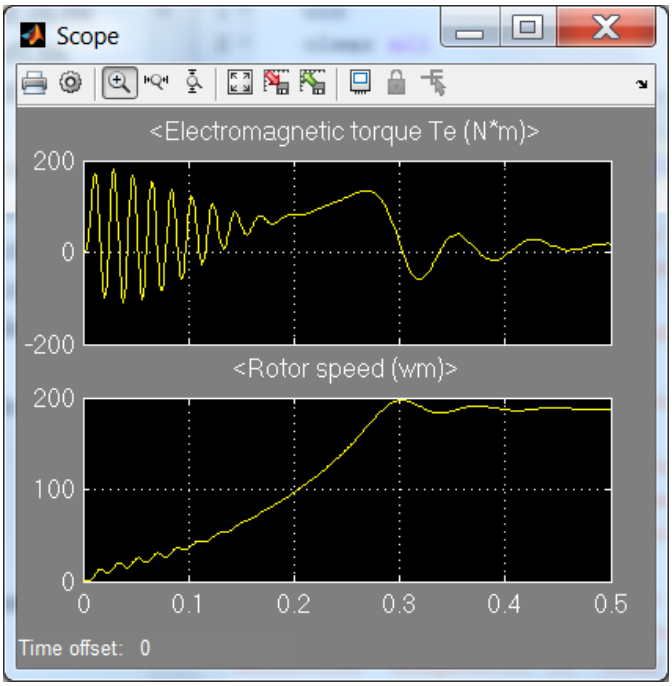


Рис. 4.2. Вікно блока *Scope*

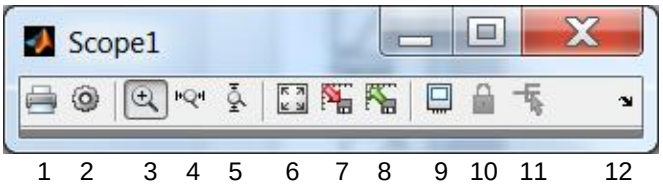


Рис. 4.3. Кнопки вікна блока *Scope*

1 – *Print* (друкування);

2 – *Parameters* (настроювання параметрів);

3 – *Zoom* (рівномірна зміна масштабу по обох осях);

4 – *Zoom X-axis* (зміна масштабу по осі *X*);

5 – *Zoom Y-axis* (зміна масштабу по осі *Y*);

6 – *Autoscale* (автоматичне масштабування);

7 – *Save current axes settings* (запам'ятовування системи координат);

8 – *Restore saved axes settings* (відновлення системи координат, наприклад, після зміни масштабу однією з клавіш *Zoom*);

9 – *Floating Scope* (плаваючий осцилограф);

10 – *Lock/Unlock axes selection* (заблокувати/розблокувати вибір ліній

зв’язку для плаваючого осцилографа);

11 – *Signal Selection* (вибір сигналів для плаваючого осцилографа);

12 – *Dock Scope* (осцилограф виводить графіки без кнопкової панелі).

Після закінчення процесу моделювання для того, щоб побачити графіки в нормальному масштабі, натискають кнопку *Autoscale*, а потім запам'ятовують діапазони систем координат кнопкою *Save current axes settings*.

Якщо результати автоматичного масштабування за якимись причинами не влаштовують користувача, то він може змінити межі координатних осей графіків вручну за допомогою кнопок 3-5 візуально або установкою границь системи координат у числовому вигляді.

Масштабування зображення в обраному виміру кнопками можна виконувати двома способами. При першому способі після активізації відповідної кнопки курсор миші підводять до бажаної ділянки графіка і клацають на ньому лівою клавішею. При кожному натисканні масштаб буде збільшуватися, що приведе до відображення у вікні всі меншого і меншого фрагмента графіка. При другому способі фрагмент графіка, що хочуть збільшити, виділяють за допомогою курсору. Повернення до результатів попереднього масштабування виконується командою контекстуального меню *Zoom out*, а повернення до вихідного масштабу – кнопками *Autoscale* або *Restore saved axes settings*.

Для установки меж графіка по осі ординат у чисельному вигляді необхідно викликати контекстуальне меню щигликом правої кнопки миші в площині системи координат, вибрати з нього функцію *Axes Properties* і у вікні, що відкрилося, установити *параметри* Y-min *і* Y-max. У цьому ж вікні можна установити заголовок системи координат (параметр *Title*), замінивши коментар *%<SignalLabel>* бажаним текстом або вставивши цей текст перед коментарем.

Діапазон часу, що є спільним для всіх систем координат, можна зміни-

ти у ***вікні Parameters***, що відкривається однойменною кнопкою й утримує 2 вкладки: *General* і *Data history.*

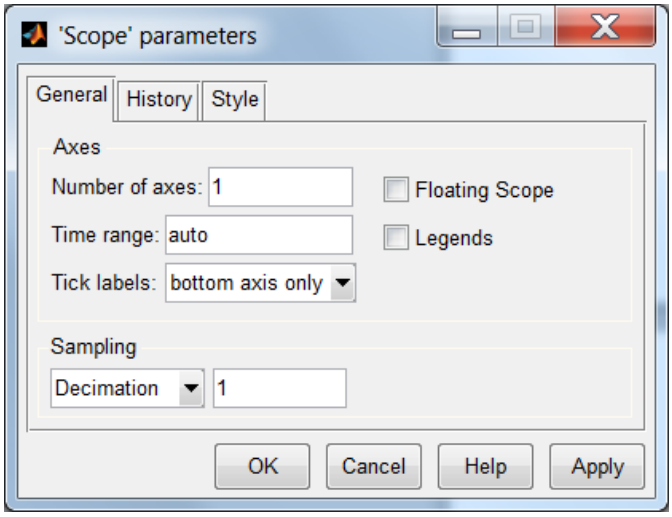


Рис. 4.4. Вкладка *General* вікна параметрів блока *Scope*

За допомогою ***вкладки General*** (див. рис. 4.4) установлюються наступні параметри:

- *Number of axes* – кількість систем координат, що визначає і кількість вхідних портів;
- *Time range* – діапазон часу;
- *Tick labels* – режим візуалізації чисельних значень міток координатної сітки, що може приймати значення *all* –значення часу виводяться в усіх підвікнах (системах координат), *bottom axis only* – значення часу виводяться тільки в самій нижній системі координат) і *none* (значення міток не виводяться, і графіки займають максимально можливу площу вікна);
- прапорці *Floating Scope* та *Legend*;
- параметри *Decimation* або *Sample Time*, що визначають дискретність відображення інформації (див. опис спільних параметрів).

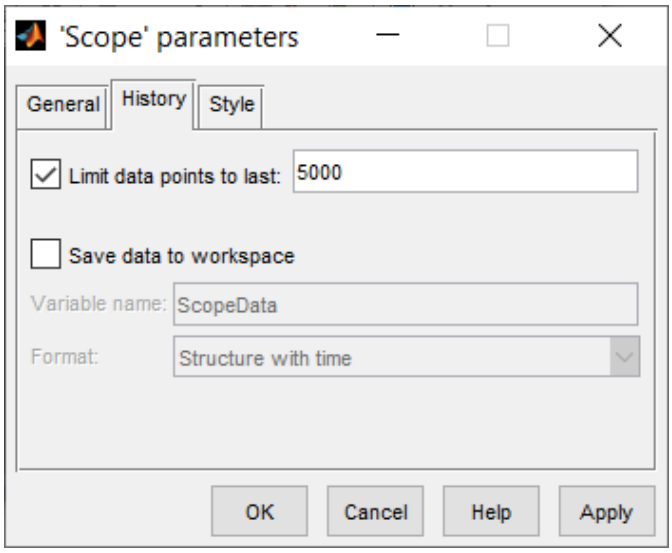


Рис. 4.5. Вкладка *History* вікна параметрів блока *Scope*

За допомогою ***вкладки History*** (див. рис. 4.5) назначаються параметри *Limit data point to last*, *Variable Name* і *Save format*. Два останніх параметри активізуються тільки при встановленому прапорці *Save data to workspace* (зберегти дані в опе-

ративній пам'яті). При установці цього прапорця блок *Scope* починає виконувати, крім власних функцій, функції ланок *To Workspace* або *Out*. За замовчуванням для запису даних пропонується змінна з ім’ям *ScopeData* у форматі *Structure with time*.

При виборі формату *Array* у перший стовпець матриці *ScopeData* записується час моделювання, а у інші – значення сигналів, приєднаних до осцилографа. Тому при побудові усіх графіків перехідних процесів в одному вікні в цьому випадку можна застосувати оператор

```
plot(ScopeData(:,1),ScopeData(:,2:end))
```

Формат *Array* не можна використовувати для збереження інформації багатоканальним *осцилографом*. Поле *signals* змінної *ScopeData* у цьому випадку є вектором структур, з кількістю елементів, рівним числу портів (каналів) *осцилографа*. На додаток до полів *dimensions*, *values* і *label* кожен елемент змінної *signals* має ще поля *title* і *plotStyle*, які містять інформацію про заголовки графіків і стилі їхнього зображення.

За допомогою ***вкладки Style*** (рис. 4.6) назначаються такі параметри:

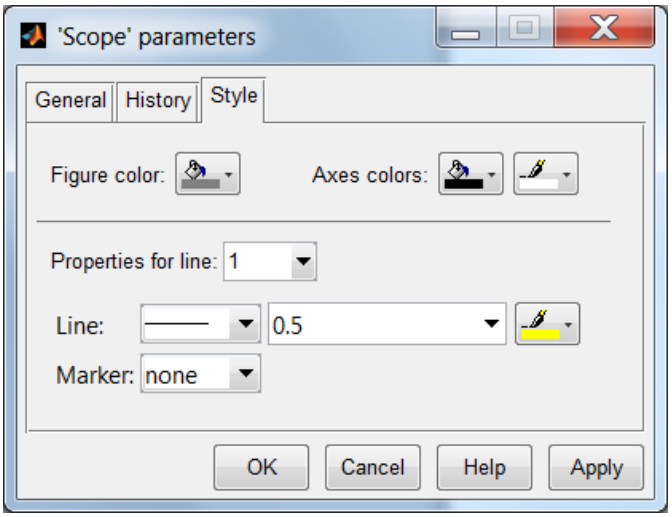


Рис. 4.6. Вкладка *Style* вікна параметрів блока *Scope*

- *Figure Colour* – колір графічної фігури;
- *Axes Colour* – колір системи координат (колір фону і колір ліній градуювання);
- *Line* – стиль зображення ліній, їх товщина та колір;
- *Marker* – стиль зображення маркерів точок графіка.

Останні три параметри встановлюються окремо для кожного графіка, вибір якого здійснюється параметром *Parameters for line:*.

Параметри *осцилографа* можна редагувати в процесі моделювання. Розмір і пропорції вікна *Scope* можна змінювати довільно.

Скалярний сигнал зображується завжди жовтим кольором, а для складових векторного сигналу використовуються за замовчанням повторювані циклічно 6 кольорів: *жовтий, малиновий, блакитний, червоний, зелений, синій*. Фон зображення за замовчанням *чорний*.

Приклади використання блока *Scope* для візуалізації двох синусоїдальних сигналів різної амплітуди і частоти показано на рис. 4.7.

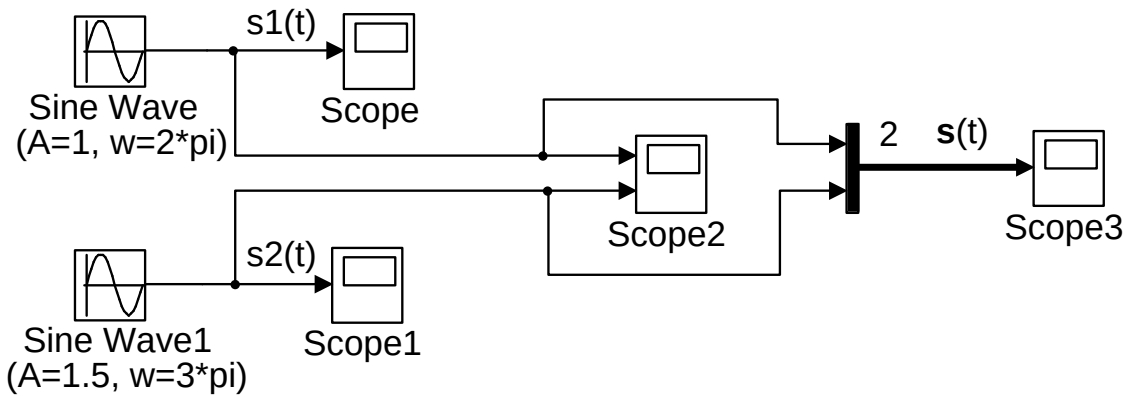


Рис. 4.7. Модель візуалізації двох сигналів блоками *Scope*

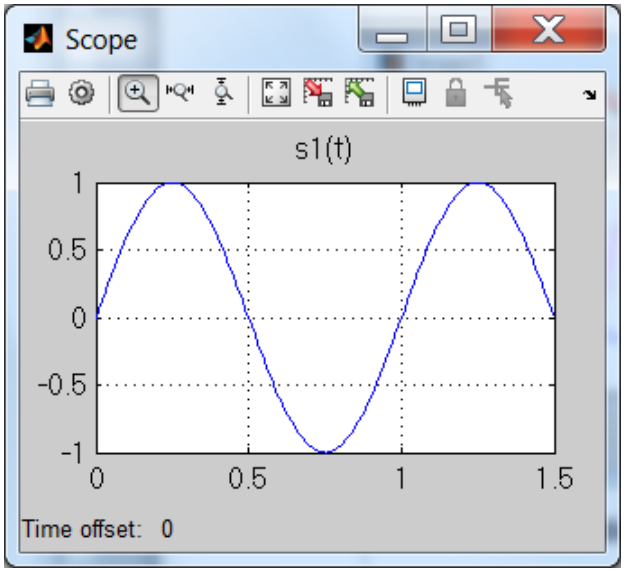
Вигляд відкритих після симуляції осцилографів показано на рис. 4.8.

Блоки *Scope* та *Scope1* відображують синусоїди у різних вікнах, а блоки *Scope1* та *Scope3* – в одному вікні. Різниця між останніми полягає в тому, що *Scope2* відкриває 2 системи координат, кожна у своєму підвікні, що досягається установленням у полі параметру *Number of axes* вкладки *General* вікна *Parameters* значення 2, а *Scope3* зображує обидва сигнали, попередньо об'єднані блоком ***Мих* (*мультиплексний канал*)** в один векторний сигнал, в одній системі координат.

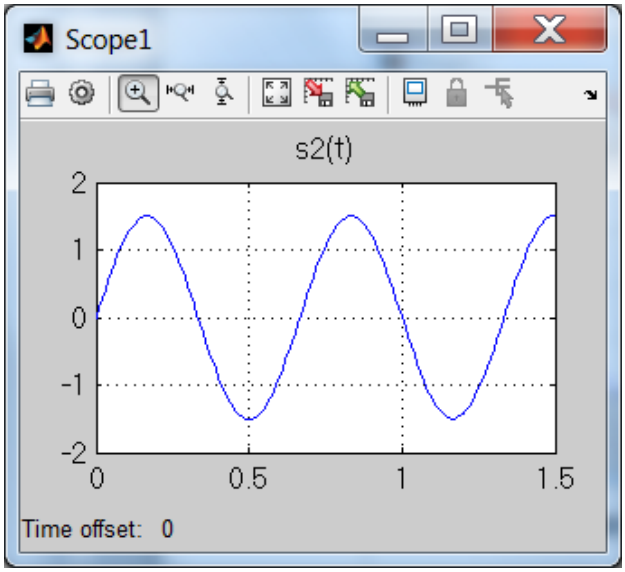
***Блок Floating Scope* (*плаваючий осцилограф*)** можна взяти з бібліотеки або переключити звичайний осцилограф *Scope* у режим, що плаває, відповідною кнопкою або установкою відповідного прапорця. *Floating Scope* не має вхідних портів. Підключення до нього реєстрованих сигналів можна ви-

конати у *вікні Signal Selector*, що відкривається кнопкою 11 (див. рис. 4.1). *Signal Selector*, завдяки наявності в ньому засобів навігації, дозволяє вибрати будь-які сигнали системи, включаючи внутрішні сигнали закритих підсистем. На відміну від блока *Scope*, ***плаваючий осцилограф не має буфера для збереження зареєстрованих даних. Тому для нього не можливі режими масштабування графіків (кнопки 3-6 не працюють)***.

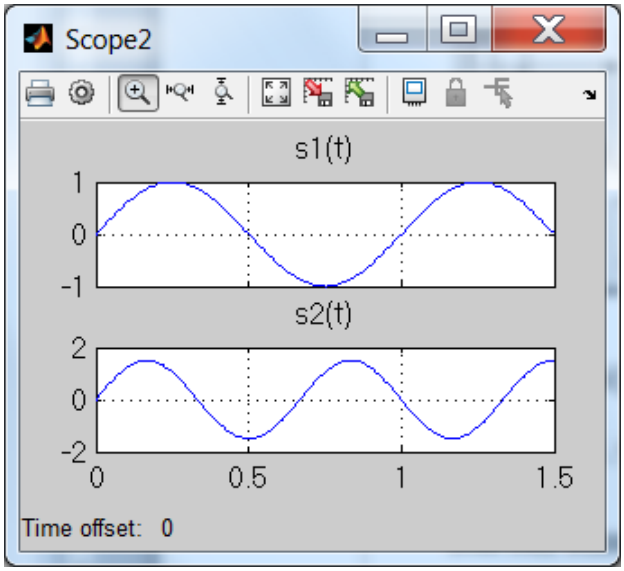
Єдиною перевагою плаваючого осцилографа є економія оперативної пам'яті.



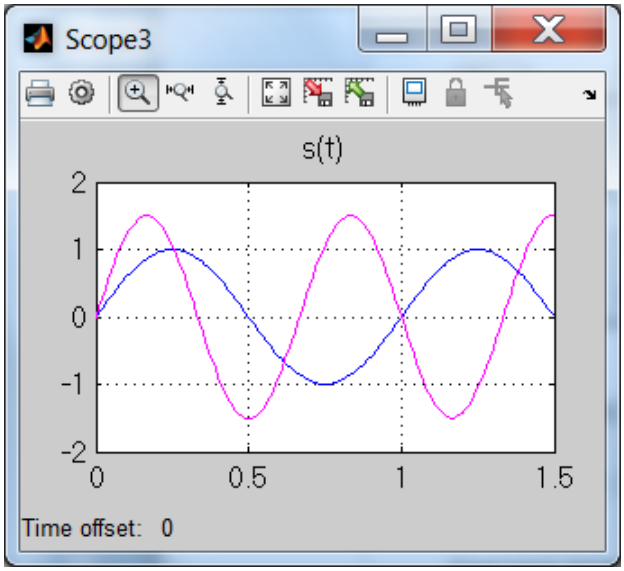
а)



б)



в)



г)

Рис. 4.8. Результати візуалізації синусоїд блоками
Scope (а), Scope1 (б), Scope2 (в), Scope3 (г)

***Блок XY Graph* (*XY-графобудувач*)** призначено для побудови фазових портретів, тобто для зображення одного зі збережених у пам'яті сигналів у функції іншого (а не у функції часу). Його вхідними параметрами є координати меж графіка : *x-min, x-max, y-min* і *y-max* (див. рис. 4.9).

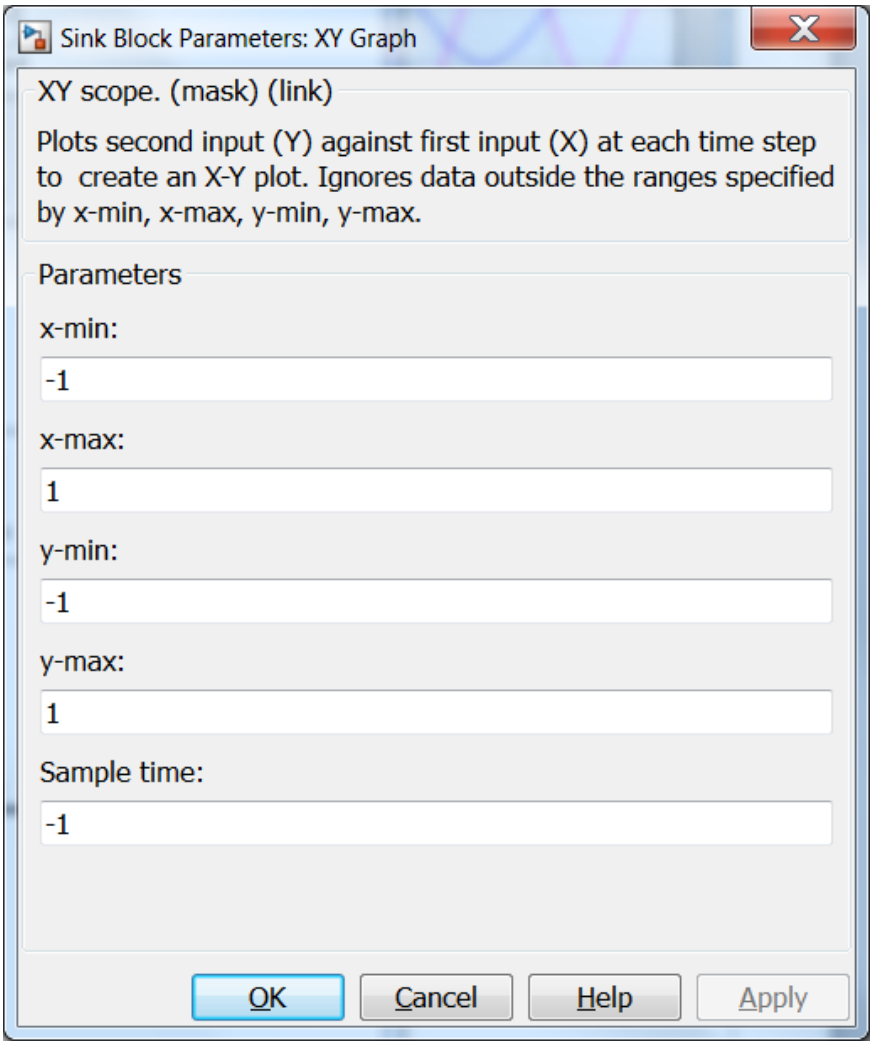


Рис. 4.9. Вікно параметрів блока *XY Graph*

Обидва входи блока є скалярними. При моделюванні *Simulink* автоматично відкриває графічне вікно *MATLAB* і відображає в ньому задану користувачем графічну залежність. Приклад використання показано на рис. 4.10.

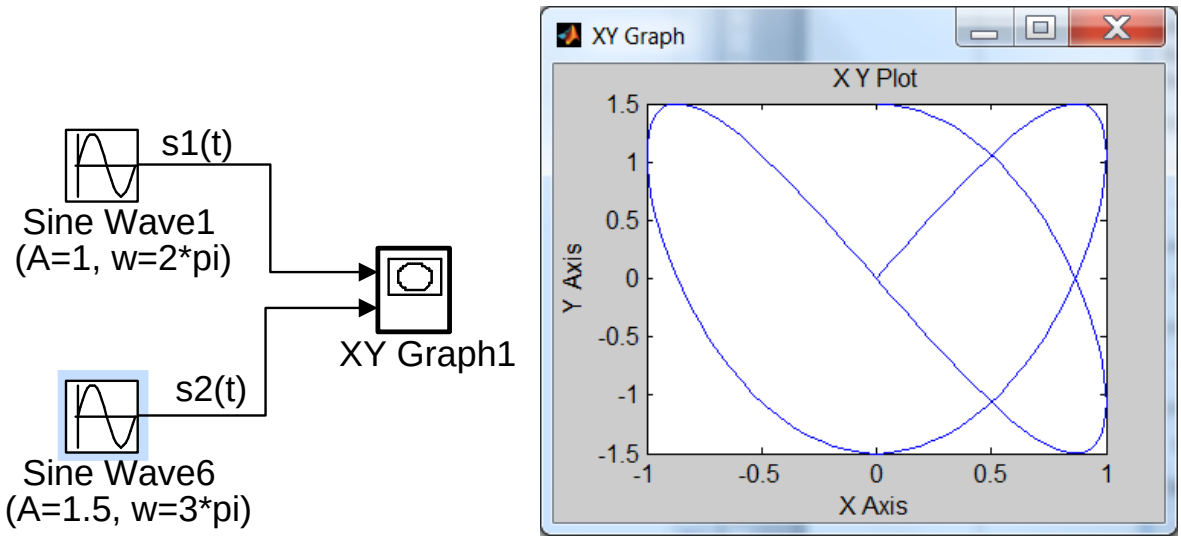


Рис. 4.10. Приклад використання блока *XY Graph*

До перегляду графіка треба встановити діапазони зміни його параметрів у вікні рис. 4.9.

Більш якісні графіки залежності одного сигналу від іншого можна побудувати оператором *plot* за результатами вимірювання сигналів [3].

Блок ***Display*** **(*дисплей*)** застосовується для виведення чисельних значень сигналів у заданому форматі, що обирається за допомогою параметра *Format* і може приймати наступні значення:

- *short* – 4 значущі цифри у форматі з фіксованою крапкою;
- *long* – 14 значущих цифр у форматі з фіксованою крапкою;
- *short-e* – 5 значущих цифр у мантисі чисел із плаваючою крапкою;
- *long-e* – 16 значущих цифр у мантисі чисел із плаваючою крапкою.

Скалярні і векторні сигнали, що надходять до блока, можуть мати як дійсні, так і комплексні значення [2].

Якщо розмір блока *Display* малий для відображення всієї інформації, то в його правому нижньому куті з'являється чорний трикутник, що вказує, у якому напрямку варто розтягти піктограму блока.

Так само, як і *осцилограф*, *дисплей* може працювати в плаваючому режимі. Приклад використання блока наведено на рис. 4.11.

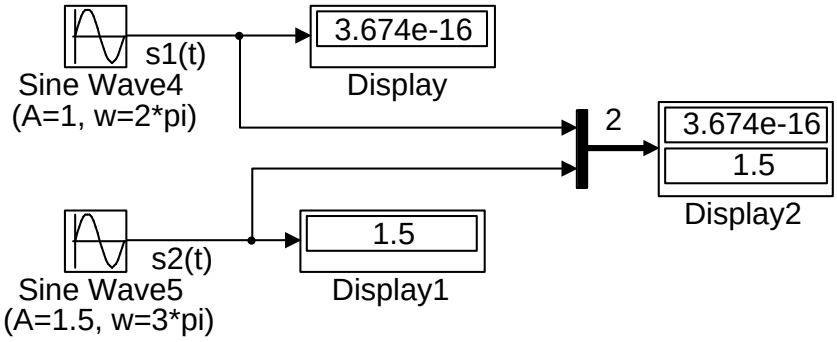


Рис. 4.11. Приклад використання блока *Display*

Оскільки зазвичай моделювання здійснюється дуже швидко і з малими проміжками часу між розрахунками, то користувач може побачити тільки останнє значення змінної після закінчення симуляції.

4.2. Блоки запам’ятовування сигналів

Блоками ***Out* (*вихідний порт*)** відзначають виходи моделей і підсистем.

У підсистемах вони необхідні для зв’язку підсистеми з моделлю більш високого рівня, а в моделях вищого рівня – для запам’ятовування приєднаних до них сигналів з метою подальшого використання для побудови графіків або для аналізу чисельних даних.

Для виконання останньої операції необхідно відкрити через функцію *Simulation* вікно *Model Configuration Parameters*, вибрати в ньому вкладку *Data Import/Export* та в розділі *Save to workspace* встановити прапорці в полях *Output* та *Time* і визначити імена змінних, у яких зберігатиметься інформація. За замовчанням у полі *Output* прописано ім’я yout , а у полі *Time* – ім’я tout, які, за бажанням користувача, можна змінити на будь-які інші (рис. 3.12).

Цей блок зручно використовувати для фіксації векторних сигналів.

Блоки *Out* використовують також для інформування додатків, призначених для аналізу і синтезу систем автоматичного керування (наприклад, *Control Toolbox*) про можливі точки знімання вихідних сигналів.

Параметри *Output when disabled* і *Initial output* мають значення тільки в підсистемах, виконуваних за умовою.

Блок ***To Workspace* (*у пам'ять*)** запам'ятовує дані, що надходять на його вхідний порт у змінній з ім'ям, завданим у поле параметра *Variable name*. Його зручно застосовувати для фіксації змінних з різними іменами.

Для запису інформації у форматі *Array* (*Масив*) треба не забути змінити значення за замовченням параметра *Save Format* (див. рис. 4.12).

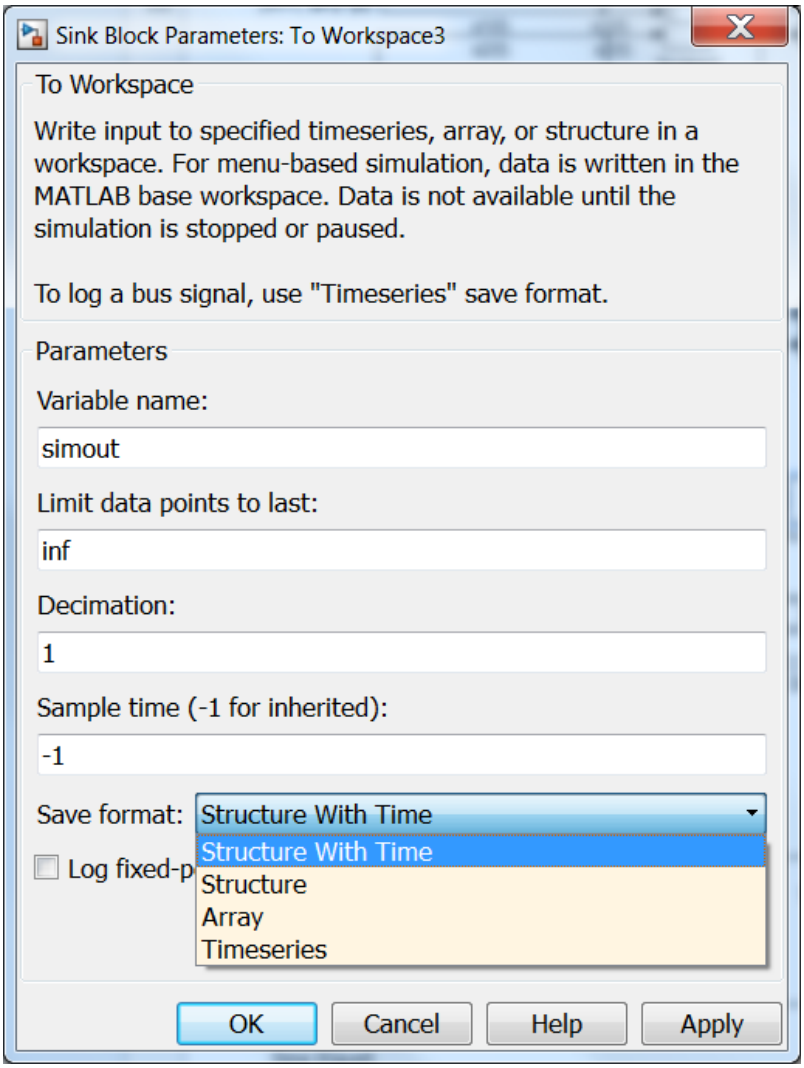


Рис. 4.12. Вікно параметрів блока *To Workspace*

Усякий раз ***при старті процесу моделювання дані, що зберігаються, обновляються***, так що цей блок не може об'єднати результати декількох послідовних розрахунків. Результати не доступні в робочому середовищі доти, поки не закінчено або не зупинено розрахунок перехідних процесів.

За замовченням параметр *Variable Name* має значення simout.

***Реєстратор To File* (*у файл*)** записує часи моделювання і вхідні сигнали у файл з ім'ям *File Name* та розширенням *mat (*.mat)*, який має структуру матриці, що містить у першому рядку монотонно зростаючий вектор часу, а в інших рядках відповідні йому вектори вихідних сигналів:

$$\begin{bmatrix} t_1 & t_2 & \dots & t_k \\ y_1 & y_2 & \dots & y_k \\ z_1 & z_2 & \dots & z_k \\ \dots & \dots & \dots & \dots \end{bmatrix}. \qquad (4.1)$$

Ім'я матриці задається параметром *Variable name*. Для дискретизації сигналів так само, як у блоці *To Workspace*, використовуються параметри *Decimation* і/або *Sample time*.

При кожному старті процесу моделювання файл даних обновляється.

Якщо якісь частини системи (наприклад, різні пристрої, що задають) працюють незалежно один від одного, то результати моделювання цих підсистем можна записати у файли і зчитувати ці данні з файлів при моделюванні частини системи, що залишилася. При багаторазовому моделюванні це дозволить скоротити час розрахунку перехідних процесів.

Цей блок також рекомендується використовувати в тому випадку, коли для запам'ятовування результатів моделювання не вистачає оперативної пам'яті або коли моделювання займає занадто багато часу. Отриманий файл результатів моделювання можна потім використовувати для побудови графіків за допомогою послідовно з'єднаних блоків *From File* і одного з пристроїв, що реєструють, наприклад, *Scope*. Піктограма блока відображає ім'я файлу, у який записуються матричні дані.

Матриця, записана у файл, стає доступною в середовищі *Matlab* після завантаження файлу командою load FileName у вигляді значення змінної, ім'я якої визначено в полі параметру *Variable name*.

На рис. 4.13 наведено приклад використання блоків *Out, To Workspace* та *To File* для реєстрації вихідних сигналів двох синусоїдальних сигналів, застосованих вище у моделях рис. 4.7, 4.10 та 4.11 [3].

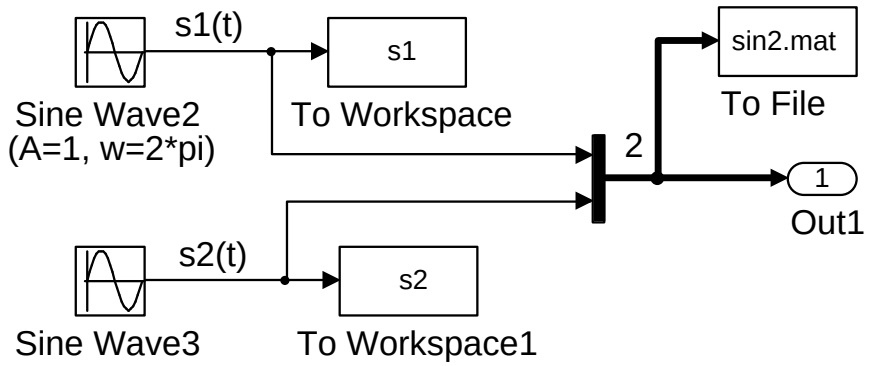


Рис. 4.17. Приклад реєстрації сигналів блоками *Out, To Workspace* та *To File*

Для того, щоб отримати графіки, аналогічні графікам, побудованим за допомогою блоків *Scope* (рис. 4.8) та *XY Graph* (рис. 4.10), використовуючи інформацію, записану у змінні tout, yout, з застосуванням блока *Out*, треба виконати таку послідовність операторів алгоритмічної мови *MATLAB*:

```
figure (1)
   plot (tout, yout (:,1)), grid on, title ('s1(t)')
figure (2)
   plot (tout, yout (:,2)), grid on, title ('s2(t)'),
figure (3)
   subplot (2,1,1)
      plot (tout, yout (:,1)), grid on, title ('s1(t)')
   subplot (2,1,2)
      plot (tout, yout (:,2)), grid on, title ('s2(t)')
figure (4)
   plot (tout, yout), grid on, legend ('s1(t)', 's2(t)')
figure (5)
   plot (yout (:,1), yout (:,2)), grid on, title ('s2(s1)')
```

Результати виконання цієї програми показані на рис. 4.14.

Такі ж само результати отримаємо, якщо у наведеній вище програмі замінимо yout (:,1) на s1, yout (:,2) на s2 та yout на [s1 s2].

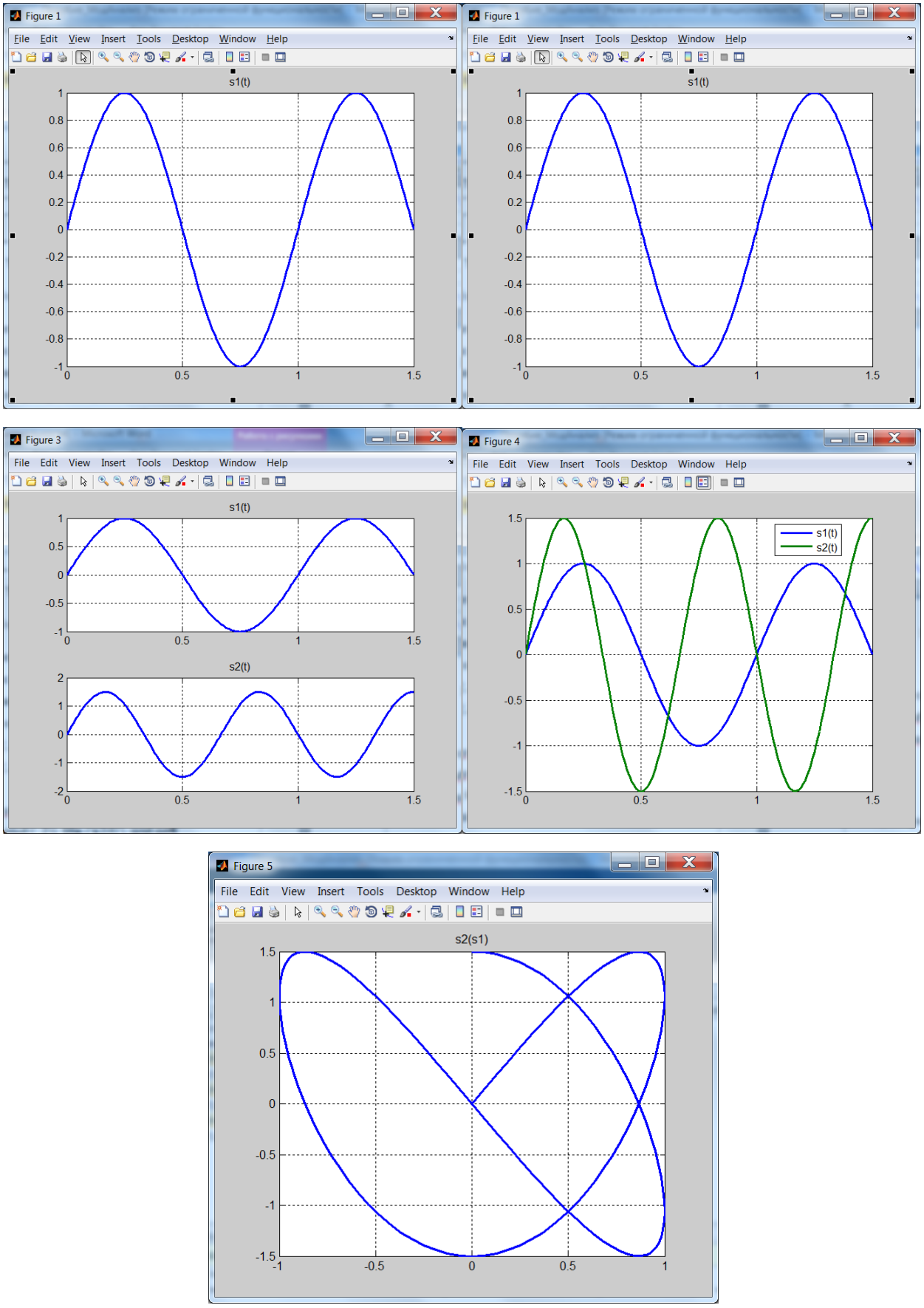


Рис. 4.14. Результати симуляції моделі рис. 4.13
та виконання наведеної вище програми

Контрольні питання та завдання

1. Охарактеризуйте бібліотечні блоки реєстрації вихідних сигналів, перелічите їхні недоліки та переваги.
2. Як створити багатоканальний осцилограф (блок *Scope*) та багатоканальний дисплей (блок *Display*)?
3. Як масштабуються графіки, відображені блоком? Як змінити межі графіків за часом та за вихідною координатою, кольори фону і ліній та інші властивості?
4. Як отримати інформацію про сигнали, приєднані до вихідних портів?
5. Опишіть структуру даних блоків *To Workspace*.
6. Як залишити у пам’яті сигнали, що візуалізуються блоком *Scope*? У чому полягає різниця між даними у форматі *Array*, записаними за допомогою блоків *Scope* та *To Workspace*?
7. Які імена за замовчанням мають дані, записані у пам’ять за допомогою блоків *Out*, *To Workspace* та *Scope*? Чи можна їх змінювати?
8. Поясніть призначення параметру *Decimation* блоків *To Workspace*, *To File* і *Scope.*
9. Поясніть призначення параметру *Limit data points to last* блоків *Scope.*
10. Які способи візуалізації одночасної візуалізації декількох сигналів ви знаєте? Як їх реалізувати?
11. Поясніть призначення блока *To File* і правила його використання.

5. ФОРМУВАННЯ ВХІДНИХ СИГНАЛІВ В СЕРЕДОВИЩІ *Simulink*

Вхідні сигнали, що діють на системи автоматичного керування, здебільш є функціями часу. Їх можна розподілити на такі групи:

- постійні;
- ступеневі;
- лінійні;
- нелінійні неперіодичні, що описуються аналітичними виразами;
- гармонічні;
- періодичні;
- випадкові.

5.1. Загальна характеристика блоків бібліотеки *Sources*

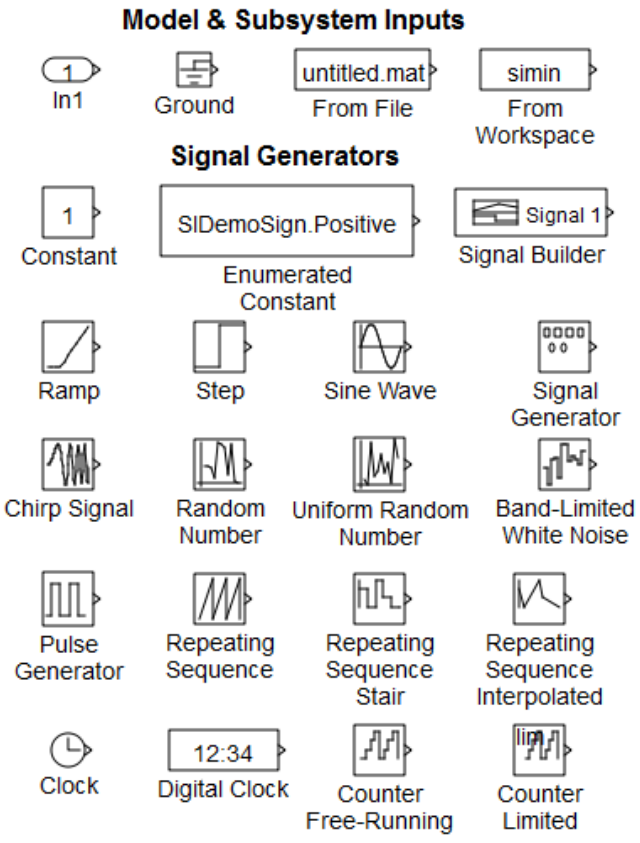


Рис. 5.1. Бібліотека джерел вхідних сигналів *Sources*

Основу формування вхідних сигналів складають блоки бібліотеки джерел *Sources*. У цю бібліотеку входить група блоків, що не мають вхідних портів, а мають тільки виходи. Їхні піктограми подані на рис. 5.1.

Блоки цієї бібліотеки можна розділити на 3 групи [3]:

1) блоки, що не формують вхідних сигналів, а тільки помічають входи (блок *In*) або зазе-

мляють їх (блок *Ground*), щоб при старті моделювання в командному вікні *MATLAB* не виводилося попередження про наявність у моделі неприєднаного вхідного порту (Warning: Input port 1 of block … is not connected);

2) блоки, що можуть формувати тільки один вихідний сигнал (*Ramp, Clock, Digital Clock* і *Repeating Sequence*);

3) блоки, що можуть формувати як один, так і декілька вихідних сигналів (*Const, Step, Signal Generator, Sine Wave, Pulse Generator, Chirp Signal, Random Number, Uniform Random Number* і *Band Limited White Noise*).

Для того, щоб блоки 3-ої групи формували матрицю, складену з одновимірний векторів-стовпців, кількість яких відповідає кількості компонент у векторах параметрів, у вікні настроювання блока має бути встановленим прапорець *Interpret vector parameters as 1-D.* Якщо цей прапорець не встановлено, то блок на кожному кроці чисельного інтегрування формує сигнали розмірність яких повторює розмірність параметрів. Для останнього випадку зберігати дані можна тільки у форматі *Structure with Time*. Візуалізація даних не залежить від стану параметру *Interpret vector parameters as 1-D.*

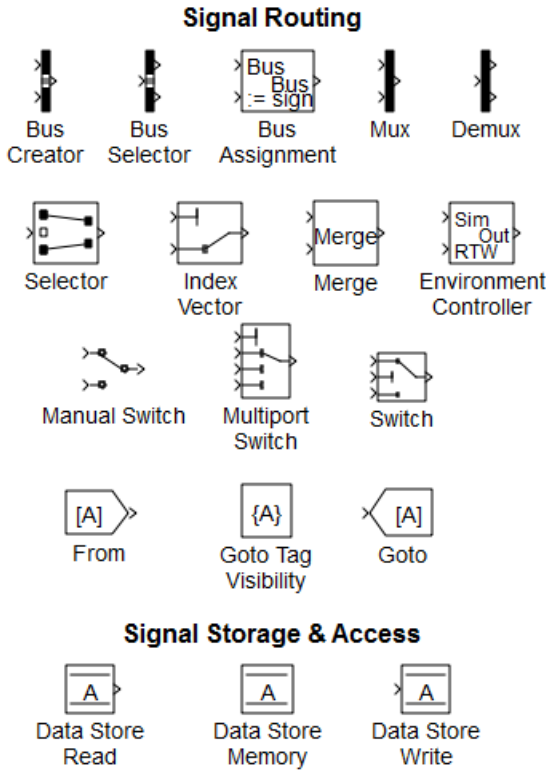


Рис. 5.2. Бібліотека *Signals Routing* (*Маршрутизація сигналів*)

Векторні параметри джерел можуть бути задані у вікнах настроювання блоків як рядками, так і стовпцями за правилами алгоритмічної мови системи *MATLAB.* Векторний сигнал може бути розділений на скалярні сигнали або на векторні сигна-

ли з меншою кількістю компонентів за допомогою блока *Demux* бібліотеки *Signal Routing* (див. рис. 5.2). Зворотну операцію виконує блок *Mux*.

Такі джерела, як *Step, Sine Wave, Pulse Generator, From File, From Workspace, Random Number* і *Uniform Random Number,* можуть змінювати значення своїх вихідних сигналів як неперервно, так і дискретно, тобто в моменти часу, кратні періоду дискретності, що задається параметром *Sample Time* (T_S). Для того, щоб джерело працювало у неперервному режимі, значення цього параметра необхідно покласти рівним нулю (T_S =0).

Якщо джерело повинне формувати дискретний сигнал, то в поле параметра *Sample Time* може вводитися одне додатне значення, що інтерпретується як період дискретності *Ts,* або два додатних значення, перше з який сприймається як період дискретності *Ts,* а друге – як зсув (запізнення) *offset.* У цьому випадку зміна вихідних сигналів ланок будуть здійснюватися в дискретні моменти часу

$$t_d = nT_s + offset\ ; \qquad |offset| \le T_s . \tag{5.1}$$

У дискретному режимі кожен блок працює так, як працював би цей же блок у неперервному режимі з приєднаним до його виходу екстраполятора нульового порядку (блок *Zer99o-Order Hold* бібліотеки *Discrete*) з тим же значенням параметра *Sample Time*. Якщо установити T_S =-1, то джерело працює в тому ж режимі, що і блок, приєднаний до його виходу.

5.2. Основні засоби формування вхідних сигналів

Виходом блока ***Constant* (*константа*)** можуть бути визначені параметром *Constant value* і не залежні від часу скалярна константа $y = c = const$, вектор констант (рядок $\mathbf{y} = [c_1 \quad c_2 \quad ... \quad c_n]$ або стовпець $\mathbf{y} = [c_1 \quad c_2 \quad ... \quad c_n]^{\mathrm{T}}$) та матриця констант. Елементи масивів можуть бути як дійсними, так і комплексними числами (в тому числі і зарезервованими константами *MATLAB*).

Джерело ***Step* (*стрибок, сходинка*)** забезпечує стрибкоподібну зміну вихідного сигналу між двома постійними рівнями y_{In} *(Initial value)* і y_{Fin} *(Final value)* у заданий момент часу t_s *(Step time)*:

$$y_{Step}(t) = \begin{cases} y_{In} \text{ при } t \le t_s, \\ y_{Fin} \text{ при } t > t_s. \end{cases} \tag{5.2}$$

Загальний вигляд скалярного ступінчатого сигналу показано на рис. 5.3

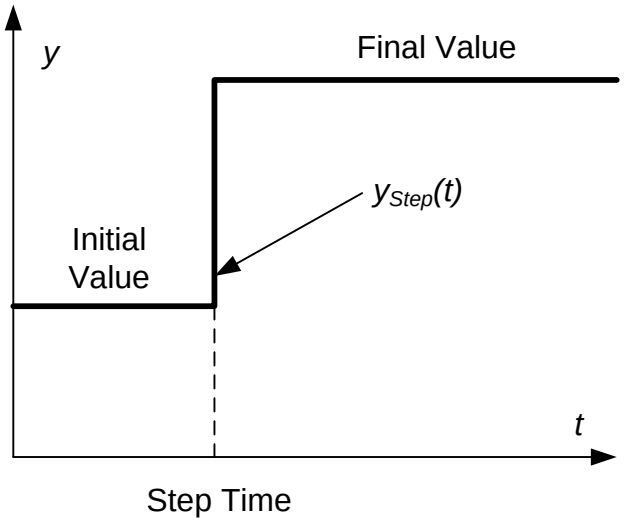


Рис. 5.3. Загальний вигляд скалярного ступінчатого сигналу

Декілька блоків *Step*, приєднаних до входів алгебраїчного суматора *Sum* з математичної бібліотеки *Math operations* створюють багатосходинковий сигнал. Враховуючи можливість блока *Step* формувати векторні сигнали, та можливість блока *Sum* сумувати усі елементи векторного сигналу, багатосходинковий сигнал можна створити послідовним з'єднанням одного блока *Step* із суматором.

Блоки ***Sum* (*суматор*)** та його аналоги ***Add (додавати), Subtract (віднімати), Sum of Elements*** можуть мати круглу (*round*) або прямокутну (*rectangular*) форму, яка визначається параметром *Icon shape*.

Вони підсумовують вхідні сигнали з відповідними знаками, які встановлюються в полі параметра *List of signs* (входи нумеруються зверху вниз для прямокутного блока та проти годинної стрілки для круглого блока). Комбі-

нація знаків “+”, ”–“ описує параметри кожного конкретного порту, де кількість портів відповідає кількості знаків, використаних у комбінації. Для пропуску позиції чергового порту у списку знаків використовують символ „|”.

Якщо замість списку знаків визначити кількість входів, то усі вони будуть підсумовуючими. При одному вході блок підсумовує з відповідним знаком елементи вхідного вектору:

$$y = sum(u) = \sum_{i=1}^{k} u_i \ . \qquad (5.3)$$

При цьому в піктограмі блока відображається не знак вхідного сигналу, а символ Σ. Блоки *Sum* та *Add* з одним входом можна замінити блоком *Sum of Elements*.

Можливі різновиди іконок блоків *Sum, Subtract, Add* та *Sum of elements* в залежності від значень їх параметрів наведені на рис. 5.4 [3].

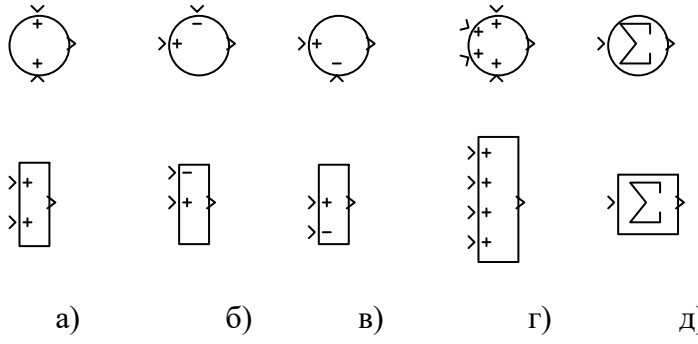


Рис. 5.4. Вигляд іконок блоків *Sum, Subtract, Add* та *Sum of elements* при таких значеннях параметру *List of signs*:
а) ++; *б*) –+|; *в*) |+–; *г*) 4; *д*) 1 або +

На рис. 5.5 показано багатоступінчатий сигнал, а на рис. 5.6 – варіанти його формування. Параметри блоків *Step* з рис. 5.6 наведені у табл. 5.1.

Блок ***Clock* (*Годинник*)** забезпечує відлік і відображення часу моделювання: $y(t) = t$. Параметр *Decimation*, що може приймати цілочислові додатні значення *d*, має сенс тільки при включеному прапорці *Display time* і при використанні для рішення диференціальних рівнянь методів чисельного інтегрування з постійним кроком *h*. У цьому випадку в піктограмі блока, що

здобуває прямокутну форму, відображаються по черзі чисельні значення часу моделювання, кратні *d*h*.

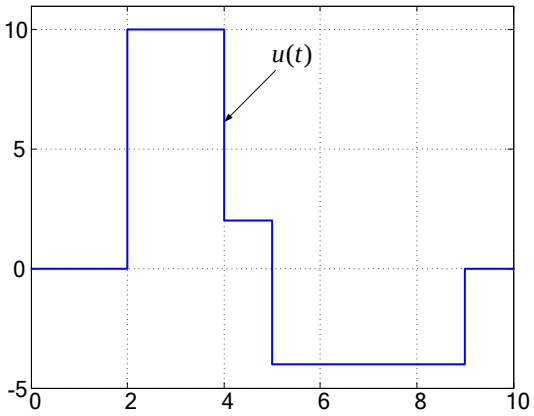


Рис. 5.5. Багатоступінчатий неперіодичний сигнал

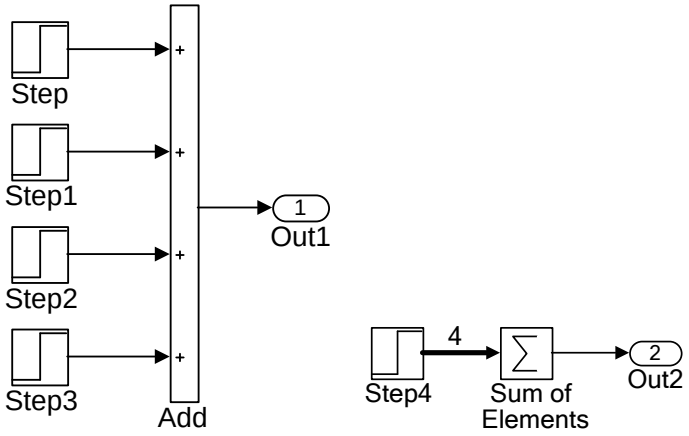


Рис. 5.6. Варіанти формування багатоступінчатого сигналу

Таблиця 5.1

Ім’я блока	*Step Time*	*Initial Value*	*Final Value*
Step	2	0	10
Step1	4	0	-8
Step2	5	0	-6
Step3	9	0	4
Step4	[2 4 5 9]	[0 0 0 0]	[10 -8 -6 4]

Блок ***Digital clock* (*цифровий годинник*)** формує сигнал, який збігається з часом моделювання у моменти часу, кратні періоду дискретності *Sample Time*. Між цими моментами вихідний сигнал має постійне значення, що дорівнює останньому обмірюваному часу моделювання.

Джерело ***Ramp* (*рампа, похила площина*)** формує сигнал, що лінійно наростає або спадає від значення y_0 (*Initial output*), починаючи з моменту часу t (*Start time*) з коефіцієнтом пропорційності k (*Slope*):

$$y(t)=\begin{cases} y_0 \ \text{при} \ t \le t_S, \\ y_0 + kt \ \ \text{при} \ t > t_S. \end{cases} \tag{5.4}$$

Загальний вигляд скалярного ступінчатого сигналу показано на рис. 5.7.

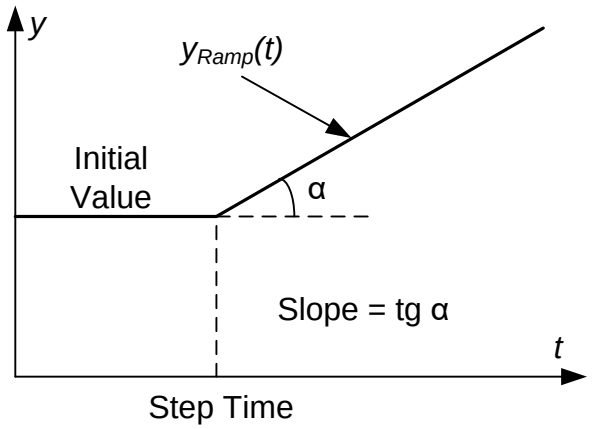


Рис. 5.7. Загальний вигляд скалярного сигналу *Ramp*

Блок ***Sine Wave* (*синусоїда*)** може працювати в двох режимах (*Time-Based Mode* і *Sample-Based Mode*), котрі встановлюються у вікні настроювання блока за допомогою випадного меню (*Parameters, Sine type*). У режимі *Time-Based* можливе формування як неперервного, так і дискретного вихідного сигналу. У неперервному режимі (T_S =0) блок генерує синусоїду у функції часу відповідно до рівняння:

$$y(t) = A\sin(\omega t + \varphi_0) + y_0, \tag{5.5}$$

де

A – амплітуда (*Amplitude*);

ω – кругова частота, рад/с (*Frequency*);

φ_0 – фазовий зсув, рад. (*Phase*);

y_0 – вертикальний зсув (*Bias*).

У дискретному *Time-Based* режимі (T_S >0) при обчисленні вихідного сигналу використовуються формули:

$$\begin{cases} \sin(t+T_S) = \sin(t)\cos(T_S) + \sin(T_S)\cos(t), \\ \cos(t+T_S) = \cos(t)\cos(T_S) - \sin(t)\sin(T_S), \end{cases} \tag{5.6}$$

які дозволяють формувати вихідне значення значно швидше, ніж у непере-

рвному режимі. Недоліком цього методу є нагромадження похибок округлення, тому що для обчислення вихідного сигналу в кожний наступний момент часу використовується значення вихідного сигналу в попередній момент часу.

Для ліквідації цього недоліку ланка *Sine Wave* передбачає можливість роботи в режимі *Sample-Based*. При цьому значення вихідного сигналу обчислюється за формулою

$$y = A\sin\left(2\pi(k+o)/p\right) + y_0, \tag{5.7}$$

де

p – кількість розрахункових крапок на періоді хвилі синусоїди (*Samples per period*);

o – зсув (відносне фазове зрушення) сигналу (*Number of offset Samples*);

k – цілочисловий параметр, що приймає на періоді синусоїди значення [0, 1, 2, …, p-1]...

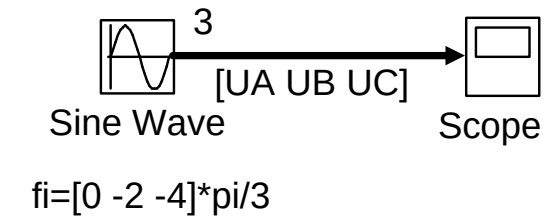
Для узгодження методів *Time-Based discrete* і *Sample-Based* необхідно забезпечити наступні співвідношення між їхніми параметрами:

$$p = \frac{2\pi}{\omega T_S}; \qquad o = \frac{p\varphi_0}{2\pi}. \tag{5.8}$$

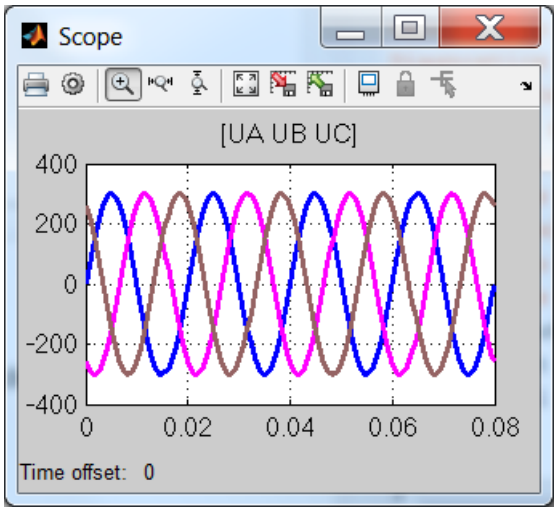
Ліквідуючи нагромадження похибок округлення, формула (5.7) має схований недолік, що виявляється при використанні цього блока в підсистемі, що виконується за умовою (*Conditionally Executed Subsystem*). Цей недолік полягає в можливості неузгодженої роботи блока *Sine Wave* з іншими блоками моделі після виконання підсистемою паузи з наступним поновленням її роботи. Отже, при виникненні потреби формування дискретної синусоїди в складі підсистеми, виконуваної за умовою, блок *Sine Wave* необхідно використовувати в режимі *Time-Based*.

Не варто забувати, що блок *Sine Wave* здатний формувати векторні сигнали. Цю властивість зручно використовувати при моделюванні трифазних

кіл змінного струму. Для прикладу на рис. 5.8 наведено модель формування трифазної напруги змінного струму частотою 50 Гц і амплітудою 300 В, зсунутих одна від одної на кут 120 градусів і результат її симуляції.



а)



б)

Рис. 5.8. Модель (*а*) формування трифазної напруги та результати її симуляції (*б*)

Блок ***Pulse Generator*** **(*генератор прямокутних імпульсів*)** також може працювати в неперервному (*Time-Based*) або дискретному (*Sample-Based*) режимах, вибір одного з яких здійснюється за допомогою параметра *Pulse Type*. В обох режимах блок формує періодичний сигнал, що складається з додатних прямокутних імпульсів. В неперервному режимі він має наступні параметри:

ht – висота імпульсу (*Amplitude*);

T – період в одиницях виміру часу (*Period(secs)*);

du – ширина імпульсу у відсотках від періоду (*Pulse width (in % of period)*);

stt – час початку пульсацій (*Phase delay (secs)*).

У дискретному режимі у вікні настроювання блока з'являється додатковий параметр *Sample Time* , а період і час початку пульсацій задаються у відносних одиницях (частках періоду дискретності – *Number of samples*).

Якщо при моделюванні обрано метод рішення диференціальних рівнянь з фіксованим кроком, то *Simulink* автоматично використовує блок *Pulse Generator* у режимі *Sample-Based* з періодом дискретності, рівним кроку інтегрування. У цьому випадку крок інтегрування повинний бути обраний та-

ким, щоб параметри *T, du*T/100* і *stt* були кратні йому.

Блок ***Repeating Sequence* (*Повторювана Послідовність*)** виконує повторення циклу, заданого таблицею двох параметрів: *Time values* (t_v) – вектор часу і *Output values* (y_v) – вектор вихідних сигналів. Вектор часу має бути монотонно зростаючим. Різниця між значеннями його останнього і першого елементів визначає період вихідного сигналу: *T*=*tv*(*end*)-*tv*(1). Блок перемальовує свою піктограму відповідно до вигляду вихідного сигналу. Він реалізований за допомогою замаскованої підсистеми зображеної на рис. 5.9.

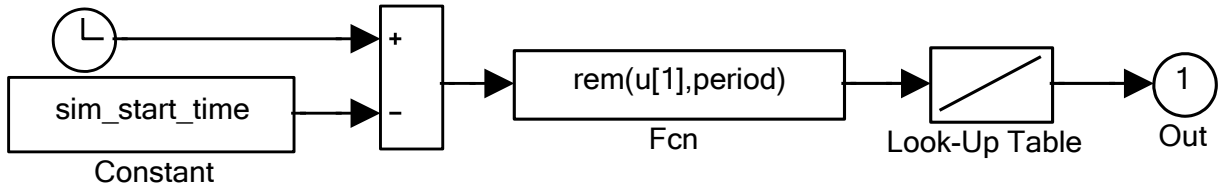


Рис. 5.9. Розгорнута модель замаскованого блока *Repeating Sequence*

Блок *Fcn* моделі рис. 5.10 визначає залишок від цілочислового ділення часу моделювання на період, а блок *Look-Up Table* виконує кусочно-лінійну апроксимацію табличної функції, заданої параметрами *Vector of input values* і *Table data*.

Джерело ***Chirp Signal* (*синусоїда зі змінною частотою*)**, реалізоване за допомогою замаскованої підсистеми рис. 5.10, формує синусоїду з одиничною амплітудою і частотою, що змінюється за лінійним законом:

$$y = \sin(At^2 + Bt), \tag{5.9}$$

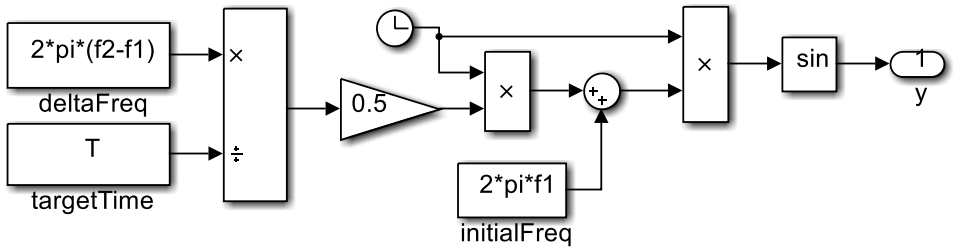
де

$$B=2\pi f_1\ , \quad A=2\pi (f_2 - f_1) / T, \tag{5.10}$$

за вхідними параметрами:

- f_1 (*Initial frequency* [*Hz*]) – початкова частота [Гц];
- T (*Target time* [*secs*]) – заданий час [с];
- f_2 (*Frequency at target time* [*Hz*]) – частота в заданий момент часу [Гц].

Піктограма джерела зображує графік функції $y=sin(t^2)$.




```
freqSlope = deltaFreq/targetTime;
instantaneousFreq(t) = freqSlope*t + initialFreq;
intantaneousPhase(t) = integral(instFreq) = ...
      0.5*freqSlope*t^2 + initialFreq*t;
outputSignal(t) = sin(instantaneousPhase(t));
```

Рис. 5.10. Розгорнута модель блока *Chirp Signal*

Випадкові сигнали створюються генераторами випадкових чисел.

Джерела випадкових чисел з нормальним **(*Random Number*)** та рівномірним **(*Uniform Random Number*)** розподілами генерують відповідно нормально розподілений (Гаусовський) і рівномірно розподілений випадкові сигнали для деяких заданих стартових чисел *Initial seed*, що ініціалізують генератори випадкових чисел.

Специфічними параметрами ланки *Random Number* є середнє значення сигналу *Mean* і середньоквадратичне відхилення *Variance*, а ланки *Uniform Random Number* – мінімальне *Minimum* і максимальне *Maximum* значення випадкового сигналу. Обидва блоки можуть працювати як в неперервному, так і у дискретному режимах.

Джерело ***Band-Limited White Noise* (*білий шум з екстраполяцією*)** забезпечує формування «білого шуму» для неперервних систем за допомогою послідовно з'єднаних блоків *Random Number*, що працює в дискретному режимі, та *Gain* (*Пропорційна ланка*) на основі таких параметрів:

Cov (Noise Power) – потужність шуму (коваріація);

T_s *(Sample Time)* – період дискретності;

Seed – стартове число генератора випадкових чисел.

Коефіцієнт підсилення ланки *Gain* обчислюється за формулою

$$k = \sqrt{Cov} / \sqrt{T_s} \,. \qquad (5.11)$$

Вектори *Noise Power* і *Seed* можуть бути однієї довжини. Для більш швидкого протікання процесу моделювання необхідно параметр *Sample Time* установлювати максимально великим, узгоджуючи однак його величину з мінімальної сталою часу системи.

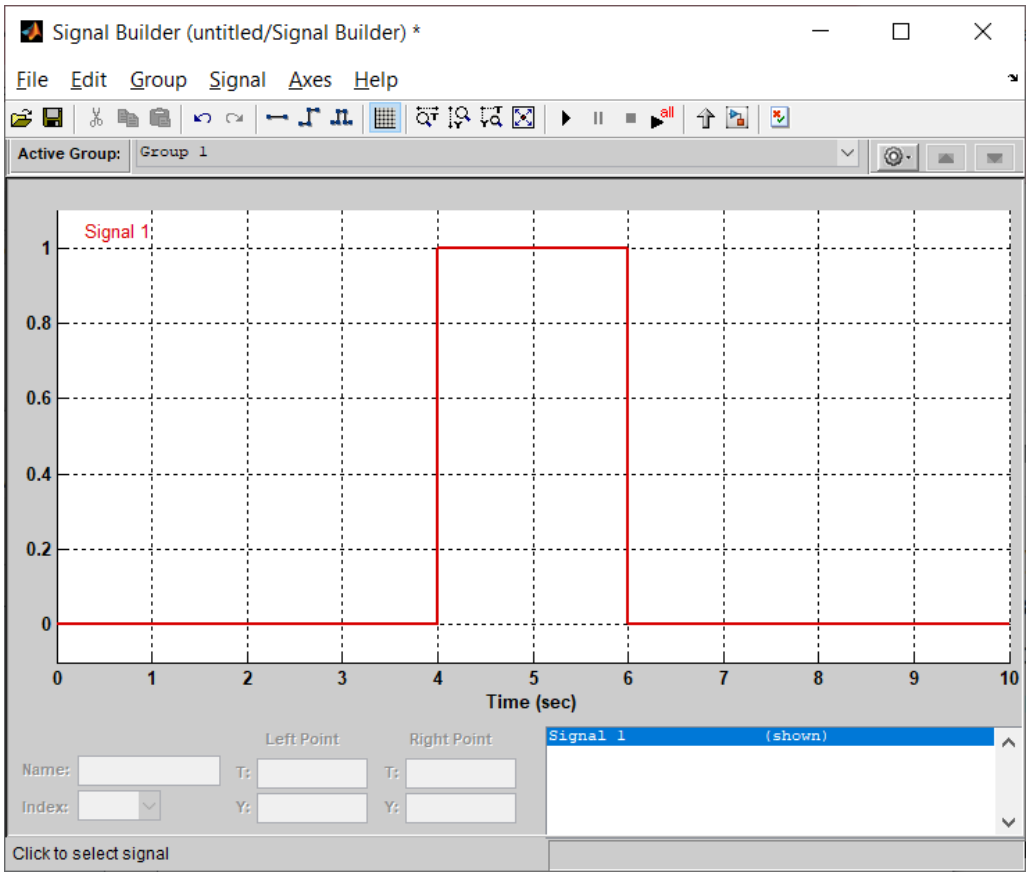
Комплексне джерело ***Signal Generator* (*генератор сигналів*)** генерує один з 4-х сигналів: синусоїду (*Sine*), прямокутний (*Square*) періодичний сигнал зі шпаруватістю 50%, пилкоподібний (*Sawtooth*) періодичний сигнал, а також випадковий сигнал (*Random*).

Тип сигналу вибирається за допомогою випадного меню (*Parameters*, *Wave form*). Чисельними параметрами являються амплітуда *A* (*Amplitude*) і частота (*Frequency*). Частота може задаватися як у Гц (*Hertz*), так і в рад/с (*rad/sec*) за допомогою параметра *Units*. Значення всіх сигналів змінюються від +*A* до –*A*. Вихідні установки блока можуть бути змінені протягом процесу моделювання, що особливо ефективно у сполученні з фіксацією результатів блоком *Scope* (*Осцилограф*), коли реакцію системи на різні типи вхідного сигналу потрібно одержати швидко.

Комплексне джерело ***Signal Bilder*** може генерувати один або декілька сигналів (див. рис. 5.11) з переліку (*Signal→New→*) *Constant, Step, Pulse…, Square…, Triangle…, Sampled Sin…, Sampled Gaussian Noise…, Pseudorandom Noise…, Poisson Random Noise…, Custom….* з різними параметрами.

Отримані сигнали можна коригувати переміщенням фрагментів та/або точок зламу кусково-лінійних функцій та змінювати їх властивості (*Signal → Rename, Color, Line Style, Line Width, Change Index*). Крім того, можна коригувати властивості системи координат (*Axes→Change Time Range, Set Y Snap Grid, Set T Snap Grid, Set Y Display Limits, Set T Display Limits*), ховати (*Signal→Hide*) та показувати (*Signal→Show*) графік сигналу, прибирати та пока-

зувати координатні сітки (кнопка *Toggle Grid Lines*).



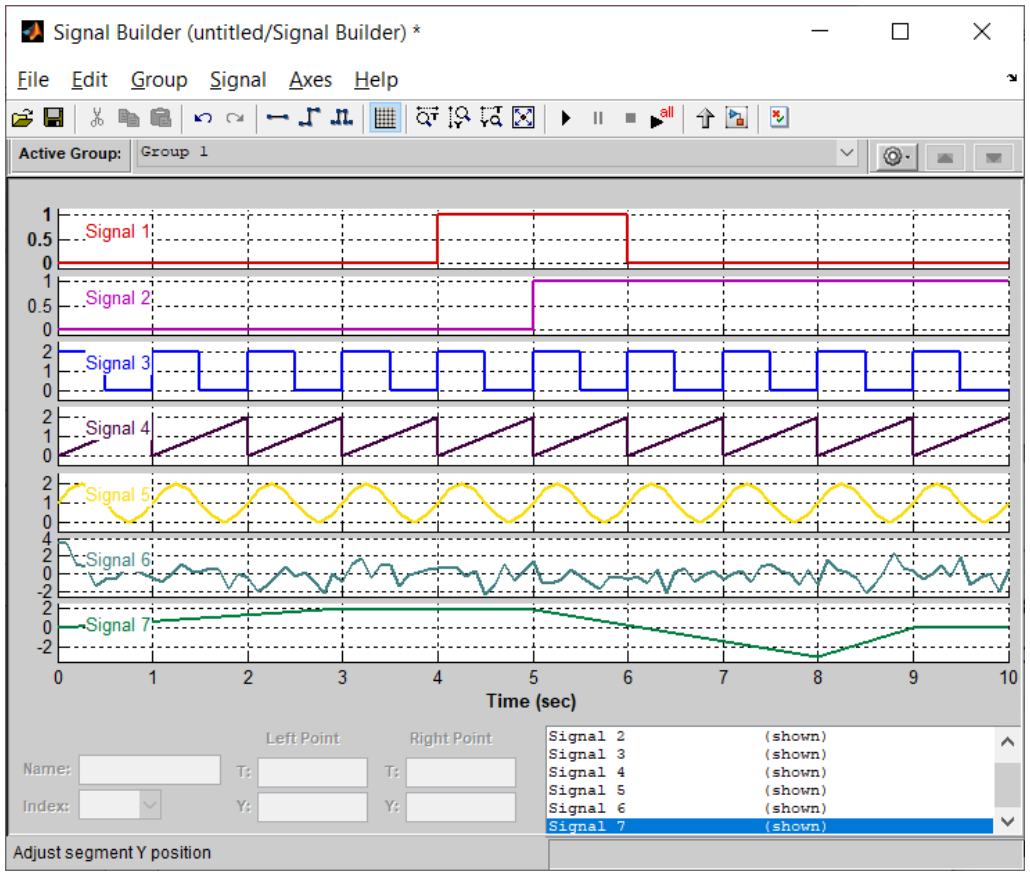


Рис. 5.11. Вікна блока *Signal Builder*

Блок ***From Workspace* (*з пам'яті*)** забезпечує читання даних з матриці, що повинна мати два або більш стовпців. Перший стовпець повинний містити монотонно зростаючі значення часу, а кожен додатковий стовпець – табличні значення функцій часу:

$$\begin{bmatrix} t_1 & y_1 & z_1 & \dots & w_1 \\ t_2 & y_2 & z_2 & \dots & w_2 \\ \dots & \dots & \dots & \dots & \dots \\ t_k & y_k & z_k & \dots & w_k \end{bmatrix}. \qquad (5.11)$$

Наприклад, для того, щоб блок *From Workspace* сформував на інтервалі $t \in [0, 2\pi]$ сигнал $y(t) = sin(t) + cos(t^2)$, у середовищі *MATLAB* до моделювання необхідно виконати послідовність операцій

```
Time = (0 : pi/50 : 2*pi)';
Out = sin (Time) + cos (Time.^2);
D = [Time  Out];
```

і установити в поле параметра *Data* у вікні настроювання блока *From Workspace* ім'я матриці D.

Значення часу використовуються блоком для обчислення вихідного сигналу за допомогою лінійної інтерполяції або екстраполяції.

Піктограма блока відображає ім'я матриці, з якої зчитуються його дані.

За допомогою випадного меню *Form output after final data value by* можна вибрати один з чотирьох способів зміни вихідного сигналу після перевищення часом моделювання значення останнього елемента першого стовпця матриці даних (максимального значення аргументу табличної функції):

- *Extrapolation* – лінійна екстраполяція;
- *Setting To Zero* – установка в нуль;
- *Holding Final Value* – затримка останнього табличного значення;
- *Cyclic Repetition* – циклічне повторення.

Для використання першого способу прапорець інтерполяції *Interpolate*

data повинний бути обов'язково встановлений, а для останнього способу – скинутий. Для інших способів стан цього прапорця не має значення. При останньому способі (*Cyclic Repetition*) дані, що зчитуються, повинні мати формат не матриці, а структури. Наприклад, щоб розглянутий у попередньому прикладі сигнал циклічно повторювався при $t > 2\pi$, необхідно сформувати структуру даних у такий спосіб:

```
D.Time = (0 : pi/50 : 2*pi)';
D.signals(1).Out=sin (Time)+cos (Time.^2);
D.signals(1).dimensions = 1;
```

Останній параметр визначає розмірність вихідного сигналу.

Джерело ***From File* (*з файлу*)** забезпечує читання даних з файлу, чиє ім'я занесено в поле параметра *Filename* і відображається в піктограмі блока.

Цей формат ідентичний формату блока *To File*, що забезпечує сумісність файлів.

Запис даних у *mat*-файл у пакеті *MATLAB* виконує команда

```
save FileName var1 var2 … .
```

Наприклад,

```
save data1 D
```

збереже значення змінної D у файлі *data1.mat.* Для визначення вихідних сигналів у моменти часу, відсутні у файлі даних, використовується лінійна інтерполяція або екстраполяція. Файл даних для блока *From File* може бути сформований блоком *To File* при моделюванні іншої системи.

Блок ***In* (*Вхідний Порт*)** помічає входи моделі. Якщо у вкладці *Workspace I/O* вікна *Simulation Parameters* установити прапорець *Input* функції *Load from Workspace*, то зовнішній вхідний сигнал можна задавати в полі параметру *Input* у такий же спосіб, як за допомогою блока *From Workspace*.

Основним параметром блока *In* є номер порту (*Port number*), що відображається в його піктограмі. Усі порти в межах одного вікна повинні мати послідовну нумерацію. Номера портів формуються автоматично в порядку

їхньої появи у вікні. При видаленні порту з вікна порти, що залишилися в ньому, перенумеровуються таким чином, щоб у послідовності номерів не було пропусків. Номера портів можна змінювати і вручну.

Параметром *Port dimensions* можна визначити розмірність вхідного сигналу. Значення за замовчуванням (-1) відповідає автоматичній установці цього параметра.

Тип сигналу (*real, complex*), тип представлення даних (*double, single, int8, ..., boolean*), прапорець *Interpolate data*, а також параметр *Sample time* мають сенс тільки при використанні зовнішніх джерел у моделях верхнього рівня.

Додаткові засоби формування нелінійних вхідних сигналів будуть розглянуті при знайомстві з нелінійними блоками і функціональними перетворювачами бібліотек *Simulink*.

Контрольні питання та завдання

1. Які типи вхідних сигналів ви знаєте?
2. Дайте стислу характеристику блоків бібліотеки Sources. Які з них можуть формувати векторні сигнали?
3. Як утворити багатоступінчатий неперіодичний сигнал?
4. Яким параметром змінюється кут нахилу лінійної характеристики, створюваної блоком *Ramp*?
5. Як сформувати одним блоком трифазну гармонічну напругу
6. Поясніть принцип роботи блока *Repeating Sequence*.
7. Поясніть принцип роботи блока *Chirp Signal*.
8. Охарактеризуйте параметри блока *Pulse Generator*.
9. Які способи генерації випадкових сигналів ви знаєте?
10. Охарактеризуйте можливості блока *Signal Generator*.
11. Охарактеризуйте можливості блока *Signal Bilder*.
12. Як можна подати на вхід системи сигнал, знятий експеримента-

но?

6. МОДЕЛЮВАННЯ НЕЛІНІЙНИХ ЕЛЕМЕНТІВ У СЕРЕДОВИЩІ *Simulink*

6.1. Загальна характеристика нелінійних блоків програми *Simulink*

Нелінійні ланки, що найчастіше використовуються при моделюванні електромеханічних систем можна розподілити на такі групи [3]:

1) блоки множення-ділення декількох сигналів;

2) функціональні перетворювачі, описувані нелінійними аналітичними виразами;

3) функціональні перетворювачі, задані таблицями вхідних та вихідних сигналів у деяких вузлових точках;

4) періодичні функціональні перетворювачі, задані аналітично або таблицею вхідних та вихідних сигналів на періоді.

Серед блоків другої та третьої груп виділяють ***типові нелінійності***. Так називають блоки з досить простими кусково-лінійними характеристиками вхід-вихід, що часто зустрічаються на практиці.

До них належать такі ідеалізовані блоки як *Обмеження координат*, *Сухе тертя*, *В'язке тертя*, *Зона нечутливості*, *Модуль*, *Люфт* (*Зазор*), *Петля гістерезису*, *Реле*, *Компаратор* та деякі інші.

З іншого боку всі нелінійності можна поділити на однозначні та багатозначні (найчастіше двозначні). У багатозначних нелінійностей зв'язок між вхідним та вихідним сигналами визначається не тільки формою статичної характеристики, але й передісторією вхідного сигналу, наприклад, від того, зменшується вхідний сигнал чи наростає. Типовими двозначними нелінійностями є блоки *Петля гістерезису* та *Зазор в кінематичній передачі*.

При розрахунку вихідних сигналів нелінійних блоків, статична характеристика «вхід-вихід» яких визначена таблицею, застосовують або апроксимацію, або інтерполювання.

При апроксимації найчастіше використовують ***метод найменших***

квадратів або ***кусково-лінійну апроксимацію***. У якості апроксимуючих аналітичних функцій здебільш застосовують степеневі поліноми, комбінації експонент та розкладення періодичних характеристик в ряд Фур'є.

Серед методів інтерполяції для більшості технічних розрахунків достатню точність можна забезпечити локальним інтерполюванням степеневими поліномами першого-третього порядків. Для підвищення точності можна використовувати глобальну інтерполяцію або інтерполяцію кубічними сплайнами. Але застосування останніх методів може значно збільшити час розрахунку перехідних процесів.

6.2. Блоки множення та ділення

Блоки, що здійснюють операції множення та ділення декількох сигналів знаходяться у бібліотеці *Math Operations* (див. рис. 6.1).

Блоки ***Product* (*Множення), Divide (Ділення*)** та ***Product of Elements (Добуток елементів)*** є одним і тим же блоком, що здійснює різні операції і змінює свою піктограму (вигляд, іконку) в залежності від значення параметрів *Number of inputs, Multiplication, Multiply over* та *.Dimension.*

У полі параметру *Number of inputs* можна вводити кількість входів або чергувати знаки операцій множення (*) і ділення (/) без пробілів між ними, подібно до чергування операцій додавання (+) та віднімання (-) у блоках *Sum, Add* і *Subtract*. Наприклад, при послідовності «*/*» для скалярних вхідних сигналів блоки розраховують $y = u_1 u_3 / u_2$, а при значенні цього параметру «4» – $y = u_1 u_2 u_3 u_4$. Значення «1» або «*» мають сенс тільки для векторних або матричних вхідних сигналів. У цьому разі до списку параметрів досліджуваних блоків додається параметр *Multiply* over, який може приймати значення або *Specified Dimension.*

Якщо обрано опцію *All Dimensions*, то блок розраховує добуток усіх елементів вхідного масиву. Якщо обрано опцію *Specified Dimension*, то

з’являється ще й параметр *Dimension*, значеннями якого є цілочислові константи, що визначають порядковий номер розмірності (для матриці значення «1» примушує блок розраховувати добутки усіх елементів кожного стовпця, а значення «2» – добутки усіх елементів кожного рядка).

Параметр *Multiplication* може приймати значення *Element-wise (.*)* та *Matrix (*)*. Опція *Matrix* має сенс при наявності мінімум двох входів.

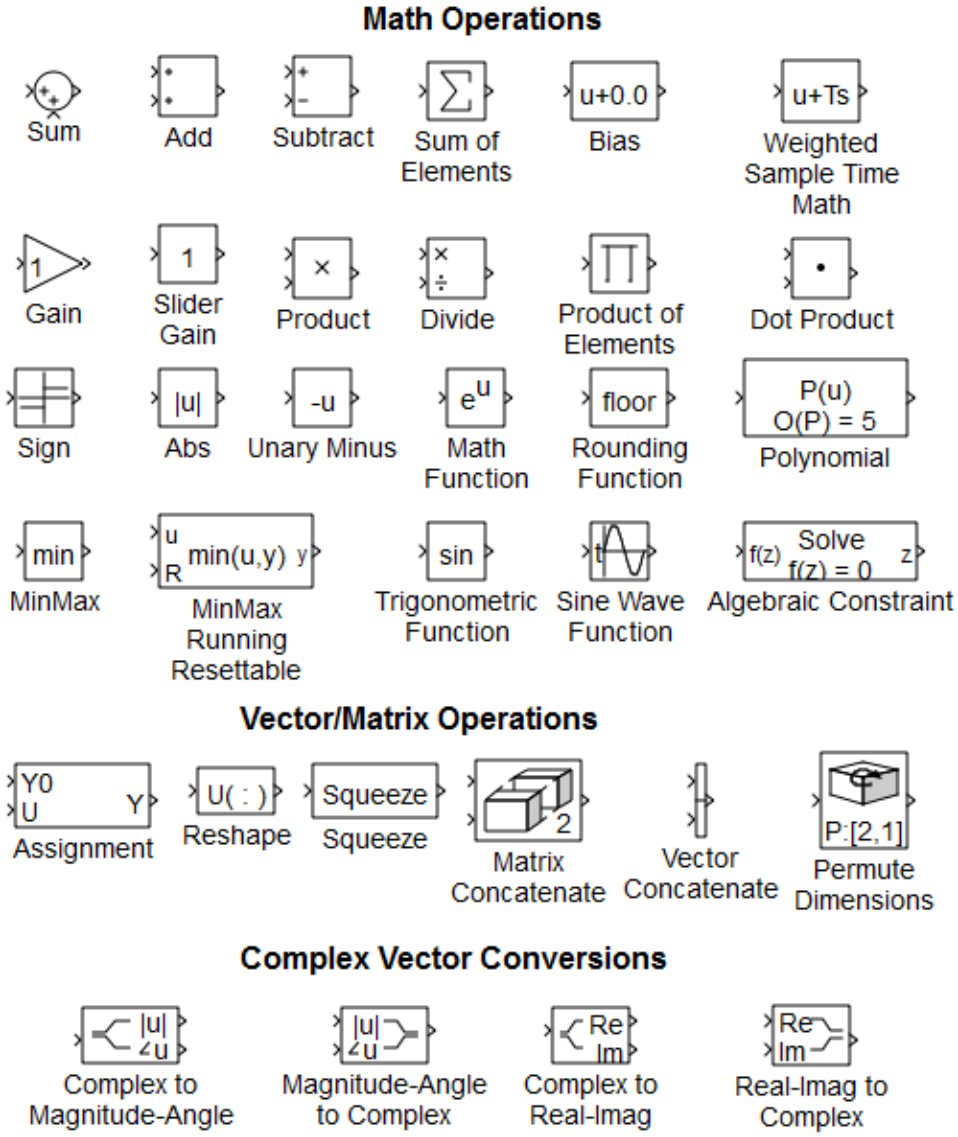


Рис. 6.1. Бібліотека математичних блоків *Math Operations*

Варіанти операцій, що можуть виконувати ці блоки, можна зрозуміти із розглядання рис. 6.2 та інформації щодо цього рисунку, наведеної у табл. 6.1.

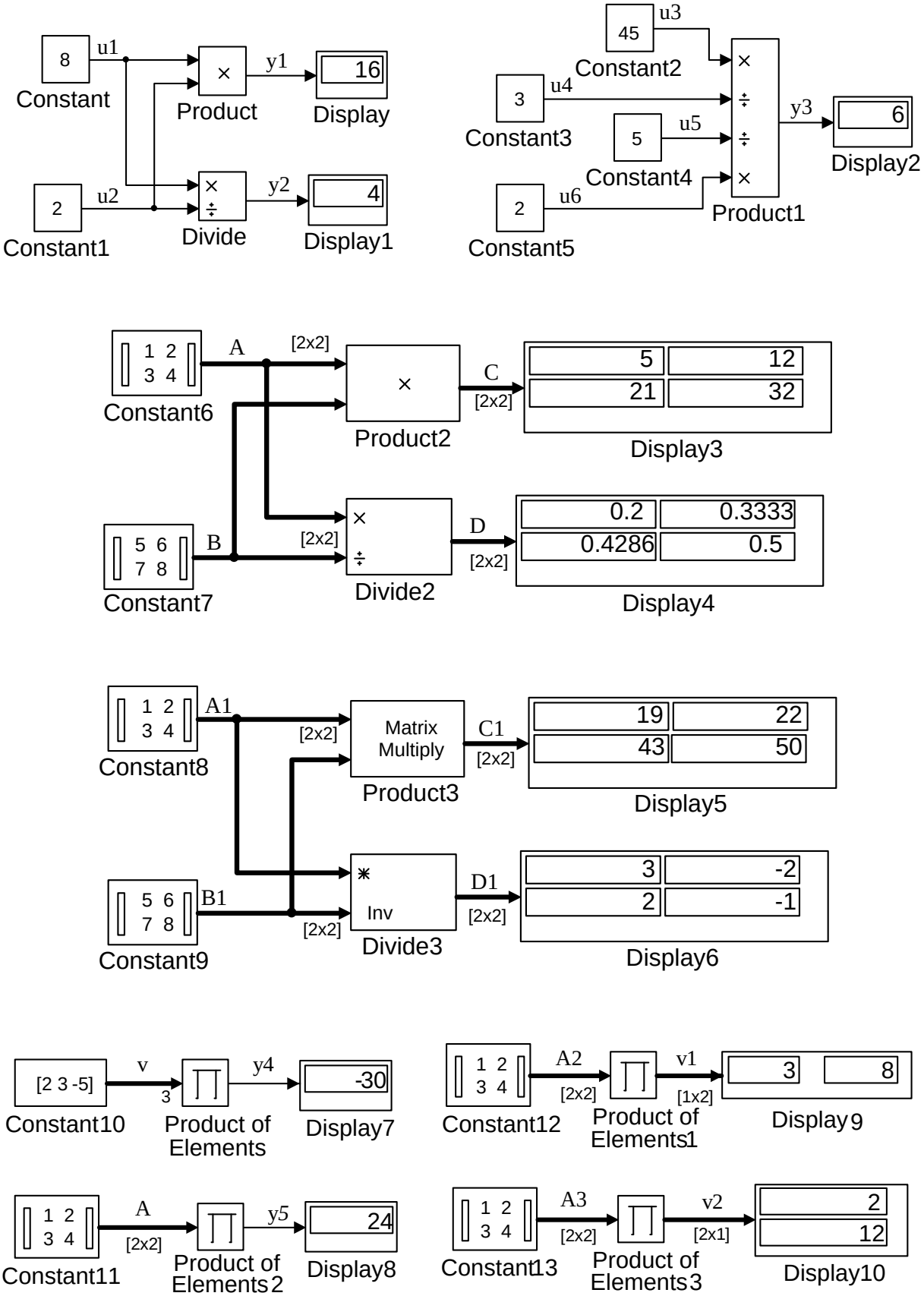


Рис. 6.2. Приклади використання блоків множення
та ділення у *Simulink*

Таблиця 6.1

Ім'я блока	*Number of inputs*	*Multiply*	*Multiplication over*	*Dimen-sion*	Операція MATLAB
Product	2	*Element-wise*	-	-	u1*u2
Divide	*/	*Element-wise*	-	-	u1/u2
Product1	*//*	*Element-wise*	-	-	u1/u2/u3*u4
Product2	2	*Element-wise*	-	-	A.*B
Divide2	*/	*Element-wise*	-	-	A./B
Product3	2	*Matrix*	-	-	A*B
Divide3	*/	*Matrix*	-	-	A/B=A*inv(B)
Product of elements	1 або *	*Element-wise*	*All Dimensions*	-	prod(v)
Product of elements1	1 або *	*Element-wise*	*All Dimensions*	-	prod(prod(A))
Product of elements2	1 або *	*Element-wise*	*Specified Dimension*	1	prod(A)
Product of elements3	1 або *	*Element-wise*	*Specified Dimension*	2	prod(A')

Блок ***Dot Product* (*скалярний добуток*)** обчислює скалярний добуток векторних вхідних сигналів **u** та **v** однакового розміру:

$$y = \mathrm{dot}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^{\mathrm{T}}\mathbf{v} = \sum_{i=1}^{n} u_i v_i \,. \qquad (6.1)$$

Складається з послідовно з'єднаних блока *Product* із двома входами і блока *Sum* з одним входом. Якщо вхідні сигнали – скаляри, то обчислюється їхній добуток.

6.3. Блоки з нелінійними статичними характеристиками, заданими аналітично

Нелінійні статичні характеристики, задані аналітичними виразами, в *Simulink* можна реалізувати блоками функціональними блоками *Abs*, *Sign*,

Sqrt, Signed Sqrt, Reciprocal Sqrt, Math Function, Trigonometric Function і *Polynomial* з бібліотеки *Math Operations* (див. рис. 6.1) і блоками бібліотеки *User defined functions*, наведеною на рис. 6.3.

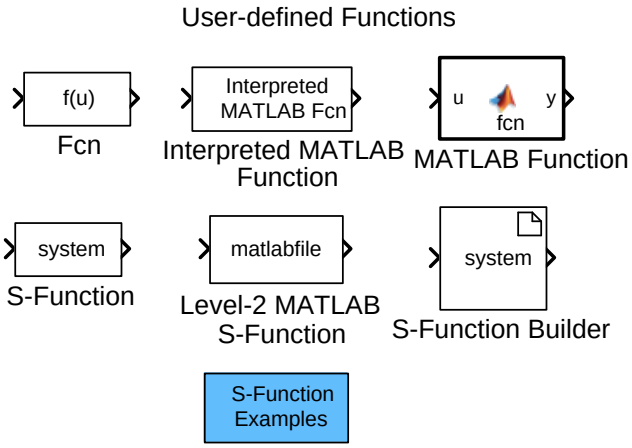


Рис. 6.3. Бібліотека *User defined functions*

Виходом ланки ***Abs* (*модуль*)** є абсолютне значення входу: $y = |u|$.

Блоки ***Sqrt, Signed Sqrt,*** та ***Reciprocal Sqrt*** – це однакові блоки, які в залежності від вибору параметру *Function*, який може приймати значення *Sqrt*, *ReSqrt* та *SignSqrt* виконують такі операції:

$$y = \sqrt{u},$$
$$y = 1/\sqrt{u}. \qquad (6.2)$$

та

$$y = \sqrt{|u|} \cdot sign(u). \qquad (6.3)$$

Блок ***MinMax* (*мінімум* / *максимум*)** визначає мінімальний або максимальний (за вибором користувача) із вхідних сигналів, кількість яких задається **параметром *Number of input ports*.**

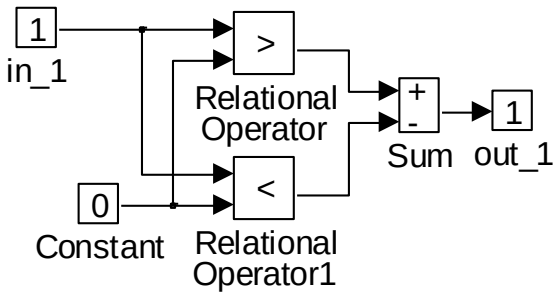


Рис. 6.4. Структура блока *Sign*

Блок ***Sign* (*знакова функція*)** формує вихідний сигнал, величина якого визначається знаком вхідного сигналу:

$$y = \begin{cases} 1 & \text{при} \quad u > 0, \\ 0 & \text{при} \quad u = 0, \\ -1 & \text{при} \quad u < 0. \end{cases} \qquad (6.4)$$

Блок ***Trigonometric Function (тригонометрична функція)*** відтворює одну із елементарних тригонометричних (*sin*, *cos*, *tan*), зворотних тригонометричних (*asin*, *acos*, *atan*, *atan2*), гіперболічних (*sinh*, *cosh*, *tanh*), та зворотних гіперболічних функцій (*asinh*, *acosh*, *atanh*) від вхідних сигналів шляхом вибору значення параметра *Function*.

Блок ***Polynomial* (*поліном*)** обчислює значення степеневого полінома, в якому незалежною змінною є вхідний сигнал u, а вектор коефіцієнтів $\mathbf{A} = [a_n, a_{n-1}, ..., a_1, a_0]$ задається у полі параметра *Polynomial coefficients*:

$$P_n(u, \mathbf{A}) = a_n u^n + a_{n-1} u^{n-1} + ... + a_1 u + a_0, \qquad (6.5)$$

що відповідає виконання в MATLAB оператора y=polyval(A,u).

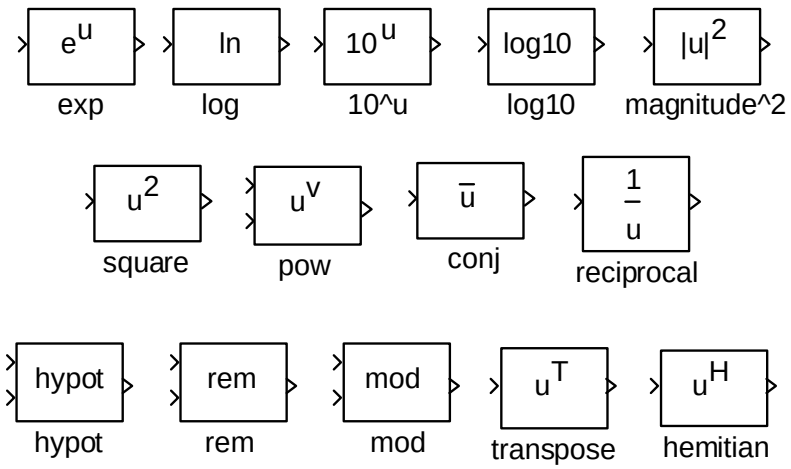


Рис. 6.5. Блоки *Math Function*

Блок ***Math Function (математична функція)*** може відтворювати елементарні одно- та двохаргументні функції від вхідних сигналів. Вибір функції здійснюється через меню. Він впливає на вигляд піктограми блока. На рис. 6.5 наведені можливі різновидності цього блока. Для наочності імена блоків замінені іменами функцій меню. Двохаргументна функція *hypot (u,v)* здійснює операцію

$$y_i = \sqrt{u_i^2 + v_i^2}, \quad i = 1, 2, ..., n, \qquad (6.6)$$

функція *mod (u,v)* округляє результат ділення першого аргументу на другий до найближчого цілого з недостачею, а функція *rem (u,v)* – визначає остачу від цілочислового ділення цих аргументів. Інші функції не потребують пояснень.

Вихідний сигнал блока ***Fcn* (*функція*)** є аналітичною функцією, при побудові якої можуть бути використані такі компоненти:

- *u*[*i*] або *u*(*i*) – значення і-го вхідного сигналу (*u* без індексу еквівалентно *u*(1));
- числа;
- операції +, -, *, /, ^;
- дужки ( );
- елементарні функції *sin*, *cos*, *tan*, *asin*, *acos*, *atan*, *sinh*, *cosh*, *tanh*, *exp*, *ln*, *log10*, *sqrt*, *floor*, *ceil*, *abs*, *atan2*, *rem*;
- операції порівняння ==, ~=, >, <, >=, <=;
- скалярні неіндексовані змінні.

Блок ***Interpreted MATLAB Fcn* (*інтерпретована МАТЛАВ-функція*)** обчислює задану *MATLAB*-функцію (стандартну або створену користувачем) від вхідного сигналу *u* (скалярного або векторного). Цей блок є пасивним двобічним інтерфейсом між середовищами *Simulink* і *MATLAB*. Наприклад, якщо задана функція *sin*, то вихідний сигнал *y* буде обчислюватися по формулі *y=sin*(*u*). Якщо вхідних сигналів декілька, то вони позначаються як *u*(*1*), *u*(*2*), *u*(*3*) і т.д.

Блок *Interpreted MATLAB Fcn* підтримує векторні входи і виходи, які перетворюються у скалярні блоком *Demux*, а навпаки – блоком *Mux*. Кількість входів повинна збігатися з кількістю вхідних аргументів *MATLAB*-функції, а розмірність виходу (*Output width*) – з кількістю її вихідних аргументів. Якщо параметр *Output width* задано рівним -1, то *Simulink* установлює розмірність вихідного сигналу рівною розмірності вхідного сигналу.

Блок ***MATLAB Fcn* (*МАТЛАВ-функція*)** включає в *Simulink*-модель *C*-код сформованої користувачем *MATLAB*-функції, текстовий редактор для формування якої відкривається при подвійному щиглику мишею на іконці

блока.

6.4. Типові нелінійності та їх формування блоками Simulink

При моделюванні нелінійних систем використовують термін ***«типові нелінійності»*** стосовно кусково-лінійних характеристик з невеликою кількістю лінійних ділянок, що часто зустрічаються на практиці, як це вже було відзначено у підрозділі 6.1.

Серед вже розглянутих вище нелінійних залежностей до них відноситься функція $y = |u|$, що реалізуються блоком ***Abs*** та має характеристику, зображену на рис. 6.6а. У якості прикладу реального пристрою з такою статичною характеристикою можна навести діодний міст (див. рис. 6.6б).

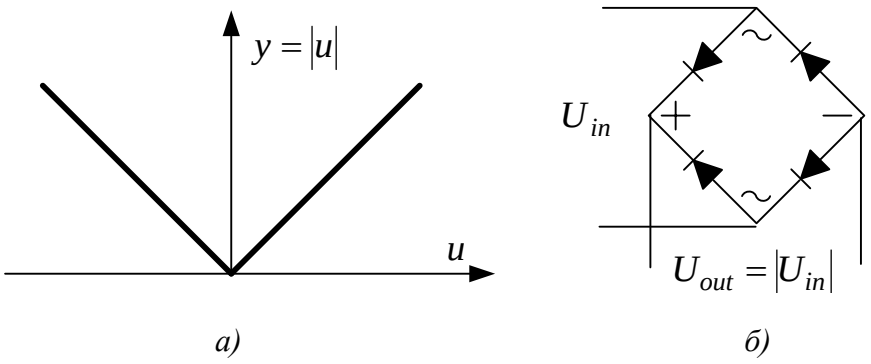


Рис. 6.6. Статична характеристика блока (*а*) та принципова схема діодного моста (*б*)

За допомогою функції $y = sign(u)$ (6.4), графік якої подано на рис. 6.7, можна описати типові нелінійності ***«Ідеальне реле»*** та ***«Сухе тертя»***.

Ідеальна релейна характеристика описується рівнянням

$$y = y_0 sign(u), \tag{6.7}$$

сила сухого тертя F_{fr} (*Coulombic friction*) для тіл з поступальним рухом – рівнянням

$$F_{fr} = F_0 sign(v), \tag{6.8}$$

а момент сухого тертя M_{fr} для тіл, що обертаються, –

$$M_{fr} = M_0 sign(\omega). \tag{6.9}$$

Отже, ***сухим (Кулонівським) тертям*** називають тертя, величина якого не

залежить від абсолютної величини швидкості (лінійної v або кутової ω). При зміні напряму швидкості сила або момент тертя також змінюють свій знак. Для прикладу на рис. 6.7 показано графік залежності моменту ідеального сухого тертя від кутової швидкості та модель сухого тертя, реалізовану за допомогою блока ***Sign***. Слід підкреслити неможливість перестановки у моделі сухого тертя блоків *Sign* та *Gain*.

Крім сухого тертя, у природі існує в'язке тертя, величина якого змінюється при варіації швидкості. При наявності і сухого, і в'язкого тертя, пропорційного швидкості, отримуємо ще одну типову нелінійність.

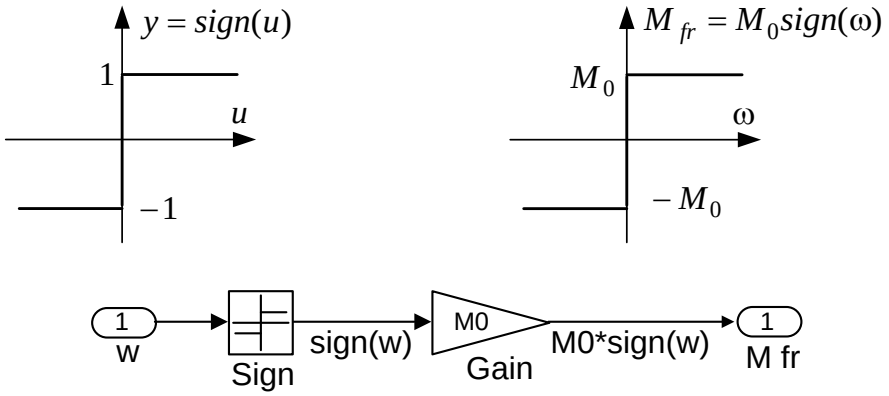


Рис. 6.7. Графіки функцій (6.7), (6.9) та модель сухого тертя

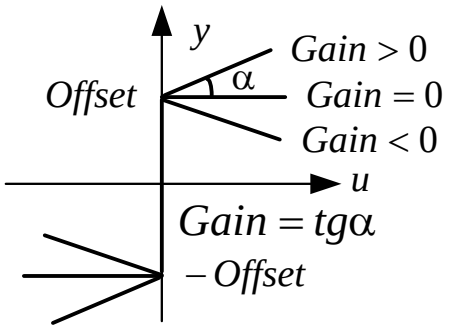


Рис. 6.8. Статичні характеристики блока *Coulombic & Viscous Friction*

Для її відображення в *Simulink* передбачено блок ***Coulombic & Viscous Friction*** **(*сухе та в'язке тертя*)** зі статичною характеристикою, поданою на рис. 6.8, яка описується рівнянням

$$y = (y_0 \pm K \cdot |u|) \cdot \text{sign}(u),$$

де y_0 – *Coulomb friction value, Offset* – (амплітуда Кулонівського тертя);

K – *Coefficient of viscous friction, Gain* (коефіцієнт в'язкого тертя).

При $K > 0$ блок реалізує ***додатне в'язке тертя***, що має місце, наприклад, при руху в рідинному або газовому середовищах, при $K < 0$ – ***від'ємне***

в'язке тертя, яке характерне для тертя ковзання між двома поверхнями, а при $K = 0$ – ***сухе тертя***.

При зміні знаку цього параметру змінюється і вигляд іконки блока (див. рис. 6.9).

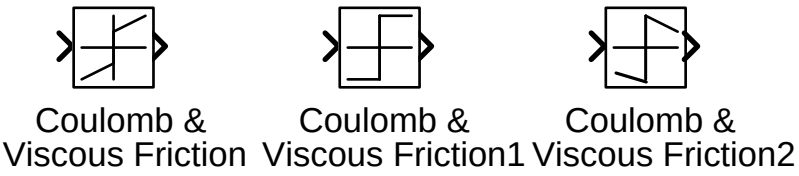
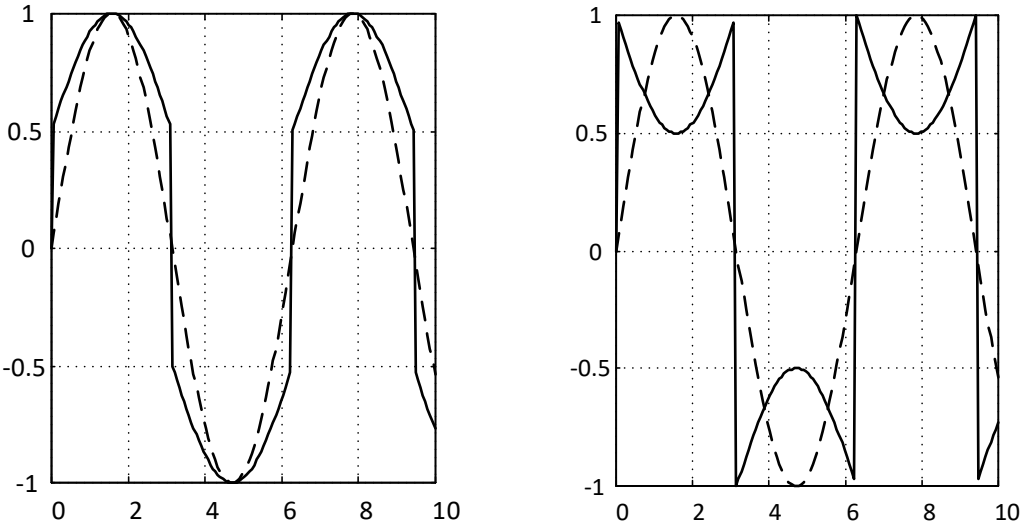


Рис. 6.9. Вигляд блоків *Coulombic & Viscous Friction* при різних значеннях параметру *Coefficient of viscous friction (Gain)*

На рис. 6.10 показано перехідні процеси перетворення синусоїди блоком *Coulumbic & Viscous Friction.*



а) *б)*

Рис. 6.10. Перетворення синусоїди ланкою *Coulumbic & Viscous Friction*: *а*) *Offset* = 0.5, *Gain* = 0.5; *б*) *Offset* = 1, *Gain* = -0.5

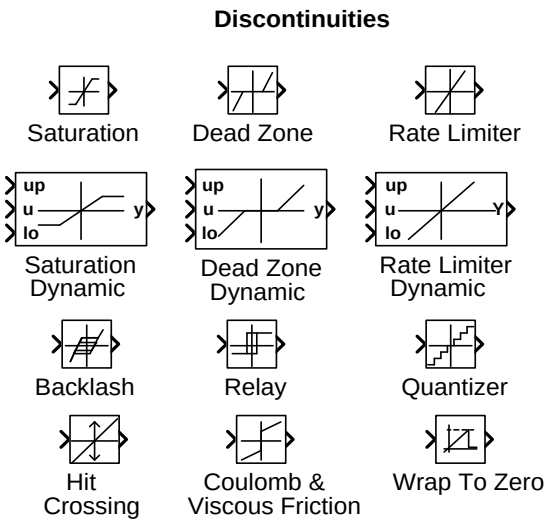


Рис. 6.11. Бібліотека *Discontinuities*

Блок *Coulombic & Viscous Friction*, як і більшість інших типових нелінійностей, знаходяться в бібліотеці ***Discontinuities***, представленій на рис. 6.11. Піктограми нелінійних блоків зображують загальний вигляд їх статичних характери-

стик.

Блок ***Saturation* (*Обмеження Координат*)** являє собою пропорційну ланку з одиничним коефіцієнтом підсилення, вихідний сигнал якої обмежений зверху на рівні U (*Upper limit*) і знизу – на рівні L (*Lower limit*):

$$y = \begin{cases} u & \text{при} \quad L \le u \le U, \\ U & \text{при} \quad u > U, \\ L & \text{при} \quad u < L. \end{cases} \tag{6.11}$$

Типова нелінійність ***«Обмеження координат»*** має надзвичайне велике поширення на практиці, тому що вихідні координати реальних пристроїв не можуть змінюватися до безкінечності. Прикладами таких пристроїв є операційні підсилювачі, тиристорні підсилювачі, різноманітні датчики. Найчастіше характеристики «Обмеження координат» є симетричними, але, на відміну від характеристики блока *Saturation*, коефіцієнт підсилення лінійної ділянки в них може бути довільним. Отже, для узгодження цих характеристик на вході блока *Saturation* треба встановити блок *Gain* з бажаним коефіцієнтом, як це показано на рис. 6.12.

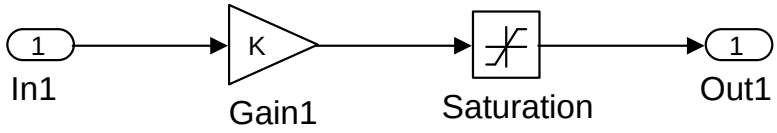


Рис. 6.12. Модель типової нелінійності *«Обмеження координат»*

Блок ***Dead Zone* (*зона нечутливості*)** має статичну характеристику, що описується формулою

$$y = \begin{cases} 0 & \text{при} \quad L \le u \le R, \\ u + L & \text{при} \quad u < L, \\ u - R & \text{при} \quad u > R, \end{cases} \tag{6.12}$$

де L, R – ліва (*Start of dead zone*) і права (*End of dead zone*) границі зони нечутливості відповідно. Ліву зону цієї ланки можна збільшувати до $-\infty$ (*-Inf*), а праву – до $+\infty$ (*Inf*). Єдине обмеження, що накладається на ці параметри –

це $L < R$.

Якщо треба змінити коефіцієнт підсилення лінійних ділянок, то блок *Gain* необхідно встановити на виході блока *Dead Zone* (див. рис. 6.13).

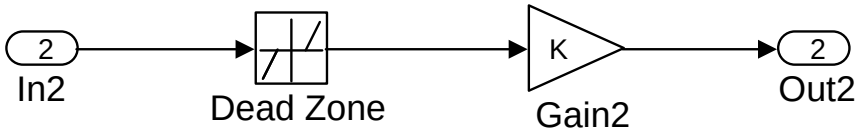


Рис. 6.17. Модель типової нелінійності *«Зона нечутливості»*

Процес перетворення синусоїдального сигналу ланками *Saturation* та *Dead Zone* відображено на рис. 6.14, 6.15.

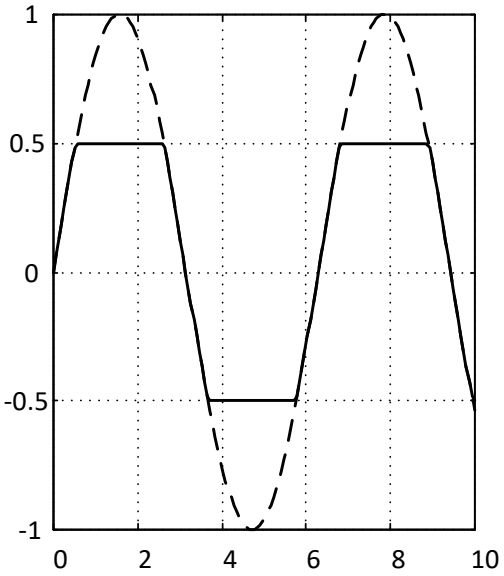


Рис. 6.14 – Перетворення синусоїди ланкою *Saturation*

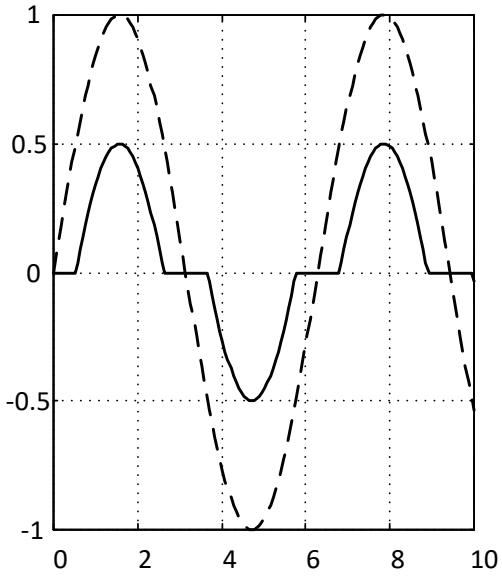
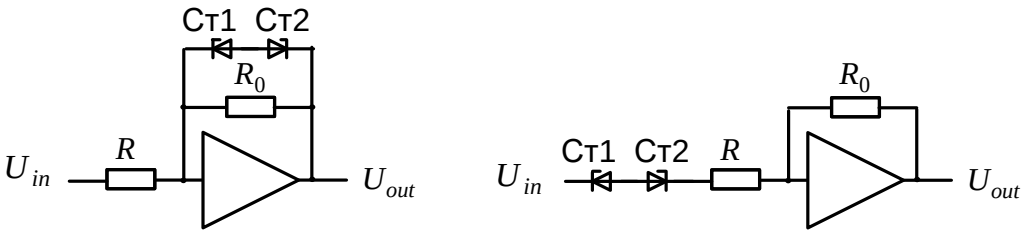


Рис. 6.15 – Перетворення синусоїди ланкою *Dead Zone*

Реалізація типових нелінійностей «Обмеження координат» та «Зона нечутливості» на операційних підсилювачах показана на рис. 6.16. Обмеження та нечутливість у цих схемах досягається завдяки відповідному включенню стабілітронів Ст1 та Ст2.



а) *б)*

Рис. 6.16. Реалізація типових нелінійностей «Обмеження координат» (*а*) та «Зона нечутливості» (*б*) на операційних підсилювачах

Щоб величини обмежень та межі зони нечутливості були варійованими, у схемах рис. 6.16 замінюють стабілітрони діодами з джерелами опорних напруг.

В залежності від параметрів характеристика «вхід-вихід» блока *Dead Zone* може змінюватися так, як це показано на рис. 6.17.

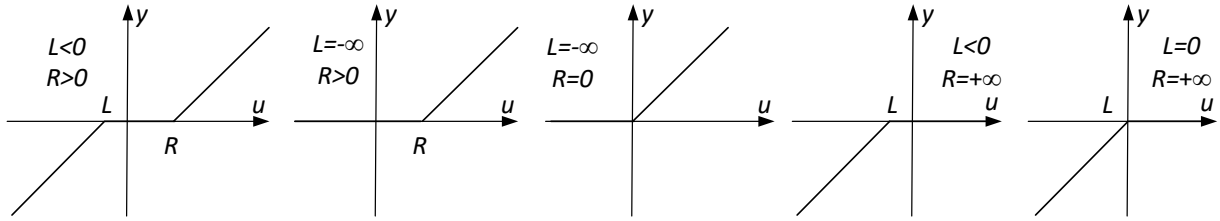


Рис. 6.17. Варіанти характеристик «вхід-вихід» блока *Dead Zone*

Для реалізації нелінійності «Зона нечутливості» з різними коефіцієнтами підсилення лінійних ділянок можна застосувати модель рис. 6.18.

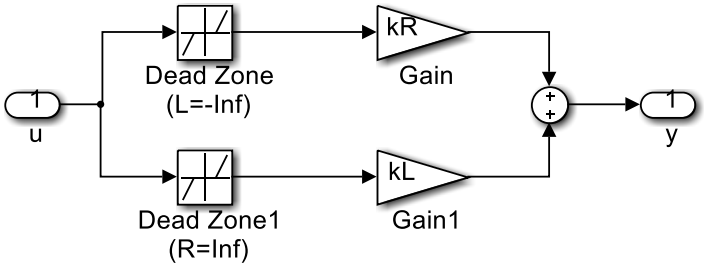
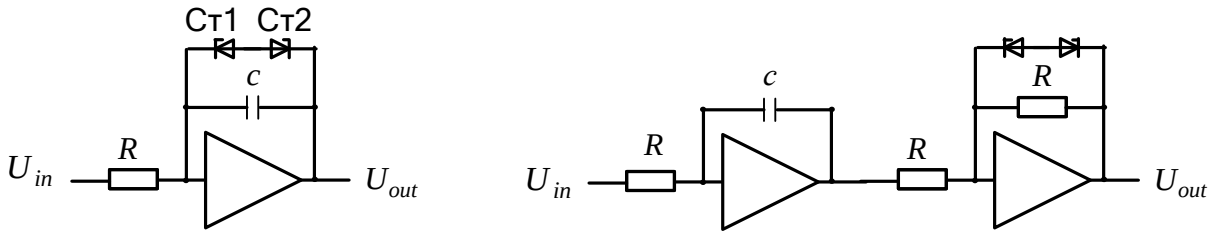


Рис. 6.18. *Simulink*-модель нелінійності «Зона нечутливості» з різними коефіцієнтами підсилювання гілок характеристики «вхід-вихід»

Зупинимося ще на особливостях обмеження динамічних ланок. Для початку розглянемо ***інтегратор з обмеженням вихідного сигналу***, принципова схема якого на операційному підсилювачі зображена на рис. 6.19а.



а) *б)*

Рис. 6.19. Принципові схеми інтегратора з обмеженням вихідного сигналу (*а*) та послідовного з'єднання необмеженого інтегратора і пропорційного підсилювача з обмеженням (*б*)

Одразу відзначимо, що ця схема не є еквівалентною схемі з послідовним з'єднанням необмеженого інтегратора і пропорційної ланки з обмеженням, показаній на рис. 6.19б.

Різниця полягає у тому, що у другій схемі навіть після обмеження сигналу U_{out} вихідний сигнал інтегратора продовжуватиме змінюватись. Тому модель інтегратора з обмеженням вихідного сигналу не можна побудувати як модель пропорційної ланки з обмеженням, зображену на рис. 6.12, в якій блок *Gain* замінюється блоком *Integrator*.

Для правильного обмеження вихідного сигналу інтегратора на заданому рівні треба при перетинанні ним цього рівня перевести блок з режиму інтегрування в режим збереження інформації підключенням до його входу замість сигналу, який інтегрується, нульової константи, як це показано на рис. 6.20. Повернення в режим інтегрування відбувається при зміні знаку вхідного сигналу.

Модель рис. 6.20 покладено в основу роботи блока *Integrator*, вікно параметрів якого подано на рис. 6.21.

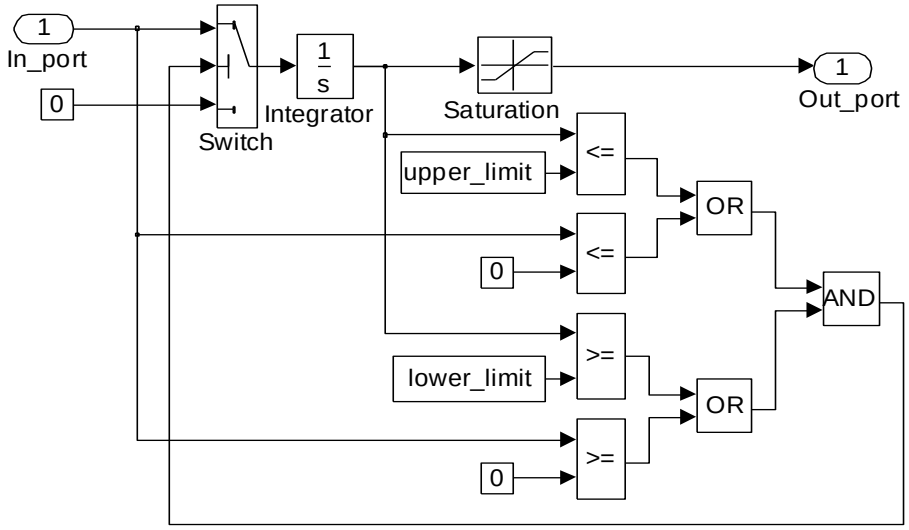


Рис. 6.20. Розгорнута модель інтегратора з обмеженням вихідного сигналу

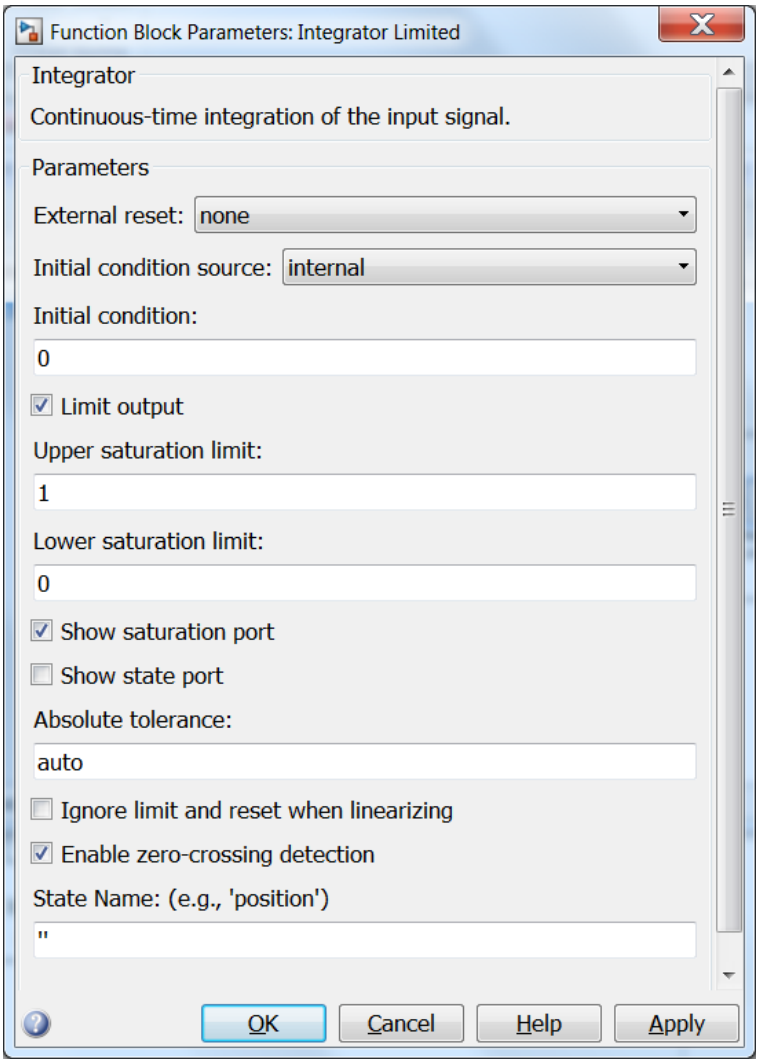


Рис. 6.21. Вікно параметрів блока *Integrator*

Для того, щоб почала працювати схема рис. 6.20, достатньо тільки

встановити прапорець у поле *Limit Output* та ввести значення параметрів *Upper saturation limit* і *Lower saturation limit*.

Після цього в іконку блока додається графічне зображення характеристики «Обмеження координат». Можна додатково візуалізувати порт обмеження, установкою прапорця *Show saturation port.* Конфігурації зовнішнього вигляду блока *Integrator* показані на рис. 6.22.

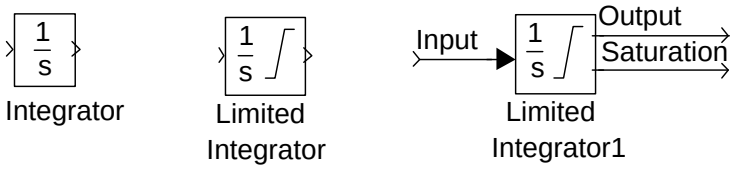


Рис. 6.22. Зовнішній вигляд блока *Integrator*

Інший підхід до моделювання інтегратора з обмеженням вихідного сигналу полягає в охопленні його від'ємним зворотним зв'язком за напругою стабілітронів, ідеалізована вольт-амперна характеристика яких являє собою «Зону нечутливості» з дуже великим коефіцієнтом підсилення лінійних ділянок, що прагне до безкінечності.

На рис. 6.23 подано схему для порівняльного аналізу розглянутих варіантів обмеження інтегратора, а на рис. 6.24 – результати моделювання.

З рис. 6.23 видно, що вихідні сигнали інтеграторів 2 і 3 змінюються майже однаково і при зміні знаку вхідного сигналу одразу виходять з режиму обмеження. При послідовному включенні інтегратора і ланки *Saturation* вихідний сигнал моделі залишається обмеженим, поки вихідний сигнал інтегратора не знизиться до рівня обмеження.

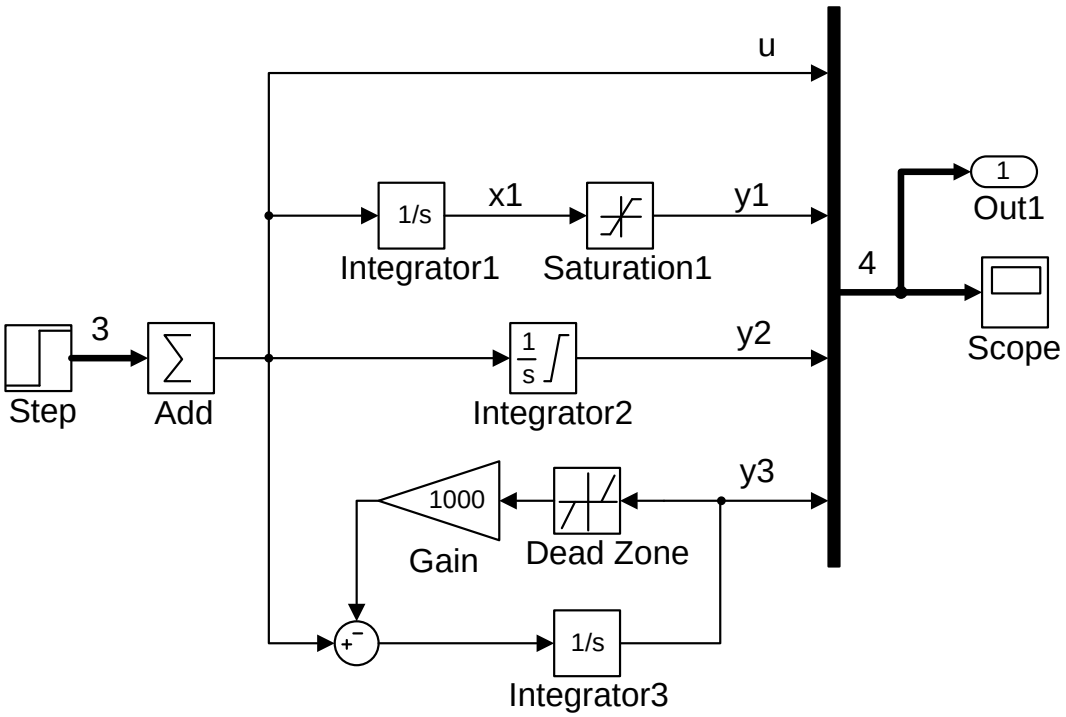


Рис. 6.23. Модель для порівняння способів обмеження інтеграторів

Для обмеження вихідних сигналів динамічних ланок довільного вигляду треба деталізувати їх передавальні функції і поряд з обмеженням вихідного сигналу ланки обов’язково обмежити інтегратори, що входять до складу деталізованої структурної схеми [3]. Рівні обмеження інтеграторів залежать від вигляду деталізованої структурної схеми.

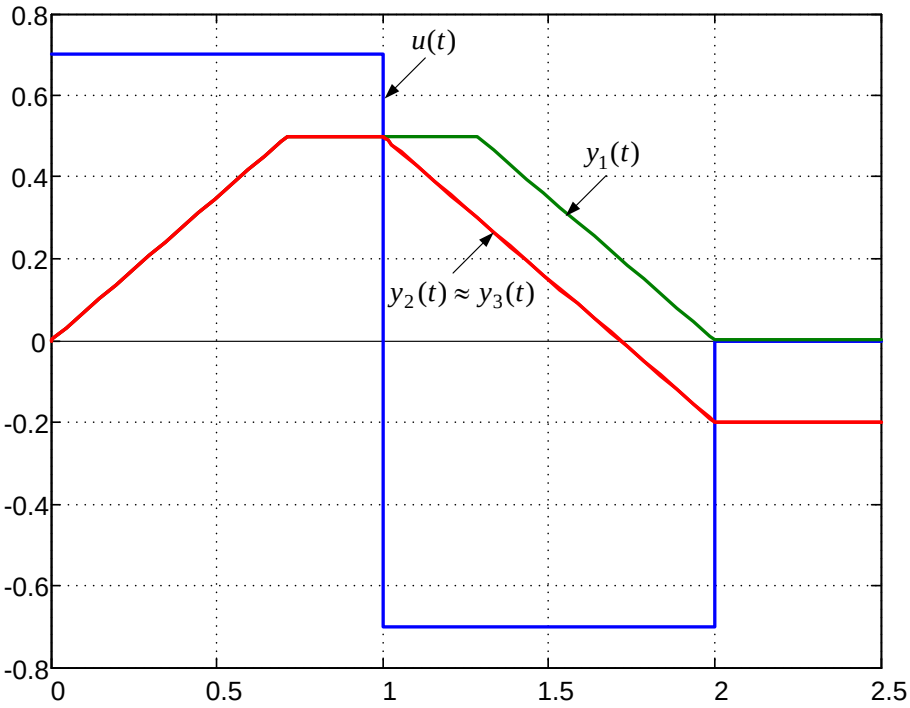


Рис. 6.24. Результати симуляції моделі рис. 6.23

Модель, що демонструє принцип обмеження вихідної координати ланки запізнювання на 1 період дискретності, що входить до складу дискретних інтеграторів, показана на рис. 6.25.

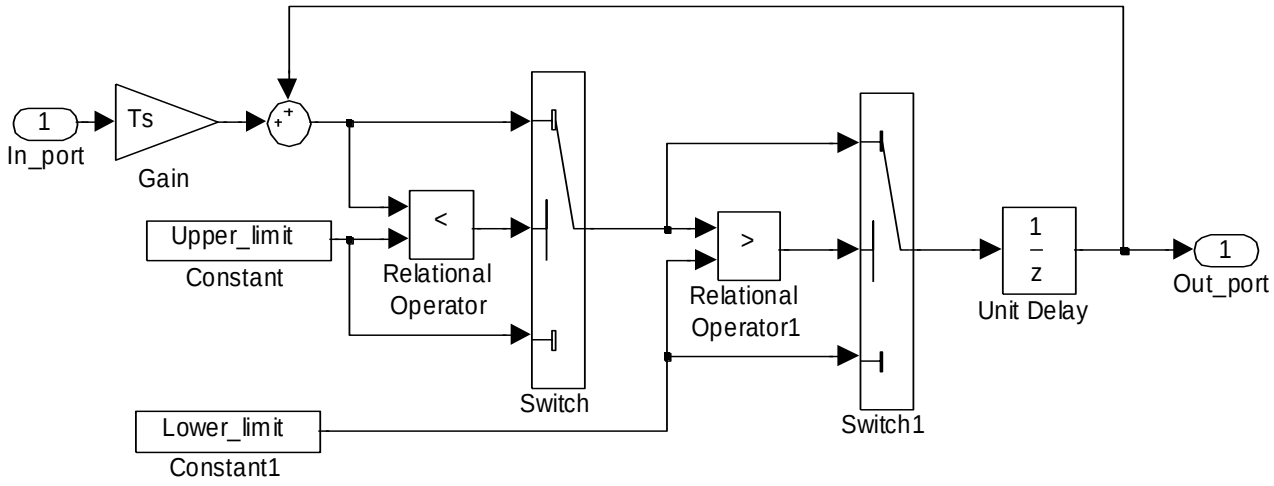


Рис. 6.25. Розгорнута модель ланки запізнювання на 1 інтервал дискретності з обмеженням вихідного сигналу

Блок ***Rate Limiter* (*обмеження темпу*)** відтворює вхідний сигнал *u(t)* з обмеженням його першої похідної *du/dt* на рівні *R (Rising slew rate)* при збі-

льшенні сигналу і *F (Falling slew rate*) – при його зменшенні:

$$y_i = \begin{cases} \Delta t \cdot \text{abs}(R) + y_{i-1} & \text{при} \quad \left|\frac{du}{dt}\right| > R, \\ -\Delta t \cdot \text{abs}(F) + y_{i-1} & \text{при} \quad \left|\frac{du}{dt}\right| < F, \\ u_i & \text{при} \quad F \le \left|\frac{du}{dt}\right| \le R. \end{cases} \tag{6.13}$$

Значення похідної обчислюється за формулою:

$$\frac{du}{dt} \approx \frac{\Delta u}{\Delta t} = \frac{u_i - u_{i-1}}{t_i - t_{i-1}}. \tag{6.14}$$

Модель та графік обмеження темпів наростання та спадання синусоїдального сигналу показано на рисунку 6.26, а стрибкоподібних – на рис. 6.27.

Отже, модель рис. 6.27а можна застосовувати для формування трапецоїдальних вхідних сигналів. Такі сигнали часто використовують в системах регулювання швидкості з обмеженням прискорення.

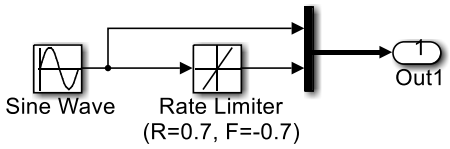
Ланка ***Backlash* (*зазор, люфт*)** моделює зазор (люфт) у кінематичній передачі:

$$y_i = \begin{cases} u - \frac{\delta}{2} & \text{при} \quad u - \frac{\delta}{2} > y_{i-1}, \\ u + \frac{\delta}{2} & \text{при} \quad u + \frac{\delta}{2} < y_{i-1}, \\ y_{i-1} & \text{при} \quad u - \frac{\delta}{2} \le y_{i-1} \le u + \frac{\delta}{2}, \end{cases} \tag{6.15}$$

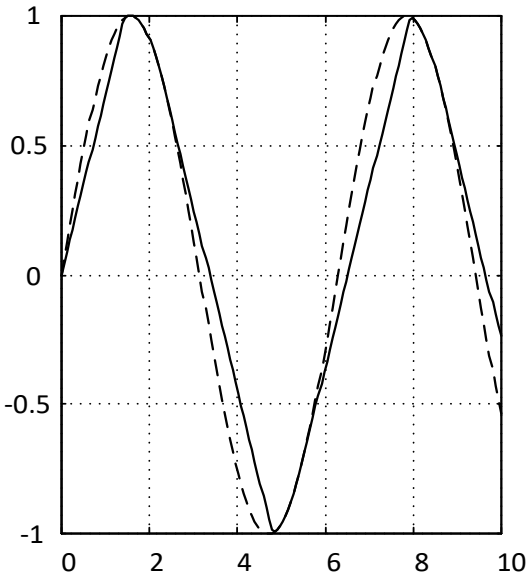
де δ – величина зазору (*Deadband width).*

Вхідним сигналом тут є положення активної (рушійної) маси, а вихідним – пасивної.

Статична характеристика блока *Backlash* та реакція його на синусоїдальний вхідний сигнал показані на рис. 6.28, 6.29 відповідно.

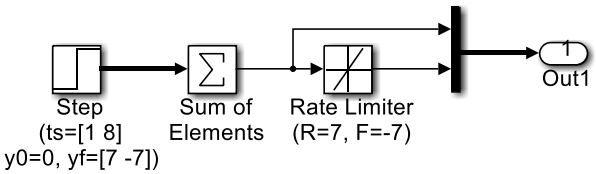


а)

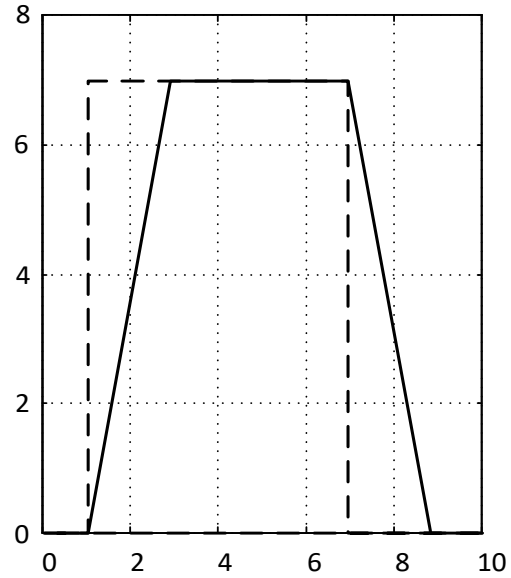


б)

Рис. 6.26. Модель(*а*) та перетворення(*б*) синусоїди ланкою *Rate Limiter*



а)



б)

Рис. 6.27. Модель (*а*) та перетворення (*б*) стрибкоподібних сигналів ланкою *Rate Limiter*

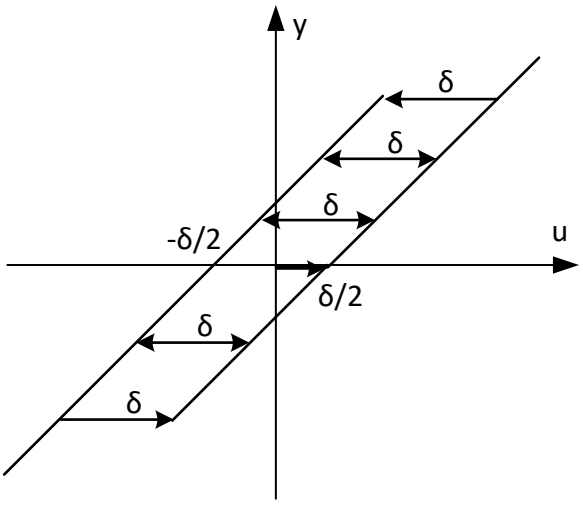


Рис. 6.28. Статична характеристика блока *Backlash*

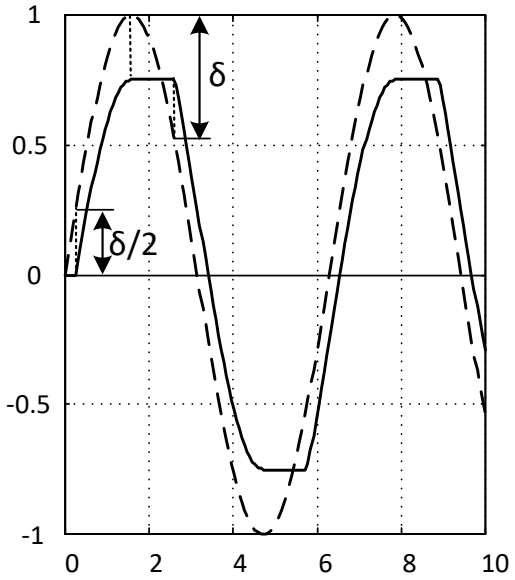


Рис. 6.29. Реакція блока *Backlash* на вхідну синусоїду

Установка початкових значень вхідного (*Initial input value*) і вихідного (*Initial output value*) параметрів дозволяють цілком визначити початковий стан системи. Початкове значення виходу повинне бути в межах 1/2 величини зазору; інакше *Simulink* сигналізує про помилку.

Блок ***Relay* (*реле з гістерезисом*)** дозволяє перемикати вихід між двома заданими значеннями *Output when on* (y_{on}) і *Output when off* (y_{off}). Коли реле включено, воно залишається в цьому стану доти, поки значення вхідного сигналу не упаде до заданого значення для вимикання реле *Input for off* (x_{off}). Коли реле вимкнено, воно залишається в цьому стану доти, поки значення входу не перевищить задане значення для включення реле *Input for on* (x_{on}). При $x_{on} > x_{off}$ моделюється реле з петлею гістерезису, а при $x_{on} = x_{off}$ – ідеальне реле (див. рис. 6.30).

Для того, щоб рівні обмеження та межі зони нечутливості можна було б задавати не тільки константами, а і змінними сигналами у бібліотеці *Discontinuities* передбачено 3 динамічних блоки: *Saturation Dynamic*, *Dead Zone Dynamic* та *Rate Limiter Dynamic*. Алгоритми роботи цих блоків можна зрозуміти, зазирнувши їм під маску (^*u*).

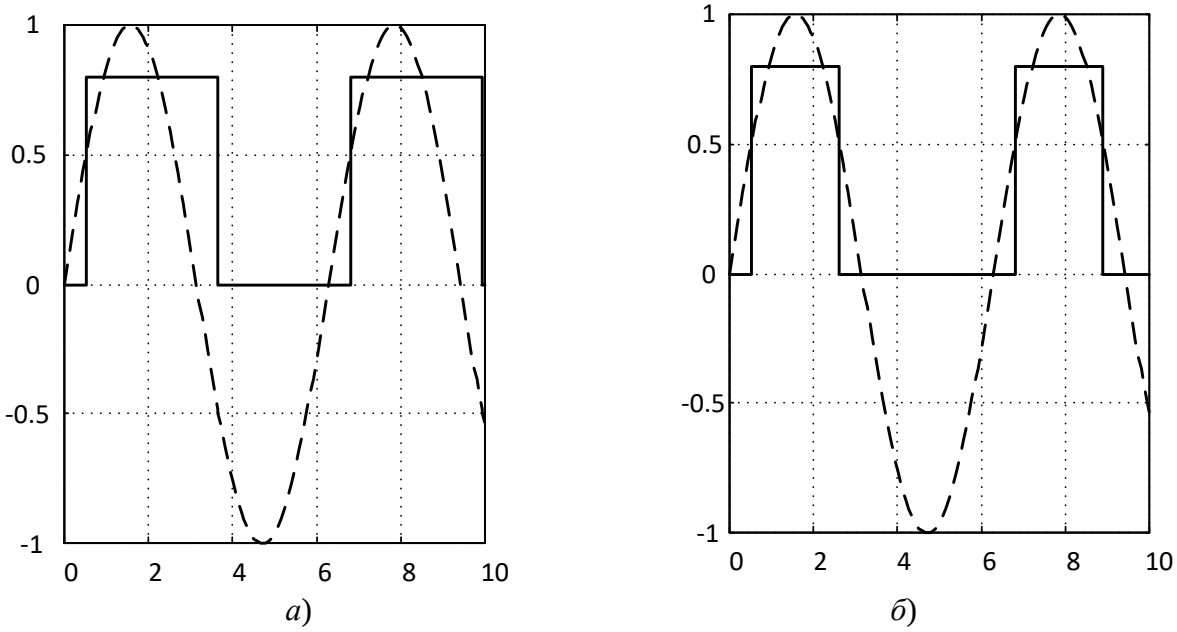


Рис. 6.30. Перетворення синусоїди ланкою *Relay*:
а) x_{on}= 0.5, x_{off} = -0.5; *б*) x_{on}= x_{off} = 0.5

На рис. 6.31, 6.32 зображені варіанти розгорнутих схем блока *Saturation Dynamic*, на рис. 6.33 – блока *Dead Zone Dynamic*, а на рис. 6.34 – блока *Rate Limiter Dynamic*.

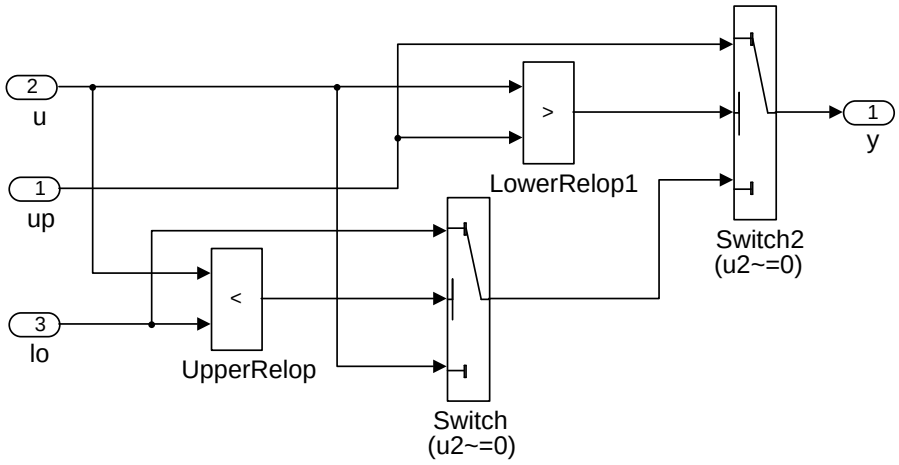


Рис. 6.31. Реалізація блока *Saturation Dynamic* з використанням операцій порівняння

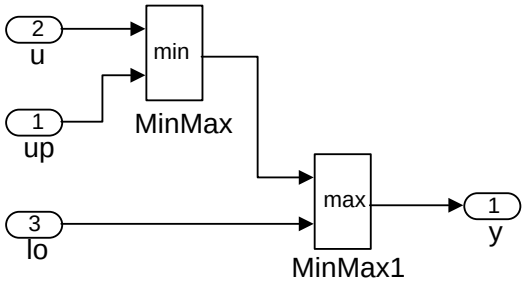


Рис. 6.32. Реалізація блока *Saturation Dynamic* з використанням блоків *MinMax*

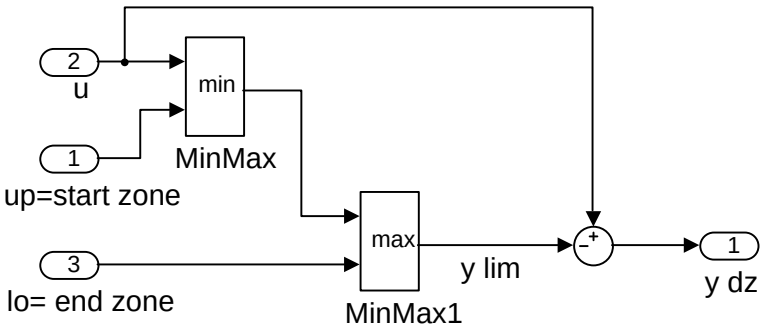


Рис. 6.33. Варіант реалізації блока *Dead Zone Dynamic*

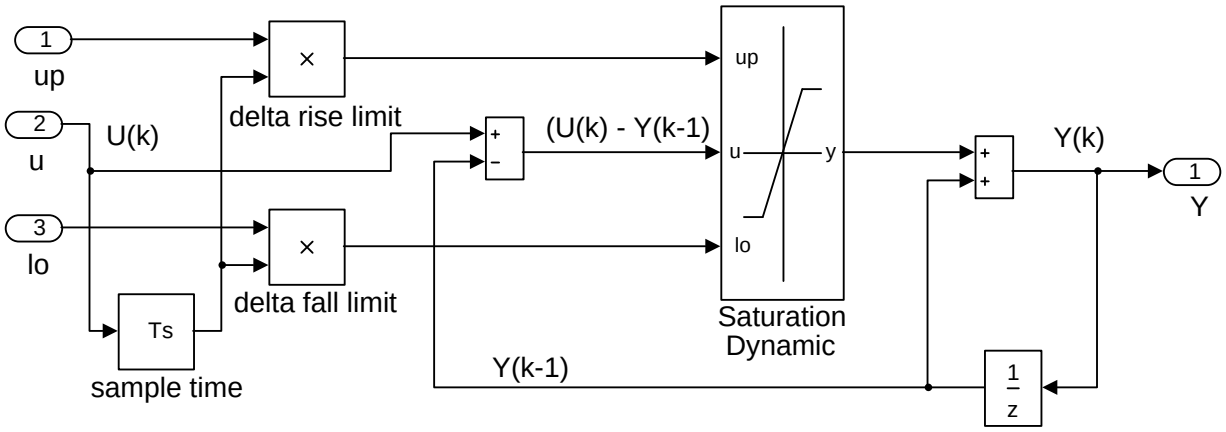


Рис. 6.34. Розгорнута структурна модель блока *Rate Limiter Dynamic*

6.5. Моделювання нелінійних функціональних перетворювачів, заданих у вигляді таблиць

У природі існують багато об’єктів з нелінійними статичними характеристиками «вхід-вихід», для яких не знайдено аналітичного опису. Зазвичай такі характеристики знімають експериментально. Інформація про них має вигляд таблиць, у яких кожному дискретному значенню вхідного сигналу відповідає певне значення вихідного сигналу. Отже, інформація про таку нелінійну залежність подається у вигляді двох одномірних масивів однакової довжини:

$$y_i = f(u_i); \qquad i=1, 2, ..., n. \tag{6.16}$$

Кожна пара чисел цих масивів визначає одну точку нелінійності. Абсциси точок таких залежностей називають вузлами таблиці. Задача полягає у розрахунку вихідних сигналів таких нелінійних залежностей у точках, відмінних від вузлових.

При розв’язанні цієї задачі використовують такі методи обчислювальної математики як інтерполювання та апроксимацію [12].

Інтерполювання рухомими степеневими поліномами різних порядків в пакеті MATLAB здійснює функція interp1*:*

y = interp1 (Ut, Yt, u, method),

де Ut, Yt – вектори аргументів та значень табличної функції;

u – точка або масив точок, у яких необхідно обчислити значення інтерпольованої функції y;

method – метод інтерполювання, може приймати наступні значення:

- 'linear' – лінійна інтерполяція (рухомими поліномами першого порядку);
- 'pchip' – кубічна інтерполяція (рухомими поліномами третього порядку);
- 'spline' – кубічна сплайн-інтерполяція (рухомими поліномами третього порядку, що забезпечують неперервність не тільки функції, але і її першої та другої похідних у вузлових точках);
- 'nearest' – інтерполяція за найближчим сусіднім вузлом (поліномами нульового порядку).

У *Simulink* ***інтерполяцію рухомими степеневими поліномами***, здійснює блок ***1D-Look-Up Table (одномірний функціональний перетворювач)*** з параметрами *Breakpoints 1* (табличні значення вхідного сигналу) та *Table data* (табличні значення вихідного сигналу)**.** Вектор аргументів має бути монотонно зростаючим.

Вікно введення параметрів цього блока зображено на рис. 6.35.

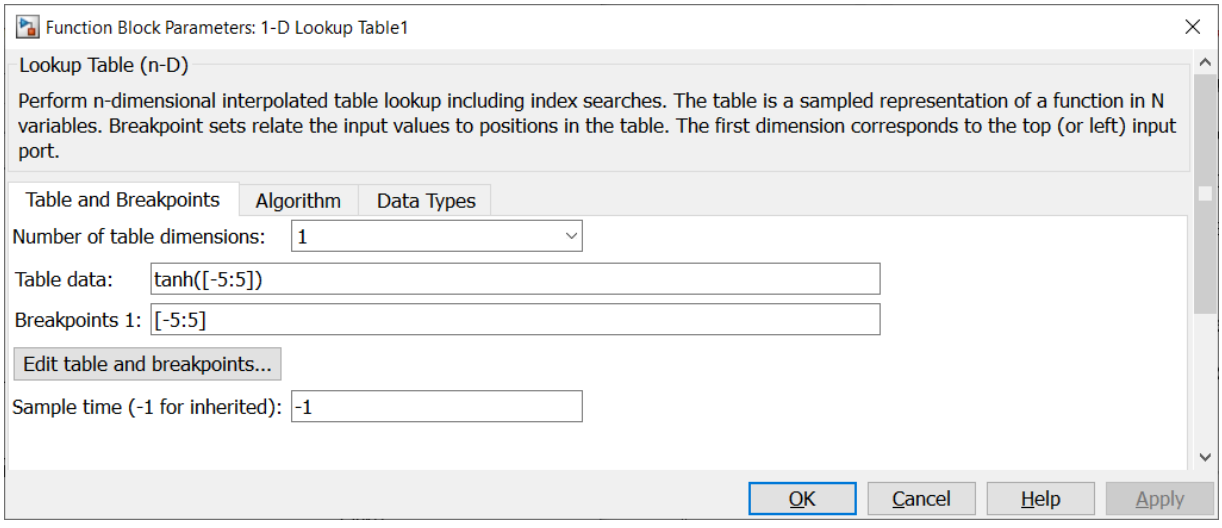


Рис. 6.35. Вікно введення параметрів блока *1D-Look-Up Table*

Цей блок утворено з блока ***nD-Look-Up Table (багатомірний функціональний перетворювач)*** за рахунок встановлення параметру ***Number of ta-***

ble dimensions (розмірність таблиці) рівним 1.

Тип інтерполювання обирається у вкладці *Algorithm* (див. рис. 6.36) з випадного меню *Interpolation method*, в якому в початковому стані встановлено метод *Linear*, що означає інтерполяцію рухомими поліномами першого порядку, яку ще називають ***кусочно-лінійною апроксимацією.***

У цьому випадку блок розраховує вихідний сигнал за формулою

$$y = \frac{y_i - y_{i-1}}{u_i - u_{i-1}} \cdot (u - u_{i-1}) + y_{i-1}, \tag{6.17}$$

де i – номер вузла, найближчого до поточного значення аргументу праворуч, тобто

$$u_{i-1} \le u \le u_{i-1}. \tag{6.18}$$

При виборі методу *Flat* виконується інтерполювання рухомими поліномами нульового порядку, прив'язаними до найближчої ліворуч вузлової точки (ступінчата апроксимація):

$$y(u) = y_{i-1}, \qquad u_{i-1} \le u \le u_i. \tag{6.19}$$

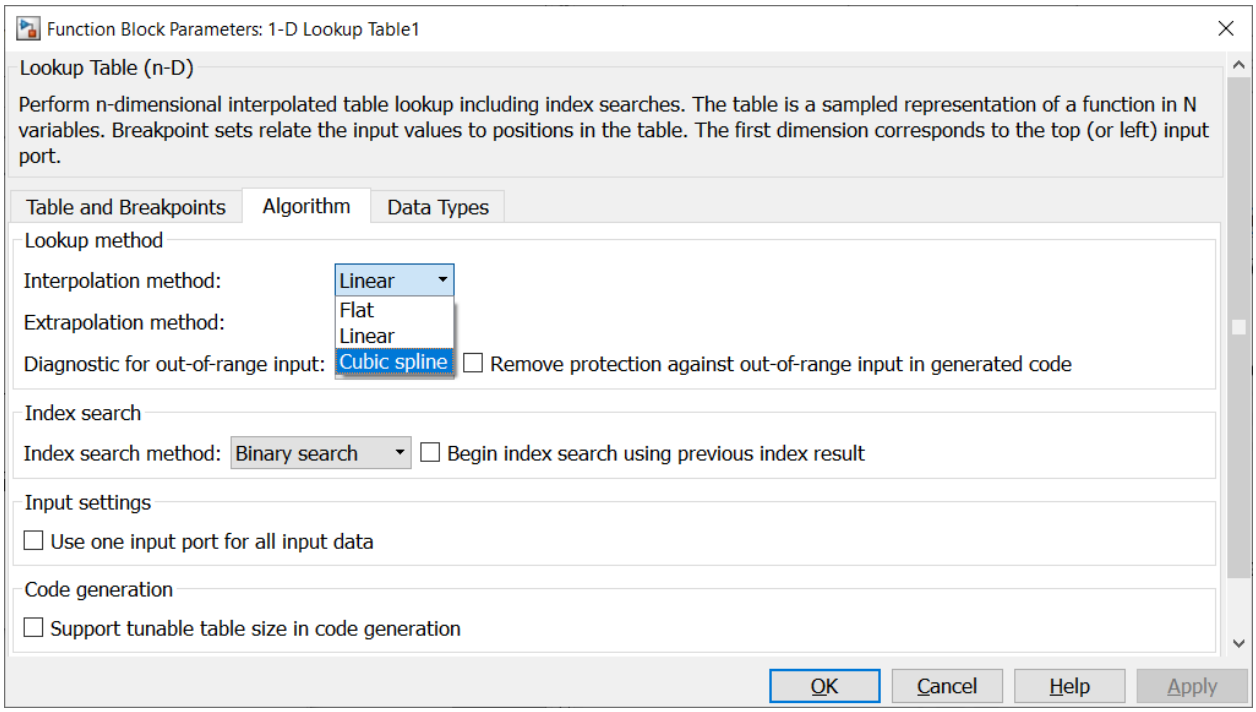
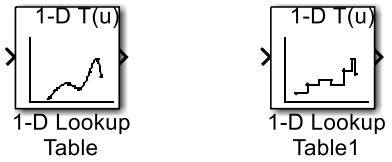


Рис. 6.36. Вкладка *Algorithm* вікна параметрів блока *1D-Look-Up Table*

При виборі методу *Cubic spline* виконується інтерполювання спеціальними рухомими поліномами третьої степені, що звуться кубічними сплайнами. Їх особливість полягає у тому, що вони забезпечують не тільки рівність значень табличної та апроксимуючої функцій у вузлових точках, а ще й неперервність першої та другої похідних у цих точках.

Вихідні значення для вхідних сигналів, що попадають за межі заданих у таблиці, виводять за допомогою екстраполювання по перших (екстраполювання ліворуч) або останніх (екстраполювання праворуч) табличних точках, кількість яких на одиницю перевищує порядок інтерполяції. Тип екстраполяції визначається параметром *Extrapolation method*, що може приймати значення: *Clip* (обрізати) – метод нульового порядку, *Linear* (продовжити лінійно) або *Cubic spline* (продовжити за рівнянням кубічного сплайна).

Піктограма блока *1D-Lookup Table* відображає графік залежності вхід-вихід тільки для кусково-лінійної апроксимації. У цьому випадку при зміні значень параметрів у діалоговому вікні блока графік автоматично перемальовується. Якщо значення параметру змінюється в робочому середовищі пакета *MATLAB*, то іконка блока перемальовується тільки після відкриття його діалогового вікна. При інтерполяції кубічними сплайнами та кусково-ступінчатій апроксимації іконка блока має вигляд, який не залежить від параметрів табличної функції (див. рис. 6.36).



а) *б)*

Рис. 6.36. Вигляд іконки блока *1D-Look-Up Table* при виборі методу інтерполяції *Cubic spline* (*а*) та *Flat* (*б*)

Недоліком інтерполяції кубічними сплайнами є складність визначення їх коефіцієнтів. Більш простим методом отримання доволі плавних функцій є

проста кубічна інтерполяція *Pchip*, яку можна реалізувати за допомогою блока *Interpreted MATLAB Fn*, звертаючись через нього до функції

interp1 (Ut, Yt, u, 'pchip').

Якщо звернутися у блоці *Interpreted MATLAB Fn* до функції

interp1 (Ut, Yt, u, 'nearest', 'extrap'),

то отримаємо ступінчату інтерполяцію поліномами 0-го порядку типу «найближчий сусід»:

$$y(u)=\begin{cases} y_{i-1}, & якщо\ (u-u_{i-1})<(u_i-u), \\ y_i, & якщо\ (u-u_{i-1})\geq(u_i-u); \end{cases} \qquad u_{i-1}\leq u\leq u_i, \tag{6.20}$$

яка дещо відрізняється від ступінчатої інтерполяції (6.19).

На рис. 6.37-6.40 подані графіки описаних вище варіантів інтерполяції та екстраполяції деякої нелінійної функції, заданої у вигляді таблиці на інтервалі $u\in[1,9]$.

Кружечками на графіки зображено вузлові точки табличної функції. Вхідний сигнал сформовано блоком *Clock*, а час моделювання дорівнює 10, отже поточне значення вхідного сигналу повністю охоплює інтервал інтерполяції і дає можливість здійснювати екстраполяцію.

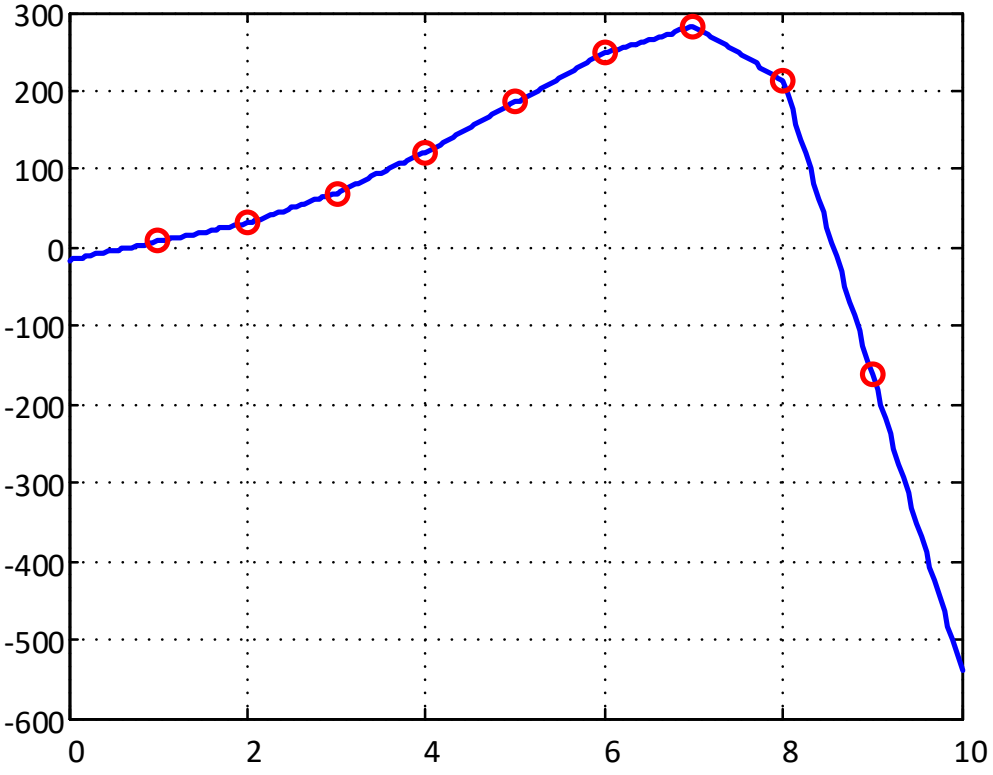
Апроксимація табличної функції (6.16) ***методом найменших квадратів*** полягає у знаходженні таких параметрів деякої аналітичної функції, які забезпечують мінімальну суму квадратичних відхилень апроксимуючої функції від заданої табличної у вузлових точках:

$$\Phi=\sum_{i=1}^{n}\left(F(u_i)-y_i\right)^2. \tag{6.21}$$

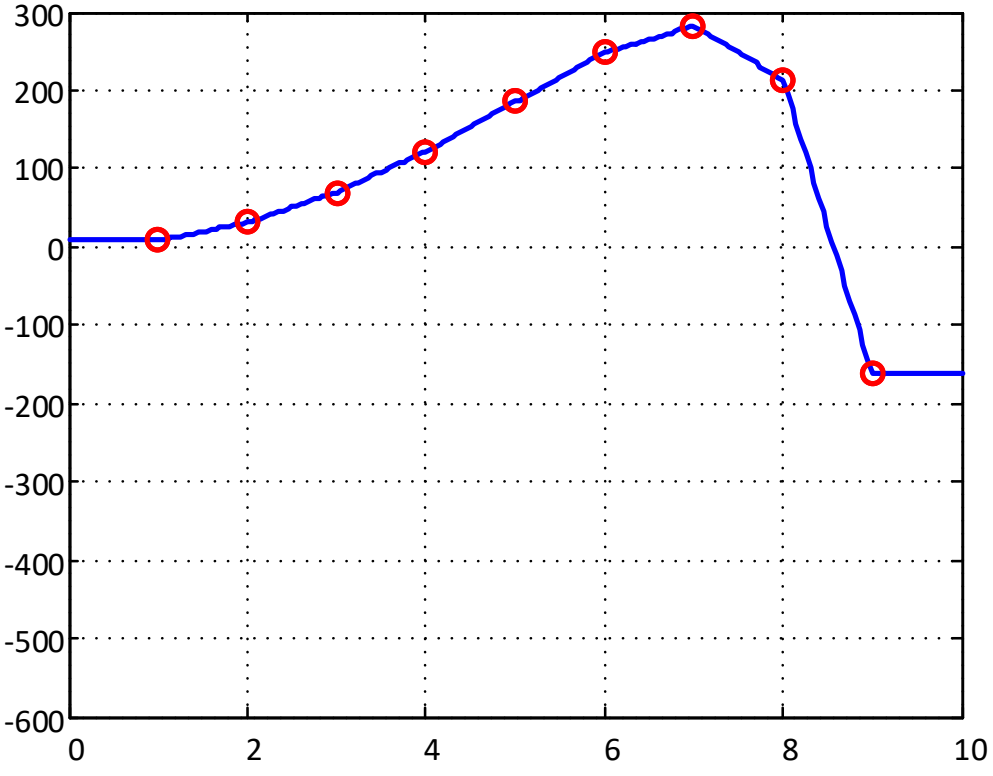
Якщо в якості функції, що апроксимує, взяти степеневий поліном

$$F(x)=P_k(x)=C_0+C_1x+\ldots+C_kx^k=\sum_{j=0}^{k}C_jx^j, \tag{6.22}$$

то задача зводиться до визначення вектору коефіцієнтів $\mathbf{C}=(C_0, C_1, \ldots, C_k)$.

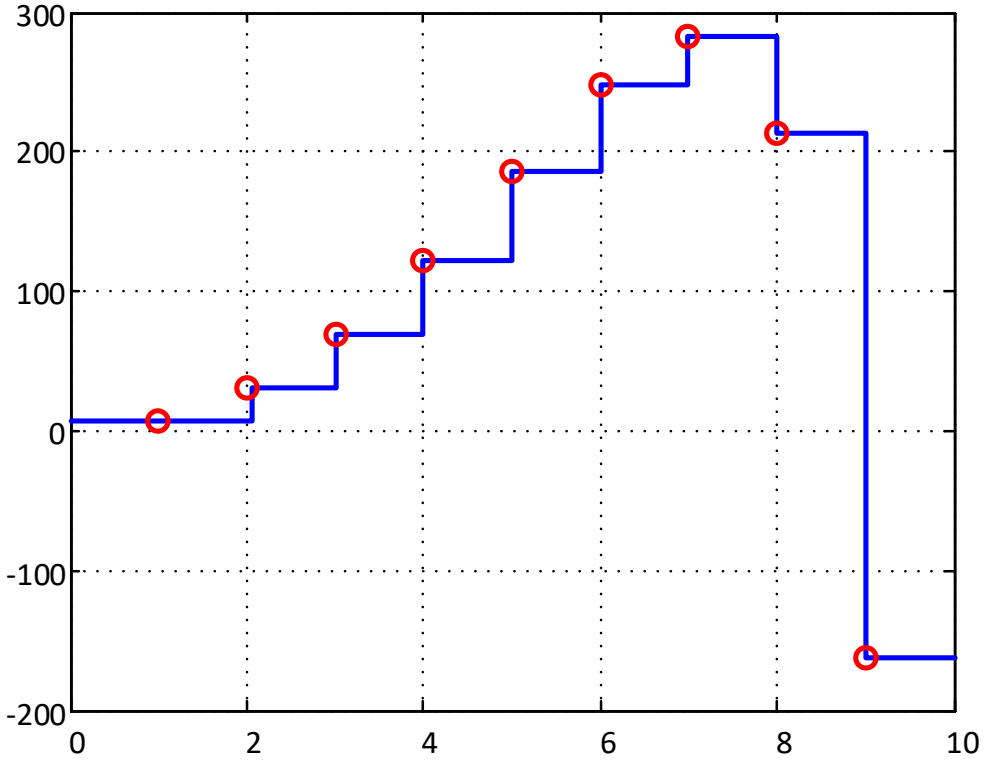


а)

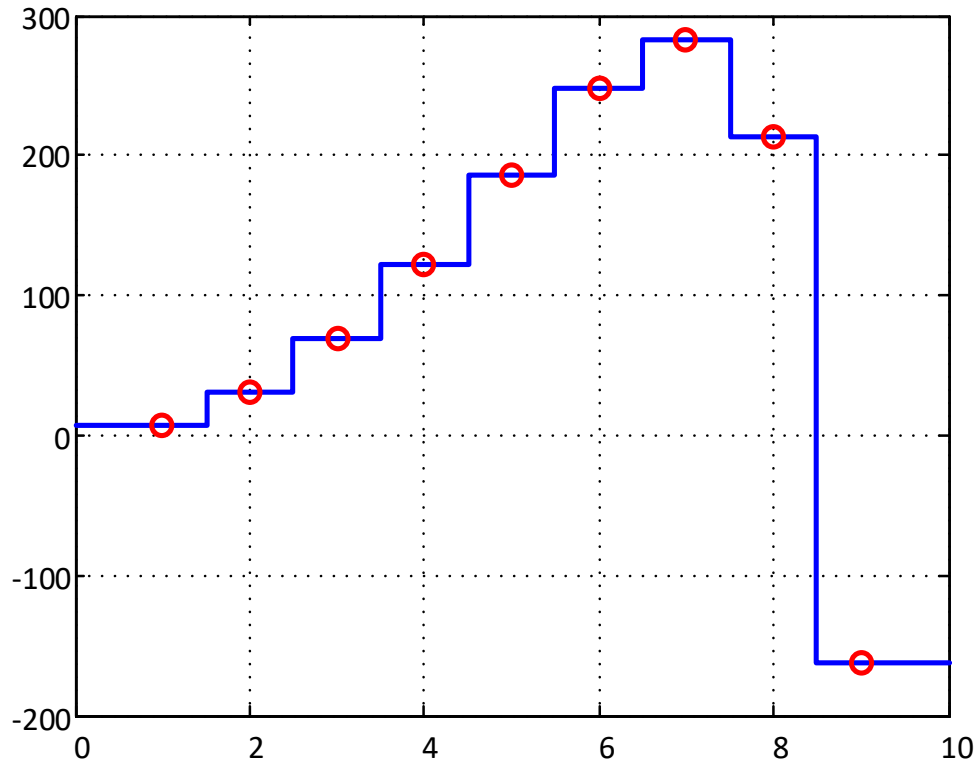


б)

Рис. 6.37. Графіки з виходу блока *1D-Look-Up Table* при *Interpolation method = Linear:*
а) Interpolation method = Linear; б) Interpolation method = Clip



а)



б)

Рис. 6.38. Кусково-ступінчата апроксимація, здійснена блоком *1D-Look-Up Table* при *Interpolation method = Flat* (*а*) та блоком *Interpreted MATLAB Fn* при зверненні interp1 (Ut, Yt, u, 'nearest', 'extrap') (*б*)

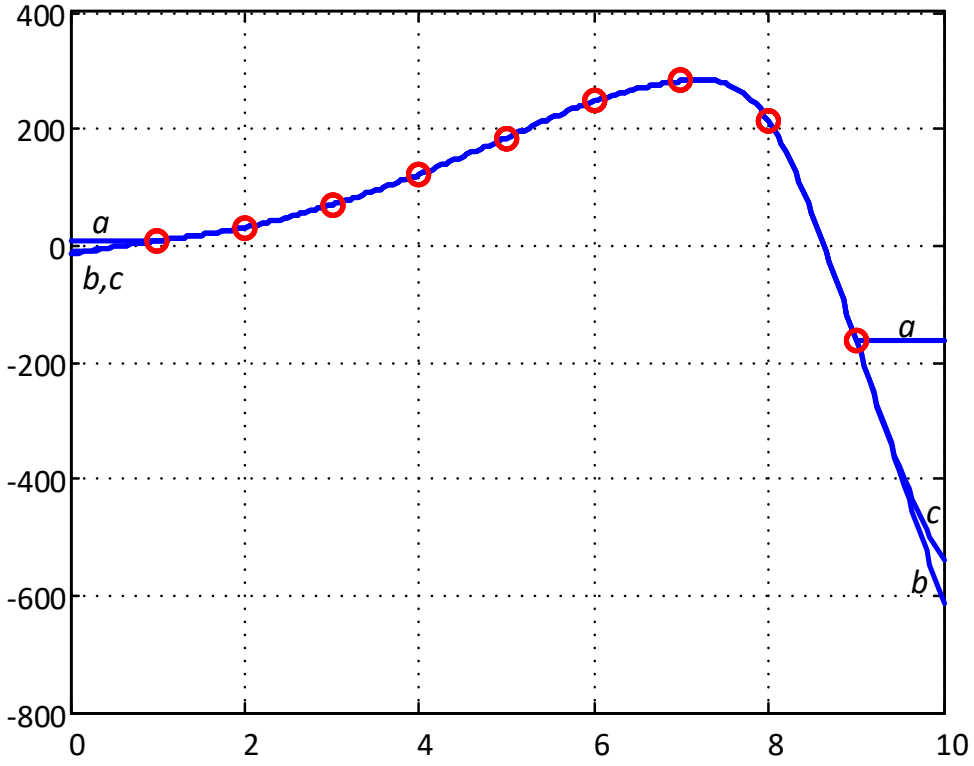


Рис. 6.39. Інтерполювання кубічними сплайнами, здійснене блоком *1D-Look-Up Table* при *Interpolation method = Cubic spline* та *Extrapolation method = Clip (a), Linear (b), Cubic spline (c)*

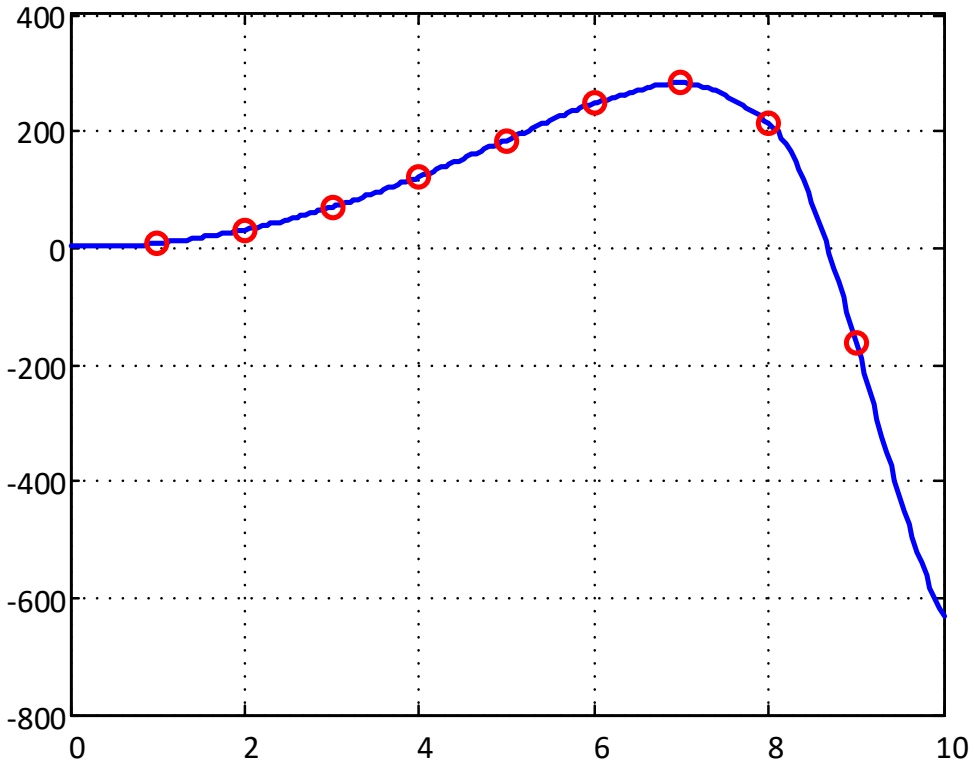


Рис. 6.40. Кусково-кубічне інтерполювання, здійснене блоком блоком *Interpreted MATLAB Fn* при зверненні
interp1 (Ut, Yt, u, 'pchip')

Цю операцію в пакеті *MATLAB* здійснює функція polyfit:

C = polyfit (Ut, Yt, k),

де C – вектор-рядок коефіцієнтів апроксимуючого полінома, упорядкований за зменшенням степені незалежної змінної x ;

k – порядок апроксимуючого полінома.

У пакеті *MATLAB* значення степеневого поліному з коефіцієнтами C у точках u вираховує функція

y = polyval (C, u),

де u – точки, у яких треба обчислити значення СП;

y – масив значень степеневого поліному, розмірність і розмір якого співпадає з відповідними параметрами вхідного аргументу u.

У *Simulink* останню операцію здійснює блок *Polynomial* (див. підрозділ 6.3).

Результати такої апроксимації поліномами 3-го, 4-го та 8-го порядків показані на рис. 6.41.

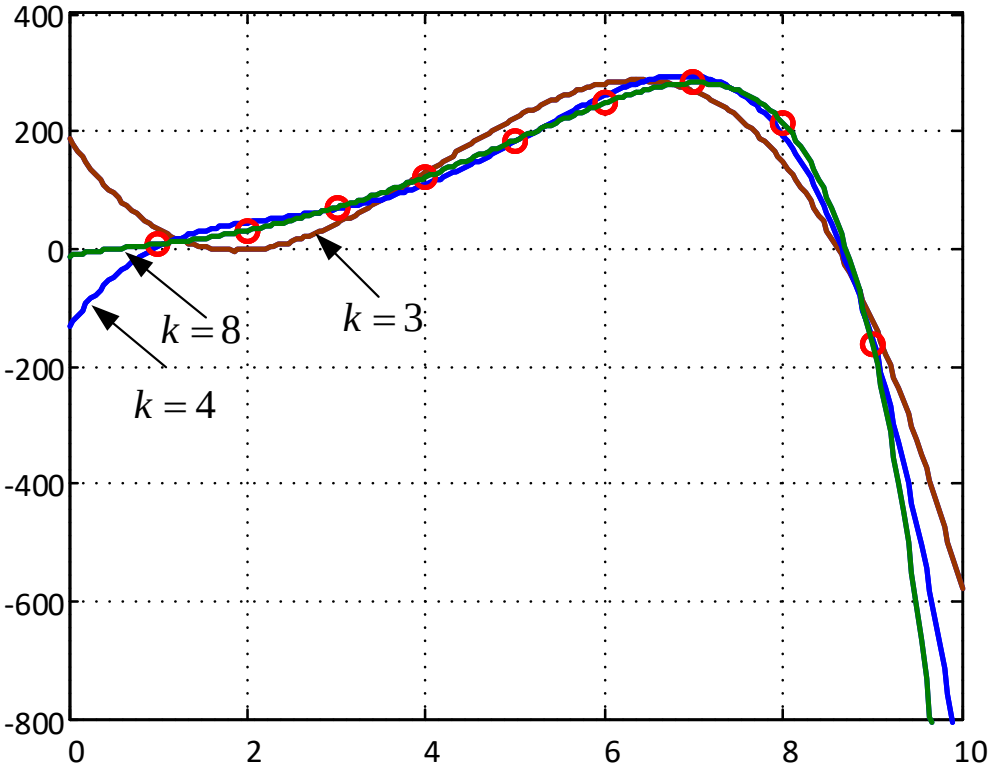


Рис. 6.41. Апроксимація табличної функції степеневими поліномами k-го порядку методом найменших квадратів

Як бачимо, при збільшенні порядку степеневого поліному k точність апроксимації підвищується. При $k = n$-1 (n – кількість табличних точок) апроксимуючий поліном стає глобальним інтерполянтом, тобто значення апроксимуючої функції у вузлових точках збігаються зі значеннями вихідної табличної функції.

6.6. Моделювання періодичних нелінійностей, заданих таблицями або аналітичними виразами на періоді

Якщо для повторюваної нелінійної залежності є відомим її математичний опис або таблиця значень абсцис і ординат на першому періоді

$$y = f(u), \quad 0 \le u \le \Delta u \tag{6.23}$$

або

$$y_i = f(u_i), \quad i = 1,2,...,n, \quad u_1 = 0, \quad u_n = \Delta u, \tag{6.24}$$

то для періодичного її повторення достатньо на вхід моделі, що формує залежність (6.23) або апроксимовані значення залежності (6.24), подати залишок від ділення вхідного сигналу u на його період Δu:

$$u_p = rem(u, \Delta u). \tag{6.25}$$

Прикладом такого підходу є блок *Repeating Sequence* бібліотеки вхідних блоків *Sources*, структуру якого можна побачити, зазирнувши під маску (див. рис. 5.9). Цей блок являє собою окремий випадок досліджуваної повторюваної послідовності, в якому вхідним сигналом є час, а закон зміни повторюваної послідовності на першому періоді утворює блок *1D Look-up-Table*. На рис. 6.42 наведено модель повторення залежності $y = (u-1)^2$ з періодом $\Delta u = 2$, а на рис. 6.43 – статична характеристика вхід-вихід цієї моделі.

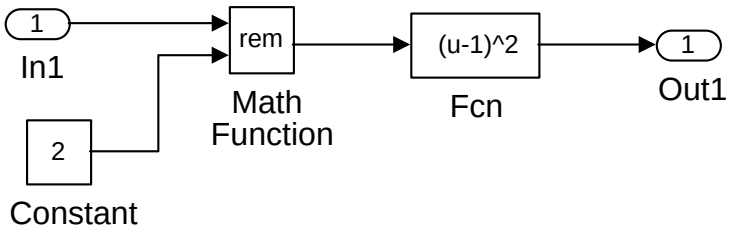


Рис. 6.42. Модель, що утворює повторювану послідовність парабол

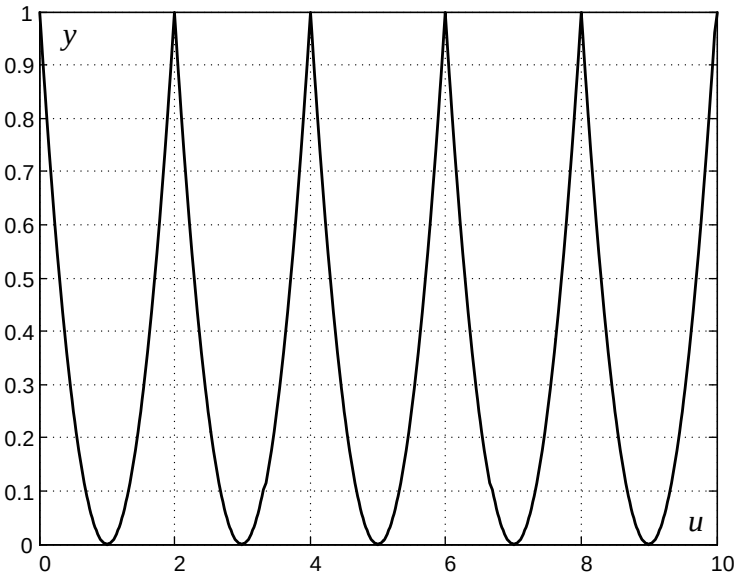


Рис. 6.43. Результат симуляції моделі рис. 6.32

Контрольні питання та завдання

1. Які типи нелінійних блоків Ви знаєте?
2. В яких бібліотеках знаходяться блоки, що роблять систему нелінійною?
3. Які математичні операції можуть виконувати блоки *Product*, *Divide* та *Product of Elements*?
4. Як можна реалізувати нелінійні статичні характеристики, визначені аналітично?
5. Які функції може виконувати блок *Math Function*?
6. Які функції може виконувати блок *Trigonometric Function*? Чим він відрізняється від блока *Sine Wave*?
7. Які операції може виконувати блок *Sqrt*?
8. Яку функцію виконує блок *Polynomial*? Як розрахувати вектор коефіцієнтів цього блока?
9. Які можливості блока *Fcn*?
10. Чим відрізняються один від одного блоки *Interpreted MATLAB Fn* та *MATLAB Fn*?

11. Які нелінійні характеристики називають типовими нелінійностями? Наведіть їх перелік з прикладами застосування.
12. Яка різниця між сухим та в’язким тертям?
13. Як реалізувати обмеження координат реалізувати зони нечутливості та ланки з неодичним коефіцієнтом підсилення?
14. Що обмежує блок *Rate Limiter*? Складіть модель, що складається з блоків *Rate Limiter, Step* та *Scope*. Наведіть і поясніть результати моделювання.
15. Яке призначення блоків *Saturation Dynamic* та *Dead Zone Dynamic*? Наведіть модель і результати обмеження синусоїди за лінійним законом.
16. Як обмежують вихідні сигнали інтегратора та неперервних динамічних ланок?
17. Як можна реалізувати нелінійні статичні характеристики, визначені таблицями?
18. Як здійснити апроксимацію табличної нелінійності степеневим поліномом методом найменших квадратів?
19. Як скористуватись функціями інтерполювання при моделюванні нелінійних функцій, заданих у вигляді таблиць?
20. Як можна організувати періодичне повторення аналітичної функції?

7. МОДЕЛЮВАННЯ СИСТЕМ ЗІ ЗМІННОЮ СТРУКТУРОЮ

Системи керування зі змінною структурою досить часто використовуються в сучасних системах електроприводу. Зміна структури може бути зумовлена особливістю технологічного процесу або особливістю алгоритму керування, що стосується насамперед адаптивних систем.

Зміна структури моделі в програмі *Simulink* може здійснюватися за допомогою ключів та використанням підсистем з бібліотеки *Subsystems*, що підключаються чи вимикаються в залежності від виконання деякої умови. Для керування ключами і підсистемами досить часто використовують логічні блоки та блоки зрівняння, що містяться у математичній бібліотеці *Math*, і блок *Hit crossing* з бібліотеки *Signals & Systems*.

7.1. Блоки, що здійснюють операції порівняння та логічні операції

Операції порівняння та логічні і бітові операції здійснюються в *Simulink* блоками, розташованими в бібліотеці ***Logic and bit operations (Логічні та бітові операції)***, вміст якої подано на рис. 9.1. Розглянемо ретельніше деякі з них.

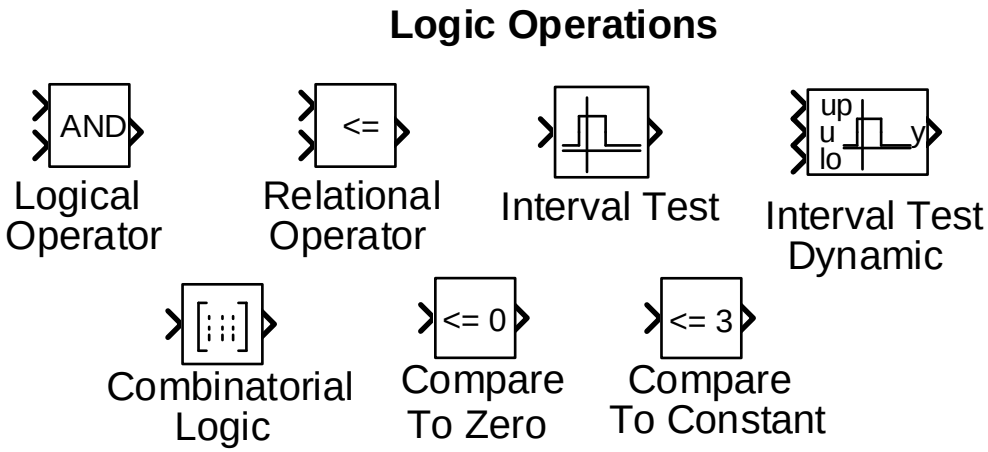


Рис. 7.1. Блоки бібліотеки *Logic and bit operations*, що здійснюють логічні операції

Блок ***Relational Operator*** **(*операції порівняння*)** виконує зазначену в полі параметра *Operator* операцію порівняння:

< – менше;

<= – менше або дорівнює;

>= – більше або дорівнює;

> – більше;

== – дорівнює;

~= – нерівно.

Результатом є одиничний (істинний) або нульовий (хибний) сигнал.

Блок ***Compare To Zero (порівняння з нулем)*** виконує обрану операцію порівняння вхідного сигналу з нулем.

Блок ***Compare To Constant (порівняння з константою)*** виконує обрану операцію порівняння вхідного сигналу з константою, визначеною параметром *Constant Value*.

Блок ***Logical Operator*** **(*логічна операція*)** виконує одну з логічних операцій *and (&), nand (~&), or (|), nor (~|), xor, nxor (~xor)* з елементами, що надходять на вхідні порти, кількість яких задається в полі параметра *Number of Input Ports.* Операндами логічних операцій є істинний (1) та хибний (будь-які числа, крім 1, але найчастіше – 0). Результатом логічної операції можуть бути тільки 0 та 1. Операція *not* (~) допускає тільки один вхідний порт і має такий результат: ~0=1, ~1=0. Значення інших логічних операцій для двох вхідних сигналів подані в табл. 7.1.

Таблиця 7.1

u	v	u & v	u ~& v	u | v	u ~| v	u xor v	u ~xor v
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

Якщо у ланці *Logical Operator* для параметру *Icon shape* вибрати значення *distinctive*, то іконка блока буде змінювати свій вигляд при зміні логічної операції згідно з IEEE-стандартом графічних символів для позначення відповідних функцій на логічних схемах, як це показано на рис. 7.2.

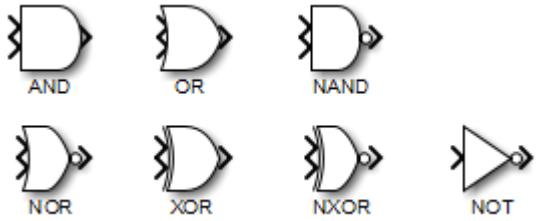


Рис. 7.2. Піктограми блока *Logical Operator* для різних логічних операцій

Блоки ***Interval Test*** та ***Interval Test Dynamic*** з параметрами *Upper limit* (U_{ap}) та *Lower limit* (U_{low}) при встановлених прапорцях у полях параметрів *Interval closed on right* та *Interval closed on left* (закритий інтервал) виконують операцію

$$y = (u \le U_{up}) \,\&\, (u \le U_{low}). \qquad (7.1)$$

Якщо прапорці не встановлені (відкритий інтервал), то відповідна операція $\le$ замінюється операцією $<$.

Блок ***Combinatorial Logic*** **(*комбінаторна логіка*)** забезпечує перетворення вхідного сигналу відповідно до заданої таблиці істинності *Truth Table*, котра являє собою список можливих вихідних сигналів ланки. При завданні цієї таблиці необхідно дотримуватись таких правил:

- кількість стовпців дорівнює кількості виходів блока;
- кількість рядків дорівнює 2^n, де n – розмірність вхідного сигналу;
- нульове значення сигналу в таблиці трактується як «хибне», будь-яке ненульове – як «істинне»;
- входи таблиці вважаються заданими; зокрема при двоелементному вхідному сигналі варіація входів визначається як [0 0; 0 1; 1 0; 1 1].

Таблиця 7.1

u_1	u_2	y_1	y_2
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

Наприклад, при введенні як параметр блока матриці [0 1; 1 0; 1 1; 0 0] повна таблиця істинності матиме вигляд табл. 7.1.

За допомогою цього блока можна описати логіку роботи будь-якого пристрою або системи, якщо їх вхідні і вихідні дані можуть бути представлені у формі булевих величин (0 – «хибне», 1 – «істинне»). Отже, розглянуту ланку можна назвати моделлю кінцевого детермінованого автомата.

Досить часто при моделюванні систем з перемиканнями використовують блок ***Hit crossing* (*Перетин*)** бібліотеки *Discontinuities*, призначений для фіксації моменту перетину вхідним сигналом *u* заданого рівня *o* (*Offset Value*) в заданому напрямку: *rising* (*при наростанні*), *falling* (*при спаданні*) або *either* (*в обох напрямках*), що задається параметром *Hit crossing direction* (*напрямок перетину*). При включеному стані вихідного порту він формує в момент, коли різниця сигналів *u-o* змінює знак на протилежний, у відповідності з обраним напрямком перетину, одиничний імпульс. При *u=o* вихідний сигнал утримує одиничне значення. У всіх інших випадках вихідний сигнал блока дорівнює нулю.

Демонстраційна модель *hardstop* ілюструє використання блока *Hit crossing* для моделювання тертя з урахуванням різниці між тертям покою та тертям руху і симуляції процесу миттєвого стопоріння рухомого органу.

7.2. Блоки, що здійснюють перемикання в моделях

Бібліотека ***Signal Routing (Маршрутизація сигналів)***, вміст якої показано на рис. 7.3, має 3 ключових елемента: *Switch, Manual Switch* і *Multiport Switch.*

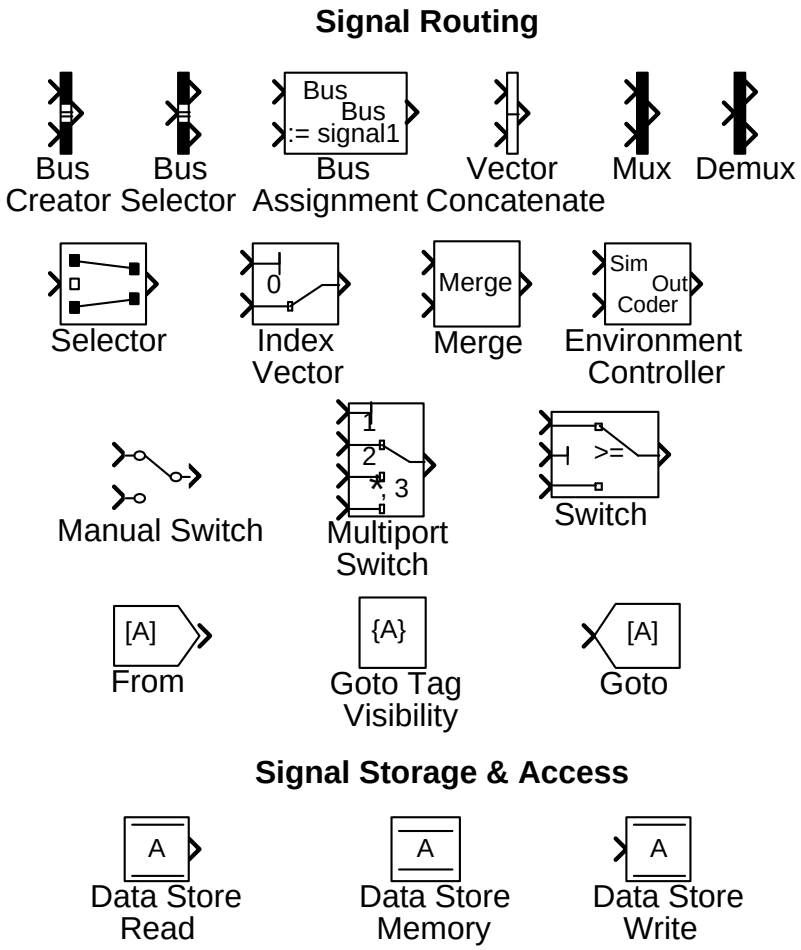


Рис. 7.3. Блоки бібліотеки *Signal Routing*

Блок ***Switch* (*керований ключ*)** пропускає на вихід один із двох вхідних сигналів, що подаються на перший і третій вхідні порти блока в залежності від стану керуючого сигналу, який надходить на другий порт, відповідно до алгоритму

$$y = \begin{cases} u_1 & при \quad u_2 \geq c \\ u_3 & при \quad u_2 < c \end{cases} \tag{7.1}$$

де c – граничне значення (*Threshold*) керуючого вхідного сигналу u_3.

Блок ***Manual Switch* (*ручний ключ*)** пропускає на вихід один із двох вхідних сигналів згідно з положенням перемикача, яке може змінюватися щигликом миші (вручну) як при підготовці до моделювання, так і в період розрахунку перехідних процесів.

Блок ***Multiport Switch* (*багатоканальний ключ*)** пропускає на вихід один із декількох вхідних сигналів, кількість яких задається параметром *Number of inputs*. Номер входу, що пропускається, визначається округленим до цілого значенням керуючого сигналу, який подається на перший вхід ключа.

Блок ***Index Vector*** схожий за призначенням на блок *Multiport Switch*. Основна різниця полягає у тому, що на керований вхід може приходити сигнал типу *real*, який присікається до цілого і здійснює переключення елементів вхідного вектору.

Блок ***Stop Simulation* (*зупинка моделювання*)** зупиняє моделювання при ненульовому скалярному вхідному сигналі. У випадку векторного вхідного сигналу досить, щоб хоча б один з його компонентів став ненульовим. Оскільки в *MATLAB* результатом логічних операцій і операцій порівняння є 1 (істинний) або 0 (хибний), то вхідний сигнал блока *Stop* зручно формувати за допомогою блоків *Relational Operator* і (або) *Logical Operator*. Наприклад, у приведеній на рис. 7.4 моделі процес моделювання перерветься, якщо вихідний сигнал інтегратора стане більш 10 або менше -5.

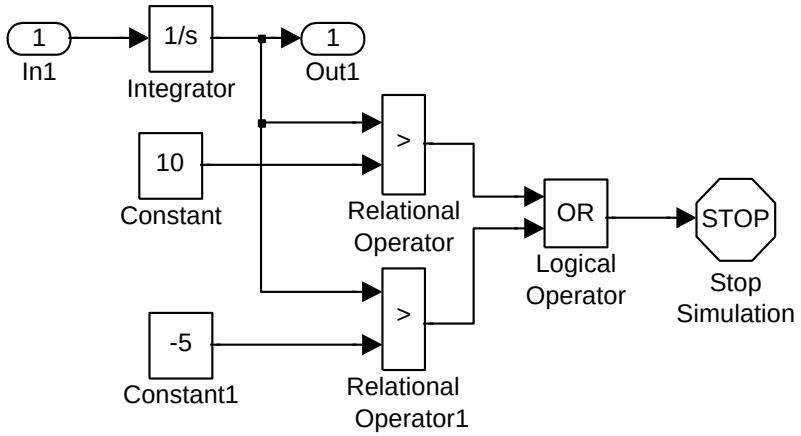


Рис. 7.4. Модель із зупинкою моделювання
за умовою

Блок ***Selector*** обирає або перегруповує елементи багатомірного вхідного сигналу. Основними параметрами цього блока є *Number of input dimen-*

sions, *Input port size* та *Index*, які для блока, що знаходиться у бібліотеці, мають значення 1, 3 та [1, 3] відповідно. Спосіб перемикання вхідних сигналів при формуванні вихідних відображається на іконці блока.

7.3. Перемикання за допомогою інтеграторів

Аналоговий і дискретний інтегратори можуть працювати у режимі скидання вихідного сигналу у початковий стан. Для включення цього режиму параметр *External reset* інтегратору треба переключити зі стану none у стан *rising, falling, either, level* або *level hold*. використовувати як внутрішні *(internal)*, так і зовнішні *(external)* початкові умови.

Початкова умова може задаватися константою в параметрах блока і може визначатися зовнішнім сигналом. Вибір здійснюється за допомогою меню опції *Initial condition source*. При виборі режиму *internal* діє початкова умова, що встановлюється в полі параметра *Initial condition*. При виборі режиму *external* блок одержує додатковий вхідний порт, до якого можна приєднати блок *IC* (*Початкова умова*) бібліотеки *Signal Attributes*. Задавати зовнішні початкові умови має сенс тільки для інтегратора зі скиданням

Для організації скидання або зміни початкових умов у функції вихідного сигналу інтегратора необхідно, щоб уникнути утворення алгебраїчної петлі, замість вихідного порту використовувати порт стану, для чого потрібно візуалізувати його установкою прапорця *Show state port*. Сигнал стану інтегратора формується раніш, ніж вихідний сигнал, хоча і має з ним однакові значення. На рис. 7.5 зображені піктограми блока *Integrator* з різними установками режимів.

Приклад використання ланки *Integrator* у режимах *Reset* і *Limited* можна подивитися в демонстраційному прикладі *Bouncing ball* (модель *bounce*).

Друга область застосування порту стану – це передача сигналів з однієї підсистеми, виконуваної за умовою, в іншу. Для демонстрації цього прикладу можна скористуватися моделлю *clutch*.

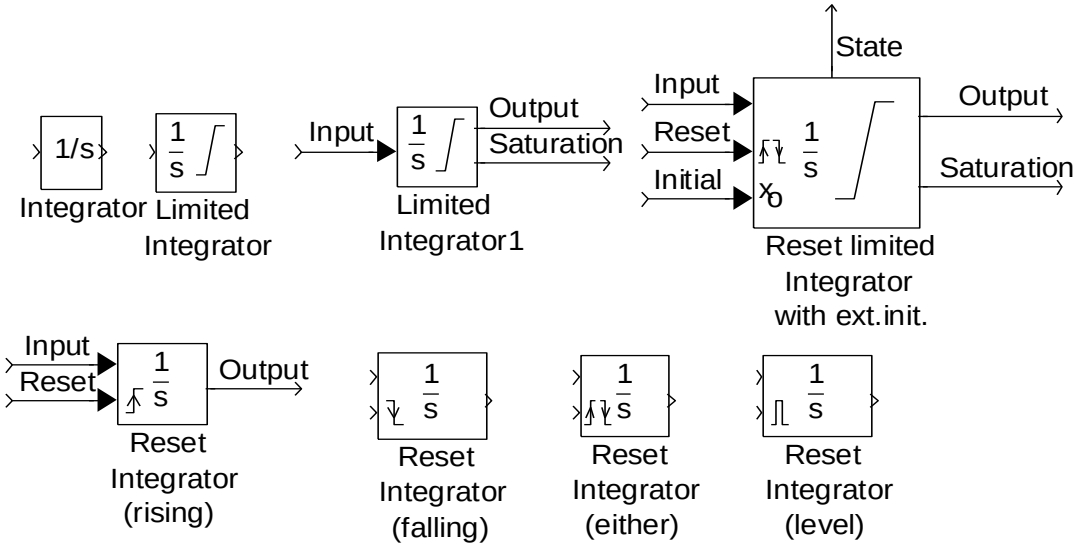


Рис. 7.5. Піктограми інтеграторів, які працюють у різних режимах

Крім знайомства із запропонованими демонстраційними моделями, для більш детального вивчення блока можна зібрати модель, зображену на рис. 7.6 і розібратися в отриманих за її допомогою перехідних процесів (рис. 7.7).

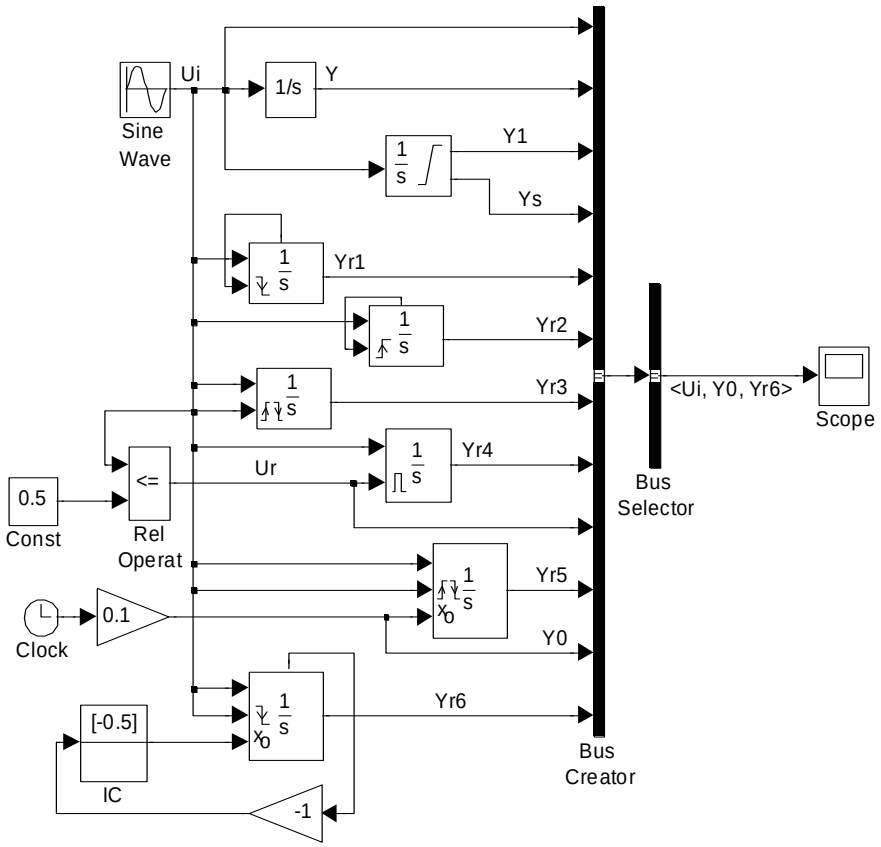


Рис. 7.6. Модель для вивчення принципу роботи інтегратора в різних режимах

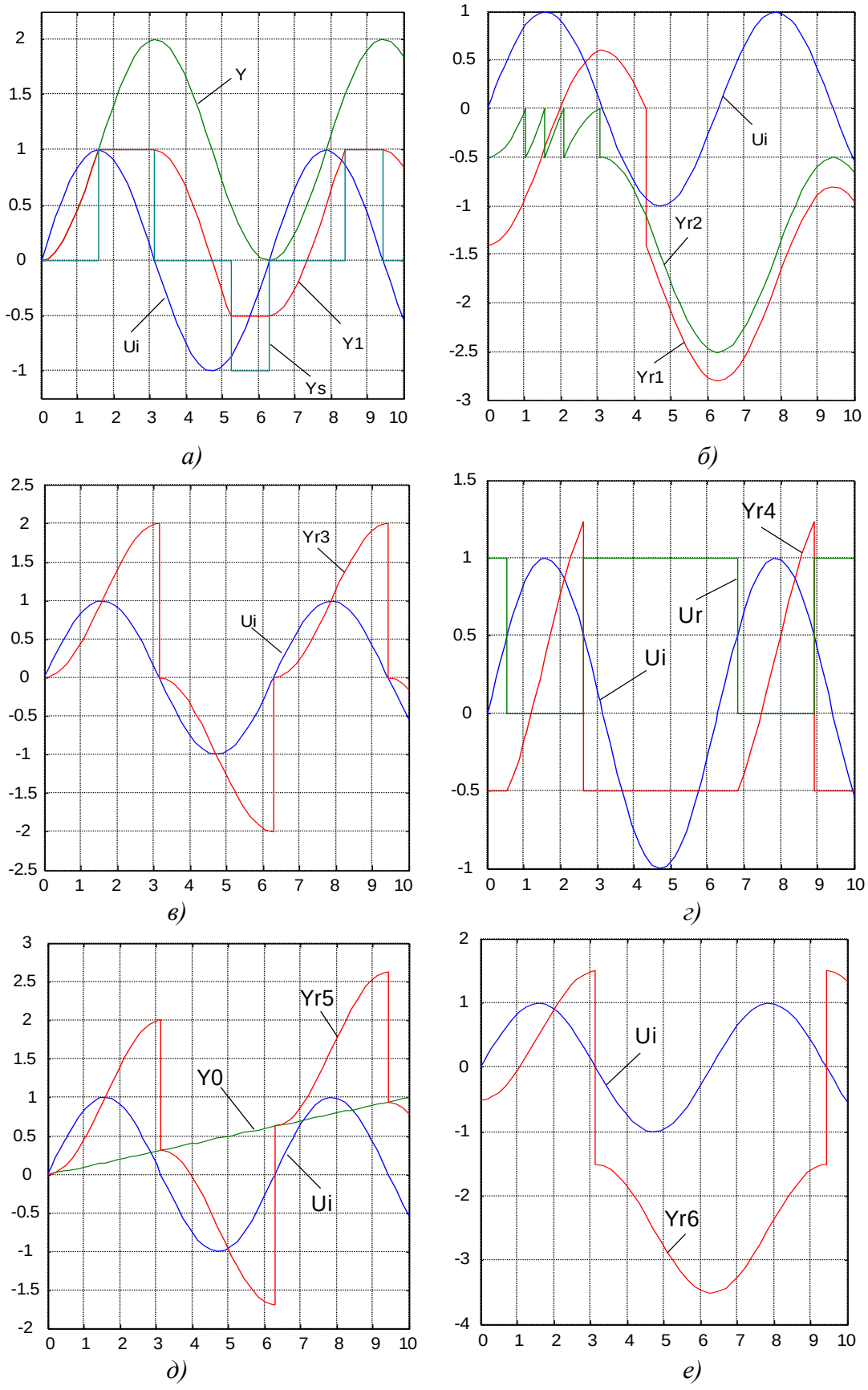


Рис. 7.7. Графіки перехідних процесів,
які відображають роботу інтегратора в різних режимах

Контрольні питання та завдання

1. Які операції порівняння та логічні операції можна здійснювати у середовищі *Simulink*? Який результат отримують при виконанні операцій порівняння та логічних операцій?
2. Які блоки можуть здійснювати перемикання в *Simulink*-моделях? Наведіть простий приклад використання блоків *Switch* та *Selector*.
3. Як виконати зупинку симуляції за заданою умовою?
4. Проілюструйте роботу блока *Hit crossing*, встановленням на його вході джерела синусоїдального сигналу, а на виході – лічильника кількості перетинів, що утворюється замиканням блока *Memory* одиничним додатним зв’язком.
5. У яких режимах може працювати блок *Integrator*?
6. Як забезпечити скидання вихідного сигналу інтегратора у початковий стан?
7. Поясніть роботу демонстраційних моделей *Simulation of the Bouncing Ball* та *Friction Model with Hardstops.*

8. МОДЕЛЮВАННЯ НЕПЕРЕРВНИХ ЛІНІЙНИХ ДИНАМІЧНИХ СИСТЕМ У СЕРЕДОВИЩІ *Simulink*

8.1. Форми математичного опису неперервних лінійних систем

У більшості випадків на першому етапі досліджень роблять такі припущення, які дозволяють вважати САУ лінійними та стаціонарними (*LTI-systems*). Методи аналізу та синтезу таких системи є найпростішими та добре розвинутими [4, 10, 11]. Як вже відзначалося у підрозділі 1.2, неперервні *LTI*-системи описуються звичайними лінійними диференційними рівняннями з постійними коефіцієнтами.

За кількістю вхідних та вихідних величин системи розподіляються на такі групи:

- *SISO (Single Input Single Output)* – системи, що мають один вхід та один вихід;
- *SIMO (Single Input Many Output)* – системи, що мають один вхід та багато виходів;
- *MIMO (Many Input Many Output)* – системи, що мають багато входів та багато виходів.

В середовищі *Simulink* застосовують три форми подання звичайних ДР, що описують неперервні *LTI*-системи:

- ***Transfer Function Polynomial*** – передавальні функції у поліноміальній формі;
- ***Zero-Pole-Gain*** – передавальні функції, розкладені за нулями та полюсами;
- ***State Space*** – математичний опис у просторі станів.

Перші 2 форми в теорії автоматичного керування (ТАК) відносяться до класичних форму запису, які застосовують для опису *SISO*-систем.

Третя форма використовується у сучасній теорії керування і є найбільш універсальною,тому що може описати як *SISO*-, так і *SIMO*- та *MIMO*-

системи.

Отже, нехай довільна неперервна стаціонарна динамічна система *n*-го порядку має *k* входів та *r* виходів. Тоді вона може бути описана у просторі стану системою лінійних диференційних рівнянь (ДР) у нормальній формі Коші

$$\dot{\mathbf{x}}(t)=\mathbf{A}\mathbf{x}(t)+\mathbf{B}\mathbf{u}(t) \tag{8.1}$$

з початковими умовами

$$\mathbf{x}(t_0)=\mathbf{x}_0 \tag{8.2}$$

та лінійним матричним рівнянням виходу

$$\mathbf{y}(t)=\mathbf{C}\mathbf{x}(t)+\mathbf{D}\mathbf{u}(t), \tag{8.3}$$

де

$\mathbf{x}=\begin{bmatrix}x_1 & x_2 & \dots & x_n\end{bmatrix}^T$ – вектор змінних стану;

$\mathbf{y}=\begin{bmatrix}y_1 & y_2 & \dots & y_r\end{bmatrix}^T$ – вектор вихідних сигналів;

$\mathbf{u}=\begin{bmatrix}u_1 & u_2 & \dots & u_k\end{bmatrix}^T$ – вектор вхідних сигналів;

$\mathbf{x}_0=\begin{bmatrix}x_{10} & x_{20} & \dots & x_{n0}\end{bmatrix}^T$ – вектор початкових умов (*Initial conditions*);

A – матриця стану розміром $n\times n$;

B – матриця входу розміром $n\times k$;

C – матриця виходу розміром $r\times n$;

D – матриця обходу розміром $r\times k$, яка характеризує прямий (пропорційний) зв’язок між входом і виходом.

Залежні змінні у таких ДР називають змінними стану. Їх кількість дорівнює кількості ДР і називається порядком системи. Якщо скласти на основі рівнянь стану скалярну структурну схему, то ***усі змінні стану будуть розташовані на виходах інтеграторів***.

Для *SIMO*-систем ($k = 1$) матриці **B** і **D** вироджуються у вектори-стовпці:

$$\mathbf{B}=\mathbf{b}=\begin{bmatrix}b_1 & b_2 & \dots & b_n\end{bmatrix}^T, \qquad \mathbf{D}=\mathbf{d}=\begin{bmatrix}d_1 & d_2 & \dots & d_n\end{bmatrix}^T,$$

а для *SISO*-систем ($k = 1$, $r = 1$) – матриця **B** залишається вектором-стовпцем, матриця **C** стає вектором-рядком розміром $1\times n$, а матриця **D** вироджується у скаляр:

$$\mathbf{C} = \mathbf{c} = \begin{bmatrix} c_1 & c_2 & \dots & c_n \end{bmatrix}, \qquad \mathbf{D} = d\,.$$

ДР (8.1) неоднорідні. Вони описують поведінку системи під дією вхідних сигналів. При відсутності таких сигналів ДР стають однорідними:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t). \tag{8.4}$$

У сукупності з рівнянням виходу

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) \tag{8.5}$$

вони описують власні рухи системи під дією відхилення початкових умов від значень, що відповідають усталеному режиму.

Для переходу до операторної форми запису рівнянь (8.1), (8.2) в них треба замінити операцію диференціювання операцією множення на оператор Лапласа s:

$$s\mathbf{x}(s) = \mathbf{A}\mathbf{x}(s) + \mathbf{B}\mathbf{u}(s), \tag{8.6}$$

$$\mathbf{y}(s) = \mathbf{C}\mathbf{x}(s) + \mathbf{D}\mathbf{u}(s). \tag{8.7}$$

Математичному опису у формі (8.6), (8.7) відповідає векторно-матрична структурна схема, що зображена на рисунку 8.1.

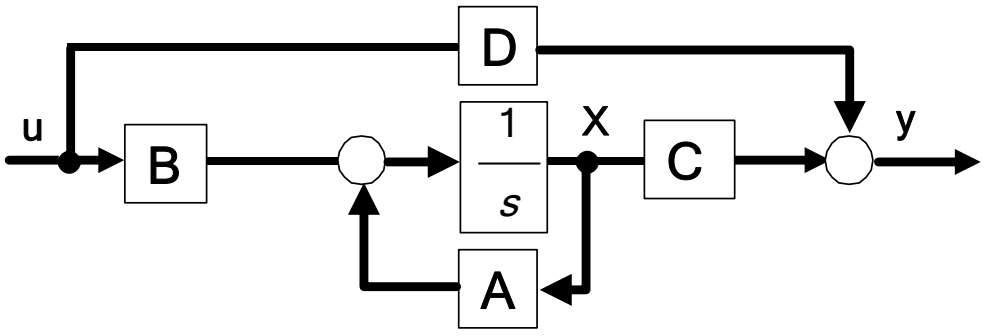


Рис. 8.1. Структурна схема *LTI-MIMO*-системи у просторі стану

З рівнянь (8.6), (8.7) можна знайти матричні передавальні функції (ПФ) від вектору входу до вектору змінних стану розміром $n\times k$ та до вектору виходу розміром $r\times k$:

$$\mathbf{W}_x(s) = \frac{\mathbf{x}(s)}{\mathbf{u}(s)} = (s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} = \frac{Adj(s\mathbf{I} - \mathbf{A})}{det(s\mathbf{I} - \mathbf{A})}\mathbf{B}, \tag{8.8}$$

$$\mathbf{W}_y(s) = \frac{\mathbf{y}(s)}{\mathbf{u}(s)} = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} = \mathbf{C}\frac{Adj(s\mathbf{I} - \mathbf{A})}{det(s\mathbf{I} - \mathbf{A})}\mathbf{B} + \mathbf{D}, \tag{8.9}$$

де **I** – одинична діагональна матриця розміром $n \times n$.

Знаменник передавальних функцій звуть ***характеристичним поліномом*** (ХП) системи:

$$G(s) = det(s\mathbf{I} - \mathbf{A}) = s^n + \alpha_{n-1}s^{n-1} + ... + \alpha_1 s + \alpha_0, \tag{8.10}$$

а рівняння

$$G(s) = 0$$

характеристичним рівнянням.

Корені характеристичного рівняння

$$\mathbf{P} = [p_1 \quad p_2 \quad ... \quad p_n] \tag{8.11}$$

називають ***полюсами системи*** або ***власними числами матриці*** **A**.

Чисельних кожної з скалярних ПФ, що є компонентами матричних ПФ (8.8) та (8.9), також являє собою степеневий поліном відносно змінної Лапласа, який називають ***поліномом впливу***:

$$H(s) = \beta_m s^m + \beta_{m-1}s^{m-1} + ... + \beta_1 s + \beta_0. \tag{8.12}$$

Його корені

$$\mathbf{Z} = [z_1 \;\; z_2 \;\; ... \;\; z_m] \tag{8.13}$$

звуть ***нулями системи***.

Порядок поліному впливу пов’язаний з порядком характеристичного поліному умовою можливості фізичної реалізації

$$m \le n. \tag{8.14}$$

Слід відзначити, що умова $m = n$ може бути виконана тільки для передавальних функцій від вхідного сигналу до вихідного, якщо вони мають між собою прямий зв’язок (без ланок затримки). Математичним виразом такого зв’язку є ненульове значення відповідного елементу матриці обходу **D**.

Отже, скалярні передавальні функції можуть бути записані у поліноміальній формі (*Transfer Function Polynomial*), наприклад:

$$W_{ij}(s)=\frac{y_i(s)}{u_j(s)}=\frac{H_{ij}(s)}{G(s)}=\frac{\beta_m s^m+\beta_{m-1}s^{m-1}+\ldots+\beta_1 s+\beta_0}{s^n+\alpha_{n-1}s^{n-1}+\ldots+\alpha_1 s+\alpha_0}. \qquad (8.15)$$

Як бачимо, ХП, визначений за формулою (8.10), і застосований у ПФ (8.15) виявляється нормованим за коефіцієнтом при старшому ступені оператору Лапласа.

У класичній теорії автоматичного керування більш прийнято нормування поліномів у чисельнику та знаменнику передавальної функції за коефіцієнтами при вільному члені або, при його відсутності, за коефіцієнтами при найменших ступенях оператора Лапласа, наприклад,

$$W_{ij}(s)=\frac{y_i(s)}{u_j(s)}=k\frac{b_m s^m+b_{m-1}s^{m-1}+\ldots+b_1 s+1}{a_n s^n+a_{n-1}s^{n-1}+\ldots+a_1 s+1}=k\frac{H_{0ij}(s)}{G_0(s)}. \qquad (8.16)$$

Коефіцієнти ПФ (8.16) і (8.17) пов’язані між собою співвідношеннями:

$$a_n=\frac{1}{\alpha_0},\qquad a_i=\frac{\alpha_i}{\alpha_0},\qquad b_i=\frac{\beta_i}{\beta_0},\qquad k=\frac{\beta_0}{\alpha_0}. \qquad (8.17)$$

Скалярній передавальній функції (8.16) відповідає ДР

$$\frac{d^n y_i(t)}{dt^n}+\alpha_{n-1}\frac{d^{n-1}y_i(t)}{dt^{n-1}}+\ldots+\alpha_1\frac{dy_i(t)}{dt}+\alpha_0 y_i(t)=$$

$$=\beta_m\frac{d^m u_j(t)}{dt^m}+\beta_{m-1}\frac{d^{m-1}u_j(t)}{dt^{m-1}}+\ldots+\beta_1\frac{du_j(t)}{dt}+\beta_0 u_j(t).$$

Використовуючи розкладення поліномів у чисельнику та знаменнику передавальної функції (8.16) або (8.17) на множники, отримуємо математичний опис у формі *Zero-Pole-Gain*

$$W_{ij}(s)=K\frac{(s-z_1)(s-z_2)\ldots(s-z_m)}{(s-p_1)(s-p_2)\ldots(s-p_n)}, \qquad (8.18)$$

де

$$K=\beta_m=kb_m/a_n. \qquad (8.19)$$

Перетворення з форми *Transfer Function* у форму *State Space* для SISO-систем з передавальними функціями загального вигляду (2.2) і (2.5) виконано у підрозділі 2.2 (див. рис. 2.1, 2.2).

Щоб визначити матриці рівнянь (8.1), (8.3) для структурної схеми рис. 2.1, записуємо її рівняння стану та рівняння виходу у скалярній формі

$$\begin{cases} sx_1 = -\alpha_{n-1}x_1 - ... - \alpha_1 x_{n-1} - \alpha_0 x_n + 1\cdot u, \\ sx_2 = \quad 1\cdot x_1 \; + \; ... + 0\cdot x_{n-1} + 0\cdot x_n \; + 0\cdot u, \\ ... \\ sx_n = \quad 0\cdot x_1 \; + \; ... + 1\cdot x_{n-1} \; + \; 0\cdot x_n \; + 0\cdot u, \end{cases} \tag{8.20}$$

$$y = 0\cdot x_1 \; + ... + 0\cdot x_{n-1} \; + 1\cdot x_n \; + 0\cdot u\,. \tag{8.21}$$

Із (8.20) та (8.21) випливає

$$\mathbf{A} = \begin{bmatrix} -\alpha_{n-1} & ... & -\alpha_1 & -\alpha_0 \\ 1 & ... & 0 & 0 \\ & ... & & \\ 0 & ... & 1 & 0 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 \\ 0 \\ ... \\ 0 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 0 & 0 & ... & 1 \end{bmatrix}, \quad D = 0\,. \tag{8.22}$$

При наявності у ПФ загального вигляду нулів (структурна схема рис. 2.2, ПФ (2.5)) у порівнянні зі структурною схемою рис. 2.1 змінюється тільки рівняння виходу

$$y_1(s) = \beta_0 x_n(s) + \beta_1 x_{n-1}(s) + ... + \beta_{m-1} x_{n-m+1}(s) + \beta_m x_{n-m}(s)\,. \tag{8.23}$$

Отже, досліджувані схеми мають однакові матриці стану та матрицю входу, а матриці виходу та прямого зв'язку входу з виходом системи рис. 2.2 при $m = n$ набувають вигляду:

$$\mathbf{C} = \begin{bmatrix} \beta_{n-1} - \beta_n\alpha_{n-1} & ... & \beta_1 - \beta_n\alpha_1 & \beta_0 - \beta_n\alpha_0 \end{bmatrix}, \quad D = \beta_n\,. \tag{8.24}$$

а при $m < n$ $(\beta_n = 0)$ -

$$\mathbf{C} = \begin{bmatrix} \beta_{n-1} & ... & \beta_1 & \beta_0 \end{bmatrix}, \quad D = 0\,. \tag{8.25}$$

Кожна структурна схема може мати декілька варіантів її подання у просторі станів. Для неперервних динамічних блоків першого порядку можна запропонувати деталізовані структурні схеми, подані в табл. 2.1.

8.2. Характеристика блоків бібліотеки *Continuous*

Блоки, призначені для моделювання неперервних лінійних динамічних ланок в середовищі *Simulink*, розташовані в бібліотеці *Continuous*.

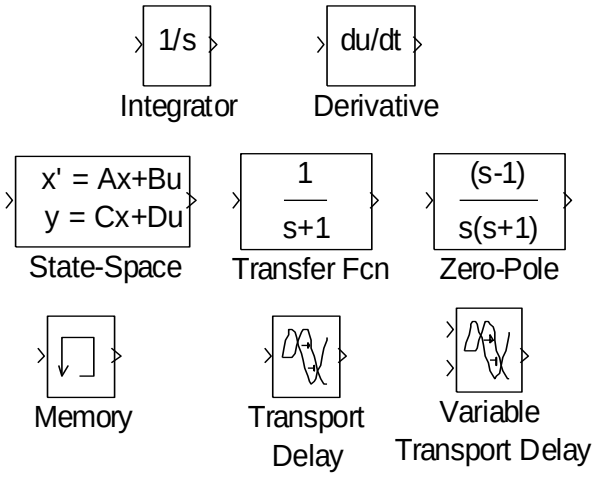


Рис. 8.2. Бібліотека неперервних динамічних блоків **(***Continuous***)**

Основні блоки бібліотеки *Continuous*, представлені на рис. 8.2. До них належать блоки аналогового інтегрування (*Integrator*) і диференціювання (*Derivative*), лінійні неперервні динамічні ланки загального вигляду в різній формі (*Transfer Fcn*, *Zero-Pole* і *State-Space*) і ланки запізнювання (*Memory, Transport Delay, Variable Transport Delay*).

8.2.1. *Integrator* (*інтегратор*)

Цей блок інтегрує вхідний скалярний u або векторний $\mathbf{u} = [u_1, u_2, ..., u_k]$ сигнал із заданими початковими умовами (*Initial condition*), тобто

$$\mathbf{x}(t) = \mathbf{x}_0 + \int_0^t \mathbf{u}(t)dt. \qquad (8.26)$$

і має передавальну функцію $W_i(s) = 1/s$.

Роботу інтегратора у режимах обмеження вихідного сигналу на заданому рівні (*Limited Integrator*) та скидання його у заданий початковий стан (*Reset Integrator*) буде розглянуто у розділах, присвячених моделюванню нелінійних систем та систем з перемиканнями відповідно.

8.2.2. *Derivative* (*похідна*)

Виконує чисельне диференціювання вхідного сигналу:

$$y(t_i) = \frac{\Delta u_i}{\Delta t_i} = \frac{u_i - u_{i-1}}{t_i - t_{i-1}} \approx \frac{du}{dt}. \qquad (8.27)$$

Точність результату залежить від розміру кроку інтегрування. Менший розмір кроку дозволяє одержати більш точний результат.

При диференціюванні дискретного сигналу з періодом переривання T_S вихідний сигнал являє собою послідовність імпульсів у моменти часу, які збігаються з моментами зміни вхідного сигналу:

$$y(t_i) = \begin{cases} \dfrac{u(kT_S) - u(kT_S - T_S)}{t_i - t_{i-1}} & \text{при } t_i = kT_S, \\ 0 & \text{при } t_i \neq kT_S. \end{cases} \qquad (8.28)$$

У цьому випадку блок може бути описаний дискретною передавальною функцією

$$\frac{y(z)}{u(z)} = \frac{1 - z^{-1}}{\Delta t} = \frac{z - 1}{\Delta t \cdot z}. \qquad (8.29)$$

8.2.3. *Transfer Fcn* (*передавальна функція*)

Блок *Transfer Fcn* реалізує передавальну функцію загального вигляду в поліноміальній формі:

$$W(s) = \frac{B_m(s)}{A_n(s)} = \frac{\sum_{i=m}^{0} b_i s^i}{\sum_{j=n}^{0} a_j s^j}. \qquad (8.30)$$

Вхідними параметрами є вектори коефіцієнтів степеневих поліномів чисельника (*Numerator*) і знаменника (*Denominator*), упорядковані за зменшенням степенів оператора Лапласа *s*. Вектор з (*n*+1) елементів задає поліном степені *n*. Порядок знаменника повинний бути більшим від порядку чисельника або дорівнювати йому. Блок має нульові початкові умови.

Коефіцієнти можуть бути введені трьома способами:

1) константами і скалярними змінними, об'єднаними квадратними дужками у вектор;

2) змінною без дужок;

3) змінною або виразом у круглих дужках.

При першому способі в піктограмі блока відображаються поліноми з коефіцієнтами у вигляді введених констант і змінних при відповідних степенях оператора Лапласа *s*. Наприклад, введення вектору [1 -2 *T* 4] відобразиться в блоці як

$$s^3 - 2s^2 + Ts + 4 ,$$

а уведення вектору [*b*0, *b*1, *b*2] – як

$$b0s^2 + b1s + b2 .$$

Змінні, що використані в цих векторах, повинні одержати значення в робочому середовищі *MATLAB* до початку процесу моделювання.

При другому способі поліном відображається у вигляді *змінна(s)*. Наприклад, введення *num* відображається як *num(s)*, введення *An* – як *An(s)* та т.і.

При третьому способі *Simulink* шукає значення змінних, які входять до виразу, у робочому просторі *MATLAB*, обчислює значення виразу і відображає його так само, як і при першому способі введення. Наприклад, якщо поліном визначено у діалоговому вікні як (*G*+*B*), а змінні *G* і *B* визначено в *MATLAB* як *G*=[4 1 1], *B*=[0 1 0], то відображення полінома в блоці має вид:

$$4s^2+2s+1.$$

Значення змінних повинне існувати в робочому середовищі в будь-який момент зображення блока, інакше в піктограмі блока відобразяться три знаки питання: *???*.

Simulink дозволяє визначити передавальну функцію не тільки для систем з один входом та одним виходом, але і для систем з одним входом та декількома виходами. Це досягається введенням чисельника передавальної функції як матриці коефіцієнтів, кількість рядків якої відповідає кількості виходів. Відображення коефіцієнтів чисельника в піктограмі блока в цьому ви-

падку не підтримується, так що чисельник представляється тільки в загальному вигляді, наприклад, *num(s)*, тобто матриця коефіцієнтів чисельника повинна задаватися тільки ім'ям змінної без дужок. Якщо наступні за розглянутою ланкою блоки не сприймають векторних вхідних сигналів, то отриманий вихідний векторний сигнал необхідно пропустити через блок *Demux* (*демультиплексор*) з бібліотеки *Signal Routing*.

8.2.4. *Zero-Pole* (*нулі-полюси*)

Являє собою ланку загального виду з передавальною функцією (5.18) та такими параметрами:

K – посилення (*Gain*);

Z – вектор нулів (*Zeros*);

P – вектор полюсів (*Poles*).

***При відсутності нулів їх задають пустою матрицею* [].**

Нулі і полюси можуть мати не тільки дійсні, але і комплексно-спряжені значення. Вони можуть вводитися тими ж трьома способами, що і вектори степеневих багаточленів блока *Transfer Fcn*.

При наявності комплексно спряжених пар нулів або полюсів в піктограмі блока співмножники передавальної функції виводяться не у вигляді добутку двох поліномів першого порядку з комплексними вільними членами, а у вигляді одного поліному другого порядку з дійсними коефіцієнтами. Наприклад, якщо

$$p_{1,2} = [\alpha + j\beta,\ \alpha - j\beta], \tag{8.31}$$

то відповідний множник характеристичного поліному знаходиться як

$$G_{1,2}(s) = (s + \alpha - j\beta)(s + \alpha + j\beta) = (s + \alpha)^2 + \beta^2 = s^2 + 2\alpha s + (\alpha^2 + \beta^2). \tag{8.32}$$

Розглянутий блок не підтримує режим *SIMO*, тобто не може бути використано для опису лінійної динамічної системи з декількома виходами.

8.2.5. *State-Space* (*простір стану*)

Забезпечує моделювання неперервної динамічної ланки загального ви-

ду за її математичному описом у просторі стану (8.1)-(8.3). Параметрами цього блока є матриці коефіцієнтів **A**, **B**, **C**, **D** та вектор початкових умов (*Initial conditions*).

8.2.6. *Memory* (*пам'ять*)

Здійснює затримку на один крок чисельного інтегрування:

$$\frac{y(s)}{u(s)} = e^{-hs}. \tag{8.33}$$

На виході утримується попереднє значення входу. Цей блок можна використовувати для розв'язки алгебраїчних контурів. Разом із блоком *Clock* його можна використовувати для визначення кроків інтегрування. У цьому блоці задається тільки один параметр: *Initial condition* – початкова умова.

8.2.7. *Transport Delay* (*чисте запізнювання*)

Передавальна функція ланки має вигляд:

$$\frac{y(s)}{u(s)} = e^{-\tau s}. \tag{8.34}$$

Вихідний сигнал відтворює вхідний сигнал із запізнюванням τ (*Time Delay*):

$$y(t) = \begin{cases} y_0 & \text{при} \quad t \le \tau \\ u(t-\tau) & \text{при} \quad t > \tau \end{cases}, \tag{8.35}$$

де y_0 – початкове значення вхідного сигналу (*Initial input*).

Ланка функціонує за рахунок збереження необхідної кількості послідовних значень вхідного сигналу в круговому буфері, початковий розмір якого задається параметром *Initial Buffer Size*. Отже, при великих часах запізнювання і порівняно малому кроці інтегрування блок може використовувати велику кількість оперативної пам'яті.

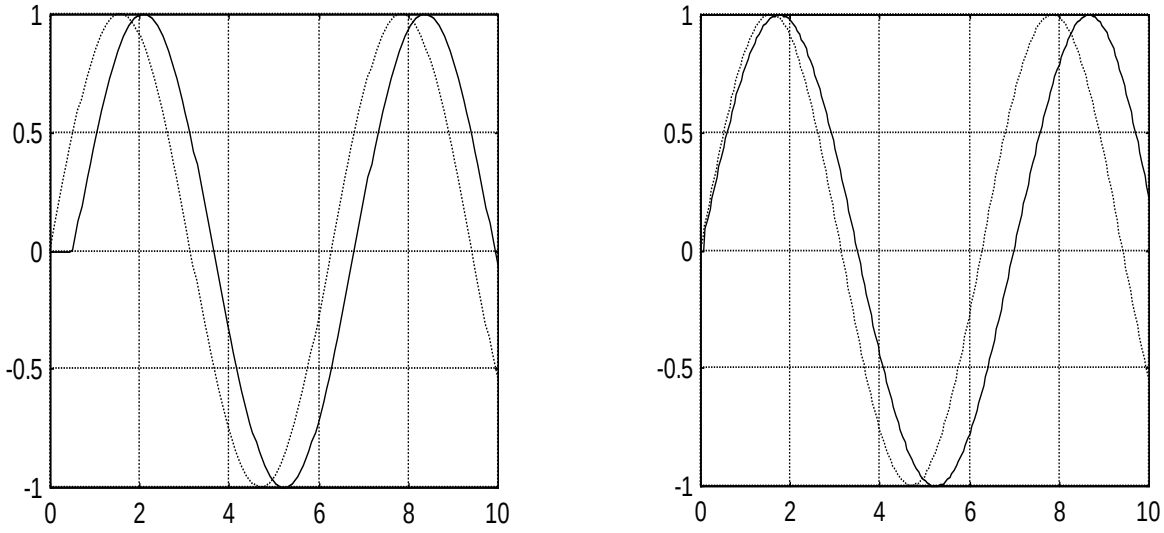
8.2.8. *Variable Transport Delay* (*змінне запізнювання*)

Ця ланка має два входи, на другому з яких формується величина запізнювання вихідного сигналу стосовно першого вхідного. Параметрами, що вводяться, *є Maximum Delay, Input і Buffer* Size. Перший параметр використо-

вується для обмеження запізнювання на заданому рівні і для розрахунку кількості точок у круговому буфері.

Діаграми відпрацьовування синусоїдального вхідного сигналу ланками постійного і змінного запізнювання наведені на рис. 8.2.

Змінне запізнювання сформоване послідовним з'єднанням джерела *Clock* і пропорційної ланки *Gain* з коефіцієнтом передачі 0,1 і обмежено на рівні $\tau = 0{,}8$.



а) *б*)

Рис. 8.2. Перетворення синусоїди ланками
а) *Transport Delay*, *б*) *Variable Transport Delay*

8.3. Характеристика лінійних арифметичних блоків бібліотеки *Math Operations*

Описані в попередньому підрозділі блоки використовуються при моделюванні лінійних динамічних неперервних систем та їх окремих елементів.

Для зв'язку динамічних елементів між собою у складі таких систем застосовуються лінійні арифметичні блоки *Sum*, *Gain*, *Slider Gain* і *Matrix Gain* бібліотеки *Math Operations*.

Блок *Sum* описано в підрозділі 5.1 (див. рис. 5.4).

Блоки ***Gain* (*підсилення, пропорційна ланка*)** та ***Matrix Gain* (*матричне підсилення*)**, починаючи з версії *MATLAB*-8.0 реалізують однакові функ-

ції. Вони можуть виконувати за вибором користувача поелементне (*Element-wise K.*u*) або матричне (*Matrix(K*u), Matrix(u*K)*) множення вхідного сигналу на коефіцієнт передачі. Тип множення задається параметром *Multiplication.* Обидва із множників можуть бути скалярами, векторами або матрицями відповідного розміру, тобто можливі такі варіанти виконання цієї операції:

$$y = K \cdot u, \quad \underset{(m,n)}{\mathbf{y}} = \underset{(m,n)}{\mathbf{K}} .* \underset{(m,n)}{\mathbf{u}}, \quad \underset{(m,n)}{\mathbf{y}} = \underset{(m,k)}{\mathbf{K}} * \underset{(k,n)}{\mathbf{u}}, \quad \underset{(m,n)}{\mathbf{y}} = \underset{(m,k)}{\mathbf{u}} * \underset{(k,n)}{\mathbf{K}}. \tag{8.33}$$

При поелементному множенні, яке виконується над масивами однакового розміру

$$y_{ij} = K_{ij} u_{ij}, \tag{8.34}$$

а при матричному –

$$y_{ij} = \sum_{l=1}^{k} K_{il} u_{lj}. \tag{8.35}$$

Враховуючи те, що сигнали моделі досить нечасто бувають двовимірними, останні три операції з (8.33) здебільш мають такий формат:

$$\underset{(n,1)}{\mathbf{y}} = \underset{(n,1)}{\mathbf{K}} .* \underset{(n,1)}{\mathbf{u}}, \qquad \underset{(m,1)}{\mathbf{y}} = \underset{(m,n)}{\mathbf{K}} * \underset{(n,1)}{\mathbf{u}}, \qquad \underset{(1,n)}{\mathbf{y}} = \underset{(1,m)}{\mathbf{u}} * \underset{(m,n)}{\mathbf{K}}, \tag{8.36}$$

$$y = \underset{(1,n)}{\mathbf{K}} * \underset{(n,1)}{\mathbf{u}}, \qquad \underset{(n,1)}{\mathbf{y}} = \underset{(n,1)}{\mathbf{K}} * u. \tag{8.37}$$

Піктограма блока відображає скалярний коефіцієнт підсилення в тій же формі, у якій він визначений при введенні (змінна або константа). Якщо коефіцієнт заданий у вигляді змінної в круглих дужках, то усередині блока відображається її значення. Якщо значення змінної або її ім’я, або вираз занадто великі для того, щоб відобразити їх у межах блока, то піктограма відображає символи «-*K*-». Щоб побачити вираз або значення коефіцієнта підсилення, необхідно збільшити розміри блока.

Блок ***Slider Gain* (*Повзункове Підсилення*)** виконує множення вхідного сигналу на коефіцієнт, що обирається в заданих межах *Low-High* повзунком віртуального реостату, розташованого у вікні вибору параметрів блока.

Контрольні питання та завдання

1. Які форми математичного опису лінійних неперервних систем Ви знаєте? Які *Simulink*-блоки використовують при їх моделюванні?
2. Як визначити залежність від часу кроків чисельного інтегрувань диференційних рівнянь?
3. Що таке характеристичний поліном, характеристичне рівняння, нулі і полюси системи?
4. Як скласти математичний опис системи у просторі стану?
5. Як отримати передавальну функцію з диференційного рівняння?
6. Як виглядає передавальна функція в іконці блока при наявності в параметрах комплексно-спряжених нулів та/або полюсів? Наведіть приклади.
7. Знайдіть параметри ланок *State Space* та *Zero-Pole-Gain* лінійної динамічної системи з передавальною функцією

$$W(p)=\frac{T_m p}{T_m T_a p^2+T_m p+1},\qquad T_m=0.2\,\text{с},\qquad T_a=0.04\,\text{с}.$$

9. СТВОРЕННЯ ПІДСИСТЕМ ТА ЇХ МАСКУВАННЯ

Щоб зменшити розмір і складність моделі, а також об’єднати фрагменти моделі, що виконують деяку спільну функцію, в єдине ціле, можна згрупувати деяку сукупність блоків у блок підсистеми.

9.1. Створення підсистем

Створити підсистему можна двома способами.

Перший з них полягає у тому, що в вікно моделі вставляється блок *Subsystem*, котрий при подвійному щиглику мишею по його піктограмі відчиняє власне вікно, в якому збирається модель підсистеми. До входів підсистеми приєднують вхідні (*In*), а до виходів – вихідні (*Out*) порти.

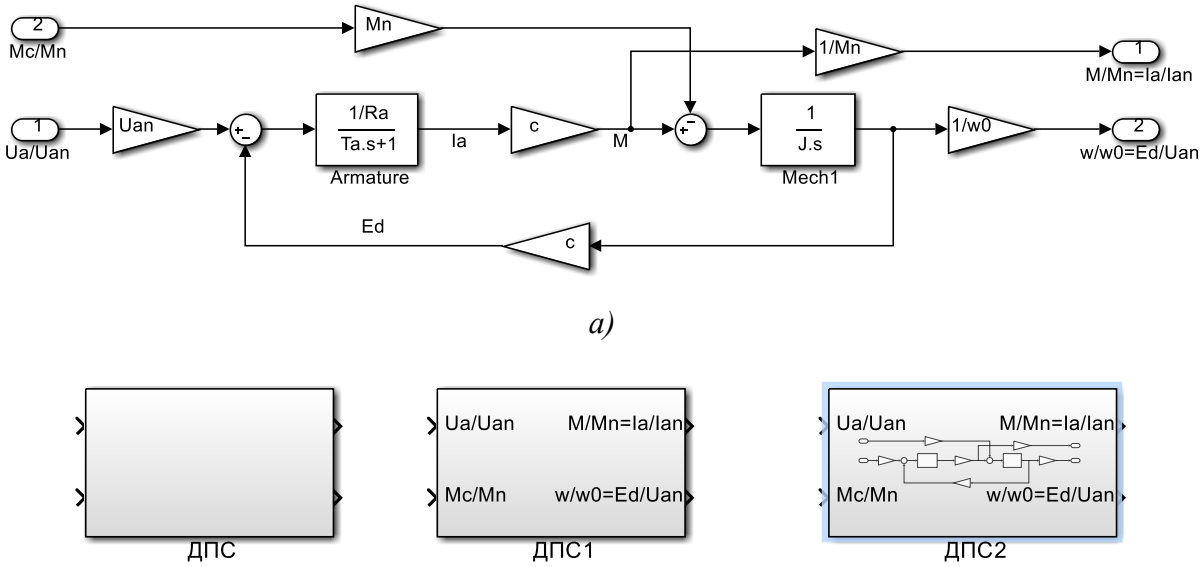
Згідно з другим способом вже існуючий фрагмент моделі виділяється та об’єднується в підсистему командою *Edit → Create Subsystem* (^*G*). В цьому разі вхідні та вихідні порти додаються до фрагменту, що утворює підсистему, автоматично.

Спосіб позначення вхідних та вихідних портів блока *Subsystem* визначається користувачем через меню *Format → Port Labels*, де можна обрати опції *None, From Port Icon, From Port Block Name* та *From Signal Name.*

Створену підсистему можна редагувати, зокрема в ній можна змінити власне ім’я та імена портів, що підвищує наочність нового макроблока. Для наочності імена портів бажано називати іменами вхідних та вихідних змінних моделі. Створену підсистему також бажано перейменувати у відповідності з назвою об’єкта, структуру якого вона відображує. Побачити зміст незамаскованої підсистеми можна подвійним щигликом миші (*double click*) по її іконці.

На рис. 9.1а показана розгорнута підсистема двигуна постійного струму, а на рис. 9.1б згорнуті підсистеми зі схованими і видимими іменами портів та підсистема в режимі *Format → Content preview*, на якій крізь іконку

згорнутої підсистеми просвічується розгорнута підсистема.



а)

б)

Рис. 9.1. Розгорнута (*а*) та згорнуті (*б*) підсистеми двигуна постійного струму

9.2 Маскування підсистем

Будь-яку підсистему можна замаскувати.

Маскування підсистеми починають командою *Edit → Mask Subsystem* (^*M*), на яку система реагує відкриттям вікна маскування *Mask Editor*, що має 4 вкладки (панелі): *Icon & Ports*, *Parameters&Dialog*, *Initialization* та *Documentation*, перша з яких показана на рис. 9.2.

Існують 2 рівня маскування.

Перший рівень полягає тільки у заміні іконки блока. При такому способі подвійний щиглик по іконці як і для незамаскованої підсистеми відчиняє вікно з розгорнутою моделлю підсистеми.

Вкладка *Icon & Ports* (див. рис. 9.2) призначена для зміни піктограми (іконки) підсистеми та позначень портів. Зовнішнім видом іконки керують команди, занесені в **поле *Icon Drawing commands.***

Принципово у піктограмі блока може бути виведеним будь-який текст, передавальна функція, графік статичної або динамічної характеристики та не

дуже складне графічне зображення.

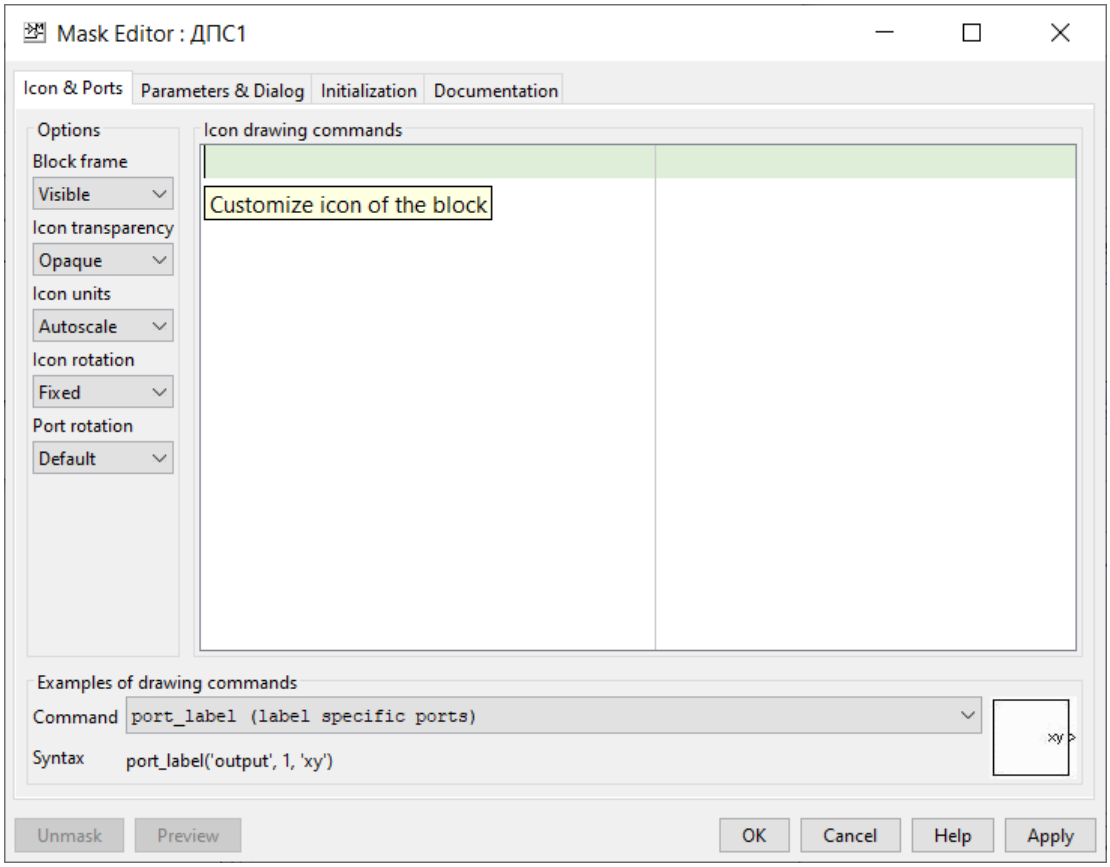


Рис. 9.2. Вкладка *Icon & Ports* редактора маскування

Для виведення тексту можна використовувати один з наступних операторів:

disp (’text’),

text (x,y,’text’),

text (x,y,’text’,’HorizontalAligment’,HorAl ’VerticalAligment’,VertAl),

port_label (port_type, port_number,label).

Для розподілу тексту на рядки можна використовувати комбінацію символів *n*, наприклад, disp(’Mask\nSubsystem’).

В операторах text параметри x,y є координатами точки в піктограмі блока, до якої прив’язується текст.

Для явного визначення системи координат, що використовується при зображенні іконки можна скористатися оператором

plot (xlb, ylb, xrt, yrt),

де xlb, ylb і xrt, yrt – координати лівого нижнього та правого верхнього кутів графічного вікна. Використання операторів xlim, ylim та axes [] з цією ж метою у полі *Drawing commands* не припустимо.

Стиль прив'язки визначається параметрами HorAl (вирівнювання по горизонталі) та VertAl (вирівнювання по вертикалі), які можуть приймати такі значення: 'center', 'left', 'right', і 'middle', 'base', 'bottom', 'cap', 'top' відповідно. За замовчанням діють опції 'left' та 'middle'. Результат використання перелічених опцій пояснюється на рис. 9.3.

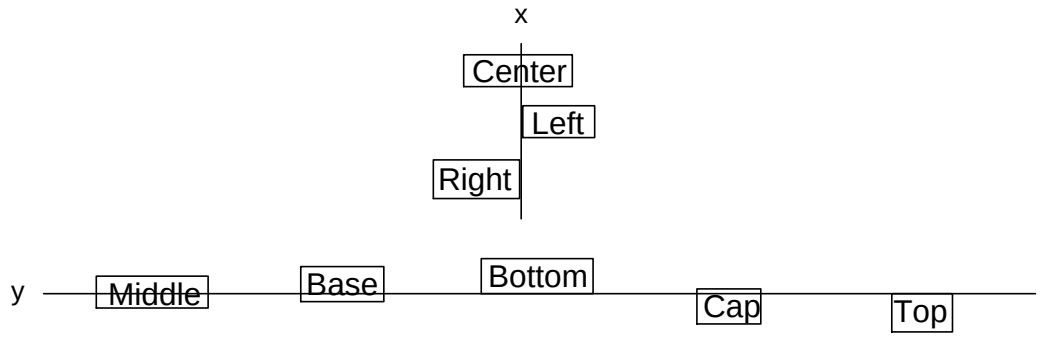


Рис. 9.3. Демонстрація дії опцій вирівнювання текстів в іконці

При застосуванні команди port_label параметр port_type може приймати значення 'input' або 'output', а параметр label – будь-якого рядка символів, наприклад, в результаті виконання команди

port_label('output',2,'a')

другий вихідний порт згорнутої підсистеми буде помічено символом a, незалежно від імені цього порта у розгорнутій підсистемі.

Для зображення на піктограмі блока його передавальної функції можна застосовувати одну із наступних команд:

dpoly (num, den) або dpoly (num, den, char),

droots (z, p, k) або droots (z, p, k, char).

Команда dpoly виводить передавальну функцію в поліноміальній формі, а droots – у вигляді розкладення на нулі-полюси. Необов'язковий символьний параметр char має за замовчанням значення 's'. Для виведення дис-

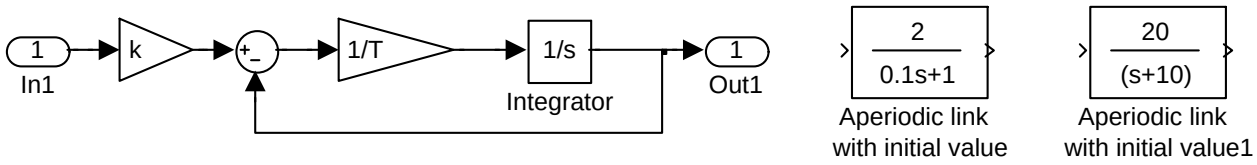
кретних ПФ його можна змінити на ’z’ або ’z^-1’.

Параметри команд dpoly і droots звичайно визначаються через параметри підсистеми у полі *Initialization commands* вкладки *Initialization*. Для прикладу на рис. 9.4 показана у розгорнутому (а) та згорнутому (б, в) вигляді підсистема *Aperiodic link with initial value* (*Аперіодична ланка з початковими умовами*). На рис. 9.4*б* іконка замаскованої підсистеми сформована командою dpoly, а на рис 9.4*в* – командою droots, для чого у полі *Initialization commands* записані оператори присвоювання

num = [k]; den = [T 1];

z = [ ]; p = -1/T; K = k/T;

відповідно.



а) *б)* *в)*

Рис. 9.4. Розгорнута (*а*) та згорнуті (*б*, *в*) підсистеми *Aperiodic link with initial value*

Як бачимо, параметри піктограм, сформованих у такий спосіб, виводяться тільки у вигляді констант. Для того, щоб результати операцій присвоювання не відображувались у командному вікні, після кожного з операторів треба поставити крапку з комою (;).

Для виведення на піктограмі графічних зображень, заданих координатами вузлових точок, можна скористуватись командами

plot (y), plot (x, y), plot (x1, y1, x2, y2, …, xN, yN),

де x, y – вектори абсцис та ординат табличної функції, або

plot (xlb, ylb, xrt, yrt, y), plot (xlb, ylb, xrt, yrt, x, y),

plot (xlb, ylb, xrt, yrt, x1, y1, x2, y2, …, xN, yN),

де xlb, ylb, xrt, yrt – явно визначають координати лівого нижнього (lb – *left, bottom*) та првого верхнього (rt – *reight, top*) кутів іконки.

Іконки у вигляді досить простих графічних зображень, що складаються з обмеженої кількості відрізків прямих, можна створити за допомогою утіліти iconedit. При виклику її з командного рядка *MATLAB* на екрані по черзі з'являються 2 запити: *Name of block diagram* (*Ім'я моделі*) і *Name of block* (*Ім'я підсистеми*). Після відповіді на ці запроси на екрані відчиняється графічне вікно, у якому можна намалювати бажане зображення, фіксуючи за допомогою миші вузлові точки рисунку та використовуючи наступні клавішні команди: *d=delete* – знищити останню з зафіксованих точок, *n=new pt* – почати нову лінію і *q=quit* – завершити створення іконки.

Для зображення зафарбованих плоских замкнених фігур використовують команду

patch (x, y, [r g b]),

де x, y – вектори абсцис та ординат точок замкненої фігури, [r g b]) – її колір, визначений інтенсивністю трьох кольорів: червоного, зеленого та синього у вигляді констант зі значеннями в діапазоні від 0 до 1. За замовчанням (при відсутньості останнього параметру) фігура зафарбовується кольором „заднього фону" (*Background color*), тобто тим кольором, яким зображені контури іконки. Наприклад, функція

patch ([-1 0 1], [0 1 0], [0 1 1])

створює піктограму у вигляді трикутника, зафарбованого бірюзовим кольором, зображену на рис. 9.5.

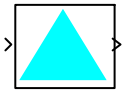


Рис. 9.5. Замаскована підсистема, іконка якої створена командою *patch*

Для зображення на іконці малюнків, створених іншими програмними продуктами, треба скористуватися командою

image (imread ('FileName.Extension')),

де FileName – ім’я графічного файла, а Extension – його розширення.

MATLAB підтримує такі типи графічних файлів: **.bmp* (*Bitmap*), **.hdf* (*Hierarchical Data Format*), **.jpg* (*Joint Photographic Experts Group*), **.pcx* (*Paintbrush*), **.png* (*Portable Network Graphics*), **.tif* (*Tagged Image File Format*), **.xwd* (*X Window Dump*).

Властивостями іконки керують опції, імена яких розташовані ліворуч, а значення обираються за допомогою *popup-menu*. Інформація про ці опції наведена в табл. 9.1.

Таблиця 9.1

Опція	Значення	Сенс
Icon Frame (*Рамка Іконки*)	*Visible*	Рамку видно
	Invisible	Рамку не видно
Icon Transparency (*Прозорість Іконки*)	*Opaque*	Іконка непрозора
	Transparent	Іконка прозора
Icon Rotation (*Обертання Іконки*)	*Fixed*	При повороті блока іконка не обертається
	Rotates	При повороті блока іконка обертається
Icon Units (*Система Координат*)	*Autoscale*	$x_{min} = \min(\mathbf{X}),\quad x_{max} = \max(\mathbf{X}),$ $y_{min} = \min(\mathbf{Y}),\quad y_{max} = \max(\mathbf{Y})$
	Normalized	$x_{min} = 0,\quad x_{max} = 1,\quad y_{min} = 0,\quad y_{max} = 1$
	Pixel	$x_{min} = 0,\quad x_{max} = block\ width,$ $y_{min} = 0,\quad y_{max} = block\ hight$

Ширину та висоту блока в пікселах при використанні стилю *Pixel* при визначенні системи координат можна розрахувати у полі *Drawing commands* за допомогою операторів

```
pos = get_param (gcb, 'Position'));
width = pos(3) – pos(1);    hight = pos(4) – pos(2).
```

Прикладом блока з невидимою рамкою є блок *Manual Switch*.

Якщо зробити іконку прозорою, то крізь неї можуть бути видні, наприклад, імена портів.

При використанні другого рівня маскування підсистема стає схожою на звичайний блок, тобто при подвійному щиглику на його піктограмі відкривається не розгорнута модель підсистеми, а вікно введення параметрів *Block Parameters*. Для того, щоб побачити розгорнуту модель такої замаскованої підсистеми треба «зазирнути під маску» виконанням команди *Edit → Look Under Mask* (*^U*).

У будь-який момент з будь-якої вкладки можна подивитися, як впливають дії, виконані в процесі маскування, на вигляд вікна *Block Parameters*. Для цього треба натиснути віртуальну кнопку *PreView* у нижній частині вікна редактора маскування.

Визначення параметрів замаскованої підсистеми здійснюється у вкладці Parameters&Dialog (рис. 9.6).

З лівого боку цієї вкладки знаходиться панель керування *Controls* параметрами *(Parameters),* дисплеєм *(Display)* та діями*(Actions)*.

Керування основними типами параметрів здійснюють 3 кнопки:

- *Edit* – введення константи, змінної або математичного виразу;
- *Checkbox* – вибір одного з двох альтернативних варіантів *on* або *off*, перший з яких діє при встановленому «прапорці», тобто при поміченому «галочкою» полі параметру, а другий – при невстановленому «прапорці», тобто при відсутності помітки в полі параметру;
- *Popup* – вибір варіанту з випадного меню.

Після вибору типу параметра у першому стовпчику *Type* таблиці *Dialog box* (центральна частина вкладки) з’являється номер параметра (напри-

клад, #1) з іконкою, що відповідає його типу. Тепер у другому стовпчику цього рядка необхідно ввести запрошення (*Prompt*) до введення параметра у вигляді тексту, а у третьому стовпчику (*Name*) – ім'я змінної, яка отримає значення після введення його у відповідь на запрошення у вікні параметрів замаскованого блока. Приклад вкладки для маскування моделі двигуна постійного струму, зображеної на рис. 9.1*а*, наведено на рис. 9.7.

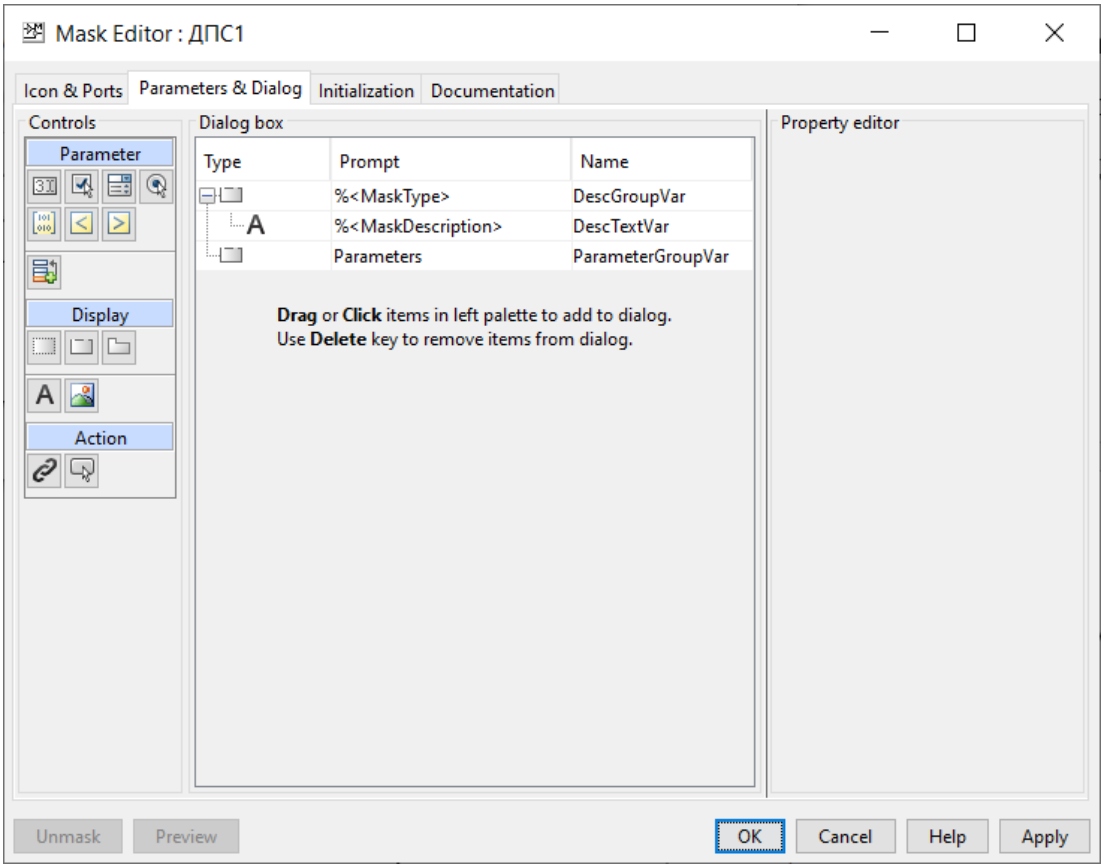


Рис. 9.6. Вкладка *Parameters & Dialog* редактора маскування

Значення обраного або введеного параметра може використовуватись як у чисельній формі, так і у форматі рядка символів. Спосіб обчислення значення параметра в режимі *Evaluate* (обчислити значення) пояснюється у табл. 9.2. Результати діалога (*Dialog box*) вкладки *Parameters & Dialog* відображаються в опції *Properties* панелі *Property editor*, розташованій праворуч. Інші опції цієї панелі *Attribute*, *Dialog* і *Layout*, призначені за замовчанням, можна змінити, але ці налаштування не є принциповими і краще їх залишити

недоторканими. Те ж саме відноситься і до опцій керування *Display* та *Actions* панелі *Controls*.

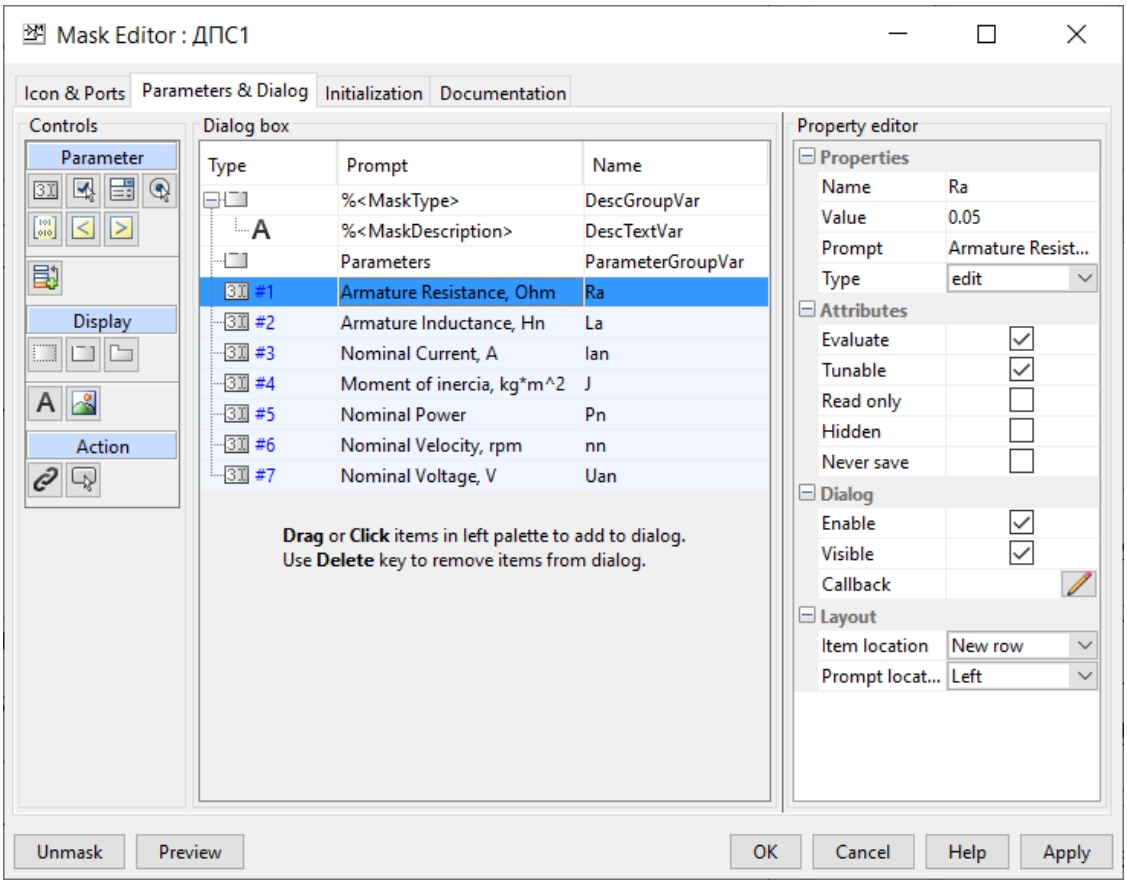


Рис. 9.7. Приклад вкладки *Parameters & Dialog* для маскування моделі двигуна постійного струму рис. 9.1*а*

Таблиця 9.2

Control type	***Numerical value***
Edit	Введена константа, обчислене значення введеної змінної або введеного виразу
Checkbox	1 – при встановленому «прапорці» (стан *on*) 0 – при відсутності «прапорця» (стан *off*)
Popup	Ціле число, що відповідає порядковому номеру (зверху вниз) обраного варіанту

У вкладці ***Initialization commands*** (рис. 9.8) вводяться команди ініціалізації, що пов’язують задані іменами змінних параметри замаскованої під-

системи з параметрами блока, що її маскує. Справа у тому, що змінні замаскованої підсистеми є локальними і не можуть бути визначені у робочому просторі (*Workspace*) пакету, якщо тільки вони не описані явно як глобальні змінні. Відповідно вони не можуть змінити значень однойменних змінних основного робочого середовища.

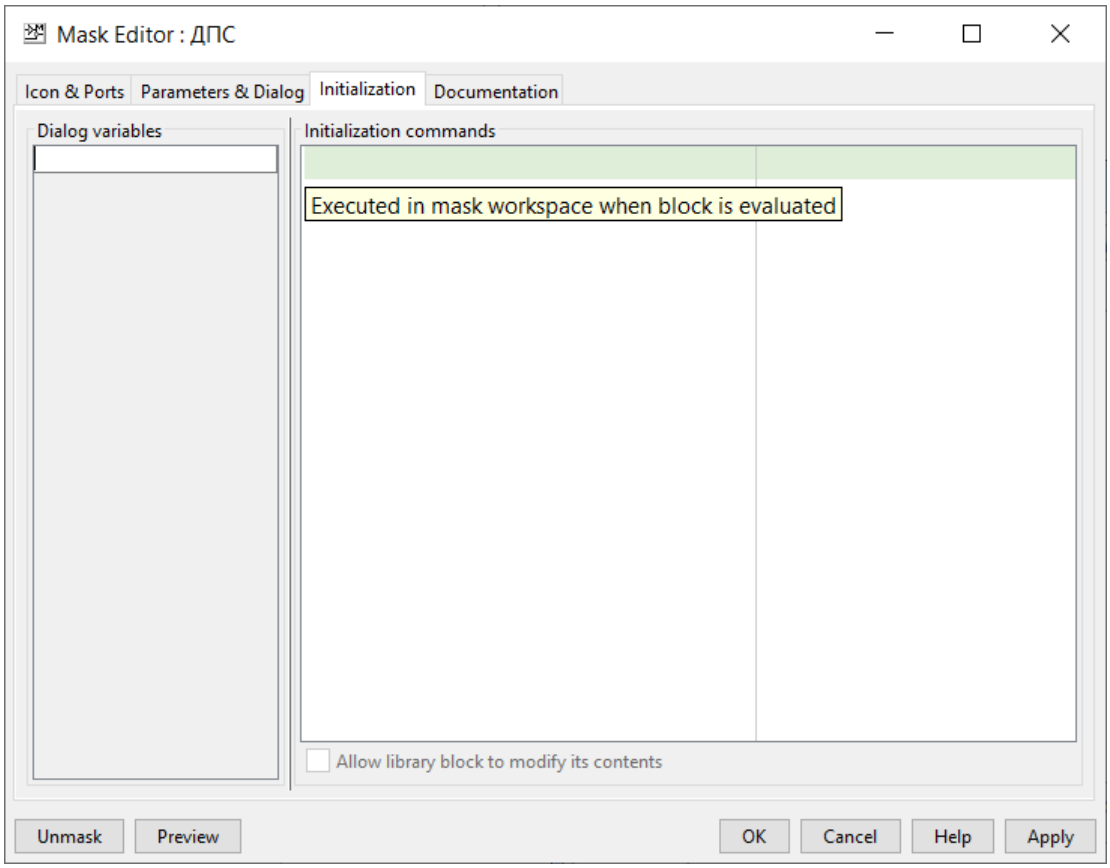


Рис. 9.8. Панель *Initialization* редактора маскування

Команди ініціалізації виконуються в таких випадках: при завантаженні моделі; при старті моделювання; при повороті блока; в усіх випадках, коли треба заново зобразити іконку блока, а команди зображення використовують результати команд ініціалізації.

Імена змінних, які використовуються в командах ініціалізації, відображаються у панелі *Dialog variables* (ліворуч) вкладки ініціалізації.

Приклад вкладки ініціалізації замаскованої підсистеми двигуна постій-

ного струму зі вхідними параметрами, визначеними на рис. 9.7, наведено на рис. 9.9.

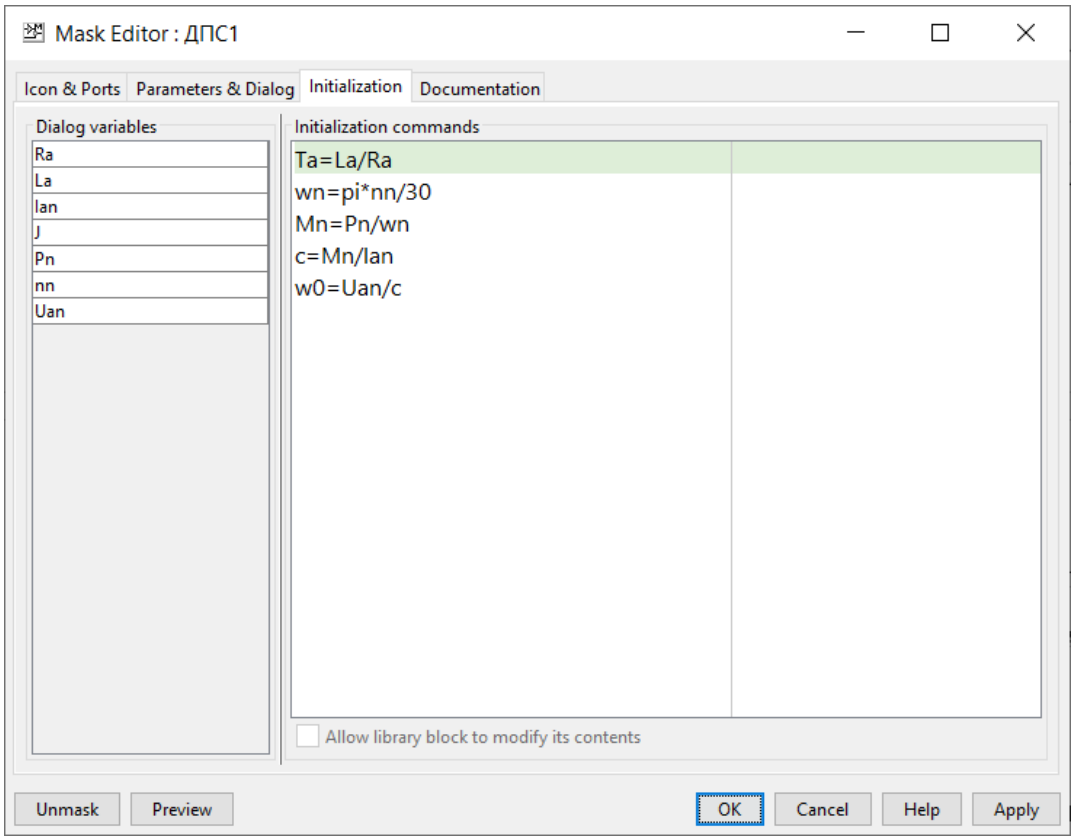


Рис. 9.9. Приклад вкладки ініціалізації замаскованої моделі двигуна постійного струму

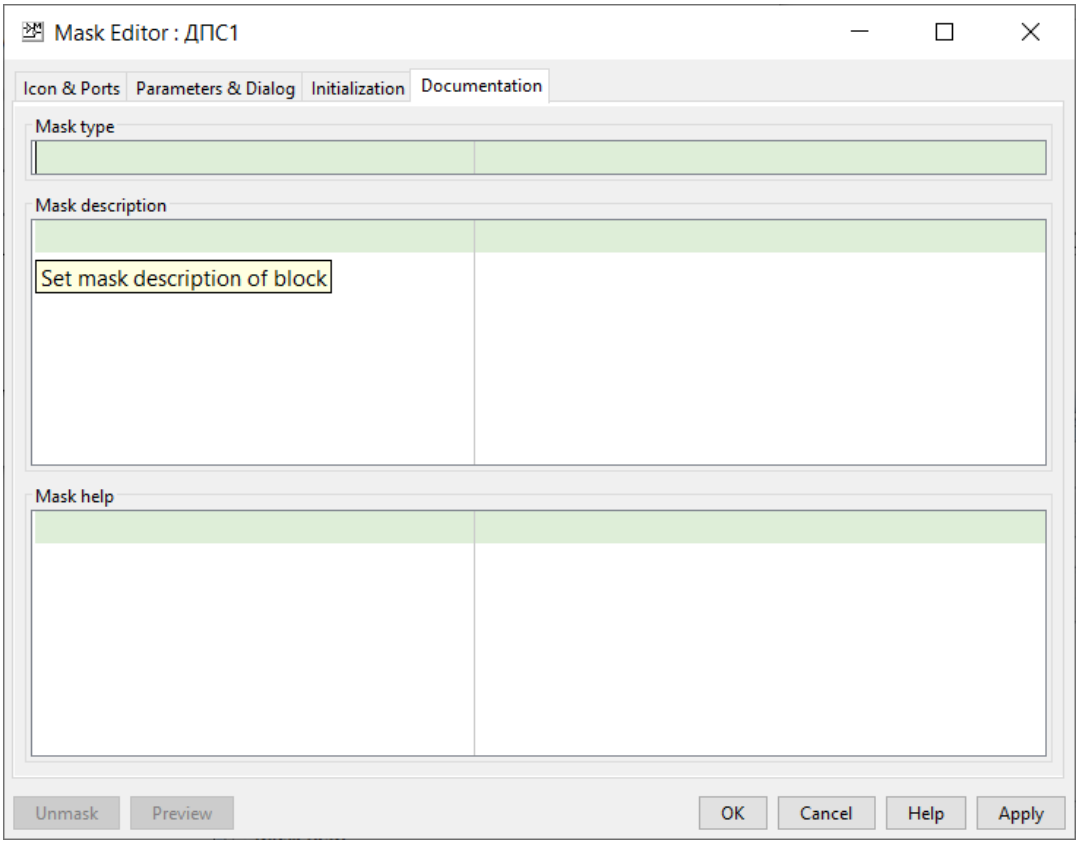


Рис. 9.10. Панель *Documentation* редактора маскування

У вкладці ***Documentation*** (рис. 9.10) визначаються 3 параметри: *Mask type*, *Mask description* та *Mask Help.*

У полі ***Mask type*** необхідно ввести тип блока. Пізніше цей тип з поміткою (*Mask*) буде відображено у вікні параметрів замаскованого блока.

В поле ***Mask description*** можна ввести короткий коментар про призначення блока та про його параметри, який буде відображений у вікні *Block Parameters*, а в поле ***Block Help*** – більш докладний опис блока, який буде виводитися на екран в *html*-форматі при натисканні клавіші *Help* в однойменному вікні.

Усі ці параметри є інформаційними і ніяк не впливають на роботу підсистеми.

Відмінити маскування можна командою *Unmask*, яку можна активізувати з кожної вкладки редактора маскування. У полі ***Mask type*** необхідно ввести тип блока. Пізніше цей тип з поміткою (*Mask*) буде відображено у вікні параметрів замаскованого блока.

Як відображаються результати маскування на вікні введення параметрів замаскованого блока, продемонстровано на рис. 9.11 на прикладі блока *From Workspace*, який має усі три перелічені вище типи визначення параметрів.

На жаль блок *From Workspace* є вбудованим, отже подивитися його вікно маскування не представляється можливим.

Як видно з рис. 9.11, вигляд вікна *Block Parameters* замаскованої підсистеми визначається змістом комірок *Mask type*, *Block description*, *Prompt*, *Variable* та *Control type*.

Відмінити маскування можна клавішею *Unmask*. При виконанні команди *Edit* → *Mask Subsystem* (^*M*) вікно маскування відновлює свої попередні установки.

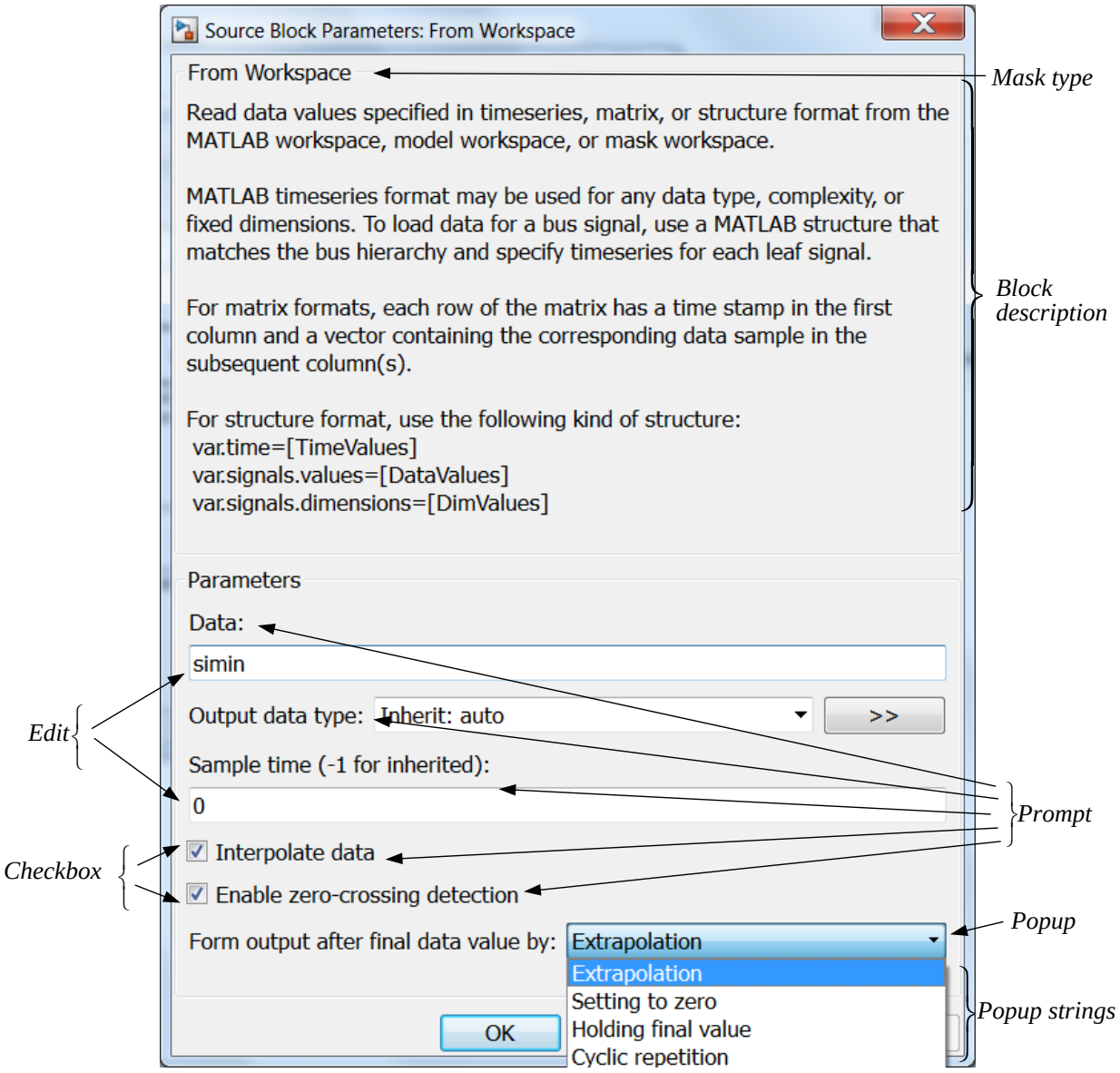


Рис. 9.11. Вікно введення параметрів блока *From Workspace*

9.3. Створення кнопок, що керують процесом виконання модельного експерименту

За допомогою блока *Subsystem* можна створити «кнопки», що керують процесом виконання модельного експерименту, тобто такі відокремлені блоки (без вхідних та вихідних портів), подвійний щиглик мишею на іконці яких дозволяє виконувати бажані оператори, функції або команди пакета *MATLAB*.

Звичайно користувачі-початківці створюють моделі, параметри яких

задають константами. В такому разі для досліджень впливу зміни якого-небудь параметру на властивості досліджуваного об’єкта доводиться відкривати вікна визначення параметрів усіх блоків, до яких входить варійований параметр, вручну змінювати значення цього параметру, виконувати моделювання через *Simulink*-меню і запам’ятовувати його результати. Таку послідовність дій багатократно повторюють, а потім обробляють результати модельного експерименту. Викладену методику не можна назвати раціональною при наявності розвинутої алгоритмічної мови і можливості виконувати моделювання в програмному режимі.

Більш ефективно дотримуватися такої методики:

- параметри моделі задавати іменами змінних;
- для ініціалізації моделі створити файл даних, в якому задаються або обчислюються значення цих змінних;
- для виконання файлу даних, не виходячи з вікна *Simulink*-моделі, створити «кнопку» з умовною назвою *Init*;
- при налагодженні моделі виконувати моделювання досліджуваної системи після процесу ініціалізації через *Simulink*-меню, фіксуючи результати блоками *Scope*;
- після того, як ви впевнитеся у працездатності моделі, треба розробити план модельного експерименту і *MATLAB*-програму або декілька програм, що його реалізують;
- для виконання розроблених програм, не виходячи з вікна *Simulink*-моделі, створити «кнопки» з назвою *Run-Plot*;
- наприкінці можна створити кнопку, активізація якої виводить в окремому вікні інформацію про модель та рекомендації щодо її застосування.

Модель, параметри якої задані іменами змінних, а не числами є більш наочною. При програмному визначенні параметрів зменшується час коригування параметрів та можливість помилок, особливо тоді, коли один і той же

параметр, як, наприклад, період дискретності, треба визначити в декількох блоках.

Програма модельного експерименту може отримувати у своєму складі цикли, в яких змінюються варійовані параметри або параметри, що керують ключовими елементами моделі. В цих циклах можна здійснювати розрахунок перехідних процесів, частотних характеристик, тощо, та обробляти отримані результати, зокрема, виводити їх у вигляді графіків.

Створення «кнопок» допомагає пов'язати модель з програмами, які її ініціюють та досліджують.

Цей процес можна виконати у такому порядку:

1) відкрити бібліотеку *Subsystems* і перетягти з неї у вікно досліджуваної моделі блок *Subsystem*;

2) відкрити блок *Subsystem*, знищити його вхідний та вихідний порти і лінію зв'язку поміж ними;

3) активізувати контекстуальне меню щигликом правої кнопки миші по піктограмі блока та обрати в ньому функцію *Block Properties*, що приведе до відчинення відповідного вікна;

4) у рядку *Open function* відчиненого вікна набрати ім'я файла або послідовність *MATLAB*-команд, які ви бажаєте виконати при натисканні створеної кнопки, після чого натиснути клавішу *OK*;

5) виділити блок і виконати з ним послідовно операції *Hide Name* і *Show Drop Shadow* функції *Format Simulink*-меню;

6) замаскувати підсистему (^*M*) виведенням в її іконці назви «кнопки», наприклад, `disp ('Init')`, остання команда набирається в полі *Drawing commands* вікна маскування;

7) зафарбувати «кнопку» яким-небудь (бажано світлим) кольором, наприклад, за допомогою операцій *Format → Background Color → Yellow.*

Власне кажучи, необхідними є тільки перша і третя операції. Усі

останні дії спрямовані на те, щоб придати «кнопці» презентабельний вигляд (див. рис. 9.12).

Для створення інформаційної «кнопки» після виконання пункту 2, тобто після знищення портів, треба у звільненому вікні набрати текст, який пояснює призначення моделі, або якусь іншу інформацію. Пункти 3 і 4 виконувати не треба, а в пункті 6, тобто при маскуванні підсистеми, треба вивести в іконці «кнопки» відповідний текст, наприклад, ‘Info’ або ‘?’.

Рис. 9.12. Вигляд „кнопки” *Init*

Контрольні питання та завдання

1. Які типи підсистем Ви знаєте?
2. Чим відрізняється замаскована підсистема від незамаскованої?
3. Як замаскувати підсистему?
4. Як змінити піктограму блока?
5. Що можна відображати в іконках?
6. Як організувати введення параметрів замаскованої підсистеми?
7. Як визначити «допомогу» для користування замаскованою підсистемою?
8. Як створити «кнопку», пов’язану з моделлю?
9. Для чого доцільно використовувати при моделюванні віртуальні кнопки?

10. МОДЕЛЮВАННЯ ДВИГУНА ПОСТІЙНОГО СТРУМУ З КЕРУВАННЯМ У КОЛІ ЯКОРЯ

10.1. Математичний опис і розробка структурних моделей

Двигун постійного струму (ДПС) з незалежним збудженням широко застосовується у регульованих електроприводах, що працюють у напружених повторно-короткочасних режимах і потребують забезпечення високої якості перехідних процесів.

При складанні його математичного опису знехтуємо розмагнічуючою дією реакції якоря, падінням напруги на щітках та тертям, а індуктивність і активний опір якірного кола та сумарний момент інерції вважатимемо постійними величинами. Регулювання швидкості здійснюватимемо зміною напруги якоря при постійній напрузі збудження. При прийнятих припущеннях ДПС описується лінійними диференційними та алгебраїчними рівняннями [13]:

$$U_{\text{я}}(t) - E_{\text{д}}(t) = \Delta U_{\text{я}}(t) = I_{\text{я}}(t)R_{\text{я}} + L_{\text{я}}\frac{dI_{\text{я}}(t)}{dt}, \tag{10.1}$$

$$M(t) = cI_{\text{я}}(t), \tag{10.2}$$

$$M(t) - M_{\text{с}}(t) = M_j(t) = J\frac{d\omega(t)}{dt}, \tag{10.3}$$

$$E_{\text{д}}(t) = c\omega(t), \tag{10.4}$$

де

$U_{\text{я}}$, $I_{\text{я}}$ – напруга та струм якоря;

$\Delta U_{\text{я}}$ – падіння напруги у якірному колі;

$R_{\text{я}}$, $L_{\text{я}}$ – активний опір та індуктивність якоря;

$E_{\text{д}}$, ω – електрорушійна сила (ЕРС) та кутова швидкість двигуна;

M, $M_{\text{с}}$, M_j, – електромагнітний момент двигуна, момент статичного опору та динамічний момент;

J – момент інерції;

$c = k\Phi_{зн}$; (10.5)

$\Phi_{зн}$ – номінальний потік збудження двигуна;

$k = \dfrac{pN}{2\pi a}$ – конструктивний коефіцієнт двигуна;

p, a, N – кількість пар полюсів, паралельних гілок та активних провідників відповідно.

Рівняння (10.1) називають ***рівнянням електромагнітної рівноваги якірного кола***, що складається за другим законом Кірхгофа, рівняння (10.3) – ***рівнянням руху*** двигуна, що складається за другим законом Кірхгофа для тіл обертального руху. Рівняння моменту (10.2) та ЕРС (10.4) свідчать про те, що при постійному потоку збудження електромагнітний момент двигуна є пропорційним струму якоря, а його ЕРС є пропорційною кутовій швидкості двигуна.

Для складання структурної схеми застосуємо 2 підходи.

Згідно з першим напишемо ДР у нормальній формі Коші:

$$\frac{dI_{я}(t)}{dt} = \frac{U_{я}(t) - E_{д}(t) - I_{я}(t)R_{я}}{L_{я}}, \qquad \frac{d\omega(t)}{dt} = \frac{M(t) - M_{с}(t)}{J},$$

або в операторній формі –

$$sI_{я}(s) = \frac{U_{я}(s) - E_{д}(s) - I_{я}(s)R_{я}}{L_{я}}, \qquad s\omega(s) = \frac{M(s) - M_{с}(s)}{J}.$$

Згідно з таким перетворенням ДР отримаємо структурну схему ДПС у просторі станів, як це показано на рис. 10.1.

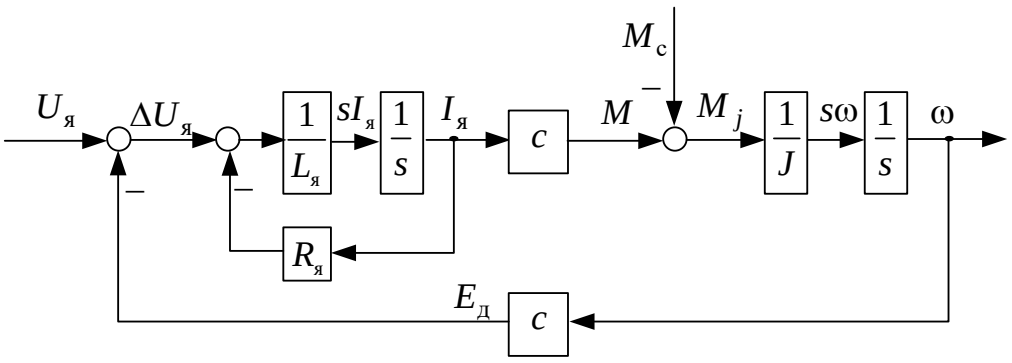


Рис. 10.1. Структурна схема ДПС у просторі станів

Об’єднаємо змінні стану та вхідні сигнали у вектори:

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \omega \\ I_я \end{bmatrix}, \qquad \mathbf{U} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} U_я \\ M_с \end{bmatrix}.$$

Тоді рівняння стану матимуть вигляд:

$$\begin{bmatrix} s\omega \\ sI_я \end{bmatrix} = \mathbf{AX} + \mathbf{BU} = \begin{bmatrix} \frac{c}{J} & 0 \\ -\frac{c}{L_я} & -\frac{R_я}{L_я} \end{bmatrix} \cdot \begin{bmatrix} \omega \\ I_я \end{bmatrix} + \begin{bmatrix} 0 & -\frac{1}{J} \\ \frac{1}{L_я} & 0 \end{bmatrix} \cdot \begin{bmatrix} U_я \\ M_с \end{bmatrix}.$$

Інший підхід полягає у тому, що ми в операторних рівняннях виконуємо приведення подібних сигналів та перегруповуємо їх так, щоб з одного боку від знаку рівності був тільки вихідний сигнал:

$$U_я(s) - E_д(s) = I_я(s)(R_я + L_я s) = R_я I_я(s)\left(1 + \frac{L_я}{R_я}\right)s, \tag{10.6}$$

$$M(s) - M_с(s) = M_j(s) = Js\omega(s). \tag{10.7}$$

З отриманих рівнянь визначаємо передавальні функції якірного кола і механічної частини двигуна як відношення зображень вихідного та вхідного сигналів:

$$W_я(s) = \frac{I_я(s)}{U_я(s) - E_д(s)} = \frac{I_я(s)}{\Delta U_я(s)} = \frac{1/R_я}{T_я s + 1}, \tag{10.8}$$

де $T_я$ – ***електромагнітна стала часу якірного кола:***

$$T_я = L_я / R_я. \tag{10.9}$$

$$W_{\text{м}}(s)=\frac{\omega(s)}{M(s)-M_{\text{с}}(s)}=\frac{\omega(s)}{M_j(s)}=\frac{1}{Js}. \qquad (10.10)$$

Передавальні функції від струму якоря до електромагнітного моменту та від кутової швидкості до проти-ЕРС двигуна є пропорційними ланками (див. рівняння (10.22), (10.24)):

$$W_{IM}(s)=\frac{M(s)}{I_{\text{я}}(s)}=c, \qquad W_{\omega E}(s)=\frac{\omega(s)}{E_{\text{д}}(s)}=c. \qquad (10.11)$$

Структурна схема, побудована з сукупності передавальних функцій (10.8)-(10.11), показана на рис. 10.2.

Обидві структурні схеми в загальному випадку є взаємозамінними. Перша з них отримує інформацію не тільки про струм якоря та кутову швидкість двигуна, а ще й про їх похідні, тобто про темп зміни струму $dI_{\text{я}}(t)/dt$ та про кутове прискорення $d\omega(t)/dt=\varepsilon(t)$. Крім того, Сімулінк-модель, побудована за першою схемою, може працювати з ненульовими початковими умовами.

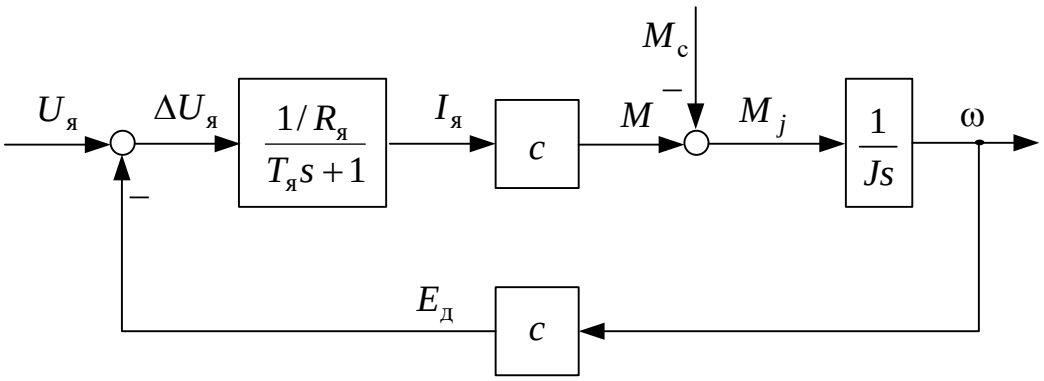


Рис. 10.2. Структурна схема ДПС, складена з передавальних функцій

10.2. Нормування структурних схем

Ще більше спростити структурну схему (СС) та зменшити в ній кількість параметрів можна шляхом нормування, тобто переходу від абсолютних одиниць до відносних. У якості базових величин обирають значення регульованих сигналів у характерних статичних режимах. Усі базові величини мають бути пов’язані між собою достатньо простими рівняннями статики.

Розглянемо статичні характеристики ДПС (рис.10.3) при постійній напрузі збудження ($\Phi_з = \Phi_{зн} = const$).

З розгляду статичних характеристик обираємо такі базові величини:

$$\omega_б = \omega_0, \quad U_{яб} = E_{дб} = U_{ян} = c\omega_0, \quad I_{яб} = I_{кз} = \frac{U_{ян}}{R_я}, \quad M_б = M_{кз} = cI_{кз}. \quad (10.12)$$

При цьому утворюється така система відносних одиниць:

$$\bar{\omega} = \frac{\omega}{\omega_0}, \quad \bar{U} = \frac{U}{U_{ян}}, \quad \bar{E}_д = \frac{E_д}{U_{ян}}, \quad \bar{I}_я = \frac{I_я}{I_{кз}}, \quad \bar{M} = \frac{M}{M_{кз}}. \quad (10.13)$$

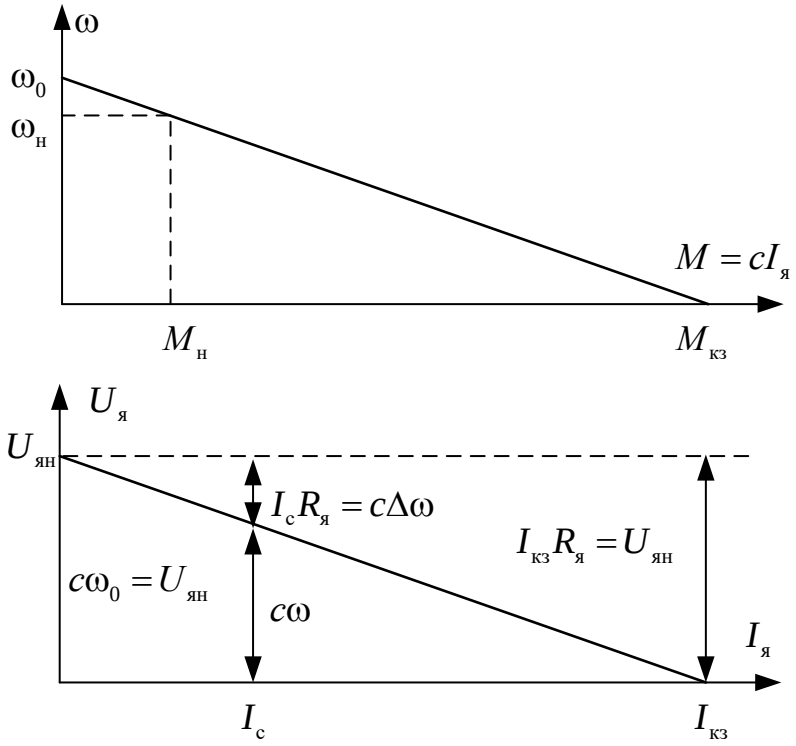


Рис. 10.3. Статичні характеристики ДПС при $\Phi_з = \Phi_{зн} = const$

Передавальні функції у в.о. розраховуємо за формулою (див. лекцію 2)

$$\bar{W}(s) = W(s)\frac{u_б}{y_б}.$$

Відповідно до неї знаходимо:

$$\bar{W}_{IM}(s) = c\frac{I_{кз}}{M_{кз}} = 1, \qquad \bar{W}_{\omega E}(s) = W_{\omega E}(s) = c\frac{\omega_0}{E_{д0}} = 1. \quad (10.14)$$

$$\bar{W}_{\text{я}}(s) = W_{\text{я}}(s)\frac{U_{\text{ян}}}{I_{\text{кз}}} = \frac{1/\cancel{R_{\text{я}}}}{T_{\text{я}}s+1}\cdot\cancel{R_{\text{я}}} = \frac{1}{T_{\text{я}}s+1}; \tag{10.15}$$

$$\bar{W}_{\text{м}}(s) = W_{\text{м}}(s)\frac{M_{\text{кз}}}{\omega_0} = \frac{M_{\text{кз}}}{J\omega_0 s} = \frac{1}{T_{\text{м}}s}. \tag{10.16}$$

В останньому рівнянні $T_{\text{м}}$ – ***електромеханічна стала часу двигуна:***

$$T_{\text{м}} = \frac{J\omega_0}{M_{\text{кз}}} = \frac{JR_{\text{я}}}{c^2}. \tag{10.17}$$

Фізичний сенс цієї сталої часу випливає із рівняння руху двигуна: електромеханічна стала часу дорівнює часу, за який двигун розігнався б до швидкості ідеального холостого ходу, якщо б до нього було прикладено момент короткого замикання. На рис. 10.4 проілюстровано наведене визначення та показано, як його можна використати для розрахунку часу розгону приводу з заданим моментом.

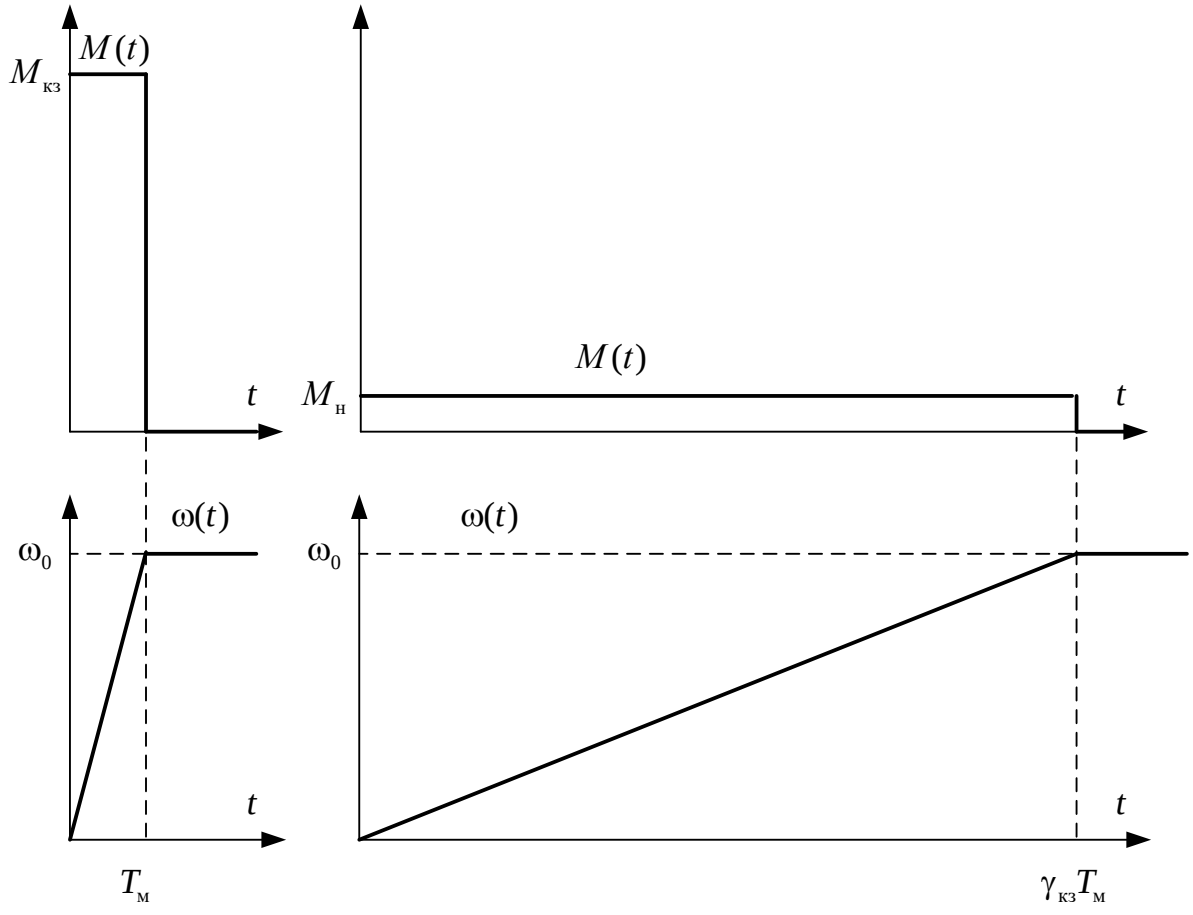


Рис. 10.4. Ілюстрація фізичного сенсу електромеханічної сталої часу

На цьому рисунку використано параметр $\gamma_{кз}$, який називають кратністю моменту (або струму) короткого замикання:

$$\gamma_{кз} = \frac{M_{кз}}{M_{н}} = \frac{I_{кз}}{I_{ян}}. \tag{10.18}$$

Цей параметр визначає жорсткість механічної характеристики:

$$\frac{\omega_0 - \omega_{н}}{\omega_0} = \frac{\Delta\omega_{сн}}{\omega_0} = \frac{1}{\gamma_{кз}}. \tag{10.19}$$

Структурна схема ДПС у в.о. зображена на рис. 10.5.

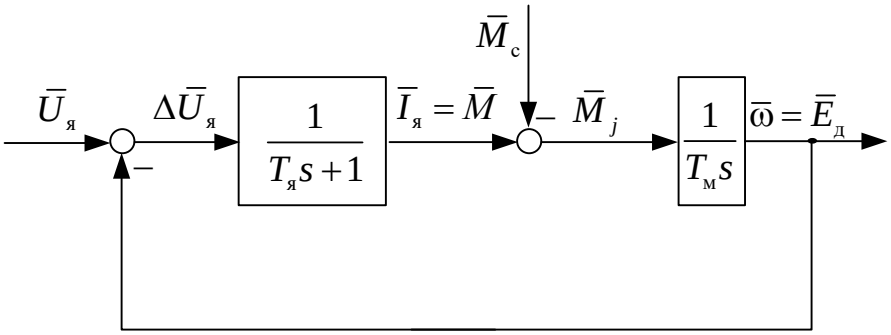


Рис. 10.5. Структурна схема ДПС у відносних одиницях (10.13)

Перевага останньої схеми – сама проста, мінімальна кількість параметрів. Недолік – струм та момент подані в долях струму та моменту КЗ. Якщо ми хочемо задавати статичний момент та візуалізувати струм і момент в долях номінальних значень, тобто

$$\bar{i}_{я} = \frac{I_{я}}{I_{ян}}, \quad \bar{m} = \frac{M}{M_{н}}, \tag{10.20}$$

то перехід від однієї системи в.о. до іншої здійснюємо за формулами:

$$\bar{i}_{я} = \bar{I}_{я}\gamma_{кз}, \quad \bar{M}_{с} = \frac{\bar{m}_{с}}{\gamma_{кз}}. \tag{10.21}$$

Структурна схема, в якій використано дві системи в.о., показана на рис. 10.6, а СС у другій системі в.о. – на рис. 10.7.

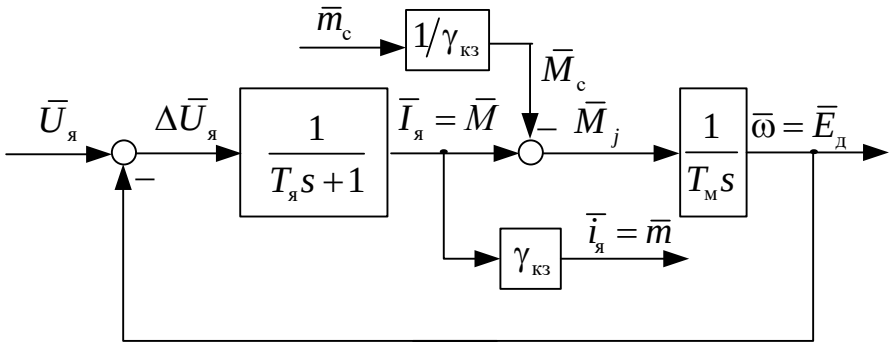


Рис. 10.6. Нормована структурна схема ДПС з двома системами в.о.

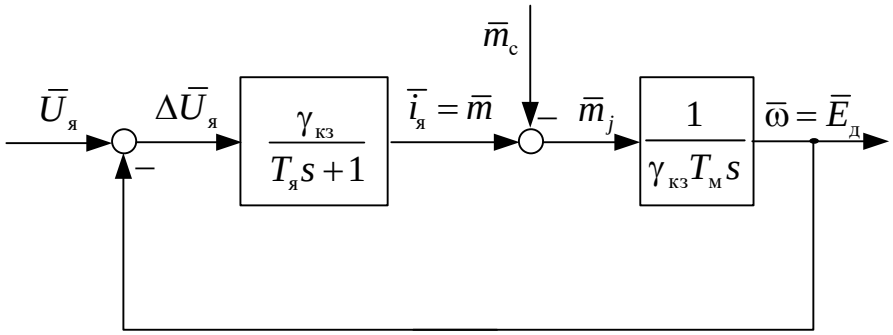


Рис. 10.7. Структурна схема ДПС у відносних одиницях (10.20)

10.3. Визначення та аналіз передавальних функцій

Для оцінювання адекватності розроблених моделей та виявлення у них логічних помилок визначимо передавальні функції досліджуваного двигуна в абсолютних одиницях:

$$\frac{E_д(s)}{U_я(s)} = \frac{M(s)}{M_с(s)} = \frac{1}{T_м T_я s^2 + T_м s + 1}, \qquad \frac{\omega(s)}{U_я(s)} = \frac{1/c}{T_м T_я s^2 + T_м s + 1}, \tag{10.22}$$

$$\frac{M(s)}{U_я(s)} = \frac{(J/c)s}{T_м T_я s^2 + T_м s + 1}, \qquad \frac{I_я(s)}{U_я(s)} = \frac{(J/c^2)s}{T_м T_я s^2 + T_м s + 1}, \tag{10.23}$$

$$\frac{\omega(s)}{M_с(s)} = -\frac{R_я}{c^2} \cdot \frac{(T_я s+1)}{T_м T_я s^2 + T_м s + 1}, \tag{10.24}$$

З (10.24) випливає, що ***реакція ЕРС і швидкості на стрибок напруги якоря збігається з реакцією електромагнітного моменту та струму якоря на стрибок статичного моменту та на лінійну зміну напруги якоря***.

В усталеному режимі (s=0)

$$\Delta\omega_c = -\frac{M_{\text{с}}R_{\text{я}}}{c^2} = -\frac{M_{\text{с}}R_{\text{я}}}{(k\Phi)^2};\ \ M_\infty = M_{\text{с}};\ \ I_{a\infty} = \frac{M_{\text{с}}}{c} = \frac{M_{\text{с}}}{k\Phi}\ \ ,$$

$$\omega_0 = \frac{U_{\text{я}}}{c};\quad \omega_\infty = \frac{U_{\text{я}}}{c} - \frac{M_{\text{с}}R_{\text{я}}}{c^2} = \omega_0 - \Delta\omega_{\text{с}};\ \ \tau_{\text{з}} = T_{\text{м}},$$

де $\tau_{\text{з}}$ – час запізнення відносної швидкості від відносної напруги при лінійній зміні напруги.

Знайдемо корені характеристичного рівняння

$$T_{\text{м}}T_{\text{я}}s^2 + T_{\text{м}}s + 1 = 0.$$

$$s_{1.2} = \frac{-T_{\text{м}} \pm \sqrt{T_{\text{м}}^2 - 4T_{\text{м}}T_{\text{я}}}}{2T_{\text{я}}} = \frac{-T_{\text{м}} \pm \sqrt{T_{\text{м}}(T_{\text{м}} - 4T_{\text{я}})}}{2T_{\text{я}}}. \qquad (10.25)$$

$$\frac{1}{T_{\text{м}}T_{\text{я}}s^2 + T_{\text{м}}s + 1} = \frac{1}{T_k^2 s^2 + 2\xi T_k s + 1}$$

$$T_{\text{м}}T_{\text{я}} = T_k^2,\ \ T_{\text{м}} = 2\xi T_k$$

$$T_k = \sqrt{T_{\text{м}}T_{\text{я}}},\quad \xi = \frac{T_{\text{м}}}{2T_k} = \frac{T_{\text{м}}}{2\sqrt{T_{\text{м}}T_{\text{я}}}} = \frac{1}{2}\sqrt{\frac{T_{\text{м}}}{T_{\text{я}}}}$$

Із (10.25) випливає, що ***при*** $T_{\text{м}} \geq 4T_{\text{я}}$ ***ДПС має 2 дійсних від'ємних полюси, тобто є аперіодичною ланкою другого порядку, для якої характерний перебіг перехідних процесів без перерегулювань. При*** $T_{\text{м}} < 4T_{\text{я}}$ ***полюси стають комплексно спряженими, що перетворює двигун на коливальну ланку з коефіцієнтом демпфірування***

$$\xi = \frac{1}{2}\cdot\sqrt{\frac{T_{\text{м}}}{T_{\text{я}}}}, \qquad (10.26)$$

який дорівнює косинусу від кутів, що утворюють радіус-вектори полюсів з дійсною віссю, і визначає величину перерегулювання перехідної функції коливальної ланки:

$$\varphi = \arccos(\xi);\qquad \sigma = \exp(-\pi\cdot ctg(\varphi))\cdot 100\%. \qquad (10.27)$$

Графік залежності перерегулювання від коефіцієнта демпфірування

коливальної ланки наведено на рис. 10.8.

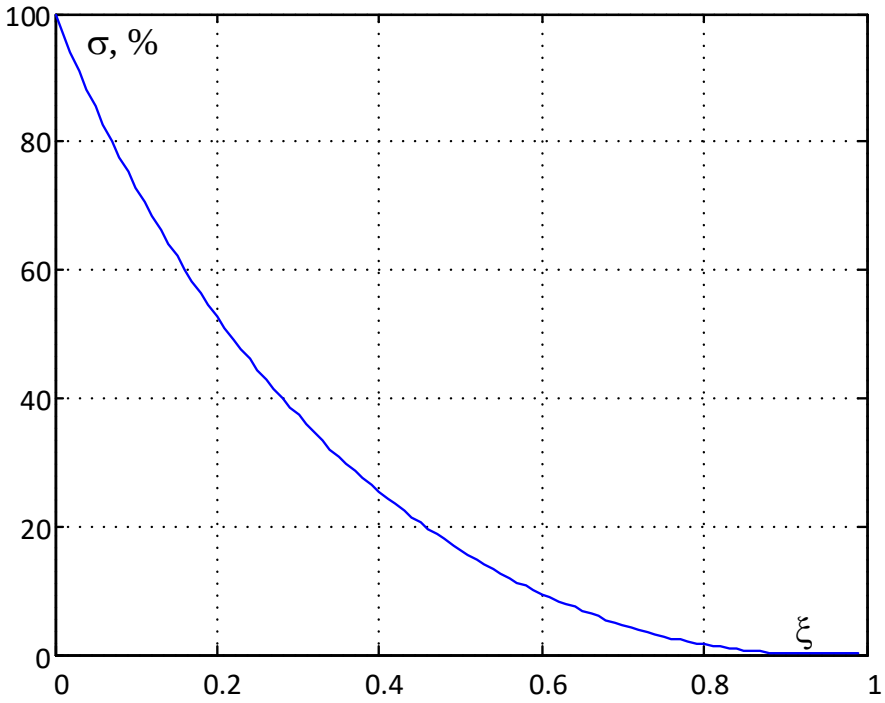


Рис. 10.8. Залежність перерегулювання перехідної функції ДПС від його коефіцієнта демпфірування

10.4. Моделювання та аналіз перехідних процесів ДПС в режимі прямого пуску

Simulink-модель ДПС в а.о. для дослідження реакції на стрибок напруги якоря з 0 до номінального значення показана на рис. 10.9, а перехідні процеси в цьому режимі – на рис. 10.10 [3].

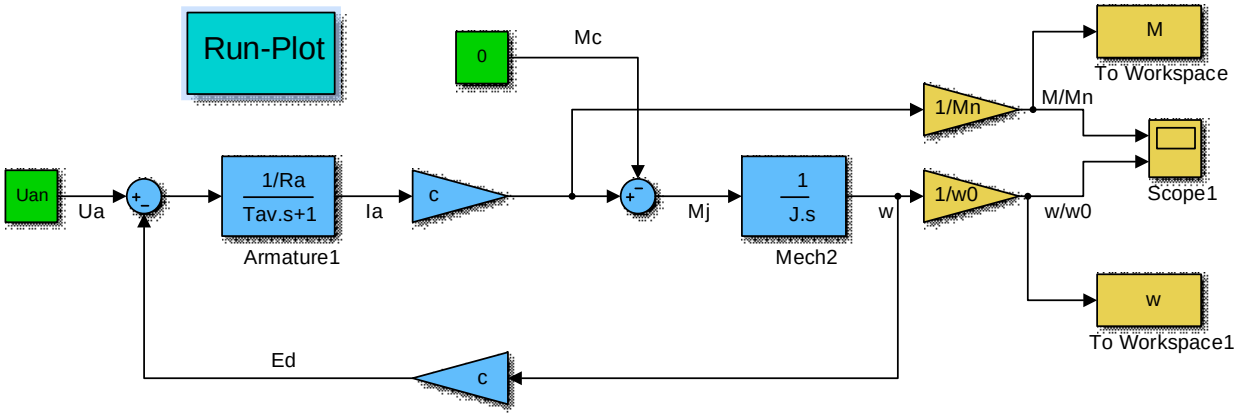


Рис.10.9. *Simulink*-модель ДПС для дослідження прямого пуску

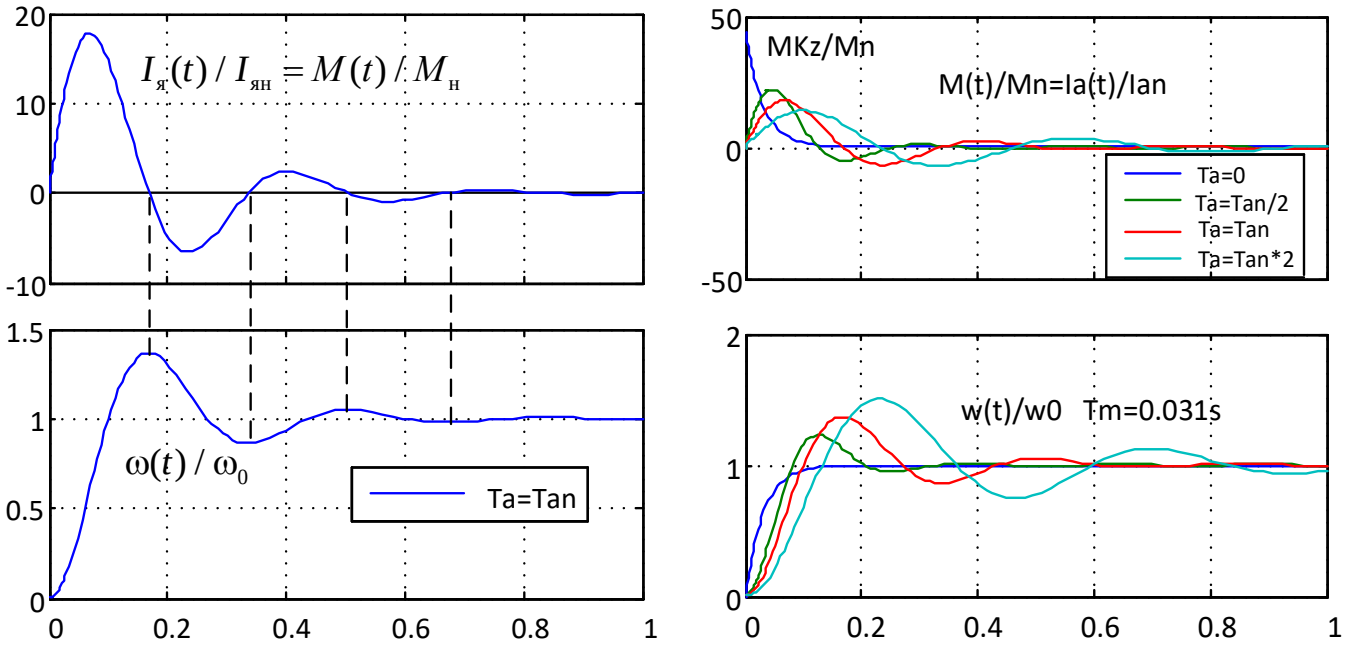


Рис. 10.10. Перехідні процеси при прямому пуску ДПС

Як бачимо, в досліджуваному режимі струм якоря і момент при зменшенні індуктивності якоря наближається до відповідних величин у режимі КЗ. Збільшення індуктивності знижує темп наростання цих величин і зменшує пікові значення, але вони все ж залишаються не припустимо великими. Цей віртуальний експеримент доводить необхідність регулювання напруги якоря.

10.5. Моделювання та дослідження властивостей ДПС в режимі реостатного пуску

Альтернативою прямому пуску в нерегульованому електроприводі є реостатний пуск. Для моделювання реостатного пуску зручну використовувати модель, побудовану на основі структурної схеми ДПС у просторі станів (рис. 10.1), в якій опор якоря присутній у явному вигляді (пропорційна ланка).

Для моделювання реостатного пуску в цій схемі достатньо пропорційну ланку з постійним коефіцієнтом $R_я$ замінити блоком множення струму на змінну величину $R_{я\ sum}$, яку необхідно сформувати при розрахунку пускового реостату у функції часу, швидкості двигуна або струму якоря.

Для імітації стрибкоподібної зміни сумарного опору якірного кола можна скористатися блоками *Look Up Table* або *Interpreted Matlab Fn*.

У першому випадку необхідно врахувати, що в таблицях не можна повторювати однакові значення аргументів двічі, тобто стрибкоподібну зміну вихідного сигналу можна сформувати тільки приблизно шляхом додавання у вектор аргументів додаткових значень.

У другому випадку необхідно аргументи і значення табличних функцій в головній програмі та в інтерпретованих функціях користувача описати як глобальні. Цей опис повинен передувати операціям з цими даними.

Модель реостатного пуску ДПС зображена на рис. 10.11, статичні характеристики подані на рис. 10.12, а перехідні процеси при пуску в функції швидкості, струму і часу подані на рис. 10.13-10.15 відповідно.

Наведені перехідні процеси демонструють різницю між ідеалізованими графіками реостатного пуску, розрахованими без урахування інерційності якірного кола, і реальними. При зменшенні активного опору підвищується електромагнітна стала часу якоря, що приводить до зменшення пікових значень струму якоря.

MATLAB-функції, застосовані для моделювання пускових реостатів, представлені у табл. 10.1

Таблиця 10.1

Пуск у функції швидкості	Пуск у функції часу	Пуск у функції струму

function Ras = Rasw(u)
% Ras=f(w)
global R W
n=length(R);
Ras=R(1);
for i=2:n
  if u>W(i), Ras=R(i); end
end
end
	function Ras = Rast(u)
	% Ras=f(Ia)
	global R tp
	n=length(R);
	Ras=R(1);
	for i=2:n
	  if u>tp(i), Ras=R(i);
	end
	end
	end
		function Ras = RasI(u)
		% Ras=f(nst)
		u=round(u);
		global R
		n=length(R);
		if u<=n, Ras=R(u);
		  else Ras=R(n);
		end
		end

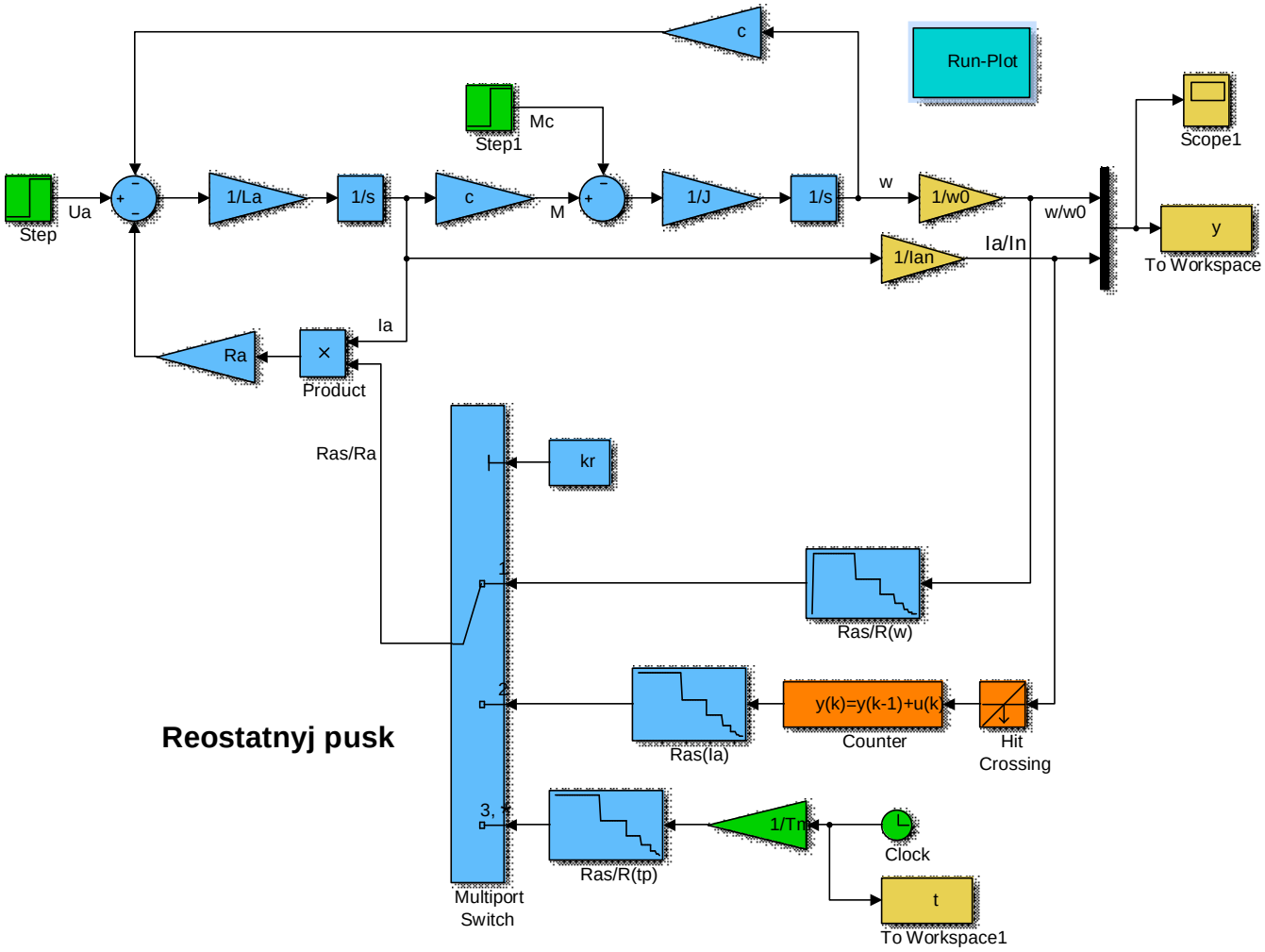


Рис. 10.11. *Simulink*-модель ДПС для дослідження реостатного пуску

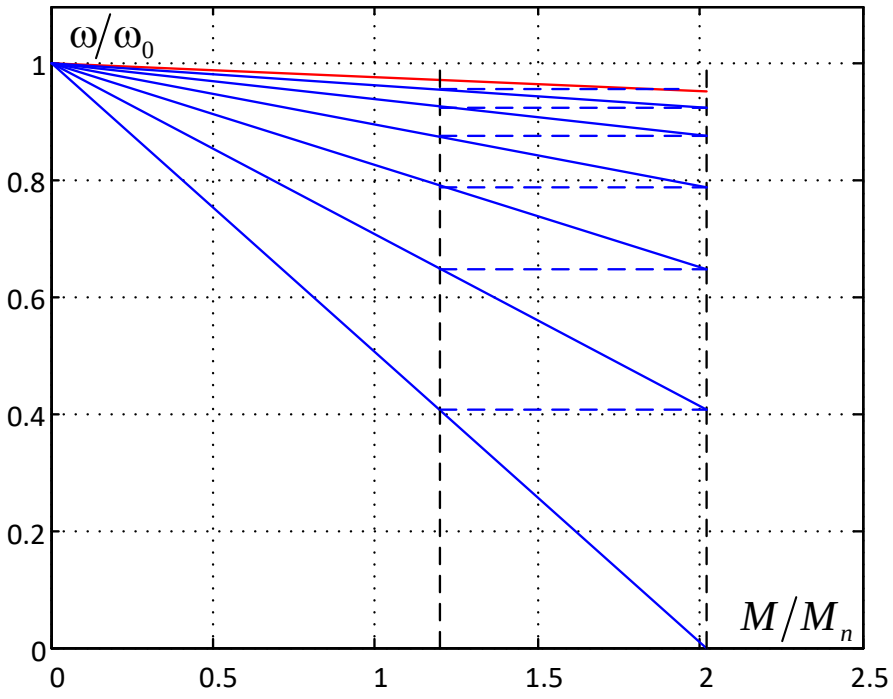


Рис. 10.12. Статичні механічні характеристики ДПС при реостатному пуску

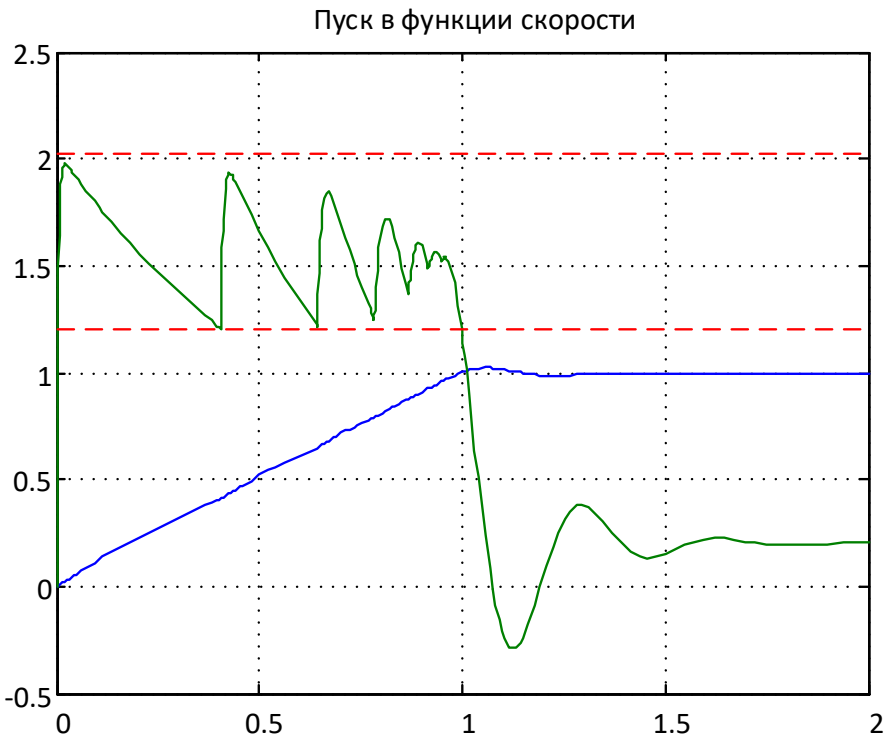


Рис. 10.13. Перехідні процеси при реостатному пуску ДПС у функції швидкості

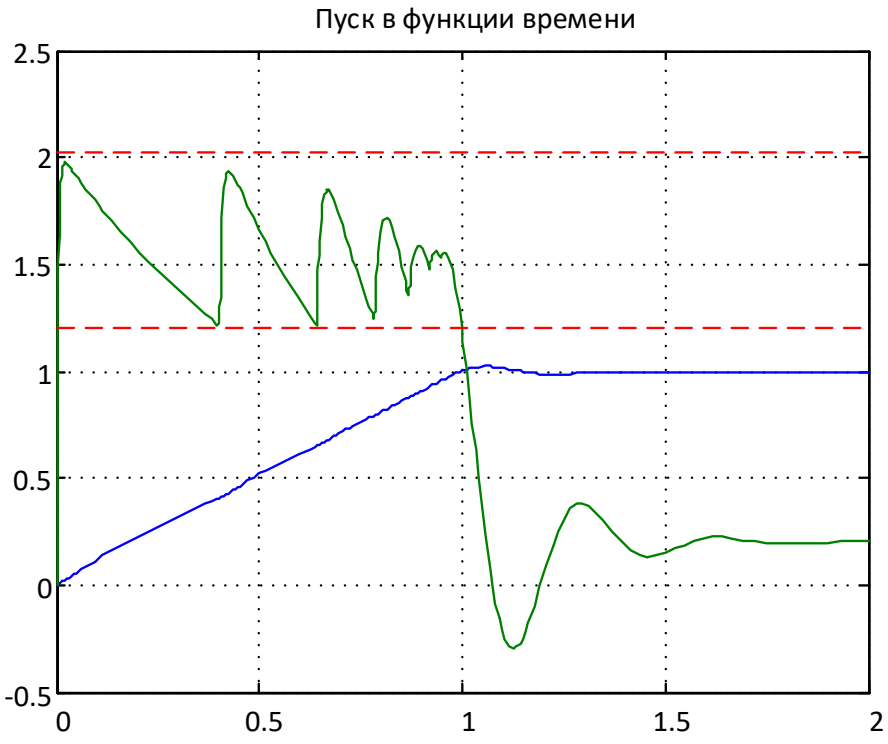


Рис. 10.14. Перехідні процеси при реостатному пуску ДПС у функції часу

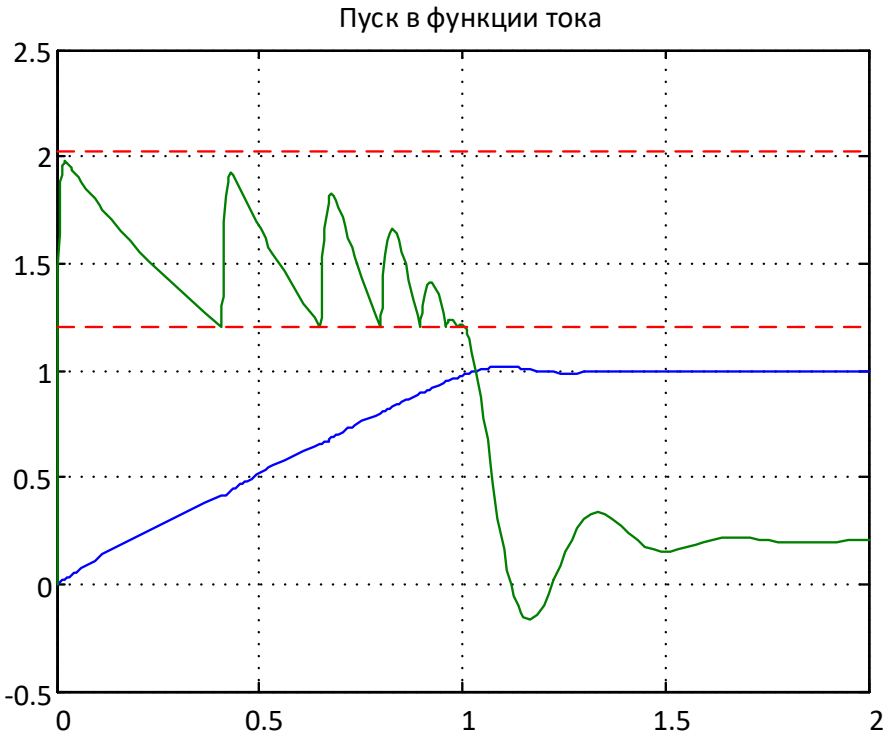


Рис. 10.15. Перехідні процеси при реостатному пуску ДПС у функції струму

Для моделювання пуску в функції струму застосовано блок *Hit Crossing*, який формує імпульси одиничної амплітуди при перетині вхідним сигналом заданого рівня в заданому напрямку. В даному випадку фіксувалися моменти перетину струмом якоря рівня нижньої межі пускового струму.

Для підрахунку кількості перетинів сконструйовано блок *Counter* (Лічильник), поданий у вигляді замаскованої підсистеми. Її зміст показано на рис. 10.16.

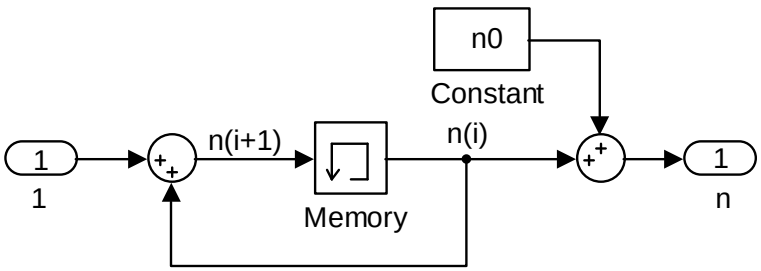


Рис.10.16. Розгорнута модель блока *Counter*

Блок *Memory* здійснює затримку на 1 крок чисельного інтегрування, що дозволяє здійснювати накопичення суми.

10.6. Моделювання ДПС при використанні задатчиків інтенсивності для формування напруги якоря

Реостатний пуск є застарілим та дуже енергозатратним способом регулювання швидкості. Альтернативою йому є регулювання швидкості шляхом регулювання напруги якоря. Щоб при такому регулюванні запобігти великим піковим значенням струму якоря, необхідно змінювати напругу не стрибком, а плавно. Одним з найпростіших законів зміни напруги, що дозволить знизити момент і струм у пуско-гальмівних режимах є лінійний закон. ***Пристрій, що формує завдання на швидкість з обмеженням прискорення, називають задатчиком інтенсивності (ЗІ).***

Для застосування такого методу керування необхідно живити двигун від підсилювача потужності. В сучасних систем регульованого електроприводу таким підсилювачем є тиристорний перетворювач постійного струму (ТП), який з точки зору ТАК можна представити у вигляді аперіодичної ланки з великим коефіцієнтом підсилювання (20-40) та малою сталою часу (4-6 мс):

$$W_{\text{ТП}}(s)=\frac{U_{\text{я}}(s)}{U_{\text{к}}(s)}=\frac{k_{\text{П}}}{T_{\text{П}}s+1}$$

При дослідженні такого способу регулювання швидкості можна знехтувати інерційністю ТП. Модель двигуна, що відпрацьовує задану трапецоїдальну тахограму показано на рис. 10.17, а перехідні процеси – на рис. 10.18.

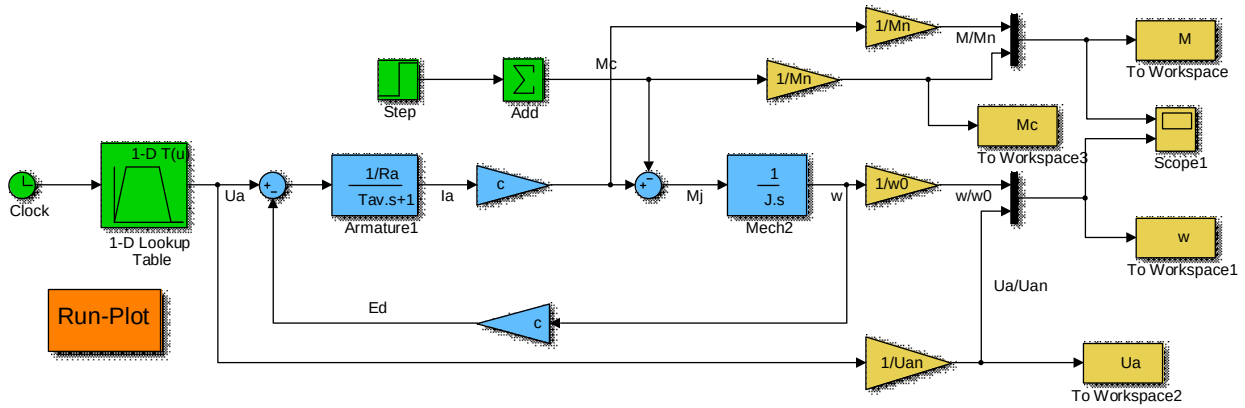


Рис. 10.17. Модель роботи ДПС за трапецоїдальною тахограмою

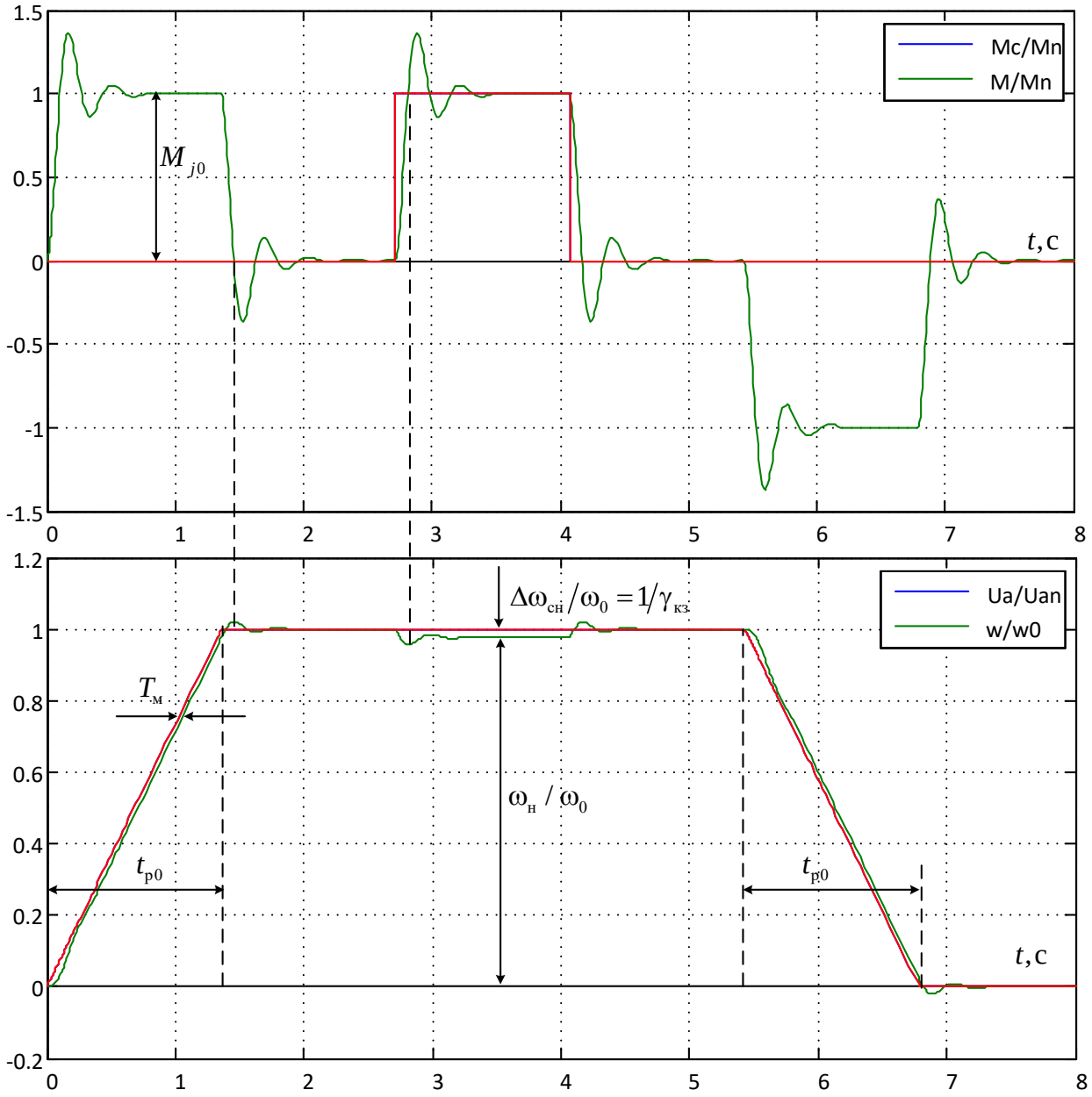


Рис. 10.18. Перехідні процеси в ДПС при лінійній зміні напруги і стрибкоподібній зміні моменту навантаження

Час зміни напруги від 0 до номінального значення при розгоні і навпаки при гальмуванні t_{p0} ***визначає задане прискорення*** $\varepsilon_0 = \omega_0 / t_{p0}$ ***і величину динамічного моменту*** $M_{j0} = J\varepsilon_0 = J\omega_0 / t_{p0}$ ***в пуско-гальмівних режимах.***

Отже

$$t_{p0} = \frac{J\omega_0}{M_{j0}}. \qquad (10.28)$$

Якщо потрібна інформація не тільки про напругу якоря, яка по суті є завданням на швидкість, а ще й про завдання на прискорення двигуна, то модель задавального пристрою може мати вигляд рис. 10.19.

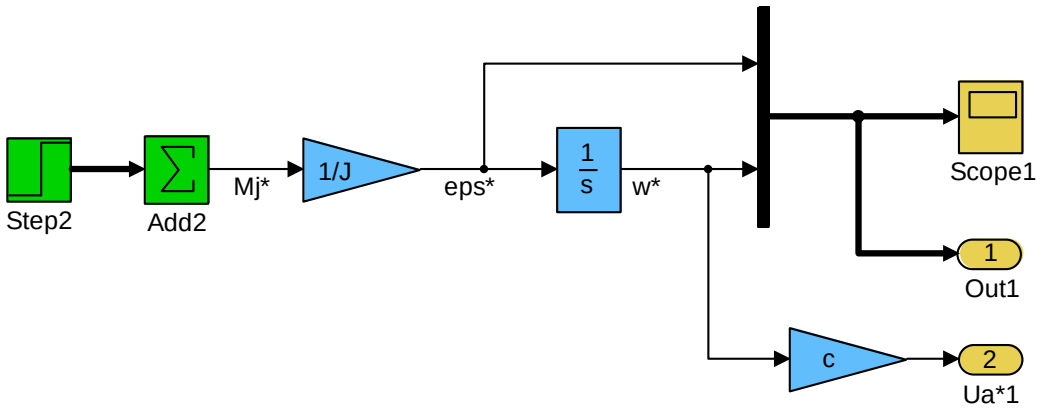


Рис. 10.19. Розімкнена модель типового ЗІ

Можна сформувати завдання на зміну напруги якоря так, щоб обмежувати не тільки прискорення, а й ривок (рис. 10.20):

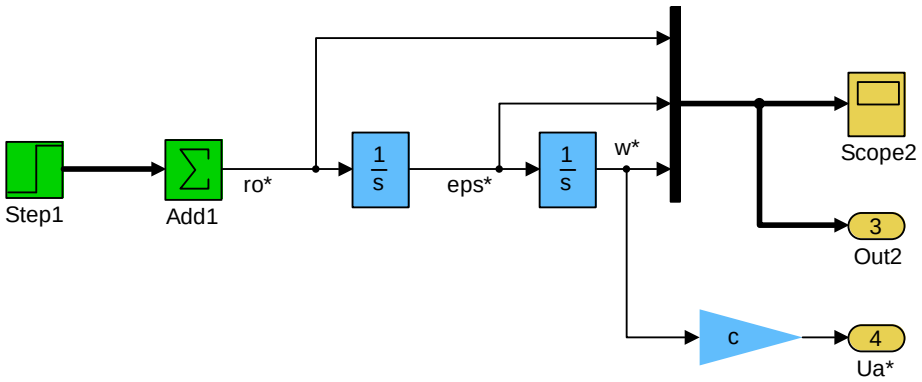


Рис. 10.20. Розімкнена модель ЗІ з обмеженням ривка

Перехідні процеси в описаних вище ЗІ показані на рис. 10.21, 10.22 відповідно.

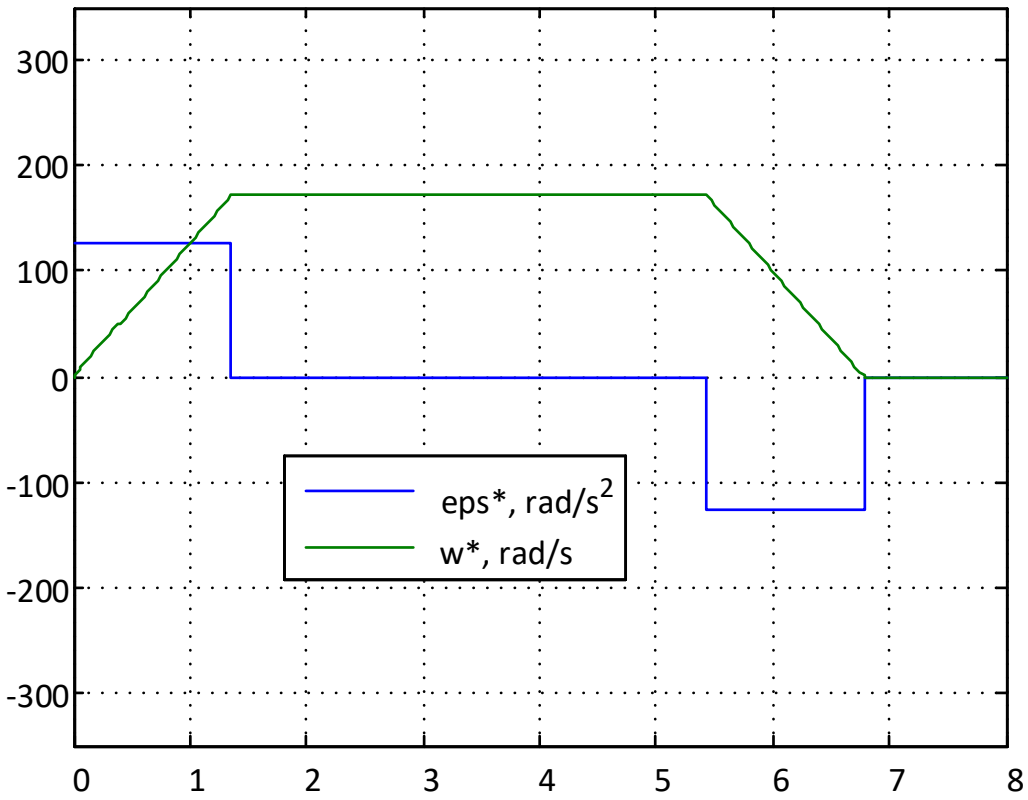


Рис. 10.21. Перехідні процеси типового ЗІ

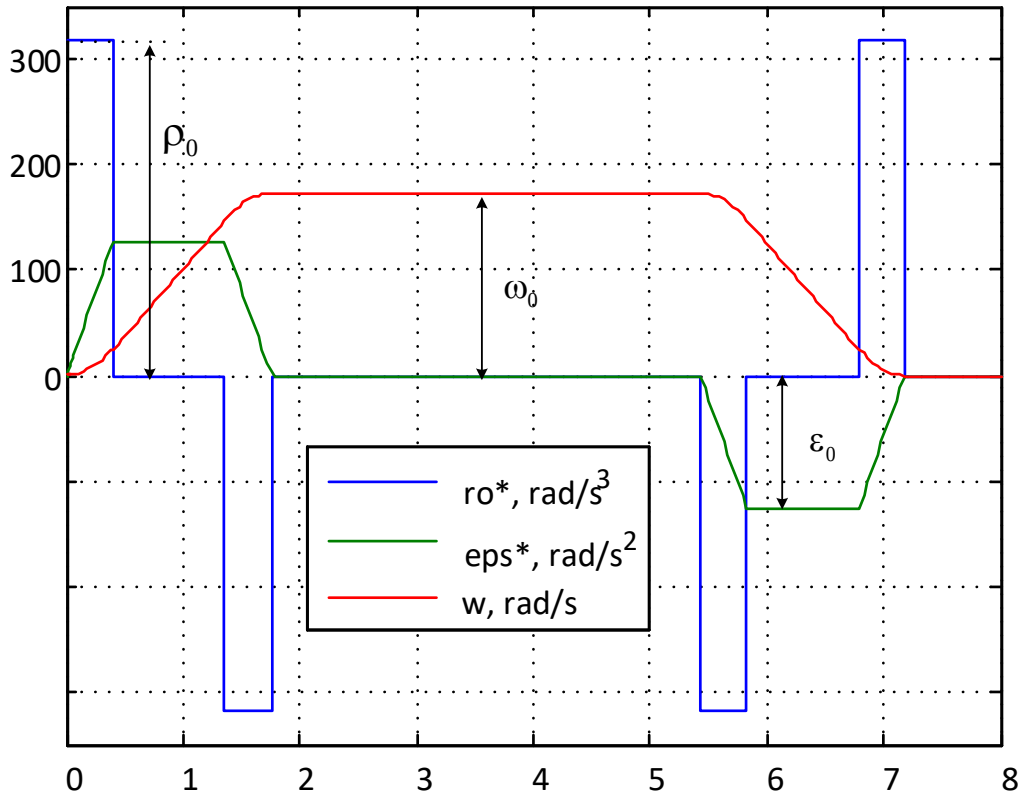


Рис. 10.22. Перехідні процеси ЗІ з обмеженням прискорення і ривка

Останню ***тахограму з комбінацією лінійних та параболічних ділянок називають також S-подібною.***

Розрахунок бажаного ривка виконується з умови обмеження похідної від прискорення і відповідно похідних від динамічних складових моменту і струму якоря:

$$\rho_0 = \frac{\varepsilon_0}{t_\varepsilon} = \frac{M_{j0}}{Jt_\varepsilon}, \tag{10.29}$$

де t_ε – час зміни завдання на прискорення за лінійним законом.

Перехідні процеси в ДПС при відпрацюванні *S*-подібної тахограми показані на рис. 10.23.

Наявність в ЗІ інтеграторів дає можливість подавати на вхід системи не тільки основне завдання, а ще і його похідні, що дає можливість компенсувати інерційність об'єкта керування і підвищити точність відпрацювання об'єктом заданого вхідного сигналу (часто кажуть «точність відпрацювання заданих траєкторій»).

Структурна схема такого ЗІ показана на рис. 10.24.

Такий спосіб керування в ТАК називають «комбіноване керування за задавальною дією» (на відміну від принципу керування за відхиленням – керування з від'ємним зворотним зв'язком без паралельних коригуючих зв'язків).

Оскільки ДПС є об'єктом другого порядку, а ЗІ має 2 інтегратори, то інерційність об'єкта можна компенсувати не частково, а повністю:

Передавальна функція представленого ЗІ має вид:

$$W_{зі}(s) = \frac{T_м T_я s^2 + T_м s + 1}{s^2}. \tag{10.49}$$

Як бачимо поліном в чисельнику дорівнює характеристичному поліному двигуна і повністю його компенсує.

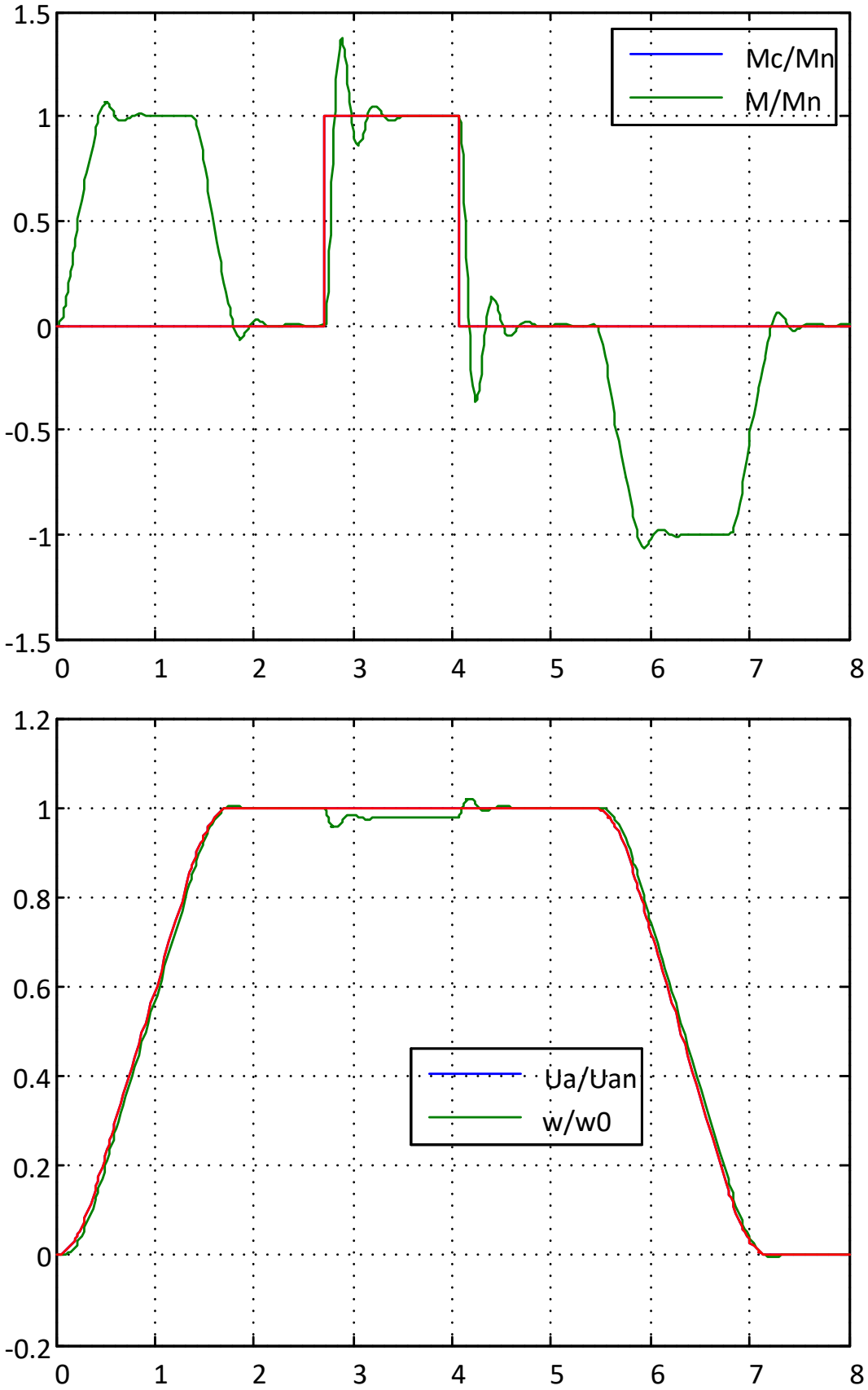


Рис. 10.23. Перехідні процеси в ДПС при відпрацюванні
S-подібної тахограми при керуванні за відхиленням

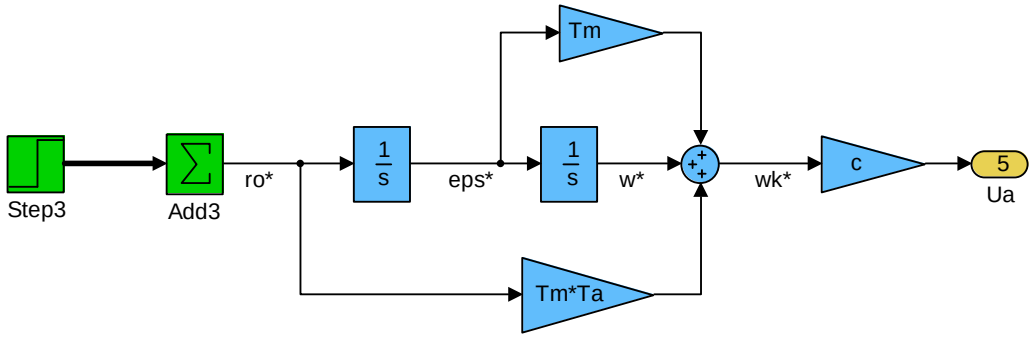


Рис. 10.24. Розімкнена модель ЗІ з обмеженням ривка і паралельними коригуючими зв'язками, що реалізують принцип комбінованого керування за задавальною дією

Для того, щоб не розраховувати часи перемикання тахограм, коли помилка в розрахунках веде до похибки формування усталеного значення швидкості, застосовують ЗІ з замкненою структурою, моделі яких показано на рис. 10.25, 10.26. Перехідні процеси в ДПС при керуванні від ЗІ рис. 10.24 або рис. 10.26 показані на рис. 10.27.

Як бачимо, у цьому випадку, задавальна дія відпрацьовується двигуном без похибки.

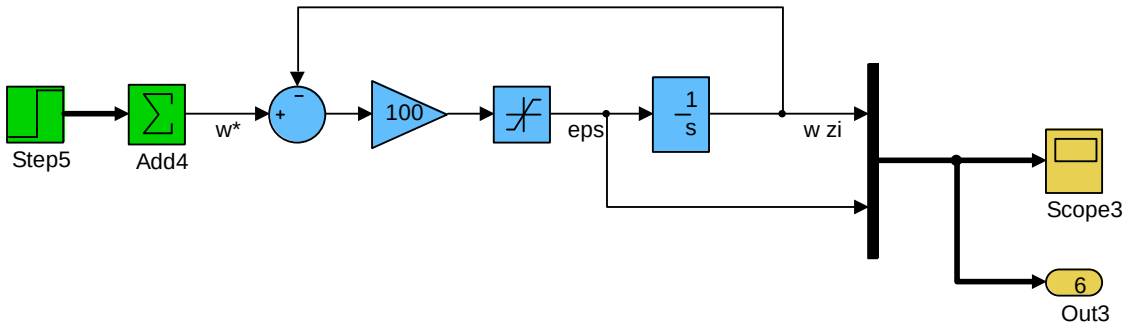


Рис. 10.25. Замкнений ЗІ з обмеженням прискорення

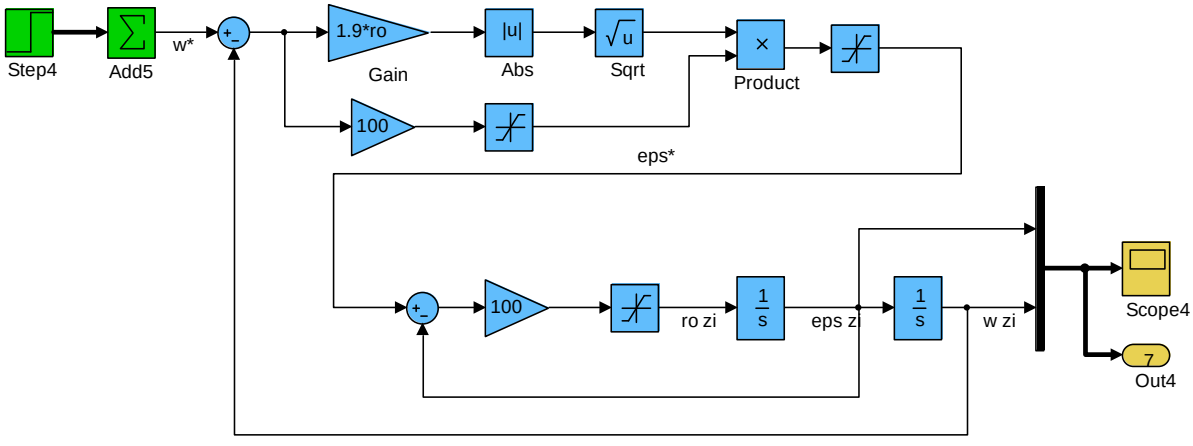


Рис. 10.26. Замкнений ЗІ з обмеженням прискорення і ривка

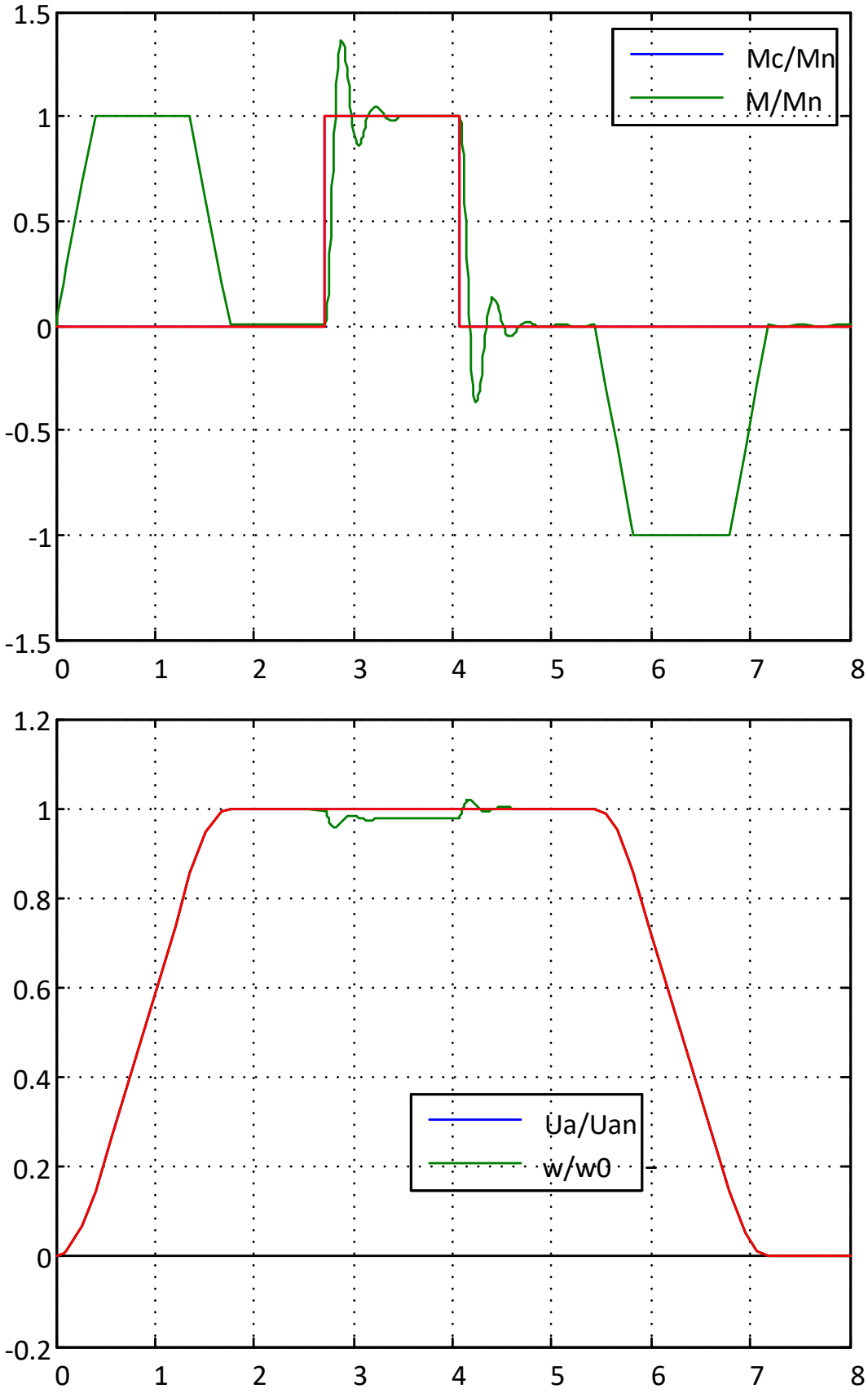


Рис. 10.27. Перехідні процеси в ДПС при відпрацюванні
S-подібної тахограми при комбінованому керуванні

10.7. Врахування тертя при моделюванні ДПС

В усіх попередніх моделях не враховано наявність ***моменту холостого ходу двигуна*** M_f***, який витрачається на подолання моменту сухого тертя*** M_{Cf} ***(Coulombic friction)***:

$$M_f = M_{Cf} \cdot \text{sign}(\omega). \tag{10.30}$$

Структурна схема механічної частини будь-якого двигуна з урахуванням моменту сухого тертя представлена на рис. 10.28*а*, де M_L – момент навантаження (від англ. *Load*), що утворюється при виконанні електроприводом корисної роботи (наприклад, підіймання вантажу, обтискання металу на прокатних станах тощо); $M_{\text{с}}$ – сумарний момент статичного опору (з урахуванням тертя).

Однак при застосуванні такої СС при цифровому математичному моделюванні через дискретність за часом та через застосування арифметики з плаваючою точкою, в якій відсутній нуль, момент холостого ходу на кожному кроці чисельного інтегрування перемикається між рівнями $+M_{Cf}$ і $-M_{Cf}$.

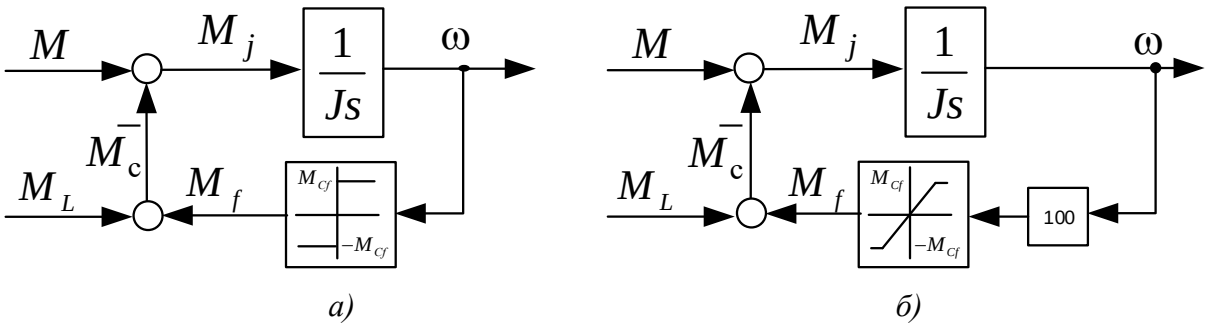


Рис. 10.28. Структурні схеми механічної частини двигуна зі врахуванням наявності моменту сухого (Кулонівського) тертя

Така різниця між значеннями однієї координати в сусідніх точках часу приводить до того, що ітераційний процес розв'язання системи диференційних рівнянь «зациклюється». Щоб запобігти цього явища треба або застосу-

вати метод чисельного інтегрування з постійним кроком, що приведе до збільшення часу моделювання, або замінити елемент з безкінечним підсиленням елементом з великим але конечним коефіцієнтом підсилення (50-100) на лінійній ділянці, тобто застосувати послідовне з'єднання блоків *Gain* та *Saturation*, як це показано на структурній схемі рис. 10.28*б*. Перехідні процеси зображені на рис. 10.29.

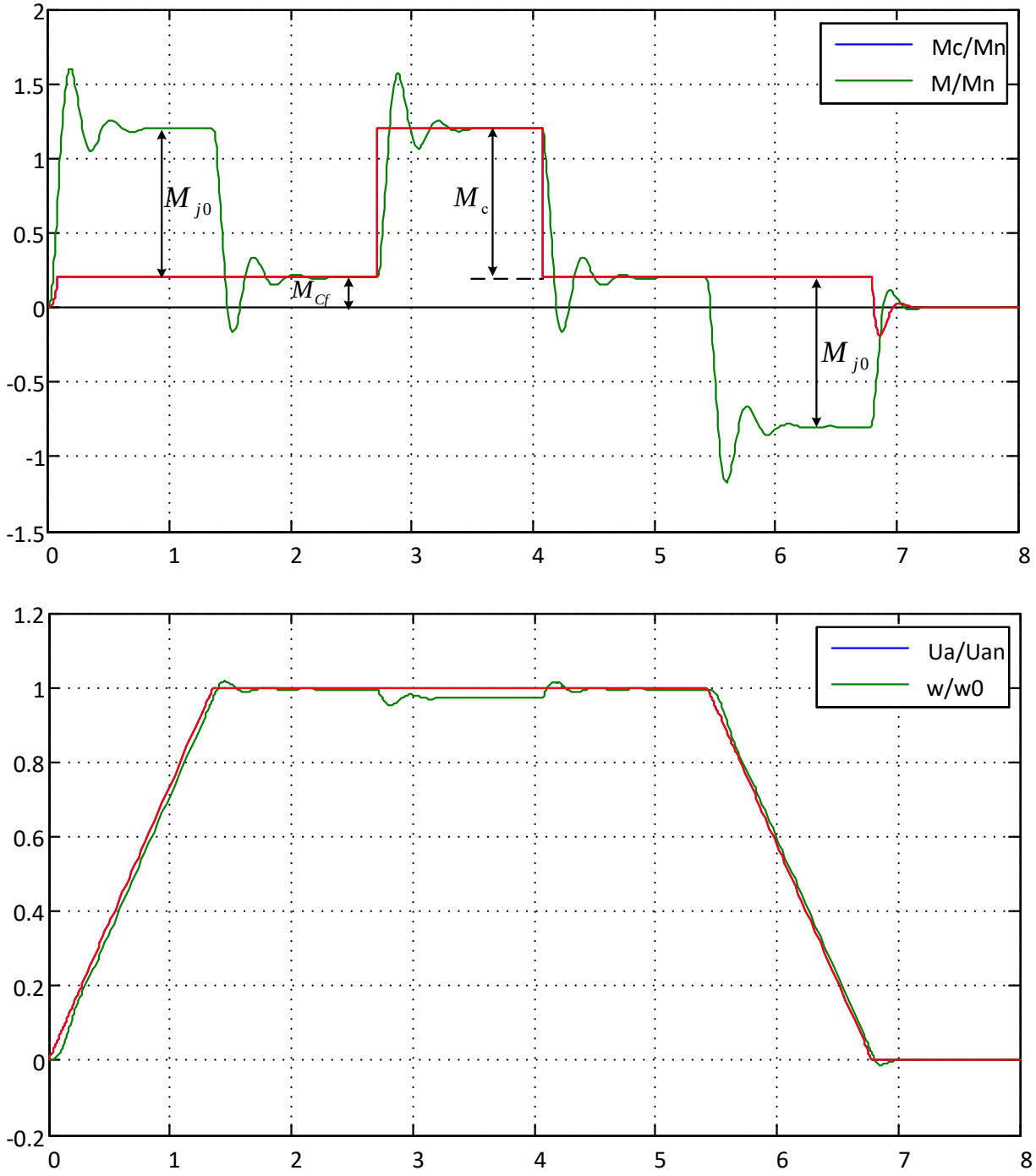


Рис. 10.29. Перехідні процеси в ДПС при врахуванні наявності сухого тертя

моделлю рис. 10.28*б*

Якщо враховувати різницю між ***сухим тертям у стані спокою*** M_{bf} ***(Breakway friction)*** та у стані руху M_{Cf}, то замість рівняння (10.30), слід застосовувати систему рівнянь

$$M_f = \begin{cases} M \ \text{ при } \ |\omega| = 0 \ \& \ |M| < M_{bf}, \\ M_{Cf} \cdot \text{sign}(\omega) \ \text{ при } \ |\omega| > 0. \end{cases} . \qquad (10.31)$$

Ще більш точна залежність моменту тертя від швидкості описується за рівняннями:

$$M_f = \begin{cases} \dfrac{M_{bf}}{\omega_{th}}\omega, \ \text{при} \ |\omega| \le \omega_{th}, \\ [M_{Cf} + M_{Sf} e^{(-c_S|\omega-\omega_{th}|)} + f_v|\omega - \omega_{th}|]sign(\omega), \ \text{при} \ |\omega| > \omega_{th}, \end{cases} \qquad (10.32)$$

де $M_{Sf} = M_{Cf} - M_{bf}$ – ***компонента Штрібека (Stribeck Fricton)***, яка існує при умові врахування різниці між тертям спокою та тертям руху [8]; c_S – ***коефіцієнт експоненціальної складової тертя (Transitional Approximation Coefficient)***; f_v – ***коефіцієнт в'язкого тертя (Viscous Friction Coefficient)***; ω_{th} – ***межа лінійної ділянки (Linear Region Velocity Threshold) статичної характеристики тертя*** $M_f(\omega)$, зображеної на рис. 10.30.

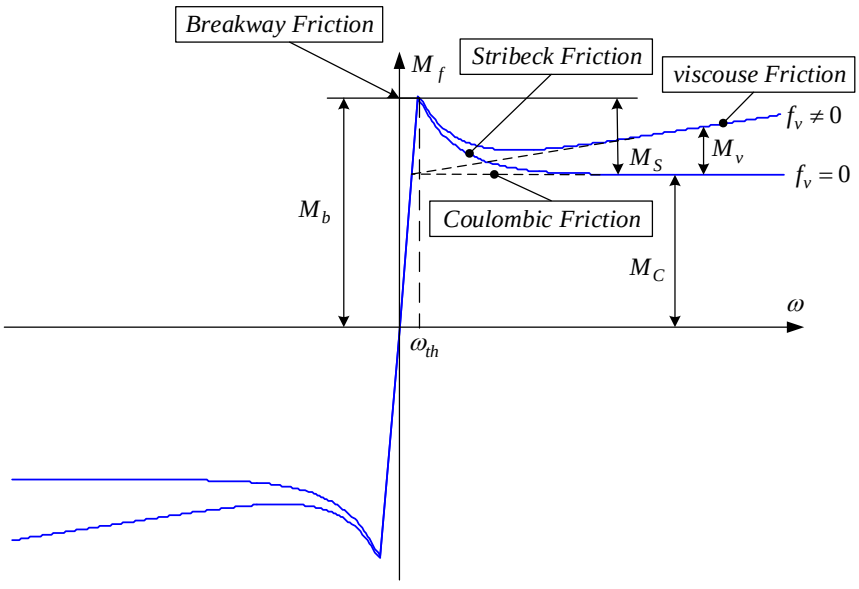


Рис. 10.30. Уточнена статична характеристика тертя

Реалізація такого математичного опису є більш складною, тому що потребує застосування та налаштування керованого ключа та блоків зрівняння і логіки. Але, зважаючи на те, що момент тертя складає невелику долю від номінального моменту двигуна, то у більшості випадків, цілком припустимо застосовувати спрощену модель тертя.

Контрольні питання та завдання

1. Які припущення приймають при розробці математичної моделі ДПС ?
2. Як перетворюють початкові диференціальні рівняння для створення структурної схеми у просторі станів (деталізована модель) та у вигляді передавальних функцій? Наведіть відповідні Симулінк-моделі.
3. Які базові величини приймають при нормуванні математичного опису ДПС? Наведіть Симулінк-моделі ДПС у в.о.
4. Які сталі часу характеризують динамічні властивості ДПС?
5. Який фізичний сенс має електромеханічна стала часу?
6. Виведіть передавальні функції ДПС за керуванням та за збуренням. Яку інформацію можна з них отримати?
7. Проаналізуйте характеристичне рівняння ДПС. Що з нього випливає?
8. Як визначити перерегулювання перехідної функції ДПС?
9. Поясніть, чому не можна здійснювати прямий пуск ДПС.
10. Як впливає електромагнітна стала часу якірного кола на максимальне значення струму при стрибкоподібній зміні напруги якоря?
11. Як промоделювати реостатний пуск ДПС?
12. Як промоделювати задатчик інтенсивності без обмеження ривка і з обмеженням ривка?
13. Як розрахувати обмеження на прискорення і на ривок? Як розрахувати час розгону приводу? На яку фізичну величину впливає значення похі-

дної від напруги якоря при розгоні та гальмуванні двигуна?

14. Якого вигляду набуває динамічний момент та швидкість при обмеженні ривка?

15. Як можна компенсувати інерційність об’єкта регулювання при наявності інтеграторів в задавальних пристроях? Що таке комбіноване керування за задавальною дією?
16. Як впливає величина активного опору якоря на просадку швидкості при накиді навантаження?
17. Як врахувати наявність моменту холостого ходу при моделюванні механічної частини двигунів?

11. МОДЕЛЮВАННЯ ДВИГУНА ПОСТІЙНОГО СТРУМУ З КЕРУВАННЯМ У КОЛІ ЯКОРЯ ТА У КОЛІ ЗБУДЖЕННЯ

11.1. Математичний опис ДПС при регулюванні потоку збудження

У ДПС зі змінним потоком збудження рівняння електричної рівноваги та рівняння руху не відрізняються від відповідних рівнянь (10.1) та (10.2) при постійному потоці, але рівняння моменту і ЕРС набувають вигляду [13]

$$M(t) = k\,\Phi_{з}(t)\,I_{я}(t), \tag{11.1}$$

$$E_{д}(t) = k\,\Phi_{з}(t)\,\omega(t). \tag{11.2}$$

До них ще додається рівняння електричної рівноваги кола збудження:

$$U_{з}(t) = i_{з}(t)R_{з} + L_{з}\frac{di_{з}(t)}{dt}, \tag{11.3}$$

де

$U_{з}$, $i_{з}$ – напруга та струм збудження;

$R_{з}$, $L_{з}$ – активний опір та індуктивність обмотки збудження.

При врахуванні ефекту насичення сталі, тобто нелінійності кривої намагнічування двигуна $\Phi_{з}(i_{з})$, індуктивність обмотки збудження змінюється за законом

$$L_{з} = k_{з}\frac{d\Phi_{з}(t)}{di_{з}(t)}, \tag{11.4}$$

де

$$k_{з} = 2p\xi w_{n}, \tag{11.5}$$

w_{n} – кількість витків обмотки збудження на полюс,

ξ=1+(0.5÷0.7)(σ-1) – коефіцієнт, що враховує вплив потоку розсіювання на індуктивність обмотки збудження і ту обставину, що частина потоку розсіювання зчеплена не з усіма витками обмотки, σ = (1.12÷1.18) – коефіцієнт розсіювання.

Після підстановки (11.4) у (11.3) отримуємо

$$U_з(t) = i_з(t)R_з + k_з \frac{d\Phi_з(t)}{dt}. \qquad (11.6)$$

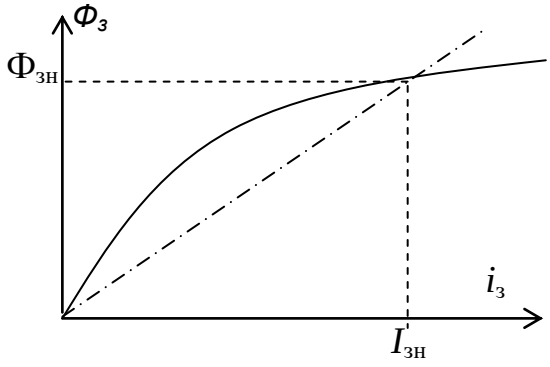


Рис. 11.1. Крива намагнічування ДПС

Загальний вигляд кривої намагнічування подано на рис. 11.1. Зазвичай проектувальники визначають її експериментально і подають у довідниках у вигляді таблиці.

Коло збудження характеризується ***сталою часу збудження***

$$T_з = \frac{L_з(i_з, \Phi_з)}{R_з} = \frac{k_з}{R_з} \cdot \frac{d\Phi_з(t)}{di_з(t)} = \text{var}. \qquad (11.7)$$

Усереднене значення сталої збудження визначають із лінеаризованої характеристики намагнічування за формулою

$$T_{зн} = \frac{L_{зн}}{R_з} = \frac{k_з}{R_з} \cdot \frac{\Phi_{зн}}{i_{зн}} = \frac{k_з k_{Фн}}{R_з} = \text{const}, \qquad k_{Фн} = \frac{\Phi_{зн}}{i_{зн}}. \qquad (11.8)$$

Треба відзначити, що, додатково до припущень, прийнятих у розділі 12, наведений математичний опис не враховує дію вихрових струмів, які виникають у масивних частинах магнітної системи потужних двигунів при зміні магнітного потоку.

Структурна схема ДПС, складена за рівняннями (10.1), (10.2), (11.1), (11.2) і (11.6) зі врахуванням позначень (11.4), (11.5) зображена на рис. 11.2.

Як бачимо модель на рис. 11.2 є двоканальною, тобто має 2 канали для регулювання швидкості.

Діапазон регулювання швидкості, що забезпечується зміною напруги якоря при постійній напрузі збудження (від 0 до номінального значення) називають першою зоною, а діапазон, що забезпечується зміною напруги збудження, при постійній напрузі якоря (вище номінального значення) – другою зоною.

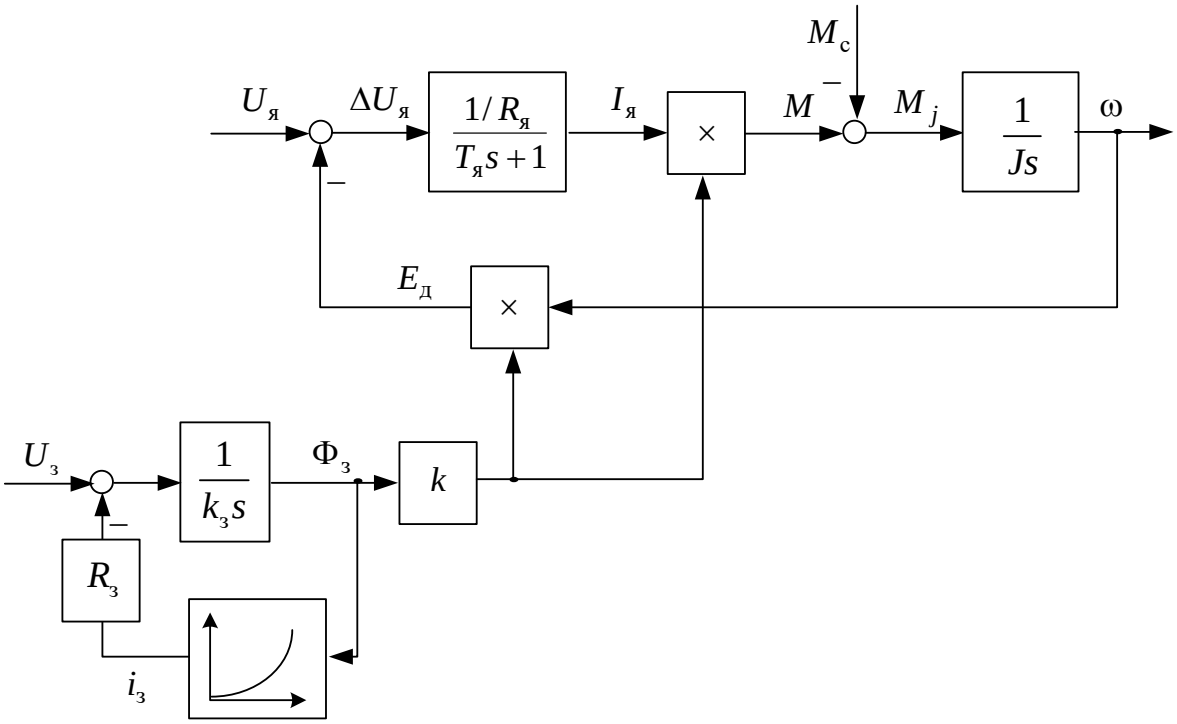


Рис. 11.2. Структурна схема двигуна постійного струму з керуванням у колах якоря і збудження

11.2. Розробка *Simulink*-моделей для дослідження ДПС з двозонним регулюванням швидкості

Для дослідження властивостей ДПС при роботі його у першій та другій зонах регулювання швидкості можна використати діаграми сигналів завдання у відносних одиницях, наведені на рис. 11.3, де позначено:

$t_з = 4T_{зн}$ – час збудження двигуна за експоненціальним законом ;

$t_{р0}$ – бажаний час розгону ДПС у першій та другій зонах;

$t_{по}$ – час початку ослаблення поля.

У відповідності з рис. 11.3 спочатку необхідно стрибком подати на двигун напругу збудження $U_з = U_{зн}$. Після того, як під дією цього сигналу потокозчеплення ДПС досягне номінального значення $Ф_з = Ф_{зн}$, можна почати змінювати за лінійним законом напругу якоря до номінального значення. Після розгону двигуна до швидкості ідеального холостого ходу спостерігаємо реакцію системи на стрибкоподібну зміну моменту статичного опору $M_с$.

Потім збільшуємо швидкість у 2 рази за рахунок зменшення напруги збудження і відповідного ослаблення поля за гіперболічним законом при постійній напрузі якоря, і на усталеній швидкості знову накидаємо навантаження.

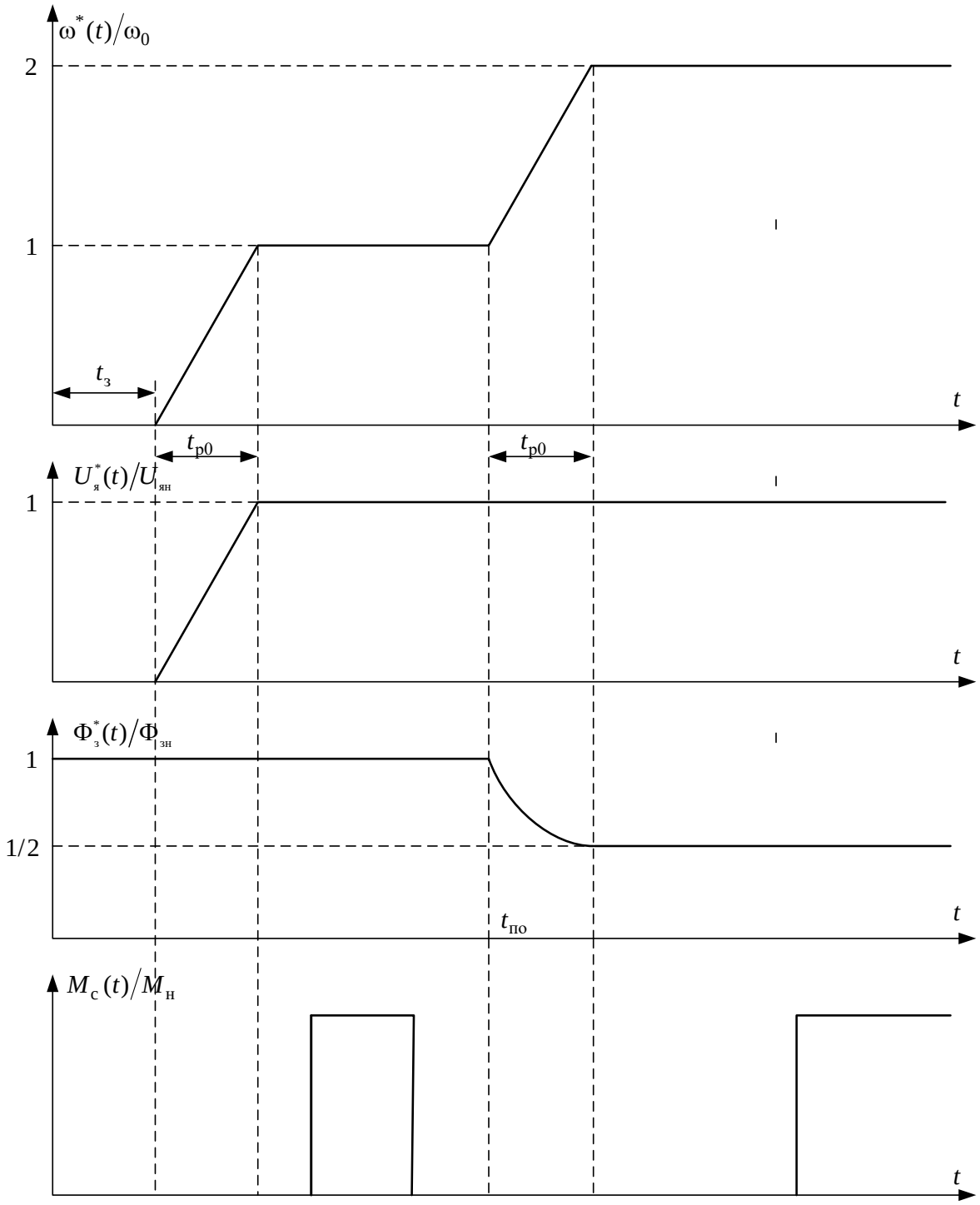


Рис. 11.3. Діаграми сигналів завдання до двозонної системи регулювання швидкості ДПС у відносних одиницях

Діаграми задавальних сигналів, зображені на рис. 11.3, можна реалізувати в *Simulink* декількома способами, один з яких представлено на рис. 11.4.

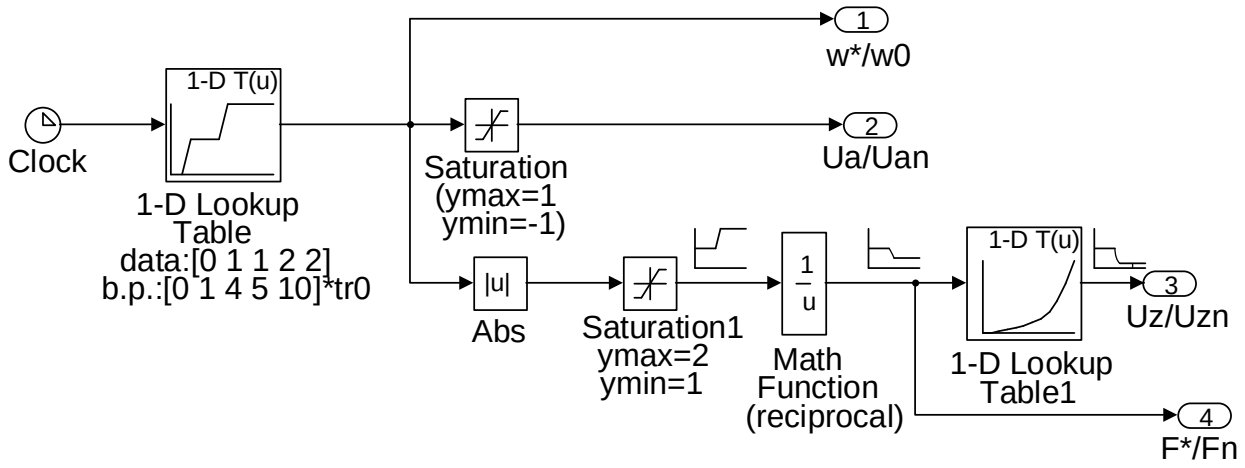


Рис. 11.4. *Simulink*-модель формування у в.о. вхідних сигналів для ДПС, що керується зміною напруги якоря та напруги збудження

Simulink-модель ДПС з перетворенням відносних вхідних сигналів у абсолютні та абсолютних вихідних сигналів у відносні зображена на рис. 11.5, а загальна структурна модель для досліджень, в якій модель двигуна рис. 11.5 подана у вигляді незамаскованої підсистеми – на рис. 11.6 [3].

Щоб підтримувати прискорення у другій зоні на тому ж рівні, що і у першій, треба завдання на магнітний потік у в.о. розраховувати за формулою

$$\Phi^*/\Phi_{\text{н}} = \begin{cases} 1 & \textit{при}\ \ |\omega^*|/\omega_0 \le 1, \\ \dfrac{1}{|\omega^*|/\omega_0} & \textit{при}\ \ |\omega^*|/\omega_0 > 1. \end{cases} \qquad (11.9)$$

Для формування нелінійної функціональної залежності $i_з^* = f(\Phi^*)$ в моделі рис. 11.6 використано кусково-лінійну інтерполяцію та кубічну інтерполяцію. Графіки перехідних процесів представлені на рис. 11.7-11.9. Можна ще здійснити апроксимацію цієї залежності поліномом 5-го порядку.

На моделі рис. 11.6 завдання на швидкість доповнено ділянкою гальмування двигуна зі швидкості $2\omega_0$ до 0.

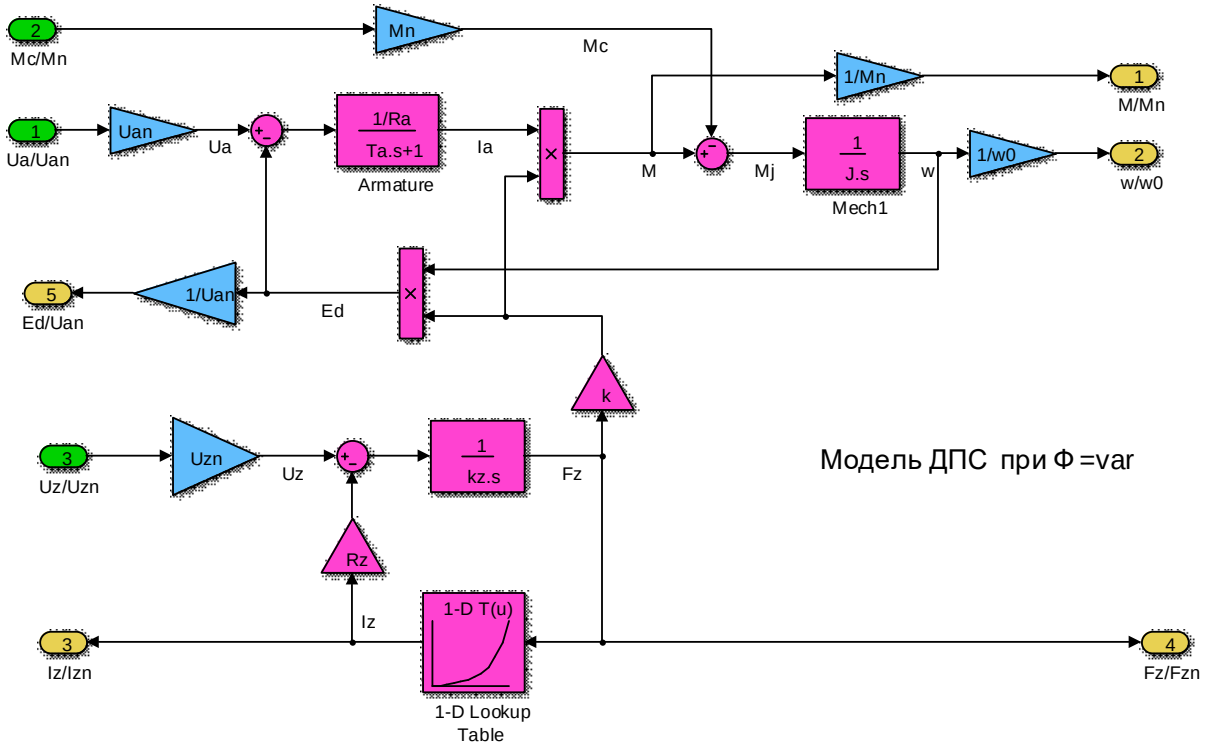


Рис. 11.5. Структурна модель ДПС, що керується зміною напруги якоря та напруги збудження

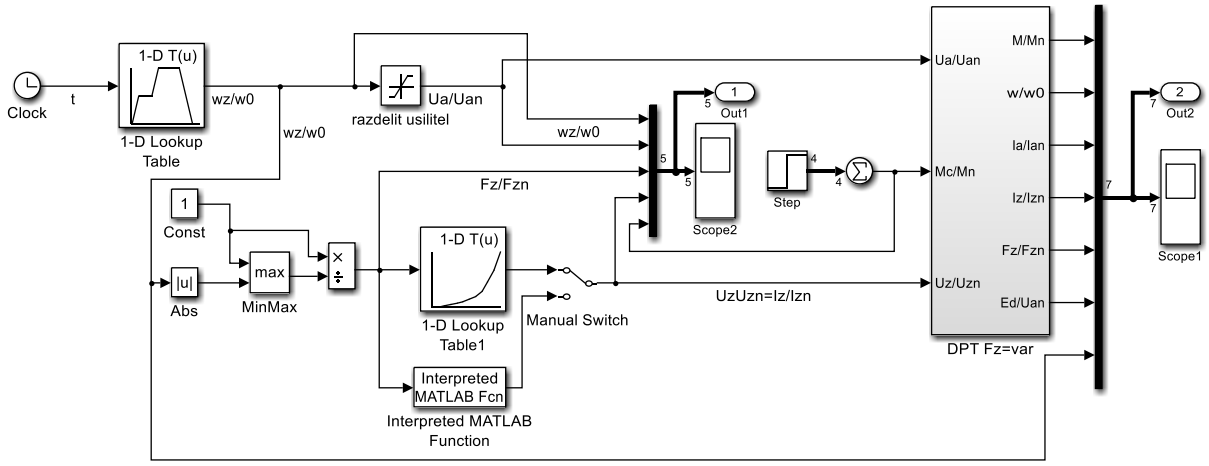


Рис. 11.6. Структурна модель для дослідження ДПС

11.3. Аналіз перехідних процесів

На рис. 11.7 зображені у відносних одиницях графіки сигналів, підключених до вихідного порту на моделі рис. 6.6: завдання на швидкість wz*/w0 (бажана тахограма), необхідні для забезпечення такої тахограми завдання на напругу якоря Ua*/Uan, струм і напругу збудження Uz*/Uzn=iz*/izn, потік збудження Fz*/Fzn та момент статичного опору

Mc/Mn.

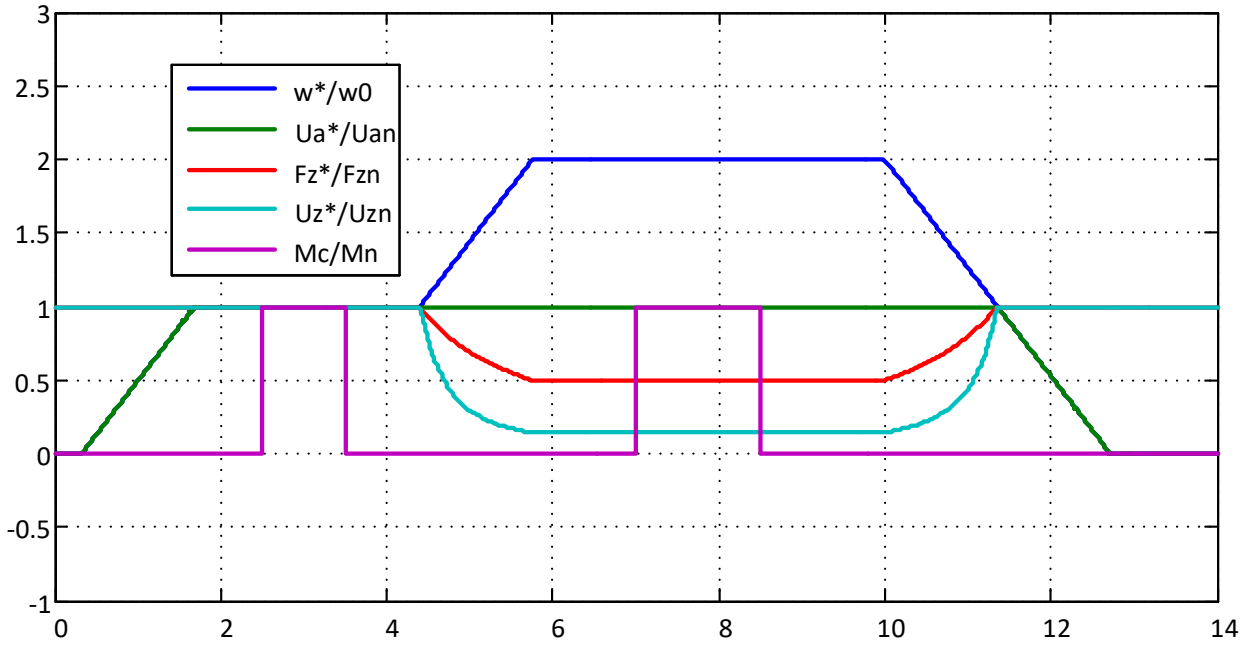


Рис. 11.7. Завдання на швидкість, напругу якоря потік збудження, напругу збудження та момент навантаження у відносних одиницях

З рис. 11.7 видно, що спочатку подається завдання на збудження двигуна, а потім завдання на швидкість. На ділянках руху з постійною швидкістю відбувається стрибкоподібний накид номінального навантаження та його скидання.

Бажана тахограма і навантажувальна діаграма на рис. 11.7 визначають послідовність таких режимів:

- намагнічування двигуна;
- розгін двигуна до швидкості ідеального холостого ходу з постійним збудженням за рахунок зміни напруги якоря від 0 до номінального значення;
- рух з постійною швидкістю $\omega = \omega_0$;
- накидання та скидання номінального навантаження при збудженні двигуна номінальним потоком;
- розгін від швидкості до швидкості ω_0 до швидкості $2\omega_0$ з постійною (номінальною) напругою за рахунок ослаблення поля;

- рух з постійною швидкістю $\omega = 2\omega_0$;
- накидання та скидання номінального навантаження при ослабленому полі;
- гальмування від швидкості $2\omega_0$ до 0.

Реакція двигуна на вхідні сигнали рис. 11.7 при кусково-лінійній інтерполяції кривої намагнічування показана на рис. 11.8

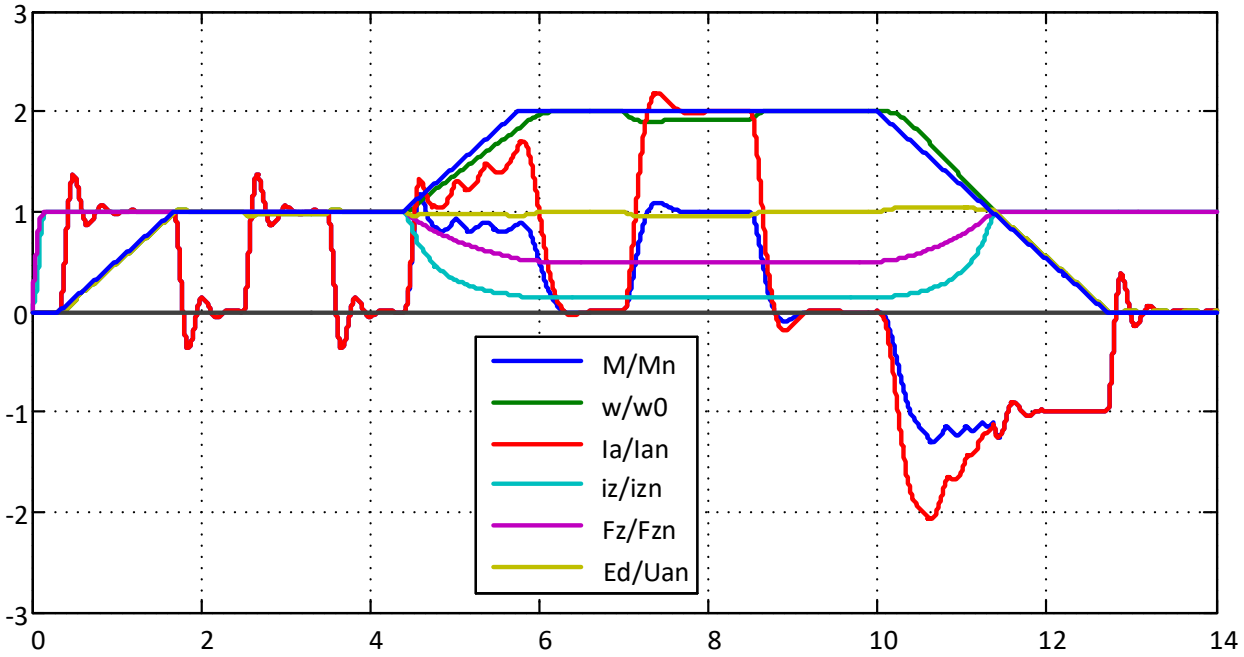


Рис. 11.8. Реакція двигуна постійного струму на вхідні сигнали рис. 11.7 при кусково-лінійній інтерполяції кривої намагнічування

Із рис. 11.8 видно, що перехідні процеси у другій зоні стають менш коливальними та більш повільними ніж у першій зоні. Це пояснюється зменшенням магнітного потоку в кінці другої зони в $d_{\text{осл}} = \dfrac{\Phi_{\text{н}}}{\Phi_{\text{осл}}}$ разів ($d_{\text{осл}}$ - діапазон ослаблення поля) та електромеханічної сталої часу – у $d_{\text{осл}}^2$ разів:

$$T_{\text{м осл}} = \frac{JR_{\text{я}}}{\Phi_{\text{осл}}^2} = \frac{JR_{\text{я}}}{\Phi_{\text{н}}^2} d_{\text{осл}}^2 = T_{\text{мн}} d_{\text{осл}}^2. \qquad (11.9)$$

Те ж саме стосується і статичної просадки швидкості при накиді навантаження:

$$\Delta\omega_{\text{с осл}}=\frac{I_{\text{с}}R_{\text{я}}}{k\Phi_{\text{осл}}}=\frac{M_{\text{с}}R_{\text{я}}}{(k\Phi_{\text{осл}})^2}=\Delta\omega_{\text{сн}}d_{\text{осл}}^2\,. \qquad (11.10)$$

Це підтверджується порівнянням статичних механічних характеристик, наведених на рис. 11.9.

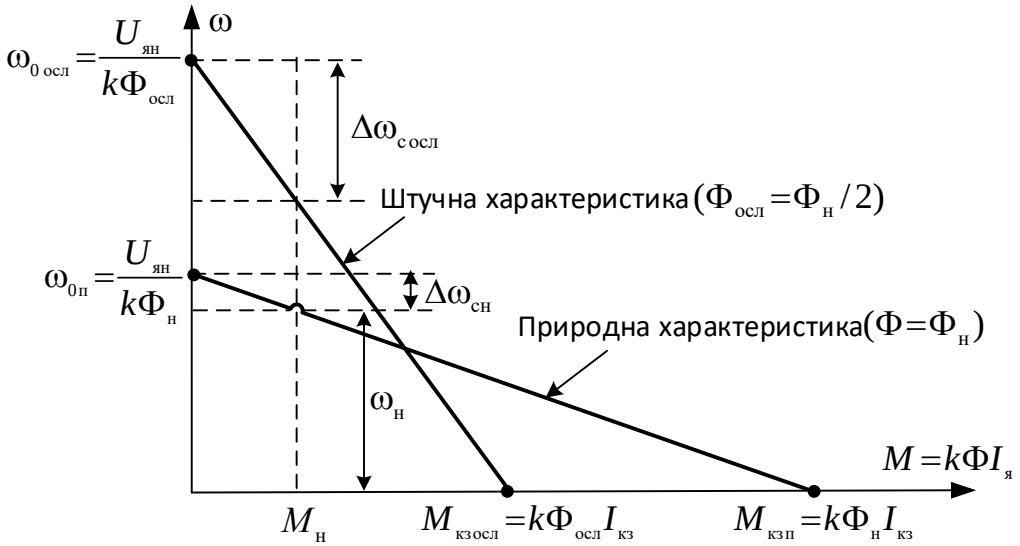


Рис. 11.9. Природна та штучна механічні характеристики ДПС

Ще варто звернути увагу на те, що в першій зоні (при $\Phi=\Phi_{\text{н}}=const$) згідно з рівняннями (11.2) і (11.4) відношення електромагнітного моменту до струму якоря та ЕРС двигуна до його кутової швидкості залишаються постійними, а відносні значення цих величин – однаковими: $M/M_{\text{н}}=I_{\text{я}}/I_{\text{ян}}$, $E_{\text{д}}/U_{\text{ян}}=\omega/\omega_0$. У другій зоні швидкість двигуна збільшується при майже постійній ЕРС, а струм якоря збільшується при майже постійному моменті за рахунок зменшення магнітного потоку.

Коливання електромагнітного моменту і струму якоря на ділянках регулювання магнітного потоку на рис. 11.8 пов'язані з кусково-лінійною апроксимацію кривої намагнічування, зумовлені розривним характером похідної від нелінійної функції в точках зламу кривої $i_{\text{з}}(\Phi_{\text{з}})$. Щоб позбутися цього явища, треба застосувати інтерполяцію кубічними сплайнами. Для цього необхідно в ланці *1-D Look up Table* якщо у змінити алгоритм інтерполювання (параметр *Algorithm → Interpolation method*), який за замовченням має зна-

чення *Linear*, на алгоритм *Cubic spline*. При цьому розрахунок здійснюється набагато швидше, ніж при використанні блока *Interpreted MATLAB Function*.

Графіки перехідних процесів у досліджуваній системі при заміні методу інтерполювання показані на рис. 11.10.

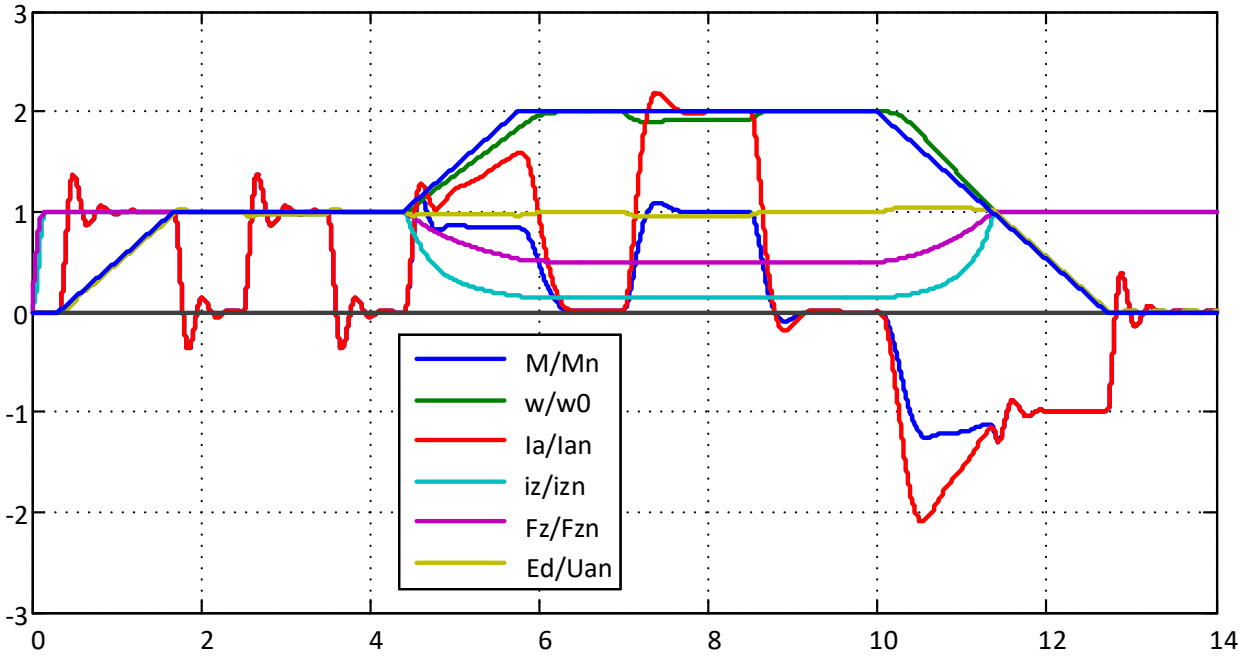


Рис. 11.10. Реакція двигуна постійного струму на вхідні сигнали рис. 11.7 при інтерполяції кривої намагнічування кубічними сплайнами

Контрольні питання і завдання

1. Запишіть рівняння, на підставі яких побудована структурна схема рис. 1.2.

2. Як визначається індуктивність обмотки збудження при врахуванні нелінійності характеристики намагнічування?

3. Поясніть збільшення струму якоря та просадки швидкості при ослабленні поля двигуна.

4. Обґрунтуйте зменшення швидкодії у зоні ослаблення поля.

5. Як змінюється електромеханічна стала часу двигуна при регулюванні потоку збудження? На що це впливає?

6. Як змінюється електромагнітна стала часу збудження при регулюванні потоку?

7. Чому потік збудження прагнуть змінювати за формулою (11.9)?

8. Поясніть різницю в перехідних процесах при розгоні приводу за рахунок зміни напруги якоря і напруги збудження.

9. Чим відрізняються процеси накиду навантаження при номінальному й ослабленому потоках збудження?

10. Як впливає на результати моделювання кусково-лінійна апроксимація кривої намагнічування? Як можна підвищити адекватність результатів?

12. МОДЕЛЮВАННЯ ЛІНІЙНИХ МЕХАНІЧНИХ ОБ'ЄКТІВ З ОДНИМ СТУПЕНЕМ ВІЛЬНОСТІ

12.1. Зв'язок між основними координатами одномасових систем

Механічні системи можна поділити на системи поступального і обертального руху.

Механічну інертність об'єктів поступального руху характеризує маса (m, кг), а основними параметрами його є ***переміщення*** (s, м), ***лінійна швидкість*** (v, м/с), ***лінійне прискорення*** (a, м/с^2) та ***ривок*** (ρ, м/с^3), що пов'язані одне з одним диференційними рівняннями

$$v(t)=\frac{ds(t)}{dt},\quad a(t)=\frac{dv(t)}{dt}=\frac{d^2s(t)}{dt^2},\quad \rho(t)=\frac{da(t)}{dt}=\frac{d^2v(t)}{dt^2}=\frac{d^3s(t)}{dt^3}. \qquad (12.1)$$

Механічну інертність об'єктів обертального руху характеризує ***момент інерції*** (J, кг·м^2), а основними параметрами його є ***кутове переміщення*** (φ, рад), ***кутова швидкість*** (ω, рад/с), ***кутове прискорення*** (ε, рад/с^2) та кутовий ривок (ρ_t, рад/с^3), що пов'язані одне з одним диференційними рівняннями

$$\omega(t)=\frac{d\varphi(t)}{dt},\quad \varepsilon(t)=\frac{d\omega(t)}{dt}=\frac{d^2\varphi(t)}{dt^2},\quad \rho_t(t)=\frac{d\varepsilon(t)}{dt}=\frac{d^2\omega(t)}{dt^2}=\frac{d^3\varphi(t)}{dt^3}. \qquad (12.2)$$

Найбільш просто математично описуються динамічні системи із зосередженими параметрами, зокрема з зосередженими масами.

Рівняння руху одномасових механічних об'єктів з постійною механічною інерційністю визначається ***другим законом Ньютона***:

$$\sum_{i=1}^{n}F_i(t)=ma(t)=m\frac{dv(t)}{dt}=m\frac{d^2s(t)}{dt^2}, \qquad (12.3)$$

$$\sum_{i=1}^{n}M_i(t)=J\varepsilon(t)=J\frac{d\omega(t)}{dt}=J\frac{d^2\varphi(t)}{dt^2}, \qquad (12.4)$$

де $\sum_{i=1}^{n} F_i(t)$ – алгебраїчна сума сил, що діють на тіло по осі його поступального руху; $\sum_{i=1}^{n} M_i(t)$ – алгебраїчна сума моментів, що діють на тіло у площині його обертання.

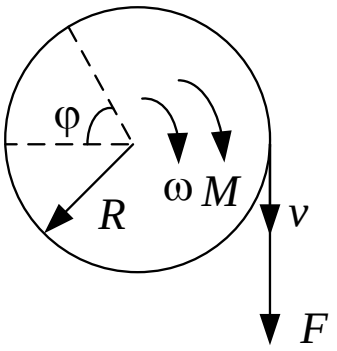


Рис. 12.1. Обертальні та поступальні параметри суцільного циліндру

Взаємозв'язок між лінійними та обертальними параметрами суцільного циліндра або диска радіусом R, що обертається навколо своєї осі (див. рис. 12.1) визначаються рівняннями

$$M = FR, \quad \omega = \frac{v}{R}, \quad \varphi = \frac{s}{R}, \quad J = \frac{mR^2}{2}. \qquad (12.5)$$

При визначенні моментів інерції тіл, що обертаються, їх форму зазвичай спрощують. Наприклад, ротор (або якір) електричного двигуна являють собою тонкостінний порожнистий циліндр, а пристрої намотки – товстостінні порожнисті циліндри, для яких відповідно

$$J = mR^2, \qquad J = \frac{m\,(R_1^2 + R_2^2)}{2}. \qquad (12.6)$$

12.2. Математичний опис двомасових механічних систем

Механізми або їх робочі органи приводяться до руху двигунами. Рух від двигуна до механізму передається різноманітними пристроями: валами, шківами, муфтами, зубчатими колесами, черв'ячними передачами, кулачковими механізмами, тощо. Такі пристрої мають пружні властивості і характеризуються ***коефіцієнтами жорсткості (коефіцієнтами пружності)*** c_{ij} ***та коефіцієнтами в'язкого (внутрішнього) тертя*** d_{ij}**.**

Лінійна пружна деформація виникає при розтягуванні (стисненні) пружин та пружинно подібних елементів, наприклад, довгих канатів. При

обертальному русі може виникнути ***пружна деформація скручування*** довгих та тонких валів.

При пружній деформації у деформованому тілі з'являється пружне зусилля або пружний момент, пропорційний величині деформації, тобто різниці між переміщеннями протилежних кінцівок передачі:

$$F_{\text{п}} = c_{ij} \cdot \Delta s = c_{ij}(s_i - s_j) \qquad M_{\text{п}} = c_{ij} \cdot \Delta\varphi = c_{ij}(\varphi_i - \varphi_j). \tag{12.7}$$

Пружній деформації перешкоджають момент або зусилля внутрішнього в'язкого тертя, зумовлені силами міжмолекулярної взаємодії деформованого матеріалу і пропорційне різниці між швидкостями протилежних кінцівок передачі:

$$F_{\text{в}} = d_{ij} \cdot \Delta v = d_{ij}(v_i - v_j) \qquad M_{\text{в}} = d_{ij} \cdot \Delta\omega = d_{ij}(\omega_i - \omega_j). \tag{12.8}$$

При взаємодії величин (12.7) та (12.8) утворюється пружно-в'язке зусилля або пружно-в'язкий момент

$$F_{ij} = F_{\text{п}} + F_{\text{в}} \qquad M_{ij} = M_{\text{п}} + M_{\text{в}}. \tag{12.9}$$

Для прикладу розглянемо кінематичну схему двомасової пружно-в'язкої механічної системи поступального руху, зображену на рис. 12.2.

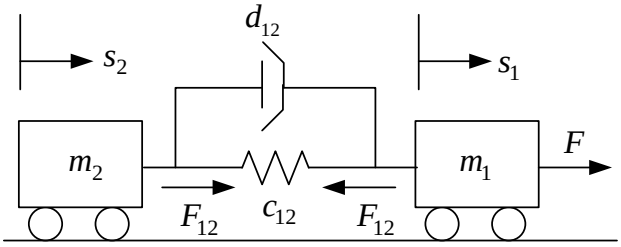


Рис. 12.2. Кінематична схема двомасової механічної системи поступального руху

Її математичний опис без урахування сухого тертя має вигляд [13]

$$\begin{cases} F_{12} = m_2 \dfrac{dv_2}{dt}, \\ F - F_{12} = m_1 \dfrac{dv_1}{dt}, \\ \dfrac{ds_1}{dt} = v_1, \quad \dfrac{ds_2}{dt} = v_2, \\ F_{\text{п}} = c_{12}(s_1 - s_2), \\ F_{\text{в}} = d_{12}(v_1 - v_2), \\ F_{12} = F_{\text{п}} + F_{\text{в}}. \end{cases} \tag{12.10}$$

Кінематична схема двомасової механічної системи обертального руху зображена на рис. 12.3.

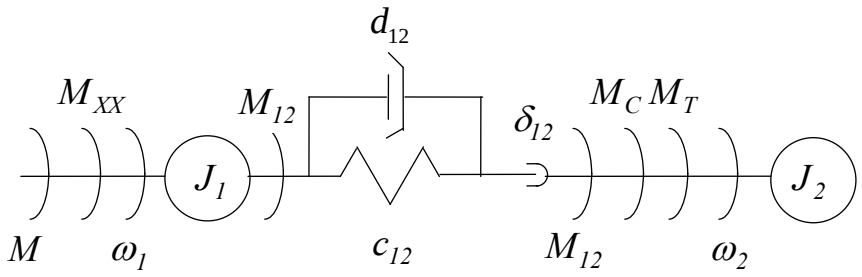


Рис. 12.3. Спрощена кінематична схема двомасової електромеханічної системи

На ній позначено:

- M_{XX}, M_{12}, M_C, M_T – електромагнітний момент двигуна, момент холостого ходу двигуна, момент скручування валу, момент статичного опору від навантаження і момент сухого тертя в механізмі відповідно;
- ω_1, ω_2 – кутові швидкості двигуна і механізму;
- c_{12}, b_{12} – коефіцієнти жорсткості і в'язкого тертя;
- δ_{12} – люфт (зазор) у кінематичній передачі;
- J_1, J_2 – моменти інерції двигуна і механізму.

Математичний опис такої системи без врахування зазору в кінематичній передачі та сухого тертя має вигляд [13]:

$$\begin{cases} M - M_{12} = J_1 \dfrac{d\omega_1}{dt}, \\ M_{12} - M_c = J_2 \dfrac{d\omega_2}{dt}, \\ \dfrac{d\varphi_1}{dt} = \omega_1, \\ \dfrac{d\varphi_2}{dt} = \omega_2, \\ M_{\text{п}} = d_{12}(\varphi_1 - \varphi_2). \\ M_{\text{в}} = b_{12}(\omega_1 - \omega_2), \\ M_{12} = M_{\text{п}} + M_{\text{в}}. \end{cases} \qquad (12.11)$$

12.3. Дослідження двомасових механічних систем

Структурна *Simulink*-модель, складена за рівняннями (12.10), зображена на рис. 12.4, а структурна схема двомасової механічної системи обертального руху – на рис. 12.5.

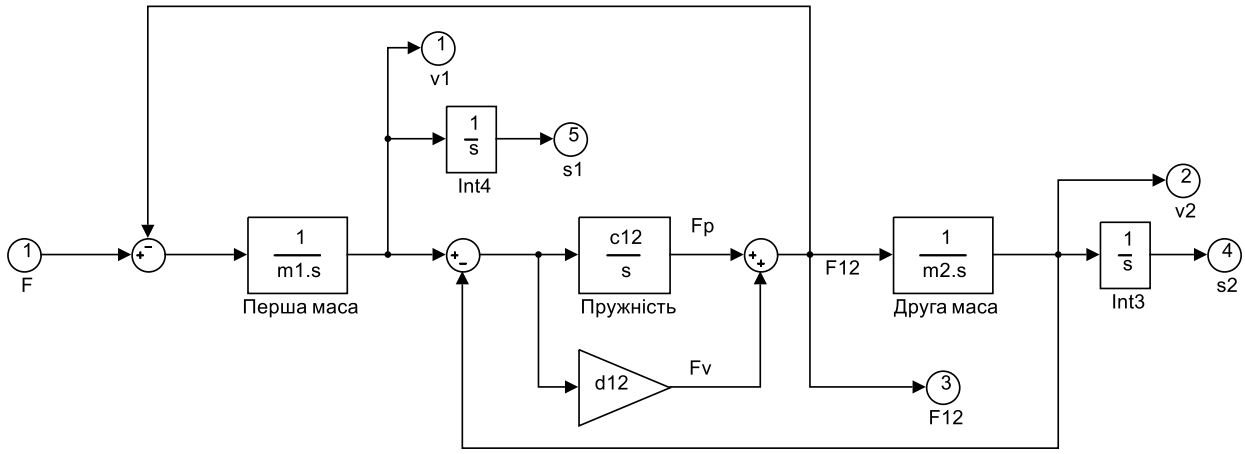


Рис. 12.4. Структурна *Simulink*-модель двомасової механічної системи поступального руху

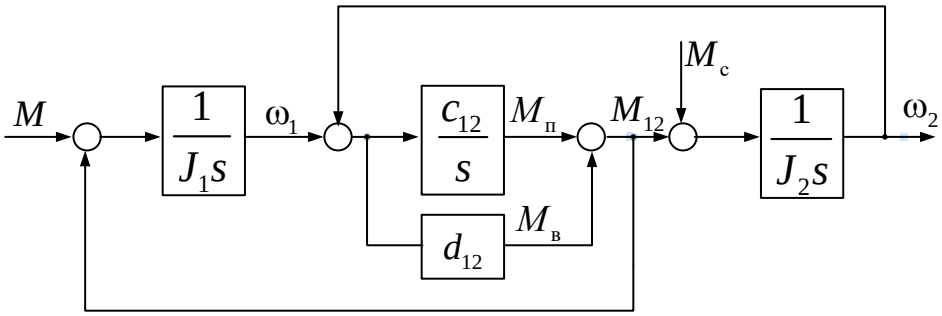


Рис. 12.5. Структурна схема двомасової механічної системи обертального руху

Щоб проаналізувати такий об’єкт, виконаємо еквівалентні структурні перетворення СС рис. 12.5 за керуванням у ланцюжок послідовно з’єднаних ПФ, як це показано на рис. 12.6.

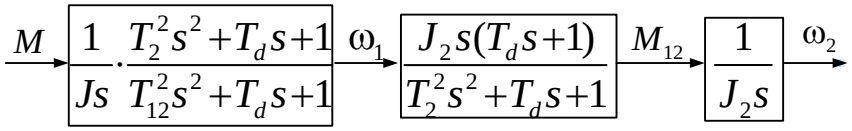


Рис. 12.6. Перетворена структурна схема системи рис. 12.5
без врахування дії статичного моменту

У схемі прийняті такі позначення:

$$J = J_1 + J_2, \quad \gamma = \frac{J}{J_1} = 1 + \frac{J_2}{J_1}, \tag{12.12}$$

$$T_{12} = \sqrt{\frac{J_1 J_2}{c_{12} J}} = \sqrt{\frac{J_2}{c_{12}\gamma}}, \quad T_2 = T_{12}\sqrt{\gamma} = \sqrt{\frac{J_2}{c_{12}}}, \tag{12.13}$$

$$T_d = d_{12}/c_{12}, \quad \xi = \frac{T_d}{2T_{12}}, \tag{12.14}$$

де

J – сумарний момент інерції;

γ – коефіцієнт співвідношення мас;

T_{12} – стала часу пружних коливань ДМО;

T_2 – стала часу пружних коливань другої маси;

T_d – стала часу демпфірування пружних коливань;

ξ – коефіцієнт демпфірування пружних коливань.

Зі СС рис. 12.6 легко знайти такі ПФ:

$$\frac{M_{12}(p)}{M(p)} = \frac{J_2}{J_1} \cdot \frac{T_d s + 1}{T_{12}^2 s^2 + T_d s + 1}, \tag{12.15}$$

$$\frac{\omega_1(p)}{M(p)} = \frac{1}{Js} \cdot \frac{T_2^2 s^2 + T_d s + 1}{T_{12}^2 s^2 + T_d s + 1}, \quad \frac{\omega_2(s)}{M(s)} = \frac{1}{Js} \cdot \frac{T_d s + 1}{T_{12}^2 s^2 + T_d s + 1}, \tag{12.16}$$

$$\frac{\omega_2(s)}{\omega_1(s)} = \frac{T_d s + 1}{T_2^2 s^2 + T_d s + 1}. \tag{12.17}$$

При дослідженні багатомасових систем часто нехтують в’язким тертям. У цьому випадку ПФ (12.15)-(12.17) набувають вигляду:

$$\frac{\omega_2(s)}{\omega_1(s)}=\frac{T_d s+1}{T_2^2 s^2+T_d s+1}, \tag{12.18}$$

$$\frac{\omega_1(s)}{M(s)}=\frac{1}{Jp}\cdot\frac{T_2^2 s^2+1}{T_{12}^2 s^2+1}, \qquad \frac{\omega_2(s)}{M(s)}=\frac{1}{Js}\cdot\frac{1}{T_{12}^2 s^2+1}, \tag{12.19}$$

$$\frac{\omega_2(s)}{\omega_1(s)}=\frac{1}{T_2^2 s^2+1}, \qquad \frac{M_{12}(s)}{\omega_1(s)}=\frac{J_2 s}{T_2^2 s^2+1}. \tag{12.20}$$

ПФ (12.18) та (12.20) – консервативні ланки, тобто коливальні ланки з нульовим демпфіруванням, яким відповідають гармонічні перехідні функції без загасання з круговими частотами та періодами коливань

$$\Omega_{12}=1/T_{12}, \qquad \Omega_2=1/T_2, \qquad T_{k12}=2\pi T_{12}, \qquad T_{k2}=2\pi T_2. \tag{12.21}$$

Тобто реакція пружного моменту на стрибок електромагнітного моменту та реакція швидкості другої маси і пружного моменту на стрибок швидкості першої маси визначаються рівняннями

$$M_{12}(t)=M\frac{J_2}{J_1}\cdot(1-\cos(\Omega_{12}t)), \tag{12.22}$$

$$\omega_2(t)=\omega_1(1-\cos(\Omega_2 t)), \qquad M_{12}(t)=J_2\frac{d\omega_2(t)}{dt}=J_2\omega_1\Omega_2\sin(\Omega_2 t) \tag{12.23}$$

Останні рівняння є ключем для перевірки адекватності моделі та відповідності її заданим параметрам. Перехідні процеси, що відповідають цим рівнянням зображені на рис. 12.7-12.8.

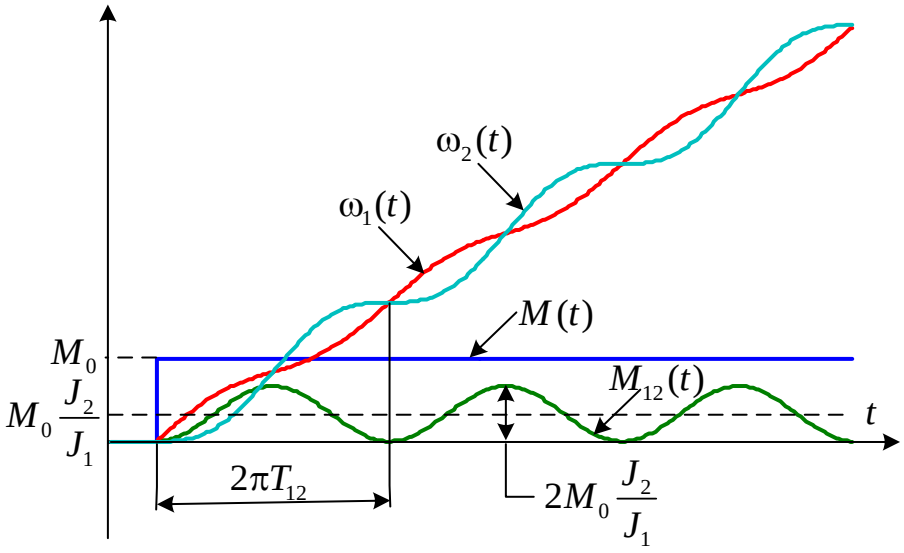


Рис. 12.7. Реакція ДМС на стрибок моменту

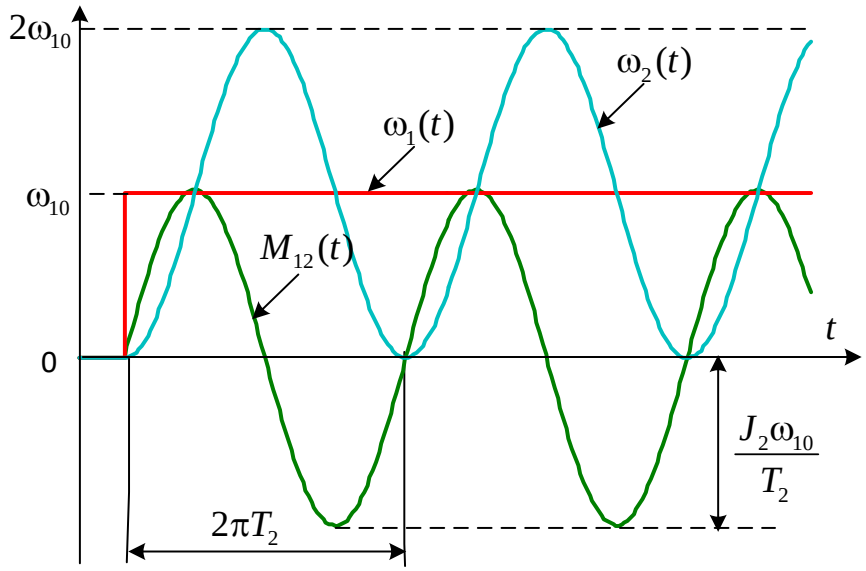


Рис. 12.8. Реакція другої маси на стрибок швидкості першої маси

Контрольні питання

1. Якому закону підпорядковуються поступальні та обертальні рухи одномасових лінійних механічних об’єктів?
2. Як пов’язані між собою координати та параметри систем обертального та поступального рухів?
3. Як розрахувати момент інерції тонкостінного та товстостінного циліндру? Яке припущення можна прийняти при розрахунку моменту інерції роторів (якорів) електричних двигунів?
4. Як визначають коефіцієнти пружності та в’язкого тертя пружно-в’язких кінематичних передач?
5. Запишіть рівняння, що описують рухи двомасових механічних об’єктів без врахування дії в’язкого тертя.
6. Як розрахувати періоди пружних коливань двомасової системи та другої маси? Як побачити ці коливання при моделюванні?
7. Яку амплітуду мають коливання пружного моменту при стрибкоподібній зміні рушійного моменту?

13. МОДЕЛЮВАННЯ НЕЛІНІЙНИХ МЕХАНІЧНИХ СИСТЕМ З ДЕКІЛЬКОМА СТУПЕНЯМИ ВІЛЬНОСТІ

До механічних систем з декількома степенями вільності належать верстати, роботи-маніпулятори, транспортні механізми (у тому числі і кранові установки) та багато інших. Більшість із них описується нелінійними диференціальними рівняннями. Керування такими комплексами здійснюється декількома двигунами, роботу яких треба узгоджувати.

Розглянемо процес моделювання механічних комплексів з декількома ступенями вільності на прикладі сукупності механізмів мостового крана, до яких належать:

- міст, що пересувається по рейках, покладених на підкранових балках уздовж прольоту відкритого або закритого вантажного майданчика;
- візок з підвішеним до неї вантажем, що переміщається уздовж моста, тобто поперек прольоту майданчика;
- підіймальний пристрій у вигляді лебідки для намотування каната, до якої прикріплений вантаж.

Оскільки вантаж є підвішеним до візка на гнучкому канаті, то він може ще відхилятися на деякий кут від вертикалі і здійснювати коливання у вертикальній площині, просторова орієнтація якої залежить від співвідношення параметрів руху моста та візка.

При дослідженні механізмів з декількома степенями вільності є сенс додавати їх поступово, щоб виявити вплив кожного механізму на рух системи в цілому.

13.1. Математичне моделювання власних рухів вантажу, підвішеного до нерухомої опори

Почнемо з моделі вантажу, підвішеного до нерухомої опори, який при виведенні його із стану рівноваги, тобто при відхиленні осі канат-вантаж від вертикалі, починає коливатися. Зазвичай при математичному описі такого

об’єкта його вважають математичним маятником – точковим тілом, підвішеним до нерозтяжної і невагомої нитки, що здійснює коливання в одній площині під дією сили власного тяжіння без врахування будь-яких сил опору. Така модель прийнятна, якщо розміри вантажу набагато менші за довжину нитки, маса вантажу набагато більша маси нитки, довжина нитки та її пружні властивості не занадто великі, щоб ними не можна б було знехтувати, сили опору коливальному руху не діють.

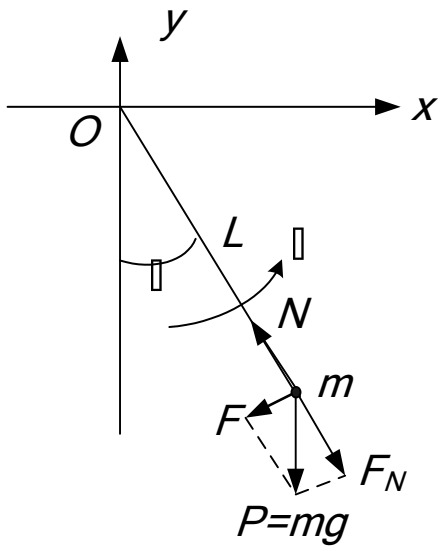


Рис. 13.1. Математичний маятник

Схематичне зображення математичного маятника масою m з довжиною нитки підвісу L подано на рис. 13.1.

Силу тяжіння $P = mg$ ($g = 9.81$ м/с^2 – прискорення сили тяжіння) можна розкласти на 2 складові, одна з яких (F_N) урівноважується силою натягу нитки N, а друга ($F = mg\sin\varphi$) створює відносно точки закріплення нитки O момент $M = -FL$, спрямований у бік зменшення кута φ. Тому його записуємо зі знаком «-».

Момент інерції матеріальної точки відносно точки підвісу дорівнює $J = mL^2$.

Тому рівняння руху математичного маятника $M = J \cdot d\omega / dt$ можна записати у вигляді $-mgL\sin\varphi = mL^2 \cdot d\omega / dt$, або

$$-g\sin\varphi = L\frac{d\omega}{dt}, \qquad \frac{d\varphi}{dt} = \omega. \tag{13.1}$$

Складаємо на їх основі структурну *Simulink*-модель (*S*-модель), показану на рис. 13.2.

При невеликих кутах відхилення маятника її можна лінеаризувати, скориставшись співвідношенням $\sin\varphi \approx \varphi$

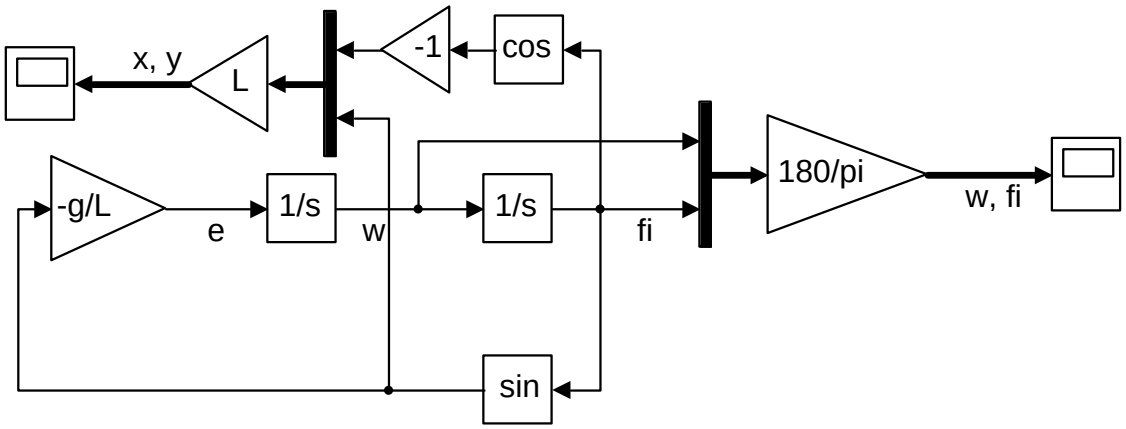


Рис. 13.2. Нелінійна структурна *S*-модель математичного маятника

Характеристичний поліном лінеаризованого об’єкта має вигляд

$$T_0^2 s^2 + 1 = 0, \qquad T_0 = \sqrt{L/g}\,, \tag{13.2}$$

а перехідний процес, зумовлений початковим відхиленням маятника на кут φ_0 ($\varphi(0) = \varphi_0$, $\omega(0) = 0$) описується рівнянням

$$\varphi(t) = \varphi_0 \cos\Omega_0 t, \qquad \omega(t) = -\varphi_0\Omega_0 \sin\Omega_0 t = -\omega_m \sin\Omega_0 t, \tag{13.3}$$

де $\Omega_0 = 1/T_0$ – кругова частота власних (невимушених) коливань лінеаризованої моделі математичного маятника, $\omega_m = \varphi_0\Omega_0 = \varphi_0 / T_0$ – амплітуда коливань кутової швидкості.

У відповідності з (13.2) період коливань становить

$$T_{k0} = 2\pi T_0 = 2\pi\sqrt{L/g}\,. \tag{13.4}$$

Період коливань нелінійної моделі при $\varphi_0 \le 60^\circ$ можна приблизно (з похибкою менше 1%) визначити за більш точною формулою

$$T_{kn} \approx 2\pi\sqrt{L/g}\left(1 + \frac{1}{4}\sin^2\frac{\varphi_0}{2}\right). \tag{13.5}$$

При $m = 4000$ кг, $L = 12$ м та при початковому куті відхилення 30° основні параметри власних коливань складають: $T_0 = \sqrt{L/g} = 1.106$ с, $T_{k0} = 2\pi T_0 = 2\pi\sqrt{L/g} = 6.95$ с, $T_{kn} \approx T_{k0}\left(1 + \frac{1}{4}\sin^2\frac{\varphi_0}{2}\right)$=7.07с., $\omega_{m0} = \varphi_0 / T_0 =$

=27.13 рад/с. Можна також розрахувати максимальні відхилення вантажу по осям x та y: $x_m = x_0 = L\sin\varphi_0 = 6\,\text{м}$, $y_m = y_0 = -L\cos\varphi_0 = -10.4\,\text{м}$.

Графіки власних коливань вантажу, описаного як математичний маятник, отримані за допомогою нелінійної структурної математичної моделі рис. 13.2, представлені на рис. 13.3, 13.4.

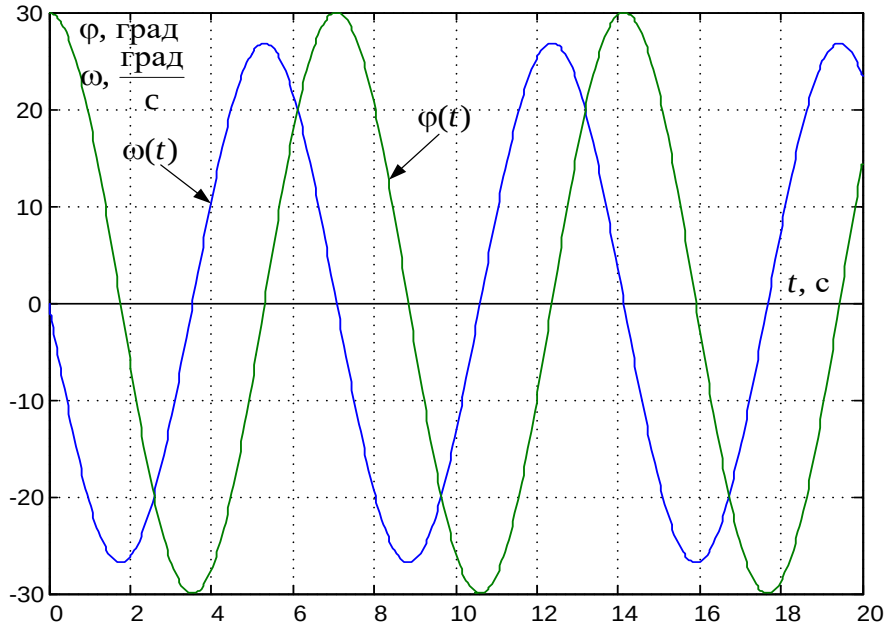


Рис. 13.3. Графіки власних рухів математичного маятника,
у кутових координатах, отримані за допомогою моделі рис. 13.1

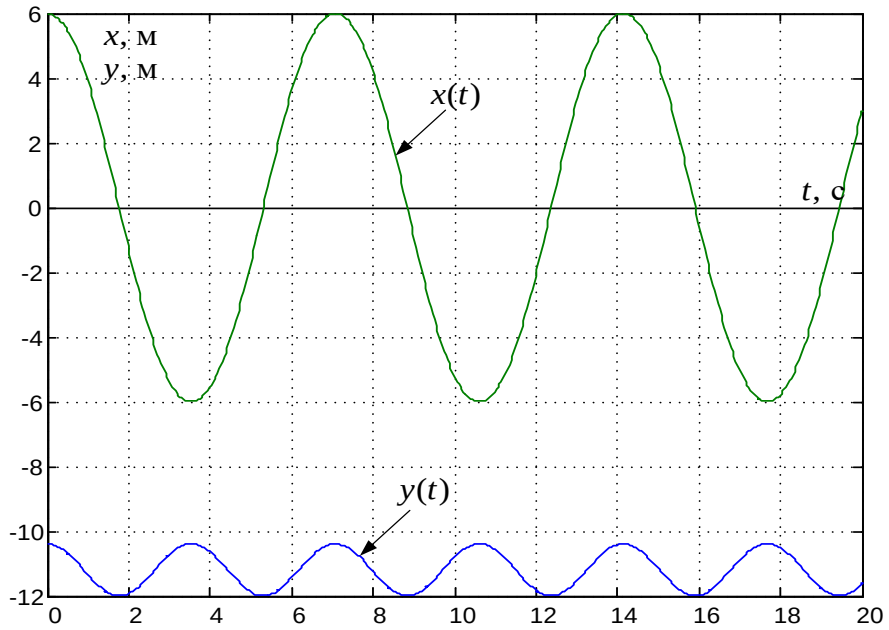


Рис. 13.4. Графіки власних рухів математичного маятника,
у декартових координатах, отримані за допомогою моделі рис. 13.1

Як бачимо, параметри характерних точок перехідних процесів повністю співпадають з розрахованими вище, незважаючи на те, що початковий кут не такий вже й малий.

13.2. Математичне моделювання механічної системи вантаж-візок

Тепер розробимо моделі для дослідження рухів вантажу, у якого точка підвісу прикріплена не до нерухомої опори, а до візка, що здійснює переміщення вантажу у напряму x. Схематичне зображення такого механічного об'єкта подано на рис. 13.5, де прийнято такі позначення: M – маса візка, m – маса вантажу, L – довжина канату, F_x – зусилля, прикладене до візка, v – горизонтальна швидкість візка, s – горизонтальне переміщення візка, φ – кут відхилення вантажу від вертикалі, ω – кутова швидкість коливань вантажу. Розподілення зусиль в цій системі показано на рис. 13.6. Сила тяжіння вантажу розкладається на 2 складових, одна з яких (F_N) урівноважує силу натягу канату N, а друга (F) утворює обертальний момент.

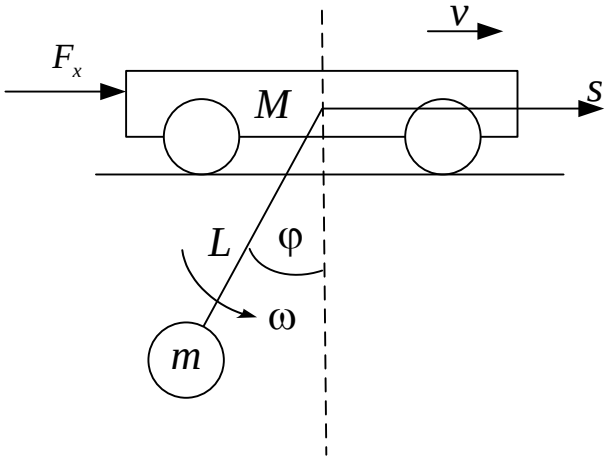


Рис. 13.5. Схематичне зображення візка з підвішеним до нього вантажем

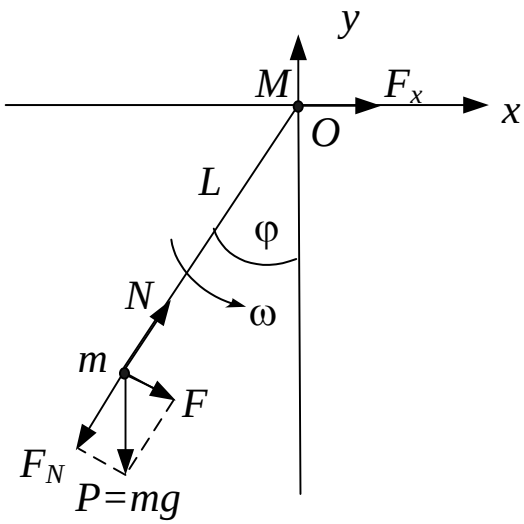


Рис. 13.6. Розподілення зусиль в системі візок-вантаж

Візок, як і вантаж, вважатимемо точковими масами, зосередженими у центрах їх тяжіння. Рух візка відбувається у горизонтальному напряму впродовж осі x під дією деякої сили F_x при постійній довжині канату.

Для складання математичного опису скористаємося рівняннями Лагранжа другого роду, яке у загальному випадку має такий вигляд:

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right)-\frac{\partial \mathcal{L}}{\partial q_i}=Q_i, \tag{13.6}$$

де q_i, $\dot{q}_i$ – узагальнені координати та їх перші похідні за часом, Q_i – узагальнені сили, $\mathcal{L}=W_k-W_p$ – функція Лагранжа, яка дорівнює різниці між кінетичною і потенціальною енергією.

Вирази для визначення просторових координат візка та вантажу в кожний момент часу мають вигляд

$$\begin{aligned}&\vec{s}_M=\left[s_{Mx}(t),\ \ 0\right]=\left[x(t),\ \ 0\right],\\&\vec{s}_m=\left[s_{mx}(t),\ \ s_{my}(t)\right]=\left[x(t)+L\sin\varphi(t),\ \ -L\cos\varphi(t)\right],\end{aligned} \tag{13.7}$$

у результаті диференціювання яких отримаємо такі вирази для проекцій векторів швидкостей руху візка і вантажу на обрані координатні осі:

$$\begin{aligned}&\vec{v}_M=\left[v_{Mx}(t),\ 0\right]=\left[v_x(t),\ 0\right],\\&\vec{v}_m=\left[v_{mx}(t),\ 0\right]\left[v_x(t)+L\,\omega(t)\cos\varphi(t),\ L\,\omega(t)\sin\varphi(t)\right].\end{aligned} \tag{13.8}$$

Для системи візок-вантаж формули потенціальної і кінетичної енергії мають вигляд

$$\begin{aligned}W_k&=\frac{m}{2}\left[\left(v_x(t)+L\omega(t)\cos\varphi(t)\right)^2+\left(L\omega(t)\sin\varphi(t)\right)^2\right]+\frac{M}{2}v_x^2(t)=\\&=\frac{m+M}{2}v_x^2(t)+\frac{mL}{2}\left(L\omega^2(t)+2v_x(t)\omega(t)\cos\varphi(t)\right),\\W_p&=-mgL\cos\varphi(t).\end{aligned}$$

Тоді функція Лагранжа для системи візок-вантаж матиме вигляд:

$$\mathcal{L}=\frac{M+m}{2}v_x^2(t)+\frac{mL}{2}\left(2v(t)\,\omega(t)\cos\varphi(t)+L\,\omega^2(t)+2g\cos\varphi(t)\right). \tag{13.9}$$

Узагальненими координатами і силами для досліджуваної системи являються:

$$\begin{cases}q_1=x,\ \ \dot{q}_1=v_x,\ Q_1=F_x;\\q_2=\varphi,\ \ \dot{q}_2=\omega,\ \ \ Q_2=0.\end{cases} \tag{13.10}$$

Визначивши послідовно усі складові рівняння Лагранжа (13.9), одержуємо такі рівняння:

$$\begin{cases} (M+m)\dfrac{dv_x(t)}{dt} + m\,L\cos\varphi(t)\cdot\varepsilon(t) - m\,L\,\omega^2(t)\cdot\sin\varphi(t) = F(t), \\ L\dfrac{d\omega(t)}{dt} + \cos\varphi(t)\cdot a_x(t) + g\sin\varphi(t) = 0, \end{cases} \tag{13.11}$$

які доповнюємо рівняннями, що пов'язують узагальнені координати з їхніми похідними:

$$\begin{cases} \dfrac{dx(t)}{dt} = v_x(t), \qquad \dfrac{dv_x(t)}{dt} = a_x(t); \\ \dfrac{d\varphi(t)}{dt} = \omega((t), \qquad \dfrac{d\omega(t)}{dt} = \varepsilon((t). \end{cases} \tag{13.12}$$

де $a(t)$ – лінійне прискорення візка; $\varepsilon(t)$ – кутове прискорення вантажу;

Структурна модель, що відповідає рівнянням (13.11), (13.12), наведена на рис. 13.6 [7].

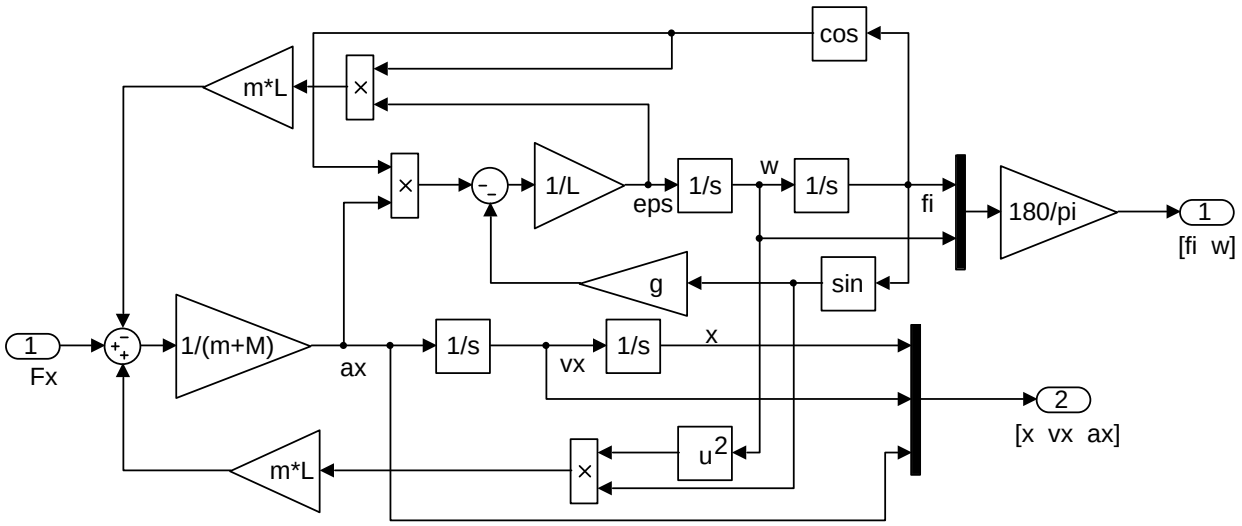


Рис. 13.6. Структурна модель системи візок-вантаж

Перехідні процеси коливань вантажу і візка при стрибкоподібній зміні зусилля, отримані при симуляції структурної моделі рис. 13.6, показані на рис. 13.7.

При моделюванні використано такі параметри: маса візка M =1000 кг, маса вантажу m=500 кг, довжина канату L=10 м, середнє прискорення $a_0 = 0.5$ м/с^2, $F_0 = a_0(M+m) = 750$ Нм.

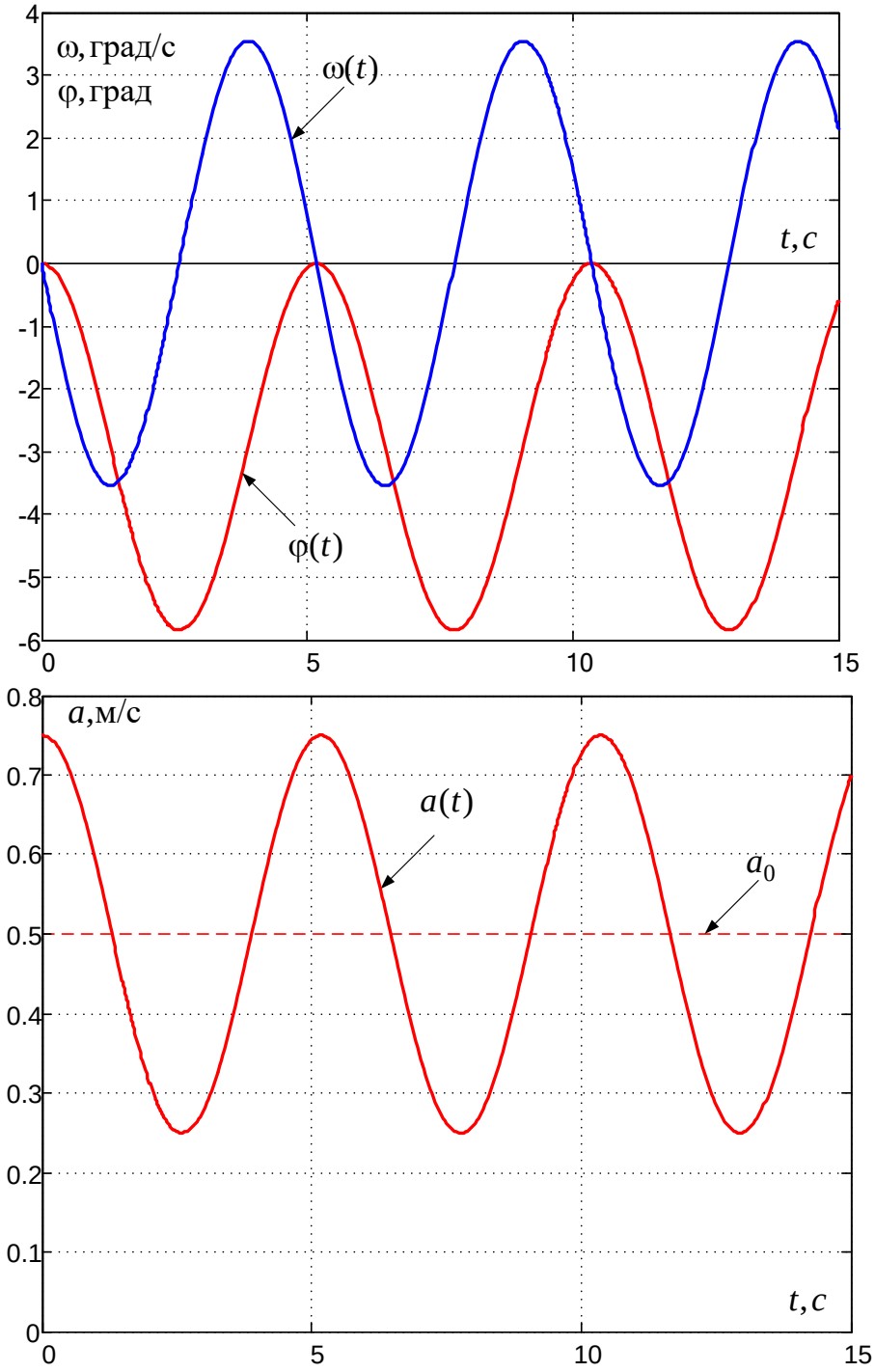


Рис. 13.7. Реакція механічної системи візок-вантаж на стрибок сили $F_x(t) = F_0 \cdot 1(t)$

Для того, щоб оцінити ступінь адекватності розроблених нелінійних моделей, їх треба лінеаризувати. З рис. 13.7 видно, що при пересуванні візка значення кута відхилення канату від вертикалі приймає досить невеликі значення, завдяки чому можна прийняти:

$$\varphi \approx 0, \quad \cos\varphi \approx 1, \quad \sin\varphi \approx \varphi, \quad \omega^2 \approx 0. \tag{13.13}$$

В такому випадку рівняння (13.11) набувають вигляду [7]:

$$\begin{cases} (M+m)\dfrac{dv(t)}{dt} = F(t) - mL\varepsilon(t), \\ L\dfrac{d\omega(t)}{dt} = -a(t) - g\varphi(t). \end{cases} \tag{13.14}$$

Такій системі рівнянь відповідає структурна модель, наведена на рис. 13.8.

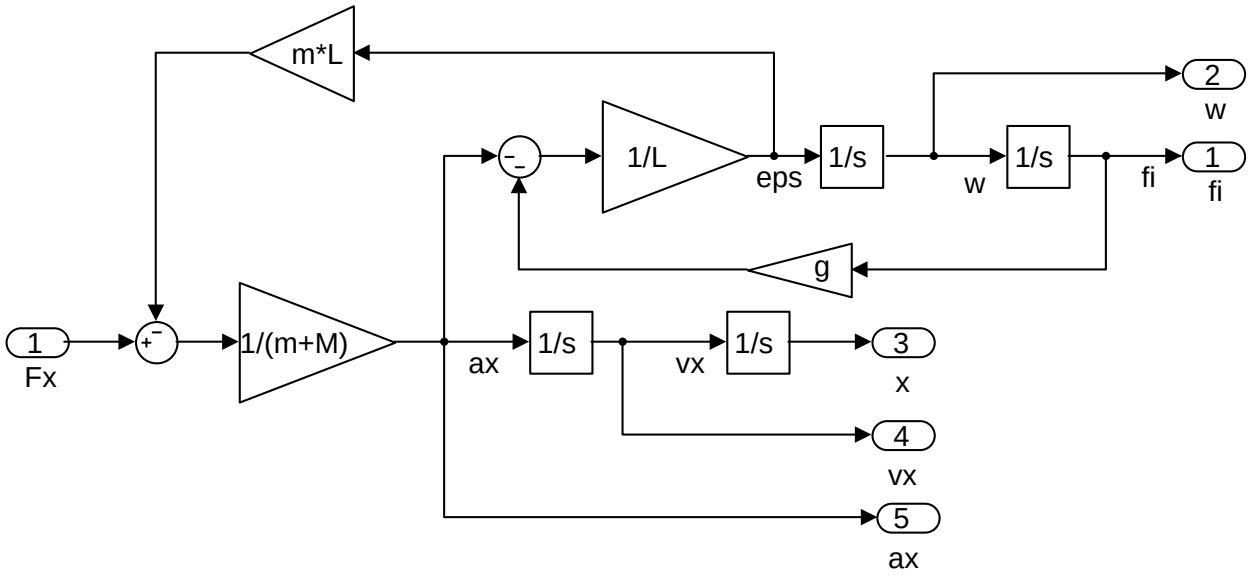


Рис. 13.8. Лінеаризована структурна модель системи візок-вантаж

Із структурної моделі рис. 13.8 знаходимо передавальні функції (ПФ) механічного об’єкта:

$$\frac{\varphi(s)}{a_x(s)} = -\frac{1}{g}\cdot\frac{1}{(T_0^2 s^2 + 1)}, \tag{13.15}$$

$$\frac{\varphi(s)}{F_x(s)} = -\frac{1}{(M+m)g} \cdot \frac{1}{(T_{12}^2 s^2 + 1)}, \tag{13.16}$$

де

$$T_{12} = \sqrt{\frac{L}{g} \cdot \frac{M}{M+m}} = T_0 \sqrt{\frac{M}{M+m}} \tag{13.17}$$

є сталою часу коливань двомасової системи візок-вантаж.

Діленням (13.16) на (13.17) отримуємо:

$$\frac{a_x(s)}{F_x(s)} = -\frac{1}{(M+m)} \cdot \frac{(T_0^2 s^2 + 1)}{(T_{12}^2 s^2 + 1)} \tag{13.18}$$

Із ПФ (13.16) знаходимо реакцію кутових координат вантажу на стрибок зусилля $F_x(t) = F_{0x} \cdot 1(t)$ при нульових початкових умовах:

$$\varphi(t) = -\frac{F_{0x}}{(M+m)g}\left(1 - \cos\frac{t}{T_{12}}\right), \tag{13.19}$$

що відповідає коливанням двомасової системи вантаж-візок з періодом

$$T_{k12} = 2\pi T_{12}$$

навколо похилої осі $\varphi_{mean} = \frac{F_0}{(M+m)g}$ від $\varphi_{\min} = 0$ до $\varphi_{\max} = -\frac{2F_{0x}}{(M+m)g}$.

Шляхом диференціювання (13.19) знайдемо закон зміни кутової швидкості:

$$\omega(t) = \frac{d\varphi(t)}{dt} = \frac{F_{0x}}{(M+m)gT_{12}} \sin\frac{t}{T_{12}} = \frac{F_{0x}}{\sqrt{Lg(M+m)M}} \sin\frac{t}{T_{12}}, \tag{13.20}$$

звідкіля маємо

$$\omega_{\max} = \frac{F_{0x}}{(M+m)gT_{12}} = \frac{\varphi_{\max}}{2T_{12}}. \tag{13.21}$$

З передавальної функції (13.15) знаходимо

$$a(t)=\frac{F_{0x}}{(M+m)}\left[\left(1-\cos\frac{t}{T_{12}}\right)+T_0^2\frac{d^2}{dt^2}\left(1-\cos\frac{t}{T_{12}}\right)\right]=$$

$$=\frac{F_{0x}}{(M+m)}\left(1-\cos\frac{t}{T_{12}}+\frac{T_0^2}{T_{12}^2}\cos\frac{t}{T_{12}}\right)=\frac{1}{(M+m)}\left[1+\left(\frac{T_0^2}{T_{12}^2}-1\right)\cos\frac{t}{T}\right],$$

або з врахуванням (13.16)

$$a(t)=\frac{F_{0x}}{(M+m)}\left[1+\frac{m}{M}\cos\frac{t}{T_{12}}\right]. \qquad (13.22),$$

З рівняння (13.22) визначаємо

$$a_{\max}=\frac{F_{0x}}{M}, \qquad a_{\min}=\frac{F_{0x}}{M+m}\cdot\frac{M-m}{M}=a_{\max}\frac{M-m}{M+m}. \qquad (13.23)$$

Отже, лінійне прискорення коливається навколо середнього значення

$$a_0=(a_{\max}+a_{\min})/2=F_{0x}/(M+m) \qquad (13.24)$$

з амплітудою

$$A_a=\frac{a_{\max}-a_{mean}}{2}=\frac{F_{0x}m}{2M(M+m)}). \qquad (13.25)$$

Розраховуючи характерні параметри графіків рис. 13.5 за наведеними вище формулами, отримуємо такі результати: $T_{12}=0{,}82$ с, $T_{k12}=5{,}18$ с, $\varphi_{\max}=5{,}8^\circ$, $\omega_{\max}=3{,}5$ град/с, $a_{\max}=0{,}75$ м/с^2, $a_{\min}=0{,}25$ м/с^2.

Результати розрахунків співпадають з параметрами перехідних процесів, що свідчить про адекватність розроблених моделей.

Як бачимо, вантаж з великою масою не тільки розгойдується сам, але й розгойдує візок.

Закінчення розгону зазвичай відбувається при майже стрибкоподібному скиданні тягового зусилля в 0. При цьому подальший вигляд перехідних процесів залежить від того в який саме момент часу здійснюється зміна зусилля. Наприклад, поведінка кута відхилення канату з вантажем від вертикалі визначається рівнянням

$$\varphi(t) = \varphi_0 \cos\frac{t}{T_{12}} + \omega_0 T_{12} \sin\frac{t}{T_{12}}, \qquad (13.26)$$

з якого видно, що амплітуда гармонічних коливань в усталеному режимі залежить від початкових значень координат в момент зміни зусилля.

На рис. 13.9, 13.10 показані перехідні процеси в досліджуваній системі під дією зусилля, що стрибкоподібно змінюється з умов забезпечення переміщення одномасової механічної системи за трапецоїдальною тахограмою з усталеною швидкістю $v_0 = 1{,}8$ м/с і усталеним прискоренням $a_0 = 0{,}5$ м/с^2.

В цьому випадку зусилля, що визначає час розгону і гальмування візка складає $t_{p0} = v_0 / a_0 = 3{,}6$ с. Як видно з наведених рисунків, елементи механічної системи в розглянутому режимі здійснюють незагасаючі коливання.

З аналізу рівняння (13.26) та графіків рис. 13.7 випливає, що позбутися цих коливань можна, якщо зробити час перемикання зусилля кратним періоду коливань двомасової системи.

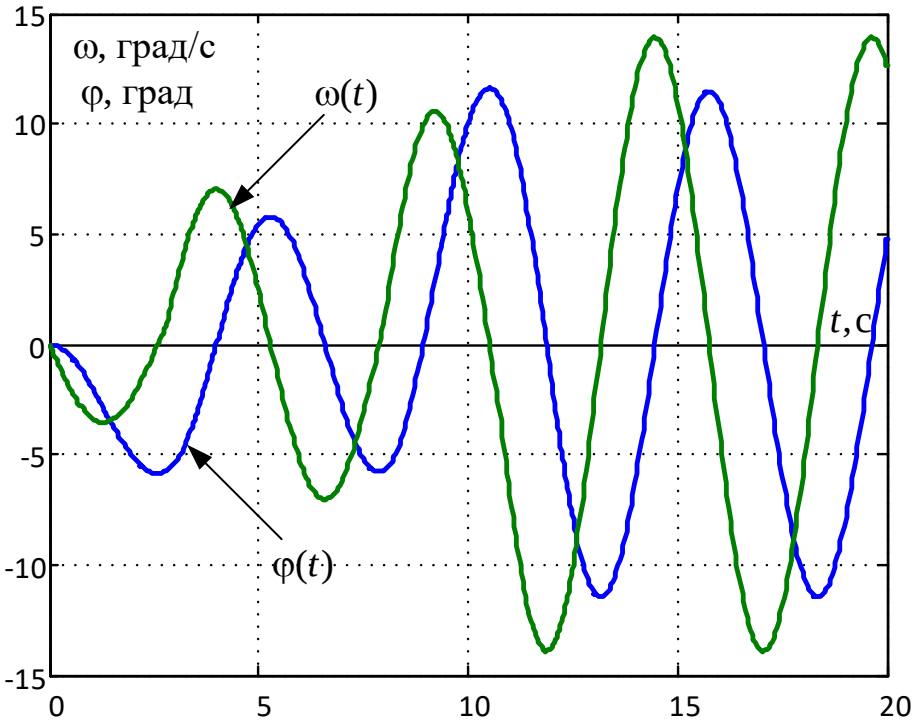


Рис. 13.9. Коливання вантажу при переміщенні візка з стрибкоподібною зміною тягового зусилля при $t_p = t_r = t_{p0}$

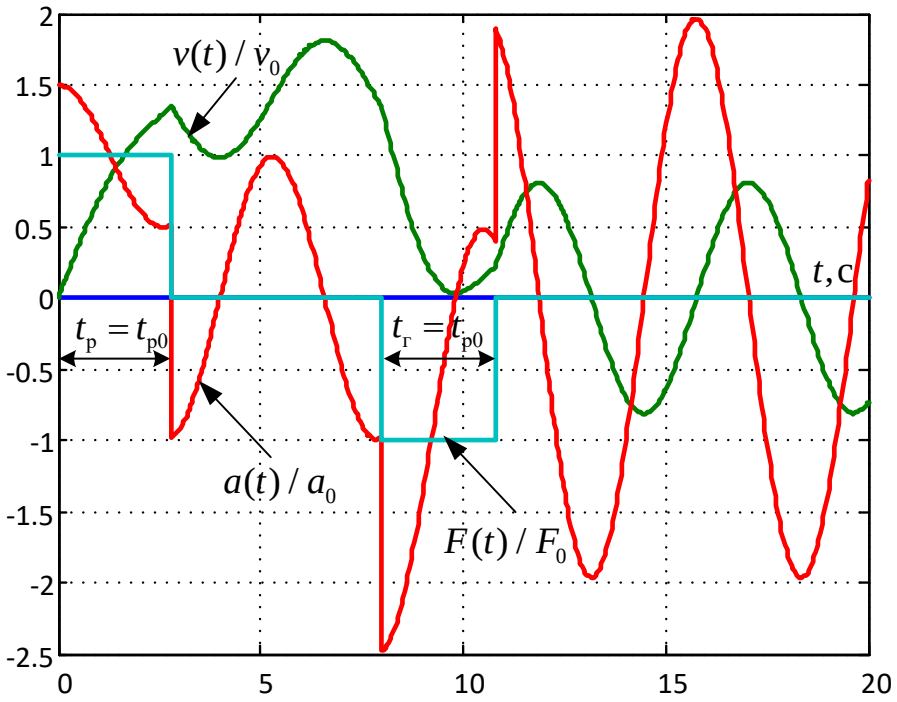


Рис. 13.10. Перехідні процеси зміни тягового зусилля, прискорення та швидкості візка при $t_{р} = t_{г} = t_{р0}$

Цей висновок підтверджується графіками перехідних процесів, представлених на рис. 13.11, 13.12. В останньому випадку система здійснює тільки одне коливання при розгоні і одне при гальмуванні.

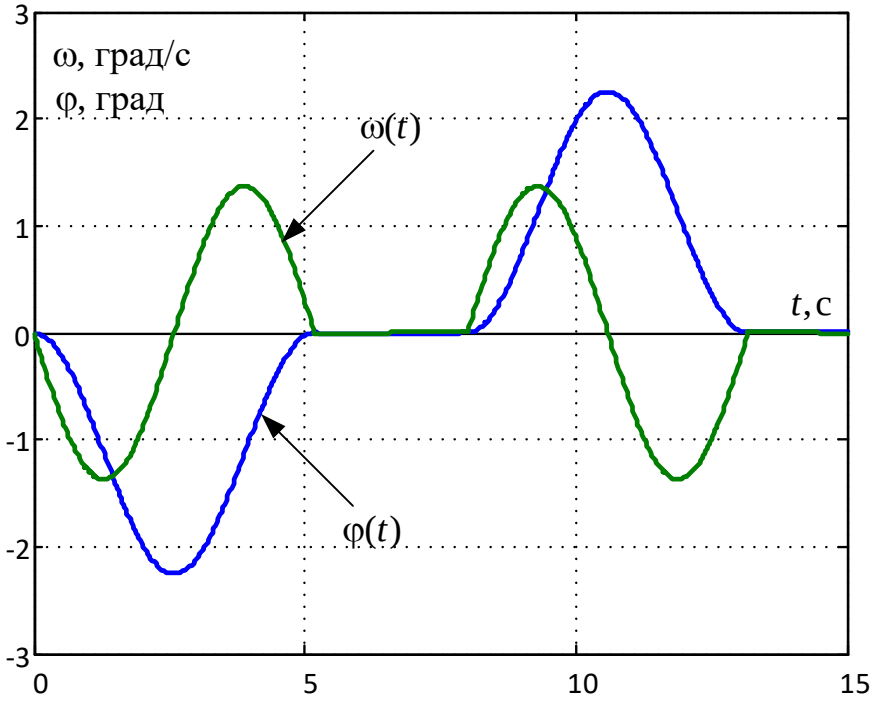


Рис. 13.11. Коливання вантажу при переміщенні візка з стрибкоподібною зміною тягового зусилля при $t_{р} = t_{г} = T_{k12}$

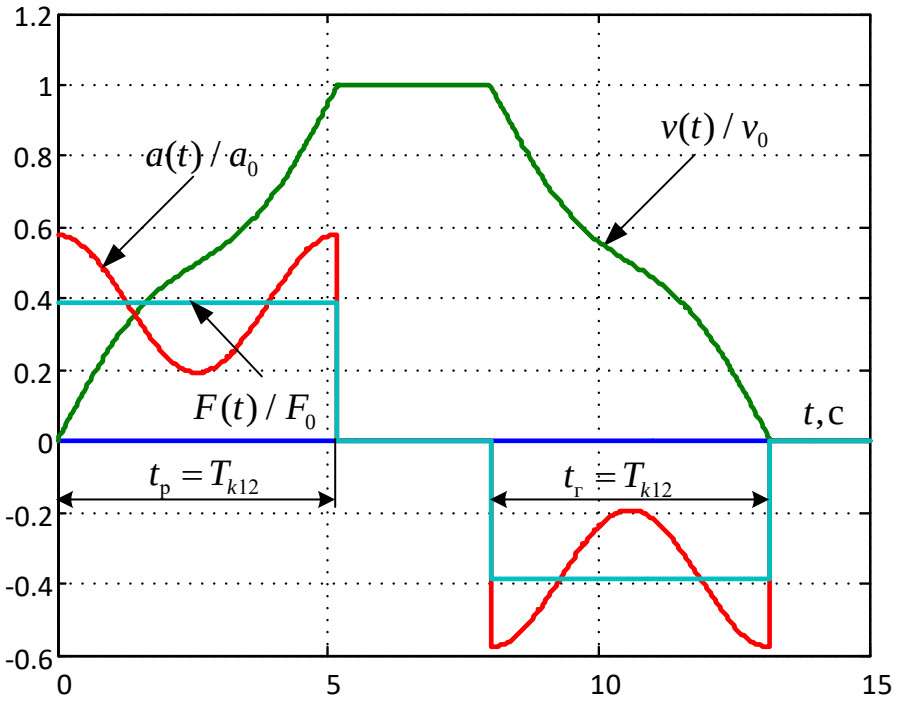


Рис. 13.12. Перехідні процеси зміни тягового зусилля, прискорення та швидкості візка при $t_р = t_г = T_{k12}$

13.3. Математичне моделювання механічної системи вантаж-підіймальний пристрій

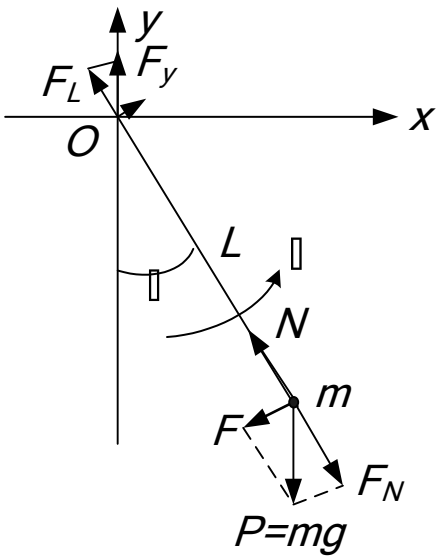


Рис. 13.17. Схематичне зображення механічної системи вантаж-підіймальний пристрій

Розглянемо тепер математичний маятник з підіймальним пристроєм, встановленим на нерухомій опорі, який може намотувати і канат на барабан і змотувати його, змінюючи у такий спосіб відстань від центру коливань маятника до його центру мас. Зміна цієї відстані відбувається під дією проекції вертикального зусилля F_y (див. рис. 13.13), спрямованого по осі y, на рухому вісь вантаж-канат, що обертається у вертикальній площині узгоджено з коливаннями

вантажу. Надалі ефект від роботи підіймального пристрою будемо називати зміною довжини канату.

Для складання математичного опису згідно з рівнянням Лагранжа другого роду (13.6) визначаємо: положення вантажу у площині *xy*

$$\vec{s}_m = \left\lfloor s_{mx}(t), \quad s_{my}(t) \right\rfloor = \left[L(t)\sin\varphi(t), \quad -L(t)\cos\varphi(t) \right],$$

потенціальну енергію

$$W_p = -m\,g\,L(t)\cos\varphi(t),$$

вектор швидкостей

$$v_{mx} = v_L(t)\sin\varphi(t) + L(t)\omega(t)\cos\varphi(t), \quad v_{my} = -v_L(t)\cos\varphi(t) + L(t)\omega(t)\sin\varphi(t)$$

та кінетичну енергію

$$W_k = \frac{m}{2}(v_L^2(t)\sin^2\varphi(t) + L^2(t)\omega^2(t)\cos^2\varphi(t) + 2v_L L(t)\omega(t)\sin\varphi(t)\cos\varphi(t) +$$
$$+ v_L^2(t)\cos^2\varphi(t) + L^2(t)\omega^2(t)\sin^2\varphi(t) - 2v_L L(t)\omega(t)\sin\varphi(t)\cos\varphi(t))$$
$$= \frac{m}{2}(v_L^2(t) + L^2(t)\omega^2(t)).$$

У якості узагальнених координат виберемо кутове переміщення та лінійне переміщення по осі, спрямованій повздовж канату:

$$\begin{cases} q_1 = \varphi, \quad \dot{q}_1 = \dfrac{d\varphi}{dt} = \omega, \quad Q_1 = 0; \\ q_2 = L, \quad \dot{q}_2 = \dfrac{dL}{dt} = v_L, \quad Q_2 = F_L = F_y\cos\varphi. \end{cases}, \tag{13.27}$$

В результаті у такий же спосіб, як у попередньому пункті, отримуємо математичний опис досліджуваної системи [7]:

$$\begin{cases} \dfrac{d\omega(t)}{dt} = \dfrac{-g\sin\varphi(t) - 2v_L(t)\omega(t)}{L(t)}, \\ \dfrac{dv_L}{dt} = \left(\dfrac{F_y}{m} - g\right)\cos\varphi(t) = a_L. \end{cases} \tag{13.28}$$

Структурна математична модель, побудована за рівняннями (13.28) з урахуванням (13.27), зображена на рис. 13.14. Графіки перехідних процесів, отримані при використанні моделі рис. 13.14, зображені на рис. 13.15, 13.16.

Графіки зміни довжини канату розраховані з умов, що усталені значення прискорення та швидкості по осі канату складають $a_u = 0.1$м/с^2, $v_u = 0.4$м/с, а довжина канату спочатку зменшується на $\Delta L = 7$ м, а потім зростає на ту ж саму величину.

З графіків видно, що при зменшенні довжину канату за лінійним законом зростає не тільки частота, а й амплітуда куту відхилення вантажу від вертикалі. Усталена частота коливань збігається з частотою, визначеною аналітично, а аналітичний вираз для закону зміни амплітуди залишається невідомим, тому що досліджувана система є суттєво нелінійною.

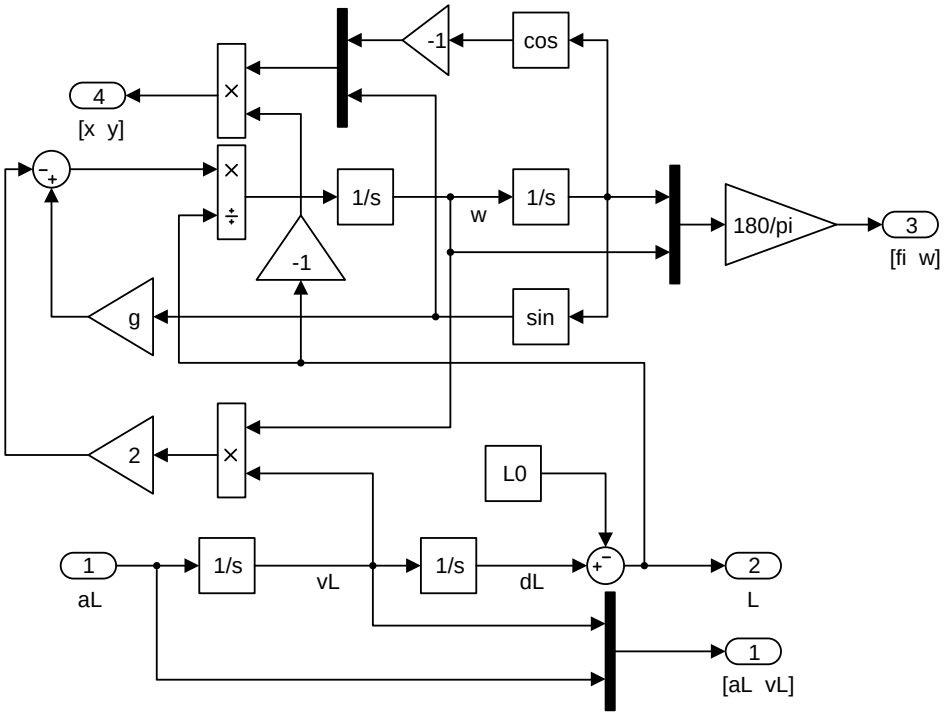


Рис. 13.14. Структурна модель вантажу з підіймальним пристроєм

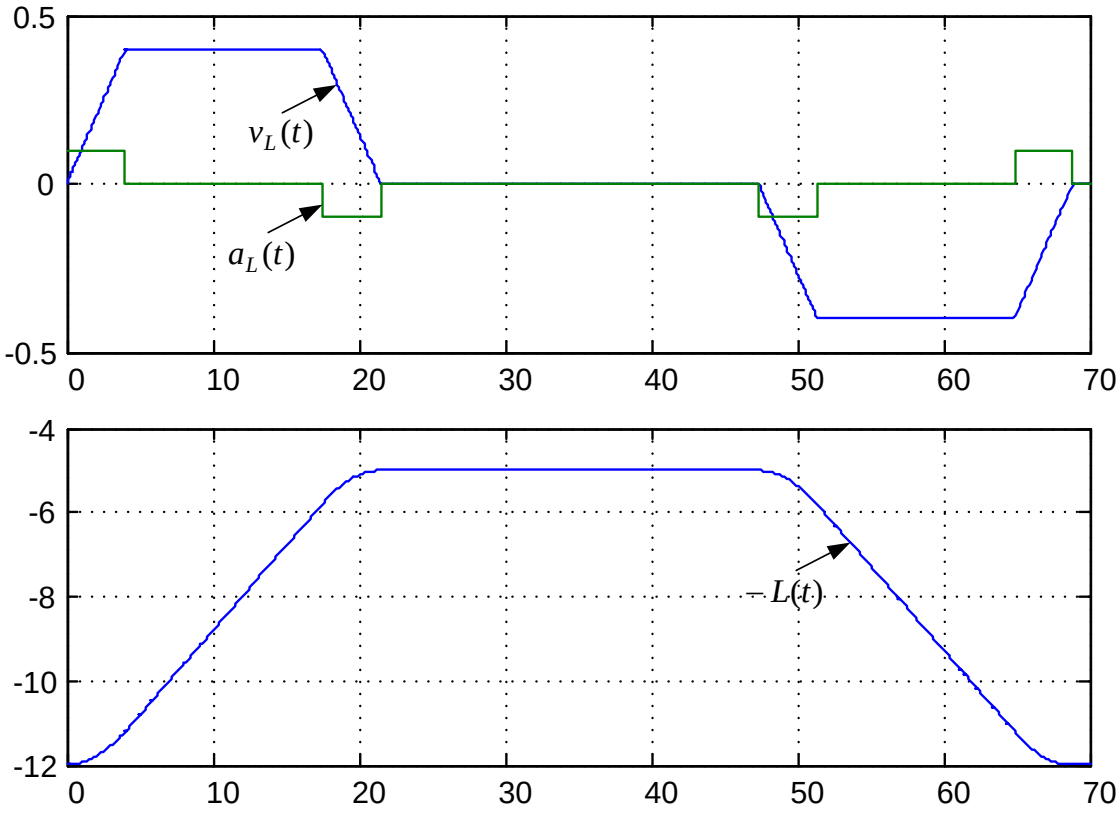


Рис. 13.15. Графіки координат руху вантажу по осі канату

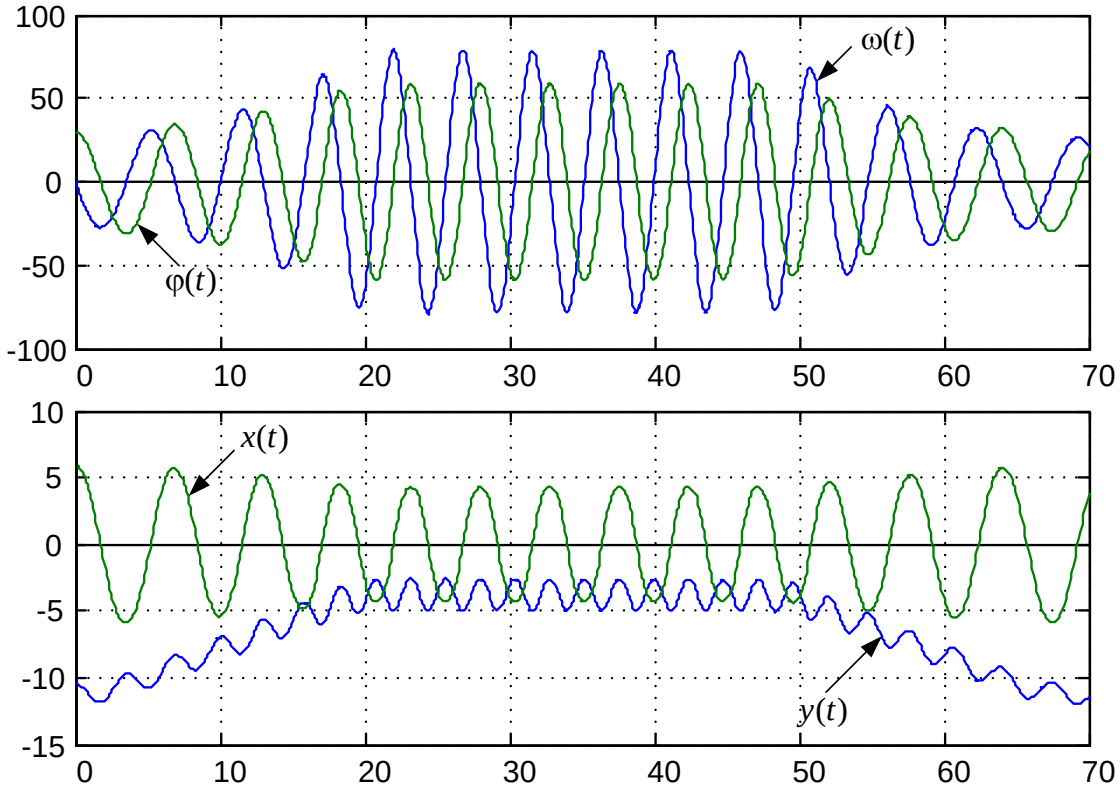


Рис. 13.16. Графіки коливань вантажу при змінній довжині канату

При об’єднанні в одну механічну систему трьох або чотирьох механізмів шляхом додавання моста та піднімача рівняння Лагранжа, математичний опис і структурні математичні моделі багатокоординатних об’єктів настільки ускладнюються, що з’являється сенс переходу від математичних моделей до віртуальних механічних моделей.

Контрольні питання та завдання

1. Яким диференційним рівнянням описуються коливання математичного маятника? Які припущення приймаються при складанні цього математичного опису? Як розрахувати момент інерції математичного маятника відносно точки підвісу?

2. При якому припущенні математична модель ідеального маятника стає лінійною? Які передавальні функції має лінеаризований маятник? Як розрахувати період коливань ідеального маятника при невеликій амплітуді коливань?

3. Як скласти математичний опис багатомасових та багатокоординатних взаємопов’язаних механічних об’єктів?

4. Наведіть у загальному вигляді рівняння Лагранжа 2-го роду. Складіть ці рівняння для механічної системи візок, що рухається по лінійній траєкторії, з підвішеним до нього вантажем. Наведіть основні передавальні функції цієї двомасової системи. Як розрахувати її період коливань?

5. Як позбутися коливань вантажу у двомасовій системі візок-вантаж після завершення руху візка?

14. МОДЕЛЮВАННЯ ДИСКРЕТНИХ ЛІНІЙНИХ ДИНАМІЧНИХ ОБ'ЄКТІВ У СЕРЕДОВИЩІ *Simulink*

14.1. Загальні поняття про дискретну інформацію

Для ***дискретної (цифрової) інформації*** характерні такі властивості: квантування за часом; квантування за рівнем; запізнення; екстраполяція.

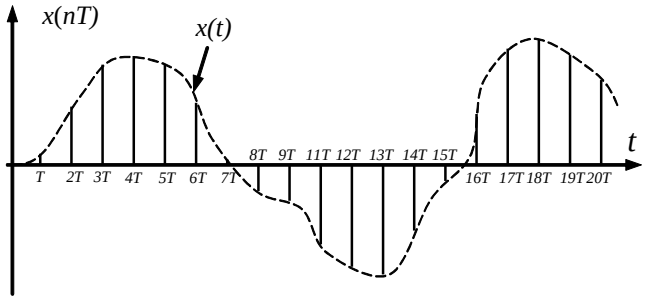


Рис. 14.1. Квантування за часом

Квантування за часом здійснюється шляхом фіксації неперервного вхідного сигналу у дискретні моменти часу $t_i = nT, \quad n = 1,2...$, як це показано на рис. 14.1. Як бачимо, вихідний сигнал елементу, що здійснює таке квантування, має характер послідовності імпульсів різної амплітуди, але постійної частоти. Величина T, що визначає час між двома сусідніми імпульсами при такому квантуванні, називається ***тактом квантування***, або ***періодом дискретності,*** або ***періодом переривання*** (в англомовній літературі цю величину називають ***Sample Time*** і позначають T_s). Дискретна функція, отримана в результаті такого перетворення називається ***решітчастою функцією***. Квантування за часом здійснюється ***імпульсними елементами***. Зображення ідеального імпульсного елементу на структурних схемах подано на рис. 14.2.

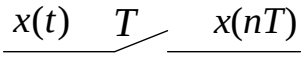


Рис. 14.2. Структурна схема ідеального імпульсного елементу

Величина періоду квантування залежить від тактової частоти цифрового пристрою та від обмежень, що накладаються на нього датчиками.

Квантування за рівнем формується шляхом фіксації неперервного сигналу на наперед визначених (фіксованих) рівнях у довільні моменти часу. Тобто, неперервний сигнал довільної форми замінюється послідовною низкою фіксованих дискретних значень відповідно до статичної характеристики перетворювача неперервного сигналу у дискретний. Рис. 14.3 пояснює процес квантування за рівнем за допомогою перетворювача з багатопозиційною релейною статичною характеристикою.

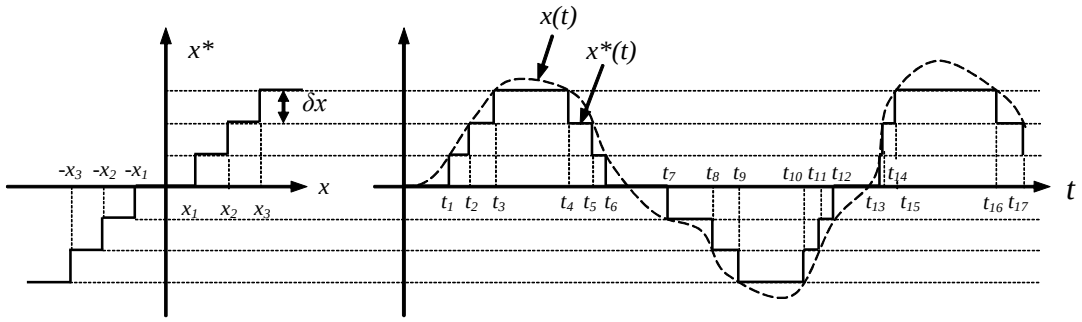


Рис. 14.3. Квантування неперервного сигналу за рівнем

З рис. 14.3 видно, що в результаті квантування за рівнем неперервного сигналу $x(t)$ довільної форми, отримуємо неперервний сигнал $x^*(t)$, що має ступінчастий характер і дорівнює вхідному $x(t)$ лише в деякі, наперед невизначені моменти часу t_0, t_1,... . Зміна величини вихідного сигналу $x^*(t)$ здійснюється лише тоді, коли вхідний сигнал $x(t)$ перетинає якійсь з рівнів релейної характеристики x_1, x_2,... . Зазвичай різниці між двома сусідніми рівнями квантованого сигналу $x^*(t)$ однакові ($x_i^* - x_{i-1}^* = \delta x = const$). Параметр δx називають ***дискретою за рівнем***. Її величина визначається або кількістю розрядів у слові цифрового або перетворюючого пристрою (при роботі їх в арифметиці з фіксованою крапкою), а саме, ціною одного розряду, або властивостями датчиків вимірюваних сигналів. При роботі цифрових пристроїв у арифметиці з плаваючою крапкою або при використанні багаторозрядних пристроїв ефектом квантування за рівнем звичайно нехтують.

При моделюванні у *Simulink* релейне квантування здійснює блок ***Quantizer*** бібліотеки *Discontinuities*, як це показано на рис. 14.4.

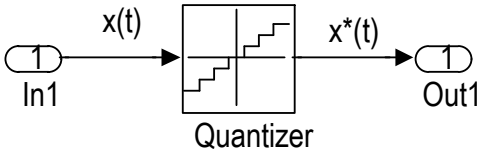


Рис. 14.4. *Simulink*-модель квантування неперервного сигналу за рівнем

Його статична характеристика має зону нечутливості $[-\delta x/2,\ \delta x/2]$. Внаслідок цього він здійснює округлювання чисел до величин, кратних дискреті квантування, заданої параметром *Quantization interval*, за правилами округлювання. Це продемонстровано графіками рис. 14.5 релейного перетворення синусоїди з одиничною амплітудою блоком *Quantizer* з $\delta x = 0.2$.

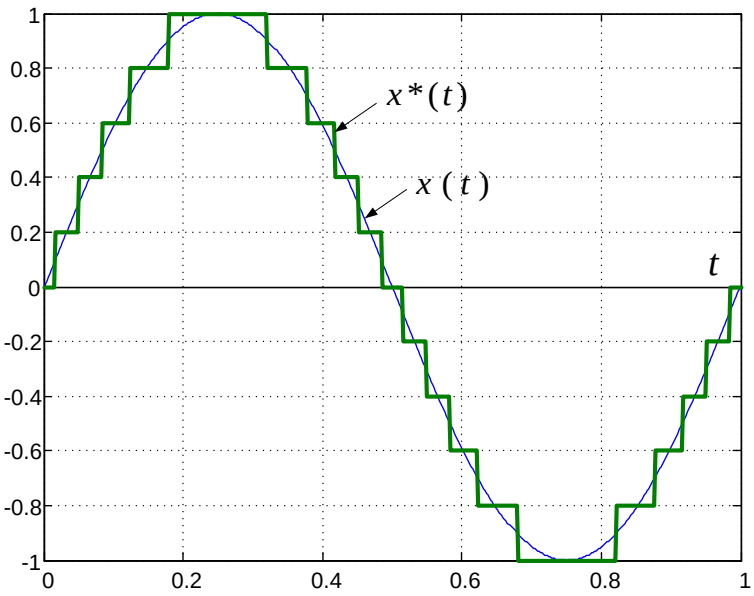


Рис. 14.5. Релейне квантування неперервного сигналу *Simulink*-блоком *Quantizer*

Ефект наявності чистого, або транспортного ***запізнювання (Transport Delay)*** у цифрових системах зумовлений часом, необхідним цифровому пристрою для виконання алгоритму керування, тобто тривалість цієї затримки визначається швидкодією обчислювального пристрою.

В цифрових системах наявність запізнення приводить до того, що дискретні сигнали змінюють свої значення не в моменти часу, кратні періоду дискретизації nT, а в моменти $nT+\tau$, тобто функція $x(nT)$ перетворюється у функцію $x(nT-\tau)$. При простих алгоритмах цим явищем нехтують, тобто вважають $\tau=0$.

Екстраполяцією називають процес визначення дискретного сигналу у проміжках часу між моментами формування ідеальних імпульсів у вигляді решітчастої функції. У решітчастої функції сигнал між імпульсами має нульове значення. Але, якщо алгоритм керування реалізується будь-яким цифровим пристроєм, то значення сигналу, розраховане у деякий дискретний момент часу, утримується (фіксується) в оперативній пам'яті впродовж усього періоду переривання. Оскільки утримуваний на періоді сигнал є незмінним, тобто таким, що описується степеневим поліномом нульового порядку, то цей спосіб екстраполяції називають фіксацією або екстраполяцією нульового порядку (англ. ***Zero Order Hold***, скорочено ***ZOH***). Ще раз підкреслимо, що цей спосіб екстраполяції є природним для цифрових пристроїв з запам'ятовуванням. Примусово можна змінити тип екстраполяції, наприклад, застосувати поліноміальну екстраполяцію першого порядку (англ. ***First Order Hold***, скорочено ***FOH***).

14.2. Форми математичного опису імпульсних лінійних систем

Лінійні стаціонарні імпульсні динамічні системи з періодом переривання T описуються у просторі стану різницевими матричними рівняннями [10, 14, 15].

$$\mathbf{x}(kT)=\mathbf{A}\mathbf{x}(kT-T)+\mathbf{B}\mathbf{u}(kT-T) \tag{14.1}$$

з початковими умовами

$$\mathbf{x}(t_0)=\mathbf{x}_0 \tag{14.2}$$

та лінійним матричним рівнянням виходу

$$\mathbf{y}(kT)=\mathbf{C}\mathbf{x}(kT)+\mathbf{D}\mathbf{u}(kT). \quad (14.3)$$

Для переходу до операторної форми сигнали, що запізнюються у часу на період T помножують на z^{-1}:

$$z=\exp(Ts), \quad (14.4)$$

де z – дискретний оператор.

В результаті такого перетворення рівняння (14.1) та (14.2) набувають вигляду [10, 14, 15]:

$$z\mathbf{x}(z)=\mathbf{A}\mathbf{x}(z)+\mathbf{B}\mathbf{u}(z), \quad (14.5)$$

$$\mathbf{y}(z)=\mathbf{C}\mathbf{x}(z)+\mathbf{D}\mathbf{u}(z). \quad (14.6)$$

Формальна аналогія математичного опису неперервних та дискретних систем в операторній формі простору часу дає можливість використовувати для їх перетворення, аналізу та синтезу однаковий математичний апарат.

Зокрема, перехід від рівнянь у просторі стану до дискретних матричних передавальних функцій здійснюється за формулами, аналогічними формулам (4.9) і (4.10) для неперервних систем:

$$\mathbf{W}_x(z)=\frac{\mathbf{x}(z)}{\mathbf{u}(z)}=(z\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}=\frac{Adj(z\mathbf{I}-\mathbf{A})}{det(z\mathbf{I}-\mathbf{A})}\mathbf{B}, \quad (14.7)$$

$$\mathbf{W}_y(z)=\frac{\mathbf{y}(z)}{\mathbf{u}(z)}=\mathbf{C}(z\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}+\mathbf{D}=\mathbf{C}\frac{Adj(z\mathbf{I}-\mathbf{A})}{det(z\mathbf{I}-\mathbf{A})}\mathbf{B}+\mathbf{D}. \quad (14.8)$$

Складові дискретних матричних ПФ записують у поліноміальній формі відносно оператора z

$$W_{ij}(z)=\frac{y_i(z)}{u_j(z)}=\frac{H_{ij}(z)}{G(z)}=\frac{b_m z^m+b_{m-1}z^{m-1}+...+b_1 z+b_0}{a_n z^n+a_{n-1}z^{n-1}+...+a_1 z+a_0}=, \quad m\le n \quad (14.9)$$

або відносно зворотної до нього величини z^{-1}

$$W_{ij}(z^{-1})=\frac{y_i(z)}{u_j(z)}=\frac{b_0 z^{-m}+b_1 z^{-(m-1)}+...+b_{m-1}z^{-1}+b_m z^{-(n-m)}}{a_0 z^{-n}+a_1 z^{-(n-1)}+...+a_{n-1}z^{-1}+a_n}. \quad (14.10)$$

Знаменник дискретної ПФ також зветься характеристичним поліномом а чисельник – поліномом впливу

Їх корені утворюють вектор полюсів $\mathbf{z} = [z_1 \quad z_2 \quad ... \quad z_n]$ та вектор нулів $\mathbf{z}_z = [z_{z1} \quad z_{z2} \quad ... \quad z_{zm}]$.

Скалярній передавальній функції (14.10) відповідає різницеве рівняння

$$a_n y_i(kT) + a_{n-1} y_i(kT - T) + \ldots + a_1 y_i(kT - (n-1)T) + a_0 y_i(kT - nT) =$$

$$= b_m u_j(kT) + b_{m-1} u_j(kT - T) + \ldots + b_1 u_j(kT - (m-1)T) + b_0 u_j(kT - mT).$$

Використовуючи розкладення поліномів у чисельнику та знаменнику передавальної функції (14.9) на множники, отримуємо

$$W_{ij}(s) = K \frac{(z - z_{z1})(z - z_{z2}) \ldots (z - z_{zm})}{(z - z_1)(z - z_2) \ldots (z - z_n)}, \qquad K = \frac{b_m}{a_n}. \tag{14.11}$$

14.3. Характеристика блоків бібліотеки *Discrete*

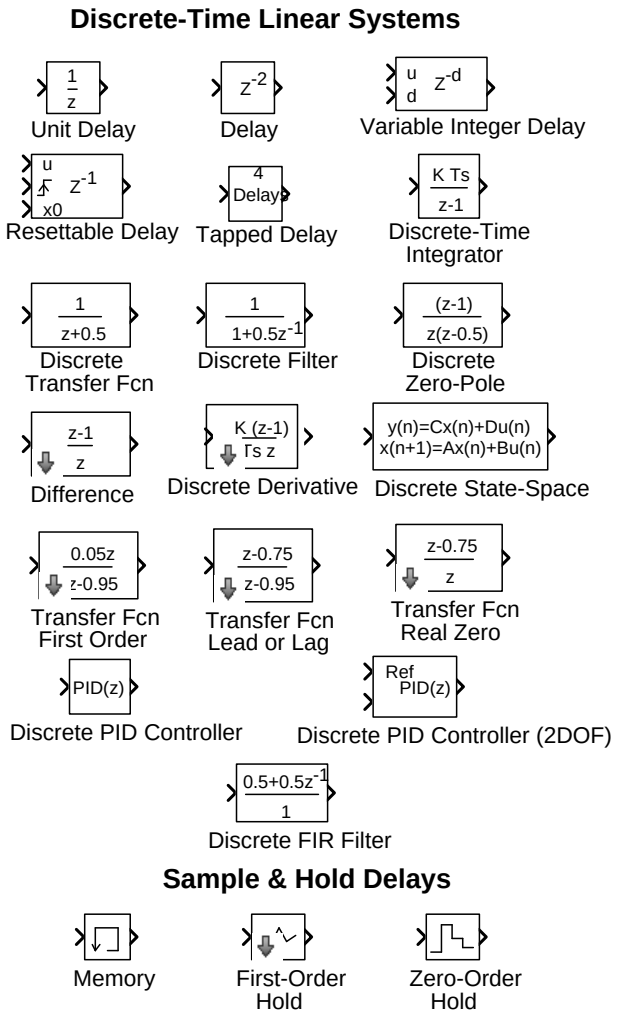


Рис. 16.6. Бібліотека *Discrete*

У цю бібліотеку, представлену на рис. 14.6, входять дискретний інтегратор, екстраполятори, ланки запізнювання на 1 або декілька періодів дискретності, дискретні динамічні ланки загального виду з різними способами їхнього математичного опису та ПІД-регулятори.

Усі ці блоки мають у діалоговому вікні введення параметрів поле з ім'ям ***Sample Time***. У цьому полі може вводитися одне значення, що інтерпретується як період дискретності T_s, або два значення, перше з який сприймається як період дискретності T_s, а друге – як запізнення *offset.*

У цьому випадку зміна вихідних сигналів ланок будуть здійснюватися в дискретні моменти часу.

Кожен дискретний блок має вбудований переривач (ідеальний імпульсний елемент) на вході і екстраполятор нульового порядку на виході, так що вхідні сигнали дискретних блоків змінюються тільки в моменти часу t_d, а вихідні сигнали між двома сусідніми перериваннями утримуються на постійному рівні.

Моделі, які отримують у своєму складі як неперервні, так і імпульсні динамічні ланки, називають ***цифро-аналоговими***.

При моделюванні цифро-аналогових систем можна використовувати методи ЧІ з автоматичним вибором кроку або методи з постійним кроком при $h = T_s$.

При моделюванні цифрових та цифро-аналогових систем, які отримують у своєму складі дискретні блоки з різними періодами дискретності, можна використовувати тільки методи ЧІ з автоматичним вибором кроку.

Для узгодження двох послідовно з'єднаних дискретних блоків, які працюють з різними періодами дискретності T_{s1} і T_{s2}, необхідно дотримуватись таких правил:

- якщо $T_{s1} > T_{s2}$, розташуйте між ними ти блок *Unit Delay* з $T_s = T_{s1}$;
- якщо $T_{s1} < T_{s2}$, розташуйте між ними ти блок *Zero-Order Hold* з $T_s = T_{s2}$.

Блок ***Unit Delay* (*одиничне дискретне запізнювання*) р**обить затримку сигналу й утримання його на цьому рівні на один період квантування:

$$W(z) = z^{-1}. \qquad (14.12)$$

Початкова умова *Initial condition* задає значення виходу блока на першому періоді моделювання системи, на якому вихід блока *Unit Delay* не визначений. Блок підтримує тільки скалярні вхід і вихід.

Блок ***Tapped Delay*** **р**обить затримку сигналу й утримання його на цьому рівні на задану параметром *Number of delays* кількість періодів квантування, а у ланці ***Variable Integer Delay*** кількість періодів запізнення визначається зовнішнім сигналом: $W(z) = z^{-d}$.

Блок ***Discrete-Time Integrator*** **(*дискретний інтегратор*)** виконує чисельне інтегрування вхідного сигналу з періодом дискретності T_s (*Sample Time*) і початковою умовою x_0 (*Initial condition*). Дискретний інтегратор може використовувати один із трьох методів чисельного інтегрування *(Integrator method)*: ***Forward Euler***, ***Backward Euler*** і ***Trapezoidal*** .

Передавальна функція цифрового інтегратора залежить від методу чисельного інтегрування. В основу цих методів покладено геометричний сенс визначеного інтеграла, який дорівнює площі фігури, обмеженою кривою підінтегральної функції $u(t)$, віссю часу та вертикалями, що виходять з точок границь інтегрування $t = t_0$ і $t = t_k$. Більшість із методів цифрового інтегрування полягає у розбиванні фігури, площу якої треба підрахувати на декілька більш простих фігур, площі яких обчислюються за простими формулами, з подальшим сумуванням цих площ. Найчастіше інтервал інтегрування поділяють на декілька рівних частин, на кожній з яких виконують локальну інтерполяцію підінтегральної функції степеневими поліномами. В залежності від степені полінома n утворюються різні методи чисельного інтегрування: при $n = 0$ – метод прямокутників, при $n = 1$ – метод трапецій, при $n = 2$ – метод параболічних трапецій, який зазвичай називають методом Сімпсона. Цей ряд можна продовжувати, але при дискретизації неперервних динамічних ланок підстановочними методами обмежуються заміною аналогових інтеграторів цифровими, синтезованими методами прямокутників та трапецій.

На рис. 14.7 наведено ілюстрації до досліджуваних методів чисельного інтегрування.

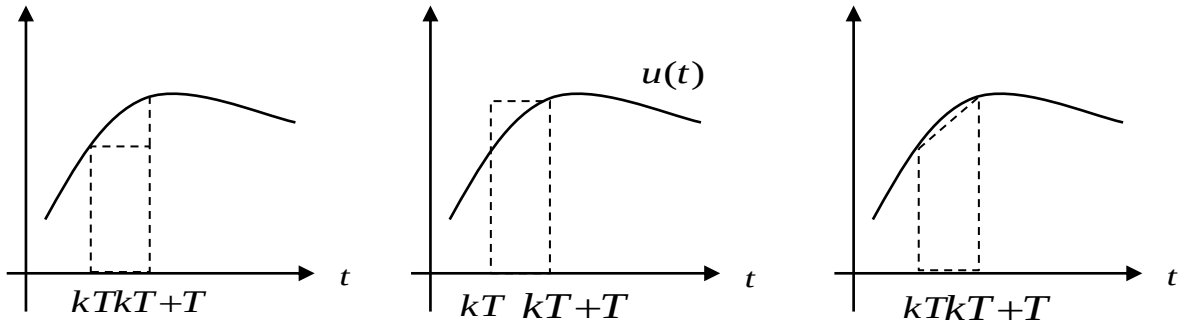


Рис. 14.7. Ілюстрація до методів чисельного інтегрування:
а) пряма апроксимація Ейлера; б) зворотна апроксимація Ейлера;
в) метод трапецій

Один крок чисельного інтегрування ***методом прямої апроксимації Ейлера (Forward Euler)***, який ще називають ***методом лівобічних прямокутників***, згідно з рис. 14.7а, описується різницевим рівнянням

$$y_{FE}(kT+T) = y_{FE}(kT) + Tu(kT), \tag{14.13}$$

яке в операторній формі набуває вигляду

$$zy_{FE}(z) = y_{FE}(z) + Tu(z). \tag{14.14}$$

Після перегрупування (14.14) легко визначаємо передавальну функцію інтегратора:

$$W_{FE}(z) = \frac{y_{FE}(z)}{u(z)} = \frac{T}{z-1} = \frac{Tz^{-1}}{1-z^{-1}}. \tag{14.15}$$

За аналогією синтезуємо цифровий інтегратор ***методом зворотної апроксимації Ейлера (Backward Euler)***, який ще називають ***методом правобічних прямокутників***. Згідно з рис. 14.7б

$$y_{BE}(kT+T) = y_{BE}(kT) + Tu(kT+T), \tag{14.16}$$

$$zy_{BE}(z) = y_{BE}(z) + zTu(z), \tag{14.17}$$

$$W_{BE}(z) = \frac{y_{BE}(z)}{u(z)} = \frac{Tz}{z-1} = \frac{T}{1-z^{-1}}. \tag{14.18}$$

Цифровий інтегратор, що інтегрує ***методом трапецій (Trapeziodal)***, який іноді називають ***методом середніх прямокутників***, синтезують згідно з рис. 14.7в:

$$y_{tr}(kT+T) = y_{tr}(kT) + T\frac{u(kT)+u(kT+T)}{2}, \tag{14.19}$$

$$zy_{tr}(z) = y_{tr}(z) + T\frac{u(z)+zu(z)}{2}, \tag{14.20}$$

$$W_{tr}(z) = \frac{y_{tr}(z)}{u(z)} = \frac{T}{2}\cdot\frac{z+1}{z-1} = \frac{T}{2}\cdot\frac{1+z^{-1}}{1-z^{-1}}. \tag{14.21}$$

З порівняння передавальних функцій (14.15), (14.18) та (14.21) випливає:

$$y_{BE}(z) = zy_{FE}(z), \qquad y_{tr}(z) = \frac{y_{BE}(z)+zy_{FE}(z)}{2}, \tag{14.22}$$

тобто вихідний сигнал інтегратора (14.18) випереджає вихідний сигнал інтегратора (14.15) на один такт квантування, а вихідний сигнал інтегратора (14.22) складається із полусуми вихідних сигналів інтеграторів (14.15) і (14.18).

Взаємозв'язок між розглянутими методами чисельного інтегрування можна продемонструвати за допомогою деталізованої структурної схеми рис. 14.8, складеної за операторними рівняннями (14.36), (14.39) та (14.42).

Як видно з рис. 14.8, основу цифрових інтеграторів складає ланка запізнювання на період дискретності $1/z$, замкнена додатним зворотним зв'язком, що здійснює накопичення сум площ прямокутників $T\cdot u(kT)$ на кожному кроці квантування за часом. Один від одного інтегратори з різними алгоритмами відрізняються точкою знімання вихідного сигналу.

Варто звернути увагу на те, що інтегратори (14.18) та (14.21) мають прямий зв'язок входу з виходом. Тому при замиканні їхнього виходу на вхід утвориться алгебраїчний контур. Така ситуація може виникнути при цифровій реалізації задатчика інтенсивності.

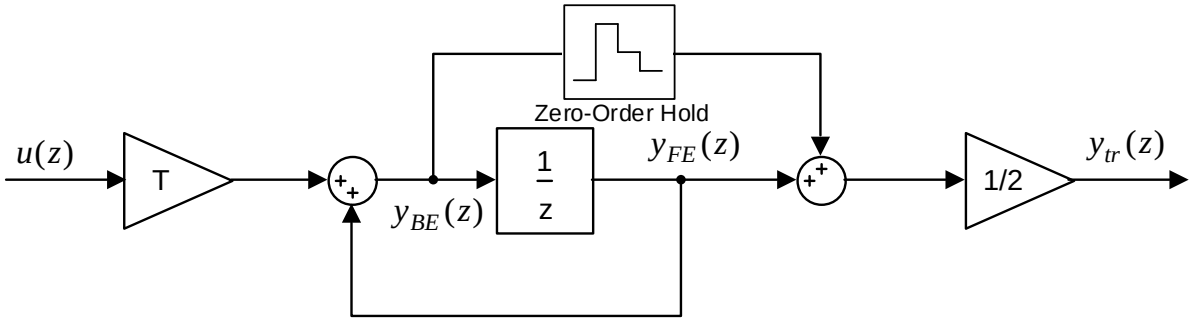


Рис. 14.8. Розгорнута модель дискретного інтегратора
з різними алгоритмами чисельного інтегрування

Цікаво порівняти розглянуті вище цифрові інтегратори з інтеграторами, отриманими методами Z-перетворення:

$$W_{imp}(z) = T \cdot Z\left\{\frac{1}{p}\right\} = \frac{Tz}{z-1} = W_{BE}(z),$$

$$W_{zoh}(z) = \frac{z-1}{z} \cdot Z\left\{\frac{1}{p^2}\right\} = \frac{z-1}{z} \cdot \frac{Tz}{(z-1)^2} = \frac{T}{z-1} = W_{FE}(z),$$

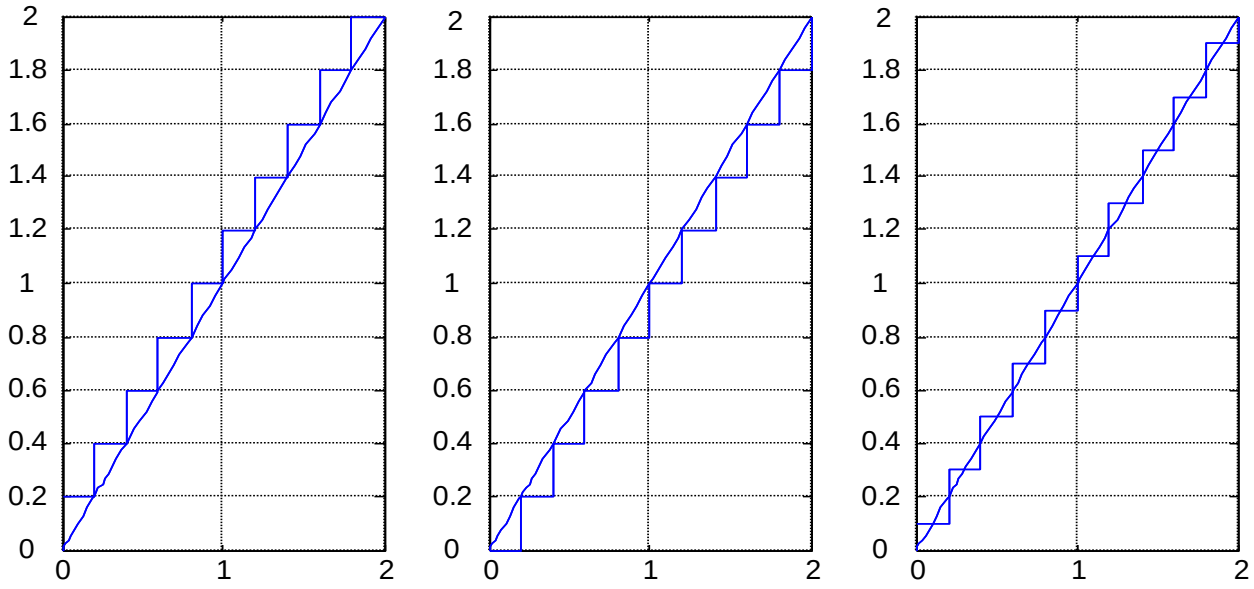
$$W_{foh}(z) = \frac{(z-1)^2}{Tz} \cdot Z\left\{\frac{1}{p^3}\right\} = \frac{(z-1)^2}{Tz} \cdot \frac{T^2 z(z+1)}{2(z-1)^3} = \frac{T}{2} \cdot \frac{z+1}{z-1} = W_{tr}(z).$$

Отже, при дискретизації неперервного інтегратора методами Z-перетворень можна отримати ті ж самі цифрові інтегратори першого порядку, синтезовані за допомогою методів чисельного інтегрування, а саме: імпульсно-інваріантне Z-перетворення аналогового інтегратора збігається з методом *Backward Euler*, ступінчато-інваріантне – з методом *Forward Euler* і лінійно-інваріантне – з методом *Trapezoidal*.

Застосування різних методів ЧІ можна продемонструвати за допомогою перехідних функцій, приведених на рис. 14.9. На останньому рисунку для порівняння приведена також перехідна функція неперервного інтегратора.

Блоки ***Discrete Filter* (*Дискретний Фільтр*)** і ***Discrete-Time Fcn* (*Дискретна Передавальна Функція*)** являють собою дискретні динамічні ланки загального виду з передавальною функцією поліноміального типу з тією ли-

ше різницею, що поліноми блока *Discrete-Time Fcn* мають незалежною змінною оператор z (див. формулу (14.9)), а поліноми блока *Discrete Filter* – z^{-1} (див. формулу (14.10)). Способи введення векторів коефіцієнтів степеневих багаточленів чисельника (*Numerator*) і знаменника (*Denominator)* і їхня інтерпретація аналогічні такими для блока *Transfer Fcn*.



а) *б)* *в)*

Рис. 14.9. Перехідні функції дискретних інтеграторів
з різними методами чисельного інтегрування:
а) *Backward Euler, б*) *Forward Euler, в*) *Trapeziodal*

Блок *Discrete-Time Fcn* базується на різновидності математичного опису дискретних ланок, що використовується у поширенні *Control System Toolbox*, а блок *Discrete Filter* – на опису, що використовується у поширенні *Signal Processing Toolbox*.

Блок ***Discrete Zero-Pole* (*дискретні нулі-полюси*)**являє собою дискретну динамічну ланку загального виду з передавальною функцією (14.11), де K – коефіцієнт підсилення (*Gain*); β_i – нулі (*Zeros*); α_i – полюса (*Poles*).

Вектори нулів і полюсів можуть уводитися тими ж трьома способами, що і вектори степеневих поліномів блока *Zero-Pole-Gain*.

Блок ***Discrete State-Spase* (*дискретний простір станів*)** забезпечує моделювання дискретної динамічної ланки загального вигляду за її описом у

просторі станів (див. рівняння (14.1)-(14.6)). Вхідними параметрами блока є матриці *A, B, C, D,* вектор початкових умов (*Initial conditions*) і період переривання (*Sample time*).

Ланка ***Zero-Order Hold* (*екстраполятор нульового порядку*)** вимірює вхідний сигнал у дискретні моменти часу і утримує значення цього сигналу на виході протягом періоду дискретності:

$$W(s) = \frac{e^{-T_S s} - 1}{e^{-T_S s} s} = \frac{z - 1}{zs}. \qquad (14.23)$$

Ланка ***First-Order Hold* (*екстраполятор першого порядку*)** вимірює вхідний сигнал у дискретні моменти часу і формує лінійну функцію, яка поєднує між собою дві сусідні точки вимірювання.

Simulink-модель перетворення неперервної синусоїди у дискретну з екстраполяцією нульового та першого порядків подана на рис. 14.10, а результат перетворення – на рис. 14.11. Слід відзначити, що адекватна робота блока можлива тільки при моделюванні з фіксованим кроком, що дорівнює періоду дискретності.

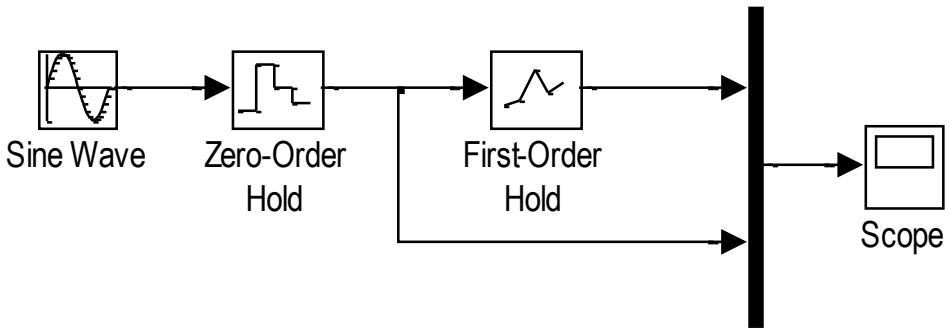


Рис. 14.10. *Simulink*-модель перетворення неперервної синусоїди екстраполяторами нульового та першого порядків

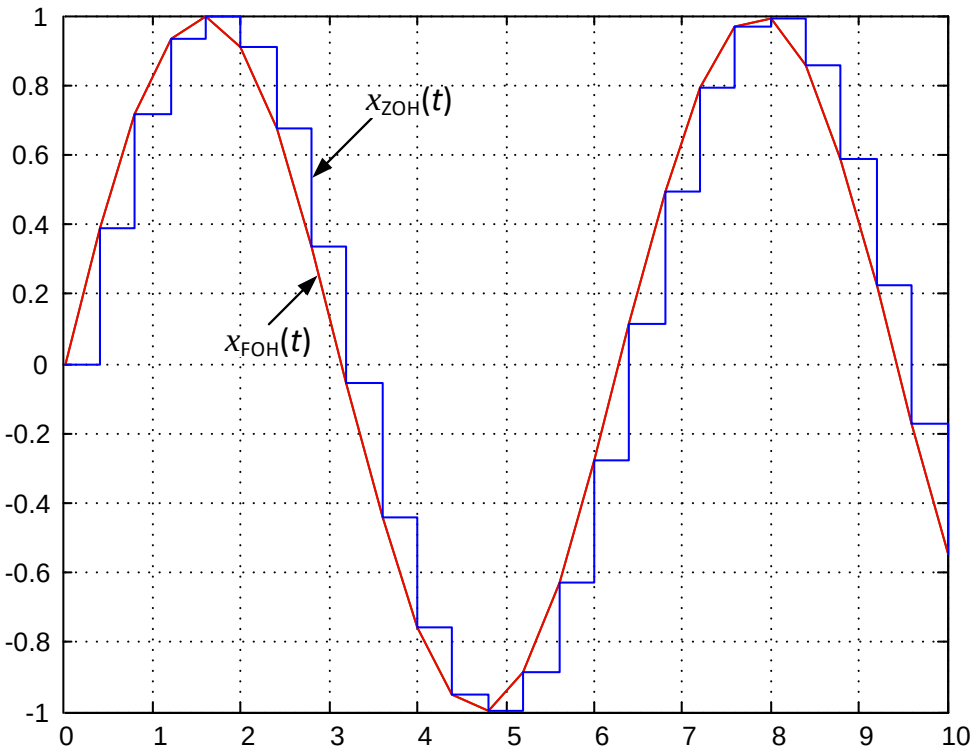


Рис. 14.11. Результат перетворення неперервної синусоїди екстраполяторами нульового та першого порядків

Контрольні питання та завдання

1. Поясніть різницю між різними методами чисельного інтегрування.
2. Поясніть, як можна знайти дискретну передавальну функцію деякого об’єкта за його різницевим рівнянням.
3. Які недоліки має бібліотечний блок *Discrete-Time Integrator*?
4. Чим відрізняються один від одного блоки *Discrete Transfer Function* і *Discrete Filter*?
5. Як треба узгоджувати між собою дискретні блоки, що мають різні періоди дискретності?
6. Які методи ЧІ можна використовувати при моделювання цифро-аналогових та цифрових об’єктів?
7. Дослідити роботу блока *Discrete-Time Integrator* з різними методами чисельного інтегрування при відпрацьовуванні ступінчатого, лінійного та синусоїдального сигналів.

15. ДИСКРЕТИЗАЦІЯ НЕПЕРЕРВНИХ ДИНАМІЧНИХ ОБ'ЄКТІВ

Як правило, в більшості електромеханічних систем автоматичного керування об'єкт керування має неперервний (аналоговий) характер. Однак, внаслідок відомих переваг цифрових коригувальних пристроїв (цифрових регуляторів) перед аналоговими, для керування неперервними об'єктами досить часто застосовуються саме такі регулятори. Тому в наступних двох главах розглянемо деякі питання, пов'язані з проектуванням цифрових регуляторів для неперервних об'єктів керування.

Синтез дискретних пристроїв керування неперервними системами виконують одним з наступних шляхів [10, 14, 15]:

- на основі неперервного об'єкта регулювання синтезують неперервні пристрої керування, а потім перетворюють їх у дискретну форму (метод неперервних моделей);
- будують дискретні моделі об'єкта регулювання і на їх основі синтезують дискретні пристрої керування.

Отже, в обох випадках треба вміти знаходити дискретні апроксимації аналогових передавальних функцій.

15.1. Постановка задачі. Класифікація методів

Отже, припустимо, що ми маємо неперервну динамічну систему з вхідним сигналом $u(t)$ і вихідним сигналом $y(t)$, що описується в області змінної Лапласа p передавальною функцією у поліноміальній формі (***Transfer Function Polynomial***):

$$W(p)=\frac{y(p)}{u(p)}=\frac{H_m(p)}{G_n(p)}=\frac{\beta_m p^m+\beta_{m-1}p^{m-1}+\ldots+\beta_1 p+\beta_0}{p^n+\alpha_{n-1}p^{n-1}+\ldots+\alpha_1 p+\alpha_0},\quad m\le n \qquad (15.1)$$

В цій ПФ характеристичний поліном (***denominator***) $G_n(s)$ є нормованим за коефіцієнтом при старшій степені оператора Лапласа.

Використовуючи розкладення поліномів у чисельнику (***numerator***) та знаменнику передавальної функції (15.1) на множники, отримуємо передавальну функцію у формі ***Zero-Pole-Cain*** :

$$W(p)=\frac{y(p)}{u(p)}=K\frac{(p-z_1)(p-z_2)\ldots(p-z_m)}{(p-p_1)(p-p_2)\ldots(p-p_n)}, \tag{15.2}$$

де

$\mathbf{Z}=\begin{bmatrix}z_1 & z_2 & \ldots & z_m\end{bmatrix}$ – вектор нулів (***Zeros***);

$\mathbf{P}=\begin{bmatrix}p_1 & p_2 & \ldots & p_n\end{bmatrix}$ – вектор полюсів (***Poles***);

$K=\beta_m$.

У просторі стану (***State Space***) такий об'єкт описується матричними рівняннями

$$p\mathbf{X}(p)=\mathbf{A}\mathbf{X}(p)+\mathbf{B}u(p), \tag{15.3}$$

$$y(p)=\mathbf{C}\mathbf{X}(p)+Du(p), \tag{15.4}$$

де

$\mathbf{X}=[x_1,\ x_2,\ ...,\ x_n]^T$ – вектор змінних стану;

A – матриця стану розміром $n\times n$;

B – вектор-стовпець входу розміром $n\times 1$;

C – вектор-рядок виходу розміром $1\times n$;

D – коефіцієнт обходу.

Задача полягає у визначенні при заданому періоді квантування T еквівалентної дискретної передавальної функції

$$W(z)=\frac{y(z)}{u(z)}=K_d\frac{(z-z_{d1})(z-z_{d2})\ldots(z-z_{dm_d})}{(z-p_{d1})(z-p_{d2})\ldots(z-p_{dn})}=$$

$$=\frac{H_{m_d}(z)}{G_n(z)}=\frac{\beta^*_{m_d}z^{m_d}+\beta^*_{m_d-1}z^{m_d-1}+\ldots+\beta^*_1z+\beta^*_0}{z^n+\alpha^*_{n-1}z^{n-1}+\ldots+\alpha^*_1z+\alpha^*_0} \tag{15.5}$$

або еквівалентних рівнянь у просторі станів

$$z\mathbf{X}(z) = \mathbf{A}_d \mathbf{X}(z) + \mathbf{B}_d u[z], \tag{15.6}$$

$$y(z) = \mathbf{C}_d \mathbf{X}(z) + D_d u(z), \tag{15.7}$$

тобто у побудові дискретної моделі неперервного об'єкта. Під еквівалентністю в даному випадку розуміють збіг реакцій неперервної системи та її дискретної моделі на будь-яку вхідну дію. Найчастіше під збіжністю реакцій розуміють, що $y[k] = y(t_k)$ при $u[k] = u(t_k)$, де $t_k = kT$, k – номер кроку квантування.

Зв'язок між рівняннями у просторі стану та передавальними функціями визначається виразами:

$$W(s) = \frac{y(s)}{u(s)} = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + D = \mathbf{C}\frac{Adj(s\mathbf{I} - \mathbf{A})}{det(s\mathbf{I} - \mathbf{A})}\mathbf{B} + D, \tag{15.8}$$

$$W(z) = \frac{y(z)}{u(z)} = \mathbf{C}_d (z\mathbf{I} - \mathbf{A}_d)^{-1}\mathbf{B}_d + D_d = \mathbf{C}_d \frac{Adj(z\mathbf{I} - \mathbf{A}_d)}{det(z\mathbf{I} - \mathbf{A}_d)}\mathbf{B}_d + D_d \tag{15.9}$$

де

I – одинична діагональна матриця розміром $n \times n$;

Adj(**X**) – союзна матриця (матриця, складена із алгебраїчних доповнень матриці-аргументу);

det(**X**) – визначник матриці.

Поставлену задачу можна розв'язати такими способами:

1) за допомогою *Z*-перетворень;
2) заміною оператора аналогового інтегрування $1/s$ одним з операторів цифрового інтегрування;
3) заміною нулів та полюсів на *s*-площині відповідними нулями та полюсами на *Z*-площині.

В загальному випадку поставлена задача не має точного розв'язку. Це пов'язано з тим, що при дискретизації вхідного сигналу втрачається інформація про його значення між вузлами квантування. Отже, вихід дискретної

моделі від цих значень залежати не може, у той час як реакція неперервної системи залежить від усіх значень вхідного сигналу.

Але все ж таки існують ситуації, у яких дискретна модель може бути точною у розумінні, викладеному вище. Для цього необхідно, щоб значення процесу $u(t)$ на інтервалі дискретності $(k-1)T \le t \le kT$) однозначно визначались послідовністю $[u(0), u(T), ..., u((k-1)T)]$. Таке має місце для імпульсних систем з амплітудно-імпульсною модуляцією першого роду та для цифрових систем керування, якщо вхідний процес, формується за допомогою ЕОМ. В останньому випадку дискретний вхідний процес перетворюється у неперервний за допомогою екстраполятора. Для таких випадків «точний» розв'язок можливий при застосуванні методів *Z*-перетворення

Отже, способи дискретизації, що використовують *Z*-перетворення, можна назвати ***точними*** в тому сенсі, що вони забезпечують збіг неперервного та дискретного вихідних сигналів у моменти часу, кратні періоду переривання, при певному виді вхідного сигналу. У зв'язку з цим розрізняють ***імпульсно-інваріантне***, ***ступінчасто-інваріантне*** та ***лінійно-інваріантне*** *Z*-перетворення.

Методи дискретизації другої групи називають ***підстановочними***. Вони за своєю сутністю є ***приблизними (неточними)***, тому що вони пов'язані з заміною неперервних інтеграторів у деталізованих структурах цифровими інтеграторами, які синтезуються за допомогою апріорі приблизних методів чисельного інтегрування.

До приблизних методів відноситься і метод відповідності нулів-полюсів. Це пов'язано з тим, що передавальні функції, отримані методами *Z*-перетворень, мають у своєму складі не тільки «системні нулі», пов'язані однозначними залежностями з нулями аналогового об'єкта, але й «нулі дискретизації», які можна визначити тільки приблизно.

Ще одним приблизним методом дискретизації, який застосовують доволі рідко із-за його невисокої точності, є перехід від диференційних рівнянь до різницевих, заснований на приблизних виразах похідних через скінченні різниці відповідних порядків.

15.2. Дискретизація методами *Z*-перетворень

З теорії автоматичного керування [10] відомі формули, за якими треба розраховувати ДПФ лінійної неперервної системи з відомою аналоговою ПФ, щоб система, отримана шляхом дискретизації, була імпульсно-інваріантною

$$W_{imp}(z) = T \cdot Z\{W(p)\}. \tag{15.10}$$

ступінчасто-інваріантною

$$W_{zoh}(z) = \frac{z-1}{z} \cdot Z\left\{\frac{W(p)}{p}\right\} = (1-z^{-1}) \cdot Z\left\{\frac{W(p)}{p}\right\}. \tag{15.11}$$

або лінійно-інваріантною

$$W_{foh}(z) = \frac{(z-1)^2}{Tz} \cdot Z\left\{\frac{W(p)}{p^2}\right\}. \tag{15.12}$$

При ступінчасто-інваріантній дискретизації використовується Z-перетворення з екстраполяцією нульового порядку (*zoh – Zero-Order Hold*), а при лінійно-інваріантній дискретизації – перетворення з екстраполяцією першого порядку (*foh – First-Order Hold*).

Для того, щоб дати рекомендацію, який із методів Z-перетворення доцільно використовувати в різних ситуаціях, порівняємо реакцію досліджуваних дискретних моделей одного з неперервних об’єктів на такі вхідні сигнали:

- ***δ-функцію Дірака*** (ідеальний імпульс одиничної площини), реакцію на яку називають ***імпульсною або ваговою функцією***;

- ***одиничну ступінчату функцію Хевісайда***, реакція на яку зветься ***перехідною функцією***:
- ***лінійну функцію з обмеженням*** її на одиничному рівні.

Розглянемо неперервний об'єкт з передавальною функцією

$$W(p)=\frac{4p(2p+1)}{24p^3+10p^2+6p+1}=0{,}33\ \frac{p(p+0{,}5)}{(p+0{,}20)(p^2+0{,}21p+0{,}21)}. \quad (15.13)$$

з трьома полюсами та двома нулями:

$$\mathbf{P}=[-0.1075\pm 0.4417i,\ -0.2016],\qquad \mathbf{Z}=[0,-0.5].$$

Дискретні передавальні функції (ДПФ), отримані з ПФ (15.12) методом розглянутих *Z*-перетворень при $T=2$, мають вигляд:

$$W_{imp}(z)=\frac{0{,}67z^3-0{,}66z^2+0{,}06z}{z^3-1{,}69z^2+1{,}33z-0{,}43}=0.67\cdot\frac{z(z-0{,}89)(z-0{,}11)}{(z-0{,}67)(z^2-1{,}02+0{,}65)}; \quad (15.14)$$

$$W_{zoh}(z)=\frac{0{,}61z^2-0{,}82z+0{,}21}{z^3-1{,}69z^2+1{,}33z-0{,}43}=0.61\cdot\frac{(z-1)(z-0{,}34)}{(z-0{,}67)(z^2-1{,}02z+0{,}65)}; \quad (15.15)$$

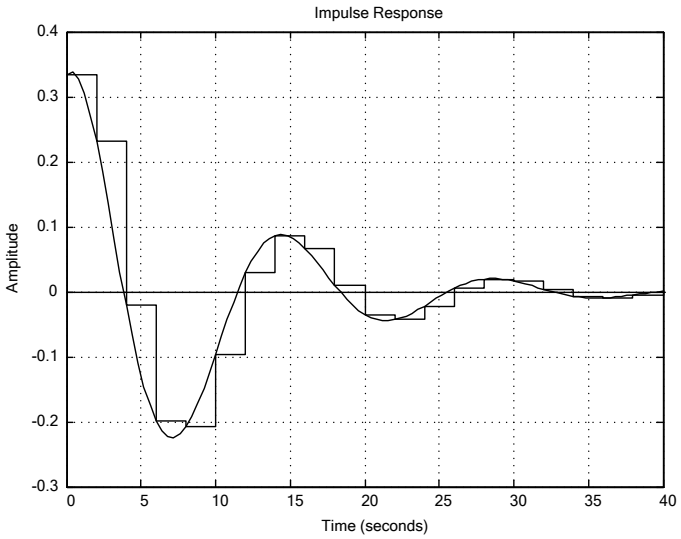
$$W_{foh}(z)=\frac{0{,}32z^3\text{-}0{,}11z^2-0{,}34z+0{,}12}{z^3-1{,}69z^2+1{,}33z-0{,}43}=0.32\frac{(z-1)(z-0{,}37)(z+1{,}03)}{(z-0{,}67)(z^2-1{,}02z+0{,}65)}. \quad (15.16)$$

Період квантування навмисне обрано великим, щоб наочніше спостерігати поведінку дискретних сигналів.

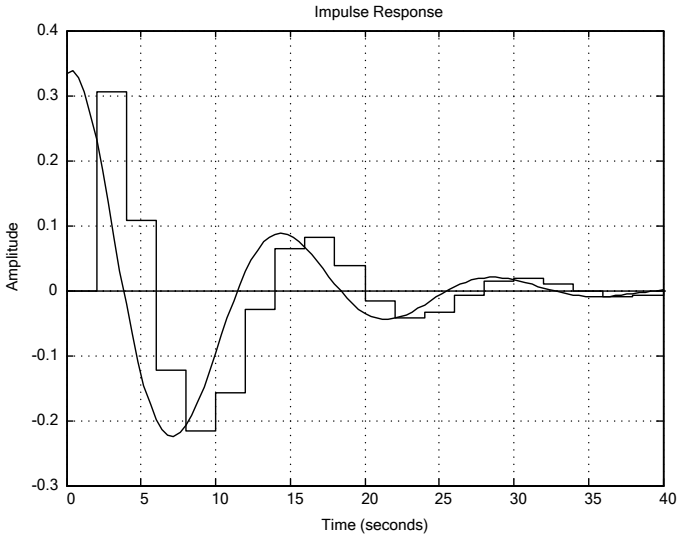
Вагові функції вихідної неперервної системи з передавальною функцією (15.12) $w(t)$ та дискретних систем з передавальними функціями (15.14) $w_{imp}(t)$, (15.15) $w_{zoh}(t)$ і (15.16) $w_{foh}(t)$ зображені на рис. 15.1.

На рис. 15.2 наведені перехідні функції тих же об'єктів, на рис. 15.3 – їх реакція на сигнал, що лінійно зростає від 0 до 1, а потім фіксується на досягнутому рівні.

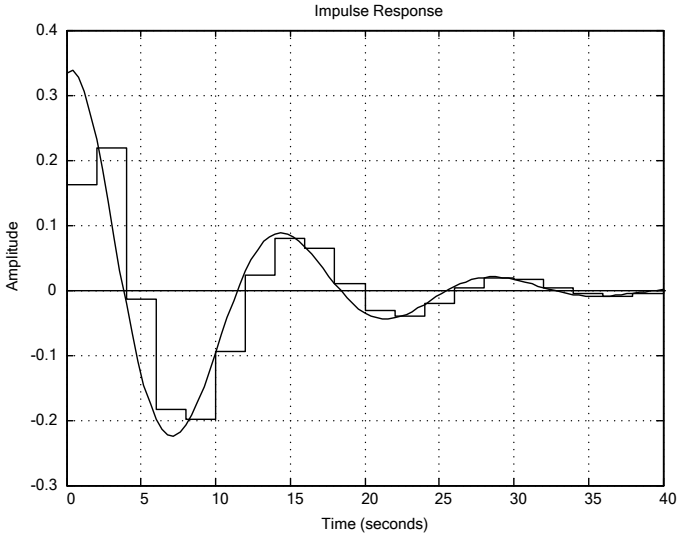
З аналізу наведених графіків видно, що значення вагової функції цифрової моделі, синтезованої за допомогою імпульсно-інваріантного Z-перетворення, як і очікувалось, на кожному інтервалі дискретності збігається зі значеннями вагової функції неперервного об'єкта на початку інтервалу.


Impulse Response
Amplitude
Time (seconds)
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
0
5
10
15
20
25
30
35
40

а)

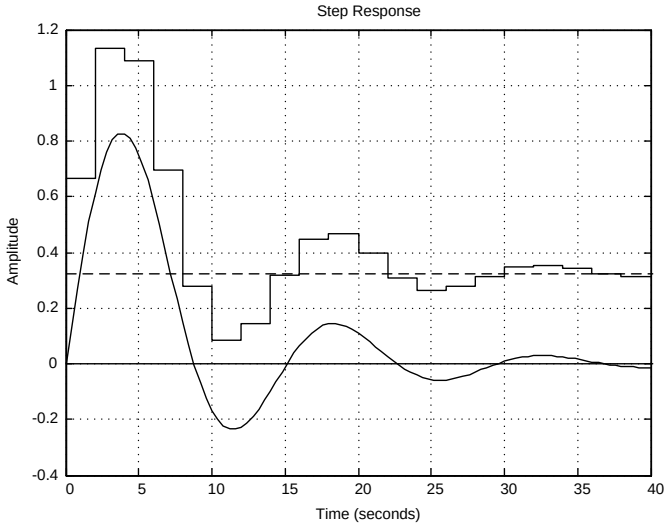

Impulse Response
Amplitude
Time (seconds)
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
0
5
10
15
20
25
30
35
40

б)

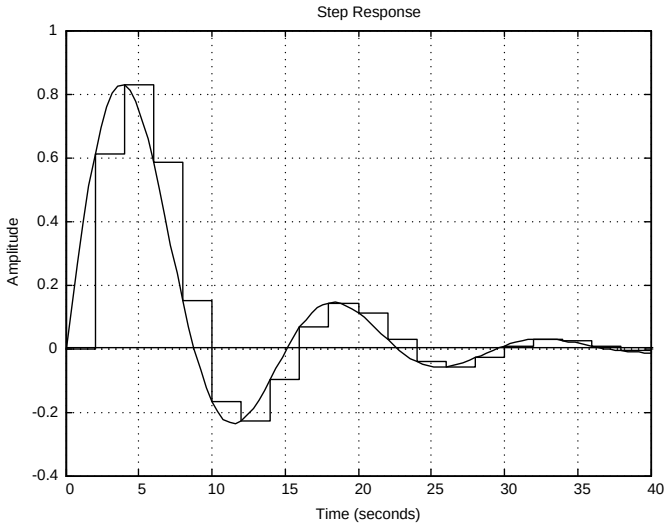

Impulse Response
Amplitude
Time (seconds)
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
0
5
10
15
20
25
30
35
40

в)

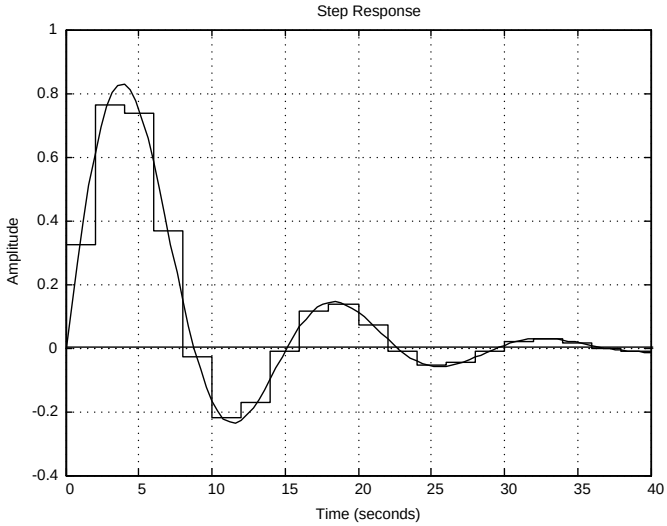
Рис. 15.1. Вагові функції неперервного об’єкта та його цифрові моделі



а)

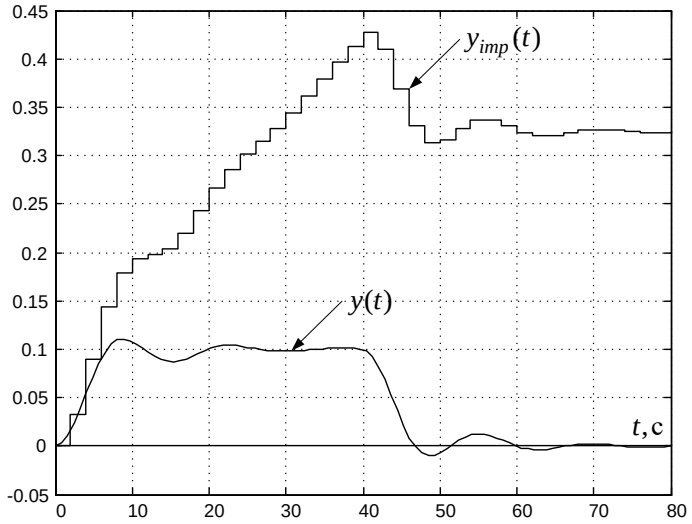


б)

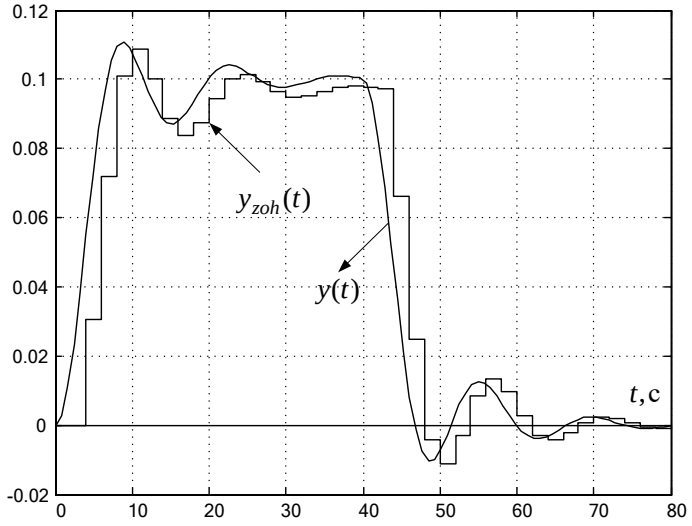


в)

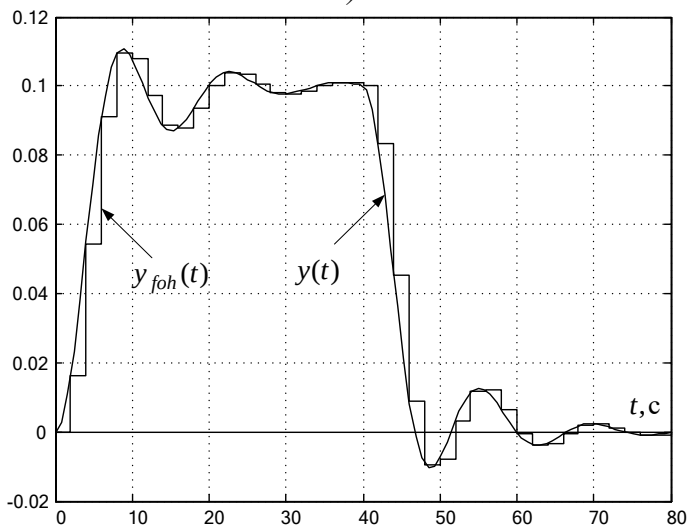
Рис. 15.2. Перехідні функції неперервного об’єкта
та його цифрових моделей



а)



б)



в)

Рис. 15.3. Реакції неперервного об’єкта
та його цифрових моделей на лінійний вхідний сигнал з обмеженням

Але перехідна функція цієї моделі суттєво відрізняється від перехідної функції дискретизованої неперервної ланки в будь-які моменти часу, включаючи початкове та усталене значення.

Відповідно до цього реакція її на лінійний вхідний сигнал має похибку, що збільшується з часом. Отже такий вид дискретної апроксимації можна використовувати тільки тоді, коли є впевненість, що сигнал на вході завжди матиме форму, що наближається до ідеальних імпульсів.

Звичайно, відхилення реакцій дискретної та неперервної ланок на однаковий вхідний сигнал зменшується при зменшенні періоду квантування. Це стосується усіх досліджень, виконаних у даному розділі.

При використанні Z-перетворення з екстраполяцією нульового порядку найкращий результат досягається при ступінчатій зміні вхідного сигналу. Реакція такої дискретної моделі на інші види вхідних сигналів характеризується запізненням на один період квантування.

При використанні Z-перетворення з екстраполяцією першого порядку найкращий результат досягається при лінійній зміні вхідного сигналу, але цілком задовільні результати отримуємо і при інших видах вхідних сигналів: реакція на імпульс спотворюється тільки на двох перших кроках квантування, а реакція на стрибок збігається з перехідною функцією неперервного об’єкта приблизно в середніх точках періодів квантування.

Аналогічні дослідження, виконані для синусоїдального вхідного сигналу, також підтверджують переваги і універсальність дискретної апроксимації неперервних об’єктів методом Z-перетворення з екстраполяцією першого порядку. Порівнюючи передавальні функції дискретних моделей видно, що усі вони мають однакові знаменники та різні чисельники.

Для аналізу особливостей отриманих дискретних апроксимацій (15.14)-(15.16) неперервного об’єкта (15.13) визначимо дискретні полюси та

дискретні системні нулі при обраному періоді квантування, що відповідають нулям та полюсам неперервного об'єкта:

$$\mathbf{P}_d = \exp(T\mathbf{P}) = [0{,}51 \pm 0{,}62i \quad 0{,}67], \tag{15.17}$$

$$\mathbf{Z}_d = \exp(T\mathbf{Z}) = [1 \;\; 0{,}37]. \tag{15.18}$$

Можна показати, що усі передавальні функції, отримані методами *Z*-перетворення, мають дискретні полюси, пов'язані з відповідними неперервними полюсами співвідношеннями (15.17), тобто усі вони мають однакові характеристичні поліноми:

$$G_{imp}(z) = G_{zoh}(z) = G_{foh}(z) = \prod_{i=1}^{n}(z - p_{di}). \tag{15.19}$$

Їхній порядок збігається з порядком неперервного об'єкта:

$$n_d = n. \tag{15.20}$$

Інакше складаються справи з нулями дискретних моделей. В більшості книжок з теорії автоматичного керування не наводиться ніякої інформації щодо їхнього визначення. Із приведених в даній роботі посилань це питання розглянуто тільки в [10], де відзначається, що дискретні передавальні функції з екстраполяцією *FOH* завжди мають порядок чисельника, що дорівнює порядку знаменника, незалежно від порядку чисельника перетворюваного неперервного об'єкта:

$$m_{foh} = n = n_d, \tag{15.21}$$

а дискретні моделі з екстраполяцією *ZOH* мають порядок чисельника на одиницю менше, за винятком того випадку, коли в неперервному об'єкті $m = n$, тобто

$$m_{zoh} = \begin{cases} n_d & \text{при} \quad m = n, \\ n_d - 1 & \text{при} \quad m \le n - 1. \end{cases} \tag{15.22}$$

Із розглянутого прикладу видно, що співвідношення (15.21) характерно і для імпульсно-інваріантної апроксимації:

$$m_{imp} = n = n_d. \tag{15.23}$$

Перевищення кількості нулів у неперервному об’єкті та його дискретних апроксимаціях зумовлена тим, що нулі цифрових моделей неперервних ланок складаються із ***системних нулів*** та ***нулів квантування*** [10]. Різниця між ними полягає в тому, що при $T \to 0$ перші наближаються до дискретного перетворення нулів неперервного об’єкта

$$\lim_{T\to 0} z_{dsi} = \exp(Tz_i), \quad i = 1,2,...,m, \tag{15.24}$$

а другі – до –1:

$$\lim_{T\to 0} z_{dkj} = -1, \quad j = m+1,\ m+2,\ ...,\ m_d\,. \tag{15.25}$$

Нулі квантування в загальному випадку можуть бути навіть немінімально-фазовими, тобто мати амплітуду, більшу за 1.

Аналіз нулів розглянутих вище дискретних моделей показує таке:

- ступінчато- та лінійно-інваріантні цифрові моделі мають близькі один до одного системні нулі $\mathbf{Z}_{ds\,zoh} = [1\ \ 0{,}34]$, $\mathbf{Z}_{ds\,foh} = [1\ \ 0{,}37]$, причому системні нулі лінійно-інваріантної моделі майже не відрізняються від нулів, отриманих за формулою (15.18);
- лінійно-інваріантна цифрова модель має у своїй передавальній функції нуль квантування $\mathbf{Z}_{dk\,foh} = -1.0368$, близький до -1, який підвищує порядок чисельника та забезпечує форсування перехідних процесів;
- при використанні імпульсно-інваріантної дискретизації системні нулі $\mathbf{Z}_{ds\,imp} = [0.8863,\ 0.1096]$ істотно відрізняються від нулів (15.18), а нуль квантування має нульове значення: $\mathbf{Z}_{dk\,foh} = 0$, що зумовлює суттєву відмінність цієї моделі від двох інших, включаючи різницю в усталених значеннях перехідних функцій.

15.3. Підстановочні методи дискретизації

У попередньому підрозділі ми розглянули, як виконується перетворення неперервних об’єктів у цифрові з умови інваріантності у часовій області.

Але існують і інші методи, засновані на взаємних перетвореннях між p та z, що є тотожним заміні оператора аналогового інтегрування $1/p$ одним з операторів цифрового інтегрування. З передавальних функцій цифрових інтеграторів, розглянутих у розділі 16, випливають такі формули підстановок:

$$p = \frac{z-1}{T}, \tag{15.26}$$

$$p = \frac{z-1}{Tz}, \tag{15.27}$$

$$p = \frac{2}{T} \cdot \frac{z-1}{z+1}. \tag{15.28}$$

Використання підстановки (15.26) часто називають ***методом Ейлера***, підстановки (15.27) – ***модифікованим методом Ейлера***, а підстановки (15.28) – ***білінійним перетворенням*** або ***методом Тастіна*** (*Tustin*).

Вже з ідеї, закладеної в ці методи, випливає, що при дискретній апроксимації інтегратора, підстановка (15.26) збігається з *Z*-перетворенням із використанням екстраполятора нульового порядку (*ZOH*), а підстановка (15.28) – з *Z*-перетворенням із використанням екстраполятора першого порядку (*FOH).* На жаль при ускладненні структури неперервного об'єкта ця збіжність та навіть прагнення до неї практично втрачаються.

Слід одразу зауважити, що перетворення (15.26) не гарантує стійкість цифрового об'єкта при стійкому неперервному. Наприклад, для неперервної ланки $W(p) = 1/(p+2)$ відповідний (за даним методом) цифровий об'єкт має дискретну передавальну функцію

$$W_{FE}(z) = \frac{1}{\frac{z-1}{T} + 2} = \frac{T}{z - 1 + 2T}.$$

Як бачимо, він буде нестійким при $T > 0.5$. Таке явище спостерігається при досить великих періодах квантування [10].

Для демонстрації підстановочних методів дискретизації на рис. 15.4 показані перехідні функції неперервної інтегрувальної ланки $W(p) = 1/p$

(верхній рядок), неперервної аперіодичної ланки $W(p) = 1/(4p+1)$ (нижній рядок) та їхніх дискретних апроксимацій при $T = 2$ [14, 15].

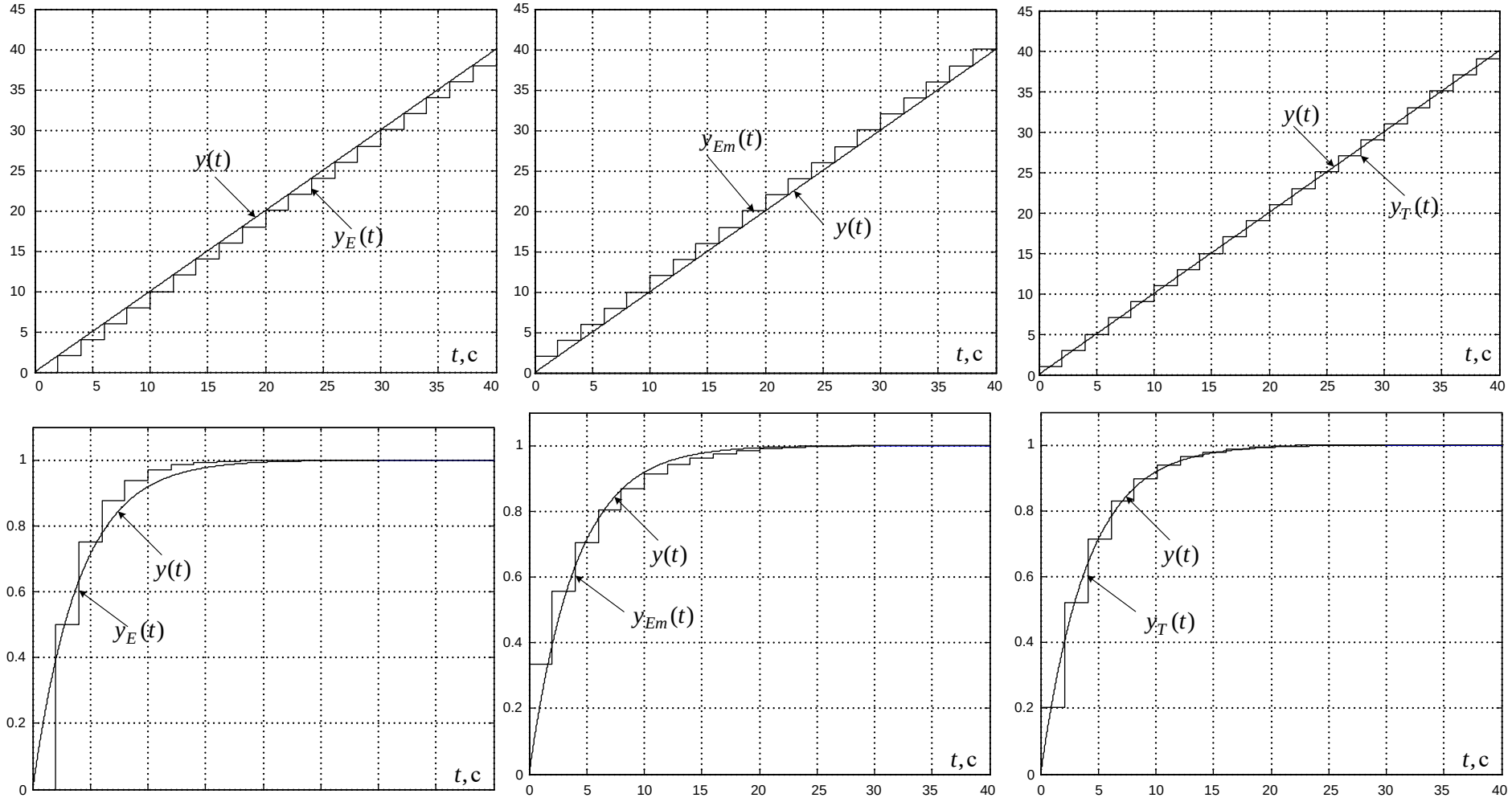


Рис. 15.4. Перехідні функції неперервного інтегратора, неперервної аперіодичної ланки та їхніх дискретних аналогів, визначених методом Ейлера (*а*), модифікованим методом Ейлера (*б*) та методом Тастіна (*в*)

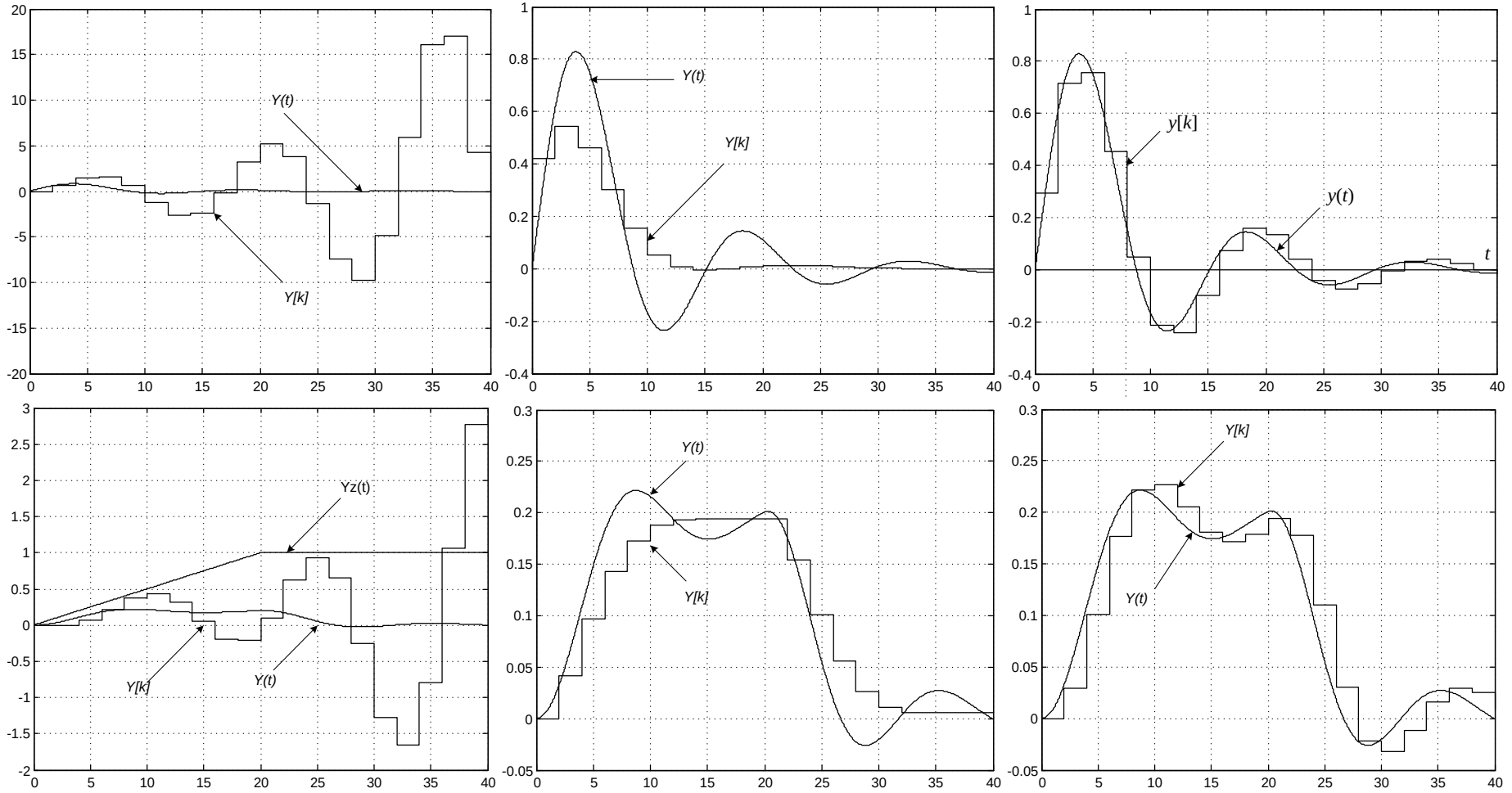


Рис. 15.5. Реакції на стрибкоподібний та лінійний впливи неперервної системи з ПФ (15.21) та її дискретних аналогів за методом Ейлера (*а*), модифікованим методом Ейлера (*б*) та методом Тастіна (*в*)

При дискретизації аперіодичної ланки отримують такі цифрові об'єкти:

$$W_{FE}(z)=\frac{0,5}{z-0,5},\quad W_{BE}(z)=\frac{0,33z}{z-0,67},\quad W_{tr}(z)=\frac{0,2(z+1)}{z-0,6}.$$

Порівняння їх з дискретними передавальними функціями цієї ж ланки, отриманими методами Z-перетворень

$$W_{zoh}(z)=\frac{0,39}{z-0,61},\quad W_{foh}(z)=\frac{0,21(z+0,85)}{z-0,61}$$

показує, що перетворення Ейлера прагне наблизитися до ступінчато-інваріантного Z-перетворення, а перетворення Тастіна – до лінійно-інваріантного Z-перетворення.

На рис. 15.5 приведені графіки перехідних процесів, що демонструють можливості підстановочних методів дискретизації неперервної системи 3-го порядку з ПФ (15.13).Відповідні дискретні передавальні функції мають вигляд:

$$W_E(z)=\frac{0,67z^2-0,67z}{z^3-2,17z^2+2,33z-0,83}=\frac{0,67z(z-1)}{(z-0,60)(z^2-1,57z+1,40)};\qquad (15.29)$$

$$W_{Em}(z)=\frac{0,42z^3\text{-}0,63z^2-0,21z}{z^3-1,79z^2+1,21z-0,32}=\frac{0,42z(z-1)(z-0,5)}{(z-0,71)(z^2-1,08z+0,44)};\qquad (15.30)$$

$$W_T(z)=\frac{0,29z^3\text{-}0,10z^2-0,29z+0,10}{z^3-1,78z^2+1,44z-0,46}=\frac{0,29(z-1)(z-0,33)(z+1)}{(z-0,66)(z^2-1,12z+0,70)}.\qquad (15.31)$$

З аналізу графіків та передавальних функцій можна прийти до таких висновків:

1) нулі та полюси усіх підстановочних перетворень відрізняються від нулів та полюсів Z-перетворень;

2) тільки перетворення Тастіна доповнює передавальну функцію приблизними значеннями нулів дискретизації $z_d=-1$; обидва з перетворень Ейлера замість нуля дискретизації додають в дискретну передавальну функцію нуль: $z_d=0,33$);

3) перетворення Ейлера утворює дискретну систему, яка при великих періодах квантування може стати нестійкою, незважаючи на стійкість відповідного неперервного об’єкта; отже, перетворення Ейлера можна використовувати тільки для неперервних систем, які описуються диференційними рівняннями невисокого порядку, та при малих періодах переривання (у порівнянні зі власними сталими часу об’єкта);

4) модифіковане перетворення Ейлера характеризується значно меншою точністю дискретизації, ніж перетворення Тастіна.

Зі сформульованих вище положень випливає, що найкращим із перестановочних перетворень є перетворення Тастіна, яке за своїми параметрами і властивостями наближується до *Z*-перетворення з екстраполяцією першого порядку. Такий висновок збігається із загальноприйнятою методикою приблизної дискретної апроксимацією неперервних об’єктів., що забезпечує компроміс між простотою і точністю дискретизації [10].

15.4. Метод відповідності нулів та полюсів

Цей метод відображає неперервні полюси p_i та нулі z_i у дискретні, виходячи із взаємозв’язку між оператором Лапласа та дискретним оператором:

$$p_{di} = e^{Tp_i}, \qquad z_{di} = e^{Tz_i}. \tag{15.32}$$

Покажемо, що застосування співвідношень (15.32) не достатньо для якісної дискретизації у розумінні подібності динамічних та статичних властивостей неперервного об’єкта та його цифрової моделі.

Припустимо, що неперервний об’єкт має 3 дійсних полюси: $\mathbf{P} = [\alpha_1\ \alpha_2\ \alpha_3]$ і жодного нуля:

$$W(p) = \frac{\beta_0}{(p-\alpha_1)(p-\alpha_2)(p-\alpha_3)}. \tag{15.33}$$

Тоді його дискретна модель, синтезована з використанням формул (15.32) матиме вигляд

$$W_{zp}(z)=\frac{\beta_0}{(z-e^{\alpha_1 T})(z-e^{\alpha_2 T})(z-e^{\alpha_3 T})} \tag{15.34}$$

Припустимо, що вихідний сигнал $y(t)$ неперервного об'єкта (15.13) є реакцією на одиничний ступінчатий вплив. Для вихідного сигналу справедливі такі початкові умови:

$$y(0)=0,\quad y'(0)=0,\quad y''(0)=0.$$

Нехай $y(kT)$ – реакція цифрової моделі (15.37) на одиничну ступінчату послідовність. Оскільки $W_{zp}(z)$ має кількість полюсів, що на 3 перевищує кількість нулів, сигнал на виході цифрового об'єкта з'явиться лише на третьому такті.

Узагальнюючи, сформулюємо такий висновок: ***якщо кількість полюсів дискретної ланки перевищує кількість нулів на*** r***, то реакція цього об'єкта на вхідний сигнал почнеться з*** r***-го такту квантування***.

До того ж передавальні функції (15.33) та (15.34) забезпечують різні усталені значення перехідних функцій.

Для ліквідації цих недоліків передавальну функцію дискретної моделі треба доповнити декількома нулями, та скоригувати коефіцієнт її підсилення.

Для адекватної дискретної апроксимації досліджуваним методом скористаємося досвідом аналізу методів перетворення, викладених у попередніх підрозділах, та відомими положеннями теорії автоматичного керування [10].

З викладених міркувань випливає така методика дискретизації неперервного динамічного об'єкта методом відповідності нулів та полюсів [14, 15]:

1) знаходимо полюси p_i $(i=1,2,...,n)$ неперервної системи розв'язанням характеристичного рівняння

$$G_n(p)=p^n+\alpha_{n-1}p^{n-1}+...+\alpha_1 p+\alpha_0=0 \tag{15.35}$$

або через пошук власних чисел матриці стану

$$\mathbf{P}=\mathrm{eig}(p\mathbf{I}-\mathbf{A}); \tag{15.36}$$

2) знаходимо нулі z_i $(i=1,2,...,m)$ неперервної системи розв'язанням рівняння

$$H_m(s)=\beta_m p^m+\beta_{m-1}p^{m-1}+...+\beta_1 p+\beta_0=0 \qquad (15.37)$$

3) розраховуємо відповідні дискретні полюси та дискретні системні нулі згідно з формулою (15.41), тобто

$$p_{di}=\exp(Tp_i), \quad i=1, 2, ..., n, \qquad (15.38)$$

$$z_{dsi}=\exp(Tz_i), \quad i=1, 2, ..., m; \qquad (15.39)$$

4) доповнюємо дискретну передавальну функцію нулями квантування

$$z_{dkj}=-1 \qquad (15.40)$$

так, щоб скоригований у такий спосіб порядок чисельника m_d був на одиницю менше порядку знаменника:

$$m_d=n-1, \quad j=m+1, m+2, ..., n-1 \qquad (15.41)$$

або, рівним із ним:

$$m_d=n, \quad j=m+1, m+2, ..., n-1, n; \qquad (15.45)$$

5) виділяємо із векторів аналогових нулів та полюсів нейтральні (тобто ті, що для неперервної системи дорівнюють 0, а для дискретної – 1) та підраховуємо їх кількість (μ та ν відповідно):

6) розраховуємо коефіцієнт передачі в усталеному режимі неперервної ланки за формулою

$$k=K\prod_{i=1}^{n_1}(-p_{1i})\Big/\prod_{j=1}^{m_1}(-z_{1j}), \qquad (15.46)$$

де $K=\beta_m$, $m_1=m-\mu$, $n_1=n-\nu$ – кількість ненульових аналогових нулів та полюсів відповідно;

7) розраховуємо коефіцієнт K_d дискретного об'єкта за формулою

$$K_d=kT^{\nu-\mu}\cdot\prod_{i=1}^{n_{d1}}(1-p_{d1i})\Big/\prod_{j=1}^{m_{d1}}(1-z_{d1j}), \qquad (15.47)$$

де p_{d1i}, z_{d1j}, m_{d1}, $n_{d1}=n_1$ – неодиничні дискретні нулі та полюси та їх кількість відповідно.

Передавальні функції, отримані при дискретизації неперервного об'єкта (15.21) за викладеною вище методикою мають вигляд: при $m_d = n-1$

$$W_{zp0}(z) = \frac{0{,}66z^2 - 0{,}90z + 0{,}24}{z^3 - 1{,}69z^2 + 1{,}33z - 0{,}43} = \frac{0{,}66(z-1)(z-0{,}37)}{(z-0{,}67)(z^2 - 1{,}02z + 0{,}65)} \qquad (15.48)$$

та при $m_d = n$

$$W_{zp1}(z) = \frac{0{,}33z^3\text{-}0{,}12z^2 - 0{,}33z + 0{,}12}{z^3 - 1{,}69z^2 + 1{,}33z - 0{,}43} = \frac{0{,}33(z-1)(z-0{,}37)(z+1)}{(z-0{,}67)(z^2 - 1{,}02z + 0{,}65)}. \qquad (15.49)$$

Вони виявляються дуже схожими на передавальні функції, отримані методами Z-перетворення з екстраполяцією нульового та першого порядків, тобто $W_{zp0}(z) \approx W_{zoh}(z)$ та $W_{zp1}(z) \approx W_{foh}(z)$.

Для демонстрації результатів двох запропонованих вище методів узгодження нулів та полюсів неперервної і дискретної систем на рис. 15.6 приведені перехідні функції досліджуваної неперервної системи та її дискретних апроксимацій, знайдених за приведеною вище методикою.

На рис. 15.7 показані реакції цих же ланок на лінійний вхідний сигнал з обмеженням.

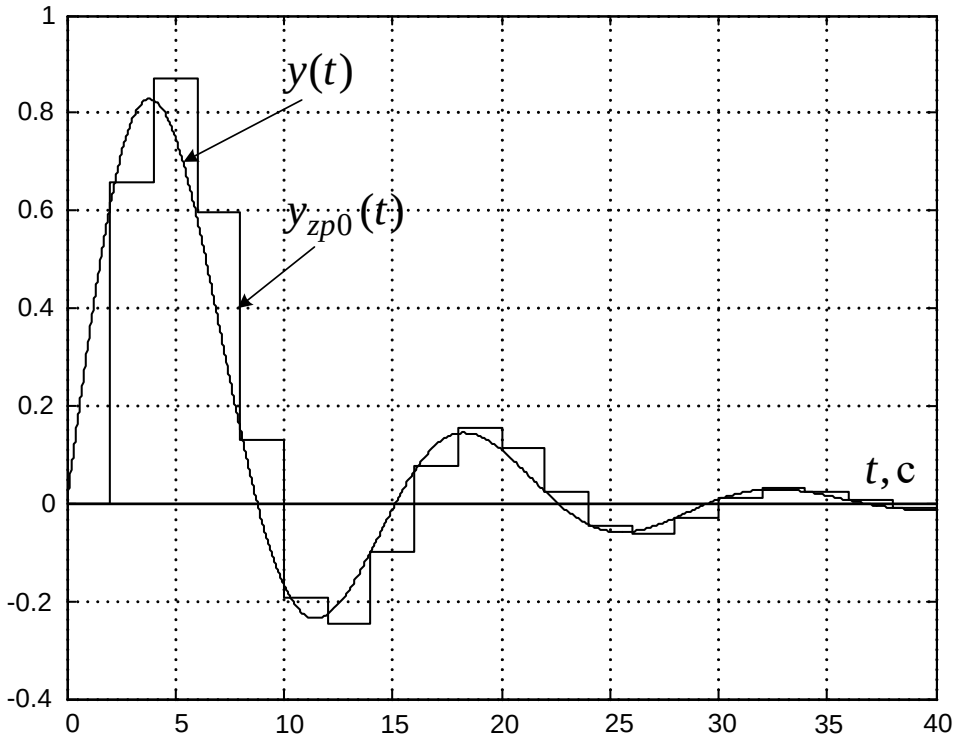
Графіки підтверджують, що навіть при такому завищеному періоді квантування отримані дискретні перехідні функції наближаються до перехідних функцій, зображених на рис. 15.2б і 15.2в.

Особливо треба підкреслити той факт, що дискретна апроксимація методом узгодження нулів полюсів при для розглянутого прикладу має значно менше відхилення від апроксимації методу Z-перетворення з екстраполяцією першого порядку, ніж при використанні методу Тастіна, який найчастіше рекомендується для застосуванні у якості приблизного методу дискретизації.

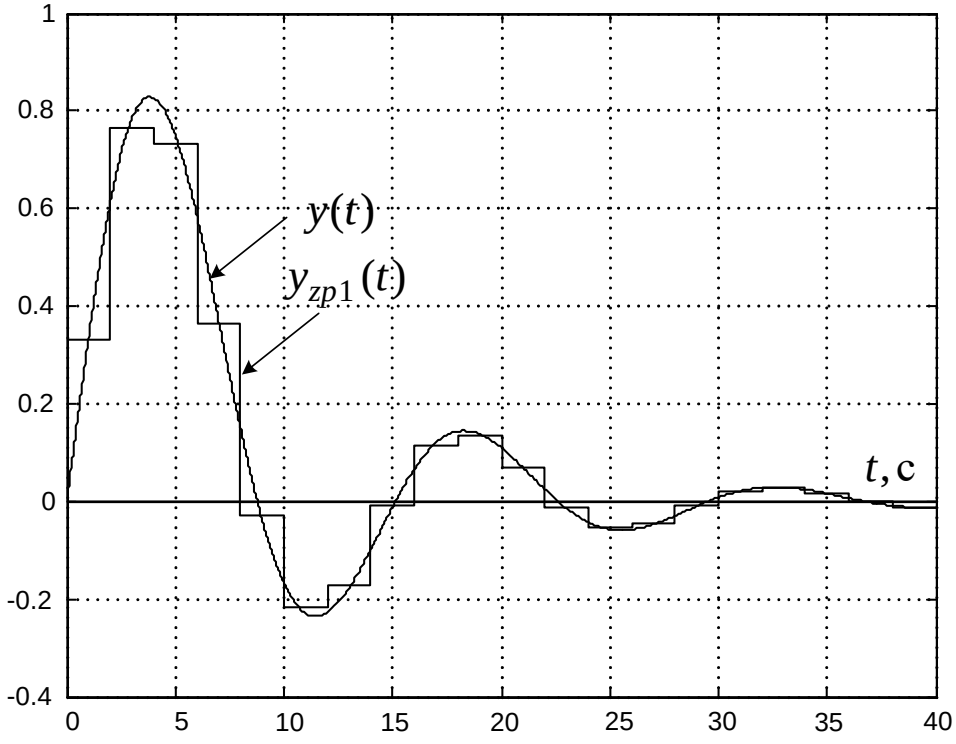
Це підтверджується графіками відповідних відхилень, зображених на рис. 15.8.

Одержані результати дають підстави для перегляду загально прийнятої рекомендації щодо приблизної дискретної апроксимації неперервних систем.

До переваг методу узгодження належить також те, що дискретні системи, отримані у такий спосіб не треба перевіряти на стійкість.

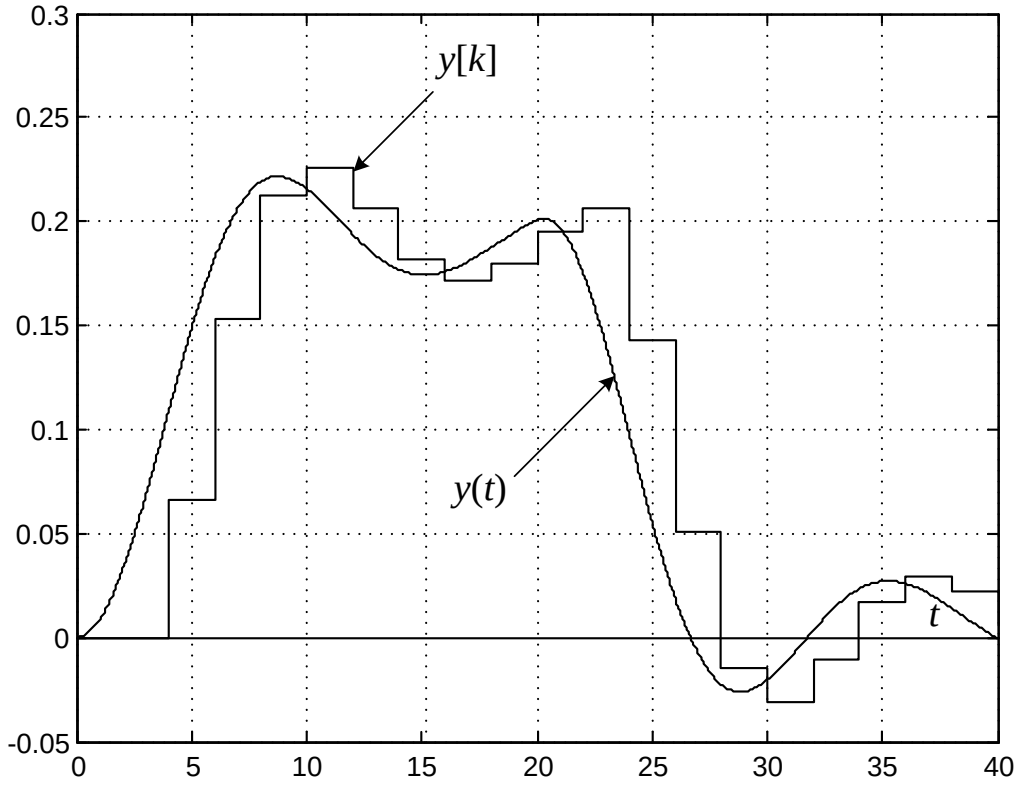


а)

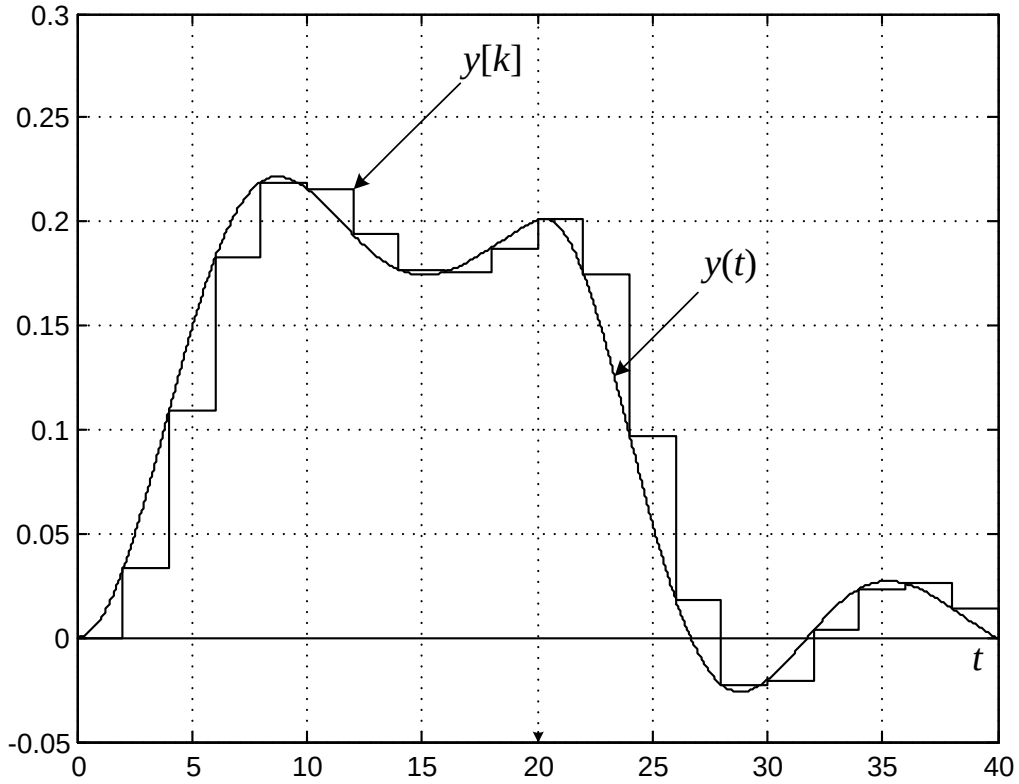


б)

Рис. 15.6. Перехідні функції неперервної системи з ПФ (15.21) та її дискретних аналогів, визначених методом узгодження нулів полюсів при $m_d = n-1$ (*а*) та $m_d = n$ (*б*)



а)



б)

Рис. 15.7. Реакція неперервної системи з ПФ (7.13) та її дискретних аналогів, визначених методом узгодження нулів полюсів при $m_d = n-1$ (*а*) та $m_d = n$ (*б*) на лінійний вхідний сигнал з обмеженням

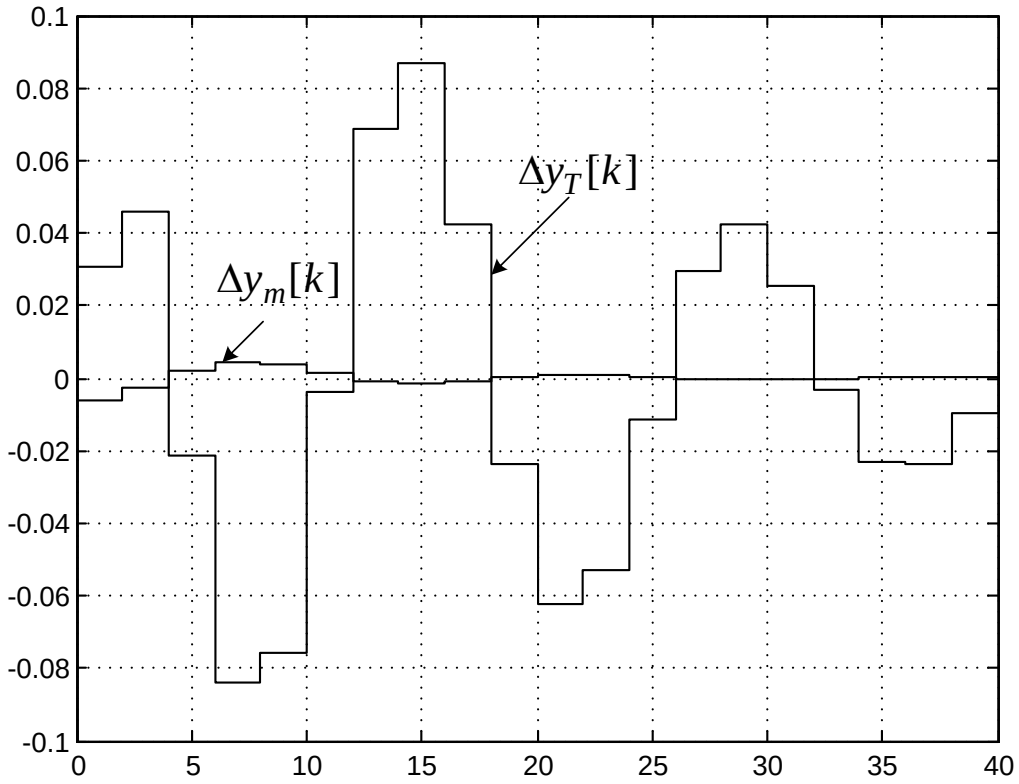


Рис. 15.8. Відхилення перехідних функцій дискретних систем, утворених методами Тастіна та узгодження нулів-полюсів від перехідної функції системи, утвореної методом *Z*-перетворення з *FOH*

15.5. Дискретизація неперервних об'єктів в середовищі пакета MATLAB

Для утворення дискретних моделей неперервних систем в MATLAB передбачена функція c2d з таким синтаксисом:

```
SysD = c2d (SysC, T)
SysD = c2d (SysC, T, 'метод')
SysD = c2d(SysC, T, options)
```

Функція SysD=c2d(SysC,T) утворює дискретну модель SysD неперервної системи SysC з періодом дискретності T методом *Z*-перетворення з використанням екстраполятора нульового порядку.

Функція

```
SysD = c2d (SysC, T, options)
```

дозволяє задати додаткові параметри за допомогою функції c2doptions.

Функція SysD=c2d(SysC,T,'метод') використовує один з методів екстраполяції, що перераховані у табл. 15.1.

Таблиця 15.1

Метод	Опис методу
zoh	*Z*-перетворення з екстраполятором нульового порядку на вході (ступінчасто-інваріантна дискретизація)
foh	*Z*-перетворення з екстраполятором першого порядку на вході (лінійно-інваріантна дискретизація)
impulse	Імпульсно-інваріантна дискретизація
tustin	Білінійна апроксимація Тастіна (трапеційна дискретизація)
matched	Метод відповідності нулів та полюсів

Для програмної реалізації методики дискретизації неперервного об'єкта методом узгодження можна рекомендувати функцію, скорочений варіант якої (без діагностики вхідних параметрів) має вигляд [15, 16]:

```
function sysd=c2d_mat(sys,Ts,method)
   [Zero, Pole, Gain, ts] = zpkdata (sys);
   z = Zero{1};   p = Pole{1};
   zd = exp (z*Ts)
   pd = exp (p*Ts)
   if method == 'zoh',
   zd = [zd; -ones (length(pd)-length(zd)-1, 1)];
   else zd = [zd; -ones (length(pd)-length(zd),1)];
   end
   p1 = p; z1 = z;
   iz = find (p==0); p1(iz) = [];
   izz = find (z==0); z1(izz) = [];
   pd1 = pd; zd1 = zd; pd1(iz) = []; zd1(izz) = [];
   k = Gain(1)*real(prod(-z1)/prod(-p1))
   if isempty (izz), izz = 0; end
   if isempty (iz), iz = 0; end
   Kd = k*real(prod(1-pd1)/prod(1-zd1))*Ts^(iz-izz);
   sysd = zpk (zd, pd, Kd, Ts);
end
```

Контрольні питання та завдання

1. Назвіть 2 шляхи синтезу цифрових пристроїв керування неперервними об'єктами.
2. Сформулюйте задачу дискретної апроксимації неперервних динамічних об'єктів.

3. Перелічіть способи дискретної апроксимації неперервних динамічних систем.
4. Які способи перетворення неперервних об’єктів у дискретні називають точними? Перелічіть їх види. Чим вони відрізняються один від одного?
5. Напишіть формули імпульсно-інваріантного, ступінчасто-інваріантного та лінійно-інваріантного *Z*-перетворень. Сформулюйте рекомендації щодо їхнього застосування.
6. Які способи перетворення неперервних об’єктів у дискретні називають приблизними? Перелічіть їх види.
7. Перелічіть підстановочні методи дискретної апроксимації. Напишіть формули підстановок.
8. Як дискретизувати Simulink-модель неперервної динамічної ланки підстановочним методом без знаходження дискретної передавальної функції?
9. Яке підстановочне перетворення не гарантує стійкість цифрової ланки при перетворенні стійкого неперервного об’єкта.
10. Сформулюйте рекомендації щодо застосування підстановочних методів.
11. Як пов’язане запізнення реакції дискретної ланки на вхідний сигнал з кількістю нулів та полюсів ДПФ?
12. У чому полягає складність застосування методу відповідності нулів та полюсів при дискретизації неперервних об’єктів.
13. Що таке нулі дискретизації та системні нулі? Як визначити системні нулі та їх кількість?
14. Які програмні засоби пакету MATLAB, призначені для дискретної апроксимації неперервних ланок у числовому вигляді Ви знаєте? Наведіть приклади.

16. ІМІТАЦІЙНЕ ФІЗИЧНЕ МОДЕЛЮВАННЯ ЕЛЕКТРИЧНИХ КІЛ З ВИКОРИСТАННЯМ БЛОКІВ БІБЛІОТЕК *SimPowerSystems*

16.1. Основні відомості про бібліотеку *SimPowerSystems* та правила її застосування

Програма *Simulink* поряд з власними блоками, розглянутими у роботах 1-8 може використовувати блоки бібліотеки *SimPowerSystems (SPS)*, призначеної для імітаційного фізичного моделювання електротехнічних, електромагнітних, електронних та електромеханічних пристроїв та схем [7-9].

В останніх версіях *Simulink* розширення *SimPowerSystems* входить до складу додатка імітаційного фізичного моделювання *SimScape* поряд з фундаментальною бібліотекою процесів *Foundation Library (Electrical, Hydraulic, Magnetic, Mechanical, Pneumatic, Thermal)* та бібліотеками *SimElectronics*, *SimMechanics*, *SimHydraulics* та *SimDriveline*.

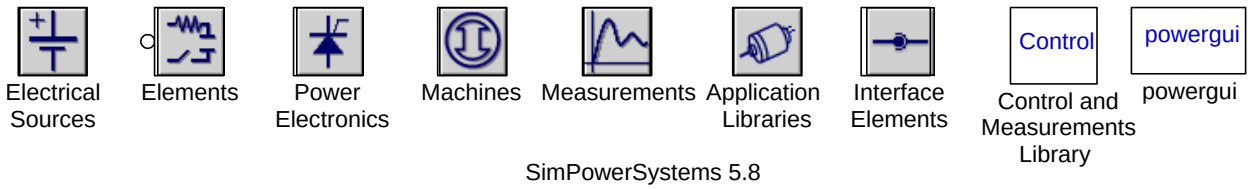
На відміну від *Simulink*-блоків, *SPS*-блоки подано у вигляді позначень відповідних елементів на принципових електричних схемах. З'єднуючись між собою, ці блоки утворюють електричні кола. Математичний опис окремих елементів приховано від користувача, завдяки чому створюється ілюзія імітаційного фізичного моделювання. Насправді ж кожному з блоків *SimPowerSystems* поставлено у відповідність неперервні та дискретні *Simulink*-моделі, які можна побачити після завантаження файлів *powerlib_models, powerlib_extras, powerlib_meascontrol*. Шлях до цих файлів у версії *R2013a* має вигляд: *ProgramFiles\MATLAB\R2013a\toolbox\physmod\powersys\powersys*. Починаючи з версії *R2014a, Simulink*-моделі блоків *SimPowerSystems* можна побачити зазирнувши під маску. Отже, не зважаючи на ілюзію фізичного моделювання, користувач повинен чітко уявляти, що *SPS*-блоки створені на основі математичного опису, складеного розробниками пакету із певними припущеннями, які необхідно знати для адекватного застосування *SPS*-моделей в процесі досліджень. Для цього треба користува-

тись довідковою інформацією за допомогою функції *help*. Якщо цієї інформації не вистачає, можна проаналізувати відповідні *Simulink*-моделі.

SPS-блоки мають такі особливості [7]:

- їх входи та виходи, на відміну від *Simulink*-блоків (*S*-блоків), не вказують напрямок передачі сигналу, бо вони фактично є еквівалентами електричних контактів;
- лінії зв'язку між *SPS*-блоками є моделями ідеальних (без опору) електричних проводів, по яким струм може протікати в обох напрямках;
- *SPS*- та *S*-блоки не можуть з'єднуватися один з іншим безпосередньо; сигнал від S-елементів можна передати до *SPS*-елементів тільки через керовані джерела енергії (*Controlled Voltage/Current Source*) *SPS*-бібліотеки *Electrical Sources*, а навпаки – через блоки бібліотек засобів вимірювання (*Measurements*);
- в моделі, яка отримує в собі *SPS*-блоки, має бути присутнім хоча б один з вимірювальних *SPS*-приборів, що пов'язано з особливостями перетворення *SPS*-моделі в еквівалентну розрахункову S-модель;
- в *SPS*-модель необхідно встановлювати блок *powergui*.

Зміст бібліотеки *SimPowerSystems* показано на рис. 16.1.



Р

Рис. 16.1 Бібліотека *SimPowerSystems*

Вона отримує такі розділи:

- *Electrical Sources* – джерела електричної енергії;
- *Elements* – пасивні електротехнічні елементи;
- *Power Electronics* – пристрої силової електроніки, керовані напівпровідникові ключі;
- *Machines* – моделі електричних машин;

- *Measurements* – вимірювальні прибори;
- *Application Libraries* – прикладні бібліотеки;
- *Interface Elements* – інтерфейсні елементи;
- *Control and Measurements Library* – бібліотека керування та вимірювання;
- *PowerGUI* – графічний інтерфейс користувача.

Перелік блоків *SimPowerSystems*, необхідний для моделювання та аналізу властивостей розгалужених лінійних електричних кіл постійного і змінного струмів, надано на рис 16.2. Вони знаходяться у різних розділах бібліотеки *SPS*.

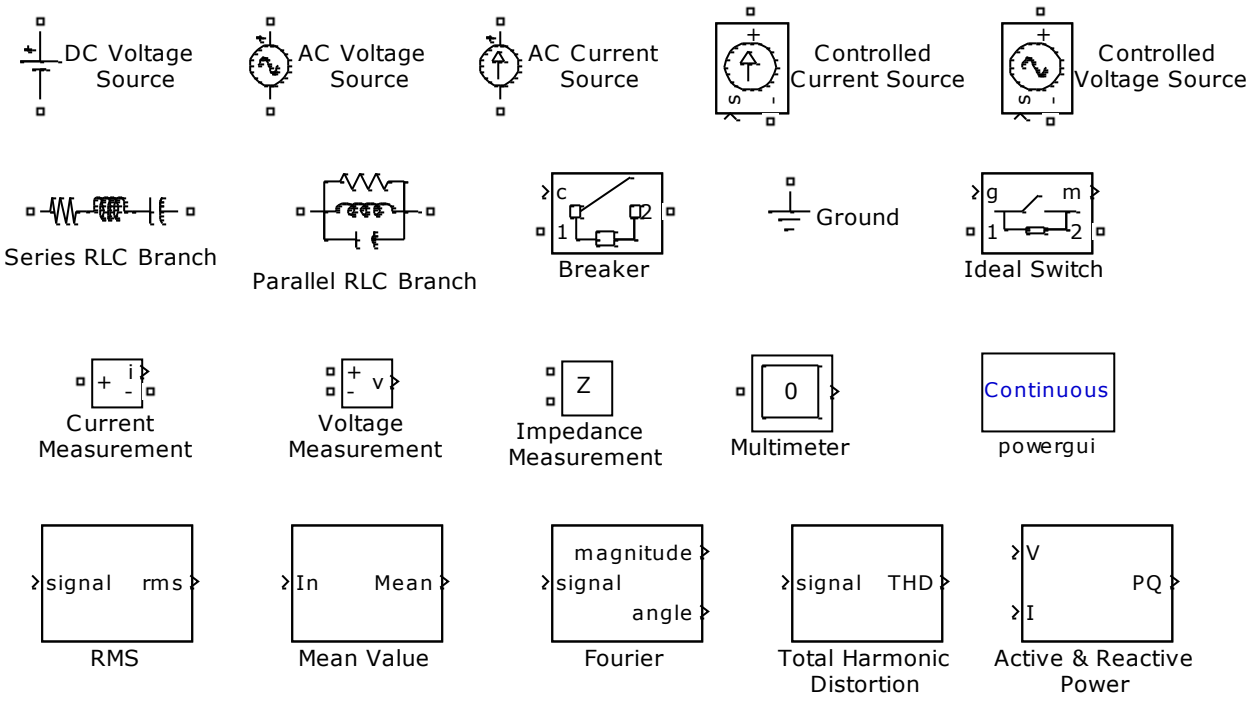


Рис. 16.2. *SPS*-блоки для імітаційного фізичного моделювання простих електричних кіл

Подані блоки мають порти двох типів: *SPS*-порти, позначені дрібними квадратиками та *S*-порти, позначені кінцівками стрілок, що допомагає запобігти помилок при з'єднанні блоків.

Параметри блоків можна розділити на ***основні***, які пов'язані безпосередньо з фізичною сутністю блоків і без встановлення яких робота блоків

принципово не можлива, і ***допоміжні***, які в багатьох випадках можна залишати такими, що встановлені за замовченням.

До допоміжних параметрів відносяться:

1. *Measurements* – здійснює вибір сигналів, які стають доступними для вимірювання блоком *Multimeter*. У меню, що випадає, можна обрати такі значення цього параметру:

- *None* – немає сигналів, доступних для вимірювання:
- *Voltage* – (для джерел напруги);
- *Current* – можна виміряти струм (для джерел струму);
- *Branch voltage* – можна виміряти напругу *RLC*-гілки або будь-якого сполучення резистора, котушки індуктивності та конденсатора (для блоків *RLC Branch* та *RLC Load*);
- *Branch current* – можна виміряти струм *RLC*-гілки;
- *Branch voltage and current* – можна виміряти струм і напругу *RLC*-гілки.

2. *Set the initial inductor current* у сполученні з параметром *Inductor initial current (A)*, дозволяють встановити початкове значення струму індуктивного елемента (для блоків *RLC Branch* та *RLC Load*).

3. *Set the initial capacitor voltage* у сполученні з параметром *Capacitor initial voltage (V)*, дозволяють встановити початкове значення напруги конденсатора (для блоків *RLC Branch* та *RLC Load*).

4. *Sample Time* – період дискретизації; встановлюється тільки при моделюванні дискретних процесів; при моделюванні неперервних процесів його залишають нульовим (0).

При створенні імітаційних фізичних моделей електричних кіл дотримуйтесь таких правил:

1. Не забувайте встановити блок *powergui*.

2. Враховуйте, що в *SPS*-моделях, на відміну від *S*-моделей, спочатку розраховується усталений режим.

3. Для вимірювання напруг та струмів треба, як мінімум, або підключити у схему амперметри та вольтметри, або скористатися мультиметром. До вихідних *S*-портів вимірювальних *SPS*-блоків приєднайте якісь із блоків *Simulink*-бібліотеки *Sinks*. Що здійснюють реєстрацію та/або візуалізацію результатів.

4. Розгалуження електричного кола не можна виконати з точок, що позначають *SPS*-порти та з точок між *SPS*-портами одного блока. Тому, якщо Ви хочете, наприклад, виміряти напруги на кожному з елементів послідовної *RLC*-гілки, то треба включити послідовно 3 блоки *RLC Branch*, один з яких зробити чисто активним, другий – чисто індуктивним, а третій – чисто ємнісним.

5. Для формування напруг та струмів складної форми використовуйте будь-які *S*-блоки, сформований сигнал подавайте на вхідні *S*-порти контрольованих *SPS*-джерел електричної енергії *Controlled Voltage*/*Current Source* з *SPS*-бібліотеки *Electrical Sources*.

6. При виборі методу та параметрів чисельного інтегрування диференційних рівнянь, що здійснюється при виконанні команди *Mogel Configurations Parameters* меню *Simulink*, ретельно аналізуйте можливий характер та тривалість перехідних процесів. При недостатній кількості розрахованих точок для якісної візуалізації переходьте від методу *ode45*(вкладка *Solver*), що пропонується за замовченням до методів *ode23s, ode15s, ode23tb* або при використанні методу *ode45* встановлюйте значення параметра *Refine Factor* (вкладка *Data Import/Export, Refine* – підвищувати якість) більшим за 1 (максимальне значення дорівнює 10), при використанні якого щільність точок збільшується в установлене число разів методом інтерполювання. У колах змінного струму для якісної візуалізації іноді достатньо встановити максимальний шаг інтегрування (*Max Step Size*) таким, щоб на періоді синусоїди розраховувалась достатня кількість точок, наприклад, $h_{\max} = 1/(50 \div 100) f$.

7. Для більшої наочності доцільно більшість блоків перейменувати у відповідності з досліджуваною електричною схемою, а імена вимірювальних блоків приховати (*Diagram→Format→Hide Block Name*), якщо їх призначення зрозуміло із піктограми блока.

16.2. Характеристика основних блоків для моделювання кіл постійного струму та однофазних кіл змінного струму

Розглянемо призначення та основні параметри блоків, наведених на рис. 16.2.

16.2.1. Джерела електричної енергії

В *SPS* присутні такі джерела:

1) ***DC Voltage Source*** – ідеальне джерело постійної напруги з основним параметром *Amplitude (V)* – величина постійної напруги U_c (В);

2) ***AC Voltage Source*** – ідеальне джерело синусоїдальної напруги (з нульовим активним опором) з параметрами

- *Peak Amplitude (A)* – амплітуда напруги U_m (А),
- *Phase (deg)* – початкова фаза φ° (град.),
- *Frequency (Hz)* – частота f (Гц), що формує напругу

$$U(t) = U_m \sin\left(2\pi f t + \frac{\varphi^\circ}{180^\circ}\pi\right); \qquad (16.1)$$

3) ***AC Current Source*** – – ідеальне джерело синусоїдального струму (з нульовою провідністю) з параметрами

- *Peak Amplitude (A)* – амплітуда струму I_m (А),
- *Phase (deg)* – початкова фаза φ° (град.),
- *Frequency (Hz)* – частота f (Гц), що формує струм

$$I(t) = I_m \sin\left(2\pi f t + \frac{\varphi^\circ}{180^\circ}\pi\right); \qquad (16.2)$$

4) ***Controlled Voltage Source***, ***Controlled Current Source*** – керовані джерела напруги і струму; використовуються для створення джерел, які вироб-

ляють струм або напругу, що відрізняються від постійної величини або від синусоїдальних сигналів (1.1), (1.2); для цього треба сформувати бажані сигнали в *Simulink* і подати їх на *S*-входи керованих джерел електроенергії, які перетворюють *S*-сигнали у *SPS*-напруги або *SPS*-струми, що живлять імітаційні електричні кола через *SPS*-контакти, позначені як «+» та «-».

При використанні керованих джерел за прямим призначенням немає сенсу виконувати ініціалізацію контрольованих джерел.

Оскільки джерело постійного струму у бібліотеках *SPS* відсутнє, то замість його можна використати блок *AC Voltage Source* з параметрами:

$$I_m = I_c, \qquad f = 0, \qquad \varphi^\circ = 90^\circ.$$

Окрім *S*-блоків, включених до складу базових *Simulink*-бібліотек, до *SPS*-бібліотеки також залучено деякі *S*-блоки, які можуть стати у пригоді при створенні та при аналізі імітаційних електротехнічних схем. Зокрема, у розділі *Control and Measurements Library* знаходиться група блоків *Pulse & Signal Generators*, які можна застосовувати при формуванні нестандартних джерел напруги. Частина з них показана на рис. 16.3.

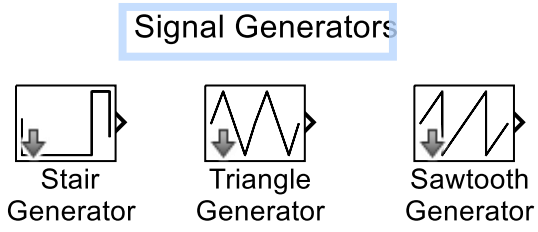


Рис. 16.3. Блоки із групи *Pulse & Signal Generators*

Блок *Stair Generator* (*генератор ступінчатих сигналів*) формує послідовність ступінчатих сигналів на основі блока *Look-Up Table*, як це показано на рис. 16.4.

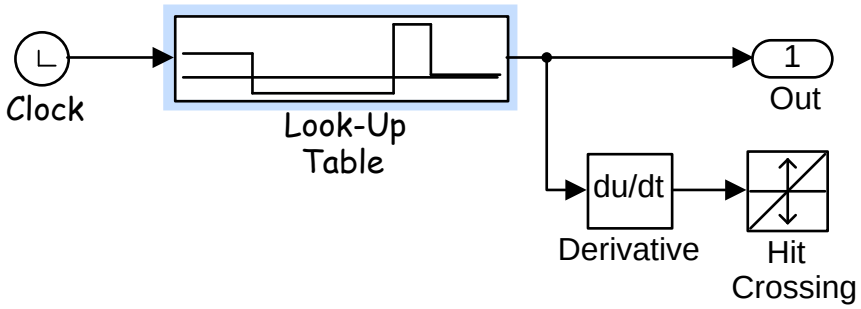


Рис. 16.4. Структура блока *Stair Generator*

Його параметрами є координати точок, в яких відбувається ступінчата зміна вихідного сигналу, а саме: вектор часу – *Time (s)* та вектор амплітуд – *Amplitude*. Його зручно використовувати для керування станом блока *Ideal Switsch*.

Блок *Triangle Generator* формує трикутні, а блок *Sawtooth Generator* пилоподібні періодичні сигнали одиничної амплітуди з заданою частотою *Frequency (Hz)* та фазою *Phase (degree)*. Структура блока *Sawtooth Generator* показано на рис. 16.5.

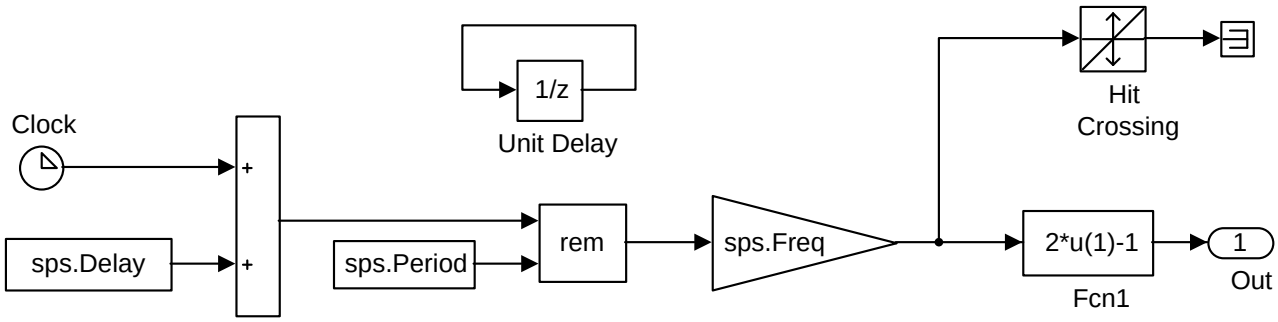


Рис. 16.5. Структура блока *Sawtooth Generator*

16.2.2. Електротехнічні елементи

1. ***Series RLC Branch*** – послідовне *RLC*-з'єднання з такими основними параметрами:

- *Branch type* – визначає комплектність з'єднання, обирається із випадного меню і може приймати одне із значень: *RLC, RL, RC, LC, R, L, C*;
- *Resistance R (Ohms)* – активний опір, Ом;
- *Inductance L (H)* – індуктивність, Гн;

- *Capacitance C (F)* – ємність, Ф.

2. ***Parallel RLC Branch*** – паралельне *RLC*-з'єднання з такими ж основними параметрами, як і блок *Series RLC Branch.*

Вікно введення параметрів блока *Series RLC Branch* відображено на рис. 16.6.

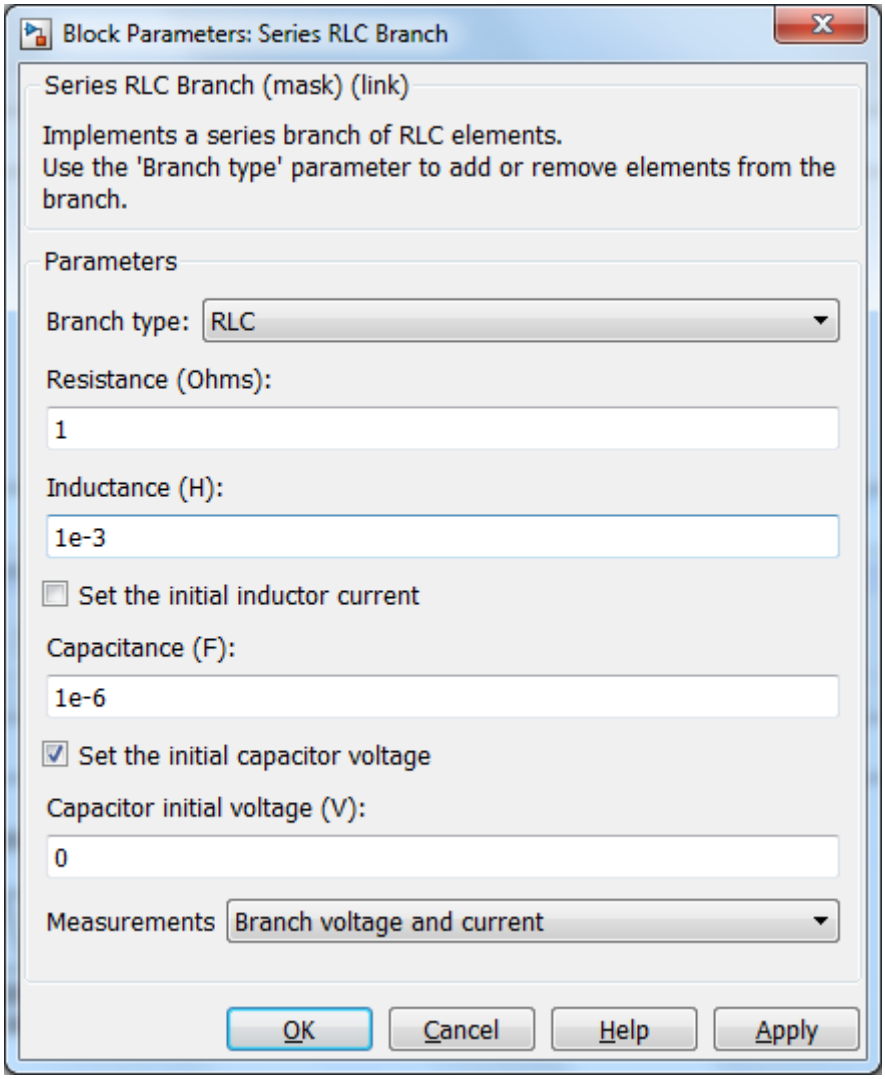


Рис. 16.6. Вікно параметрів блока *Series RLC Branch*

16.2.3. Основні вимірювальні прилади

1. ***Voltage Measurement*** – ідеальний вольтметр.

Перетворює різницю потенціалів точок, приєднаних до входів «+» і «–», в еквівалентний *S*-сигнал на виході «v», який можна перетворювати, фіксувати або візуалізувати будь-яким блоком *Simulink*-бібліотек.

2. ***Current Measurement*** – ідеальний амперметр.

Перетворює *SPS*-сигнал електричного струму, що протікає у гілці, в яку включено прибор через контакти «+» і «–», в еквівалентний *Simulink*-сигнал на *S*-виході «i», який можна перетворювати, фіксувати або візуалізувати будь-яким блоком бібліотек *Simulink*.

3. ***Multimeter*** – багатоканальний вимірювач струмів та напруг.

Сигнали, доступні для вимірювання цим прибором мають бути обрані у блоках елементів схеми за допомогою допоміжного параметра *Measurement*, як це описано у пункті 16.2.

Якщо такі сигнали існують, то після подвійного щиглика по піктограмі (іконці) блока відчиняється вікно, зображене на рис. 16.7.

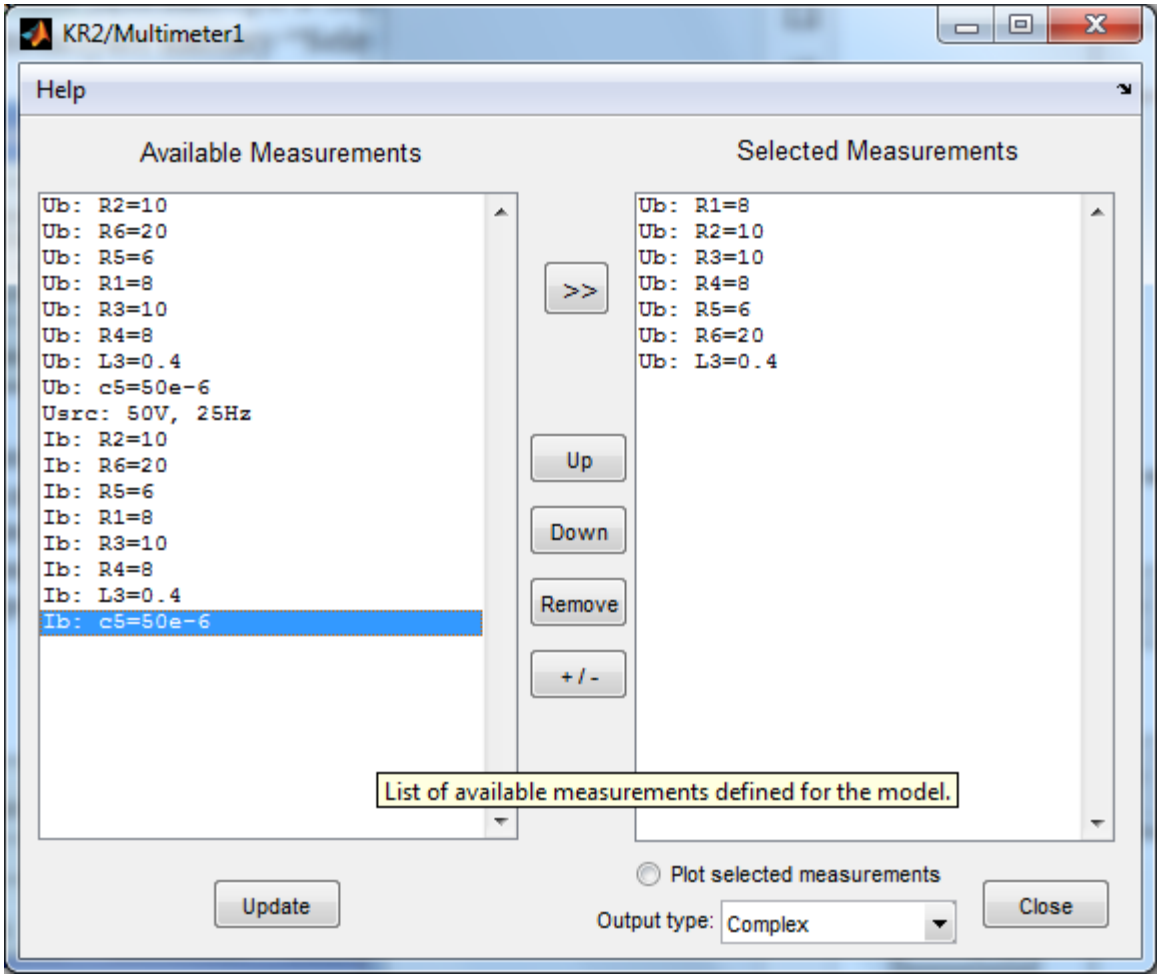


Рис. 16.7. Вікно настроювання блока *Multimetr*

У колонці *Available Measurements* відображається список величин, доступних для вимірювання, а у колонці *Selected Measurements* – список величин, обраних для вимірювання.

Для вибору колонці *Available* виділяють мишкою потрібний сигнал і кнопкою >> дублюють його у колонку *Selected Measurements*.

Видалити сигнал з колонки *Selected Measurements* можна кнопкою *Remove*. Змінити порядок сигналів у колонці *Selected Measurements* можна кнопками *Up* і *Down*. Кнопкою «+/-» можна міняти полярність обраних сигналів. На піктограмі блока автоматично відображується кількість обраних сигналів. Оновити список доступних сигналів після внесення змін у імітаційну модель можна кнопкою *Update*.

Якщо встановити прапорець у полі параметру *Plot selected measurements*, то після розрахунку перехідних процесів відкриється вікно з графіками обраних сигналів.

4. ***Impedance Measurement*** – вимірювач повного (комплексного) опору (імпедансу) ділянки електричного кола у заданому діапазоні частот.

Відображення амплітудної та фазової частотних характеристик імпедансу відбувається за допомогою блока *Powergui* у вікні, що відчиняється кнопкою *Impedance vs Frequency Measurements* (див. рис. 16.8). У цьому вікні можна скоригувати діапазон частот, нанести координатні сітки, обрати логарифмічний або лінійний масштаби зображення частотних характеристик, запам’ятати результати розрахунків.

При використанні вимірювача повного опору слід мати на увазі, що цей блок виконано на основі джерела струму. Тому його не можна включати послідовно з індуктивними елементами. Для усунення цього обмеження ***блок Impedance Measurement при наявності у вимірюваній ділянці котушок індуктивності слід шунтовати резистором із достатньо великим опором.*** Величину опору треба обирати такою, щоб властивості схеми суттєво не

змінювались.

Для вимірювання комплексних опорів пасивних елементів необхідно вилучити зі схем джерела енергії.

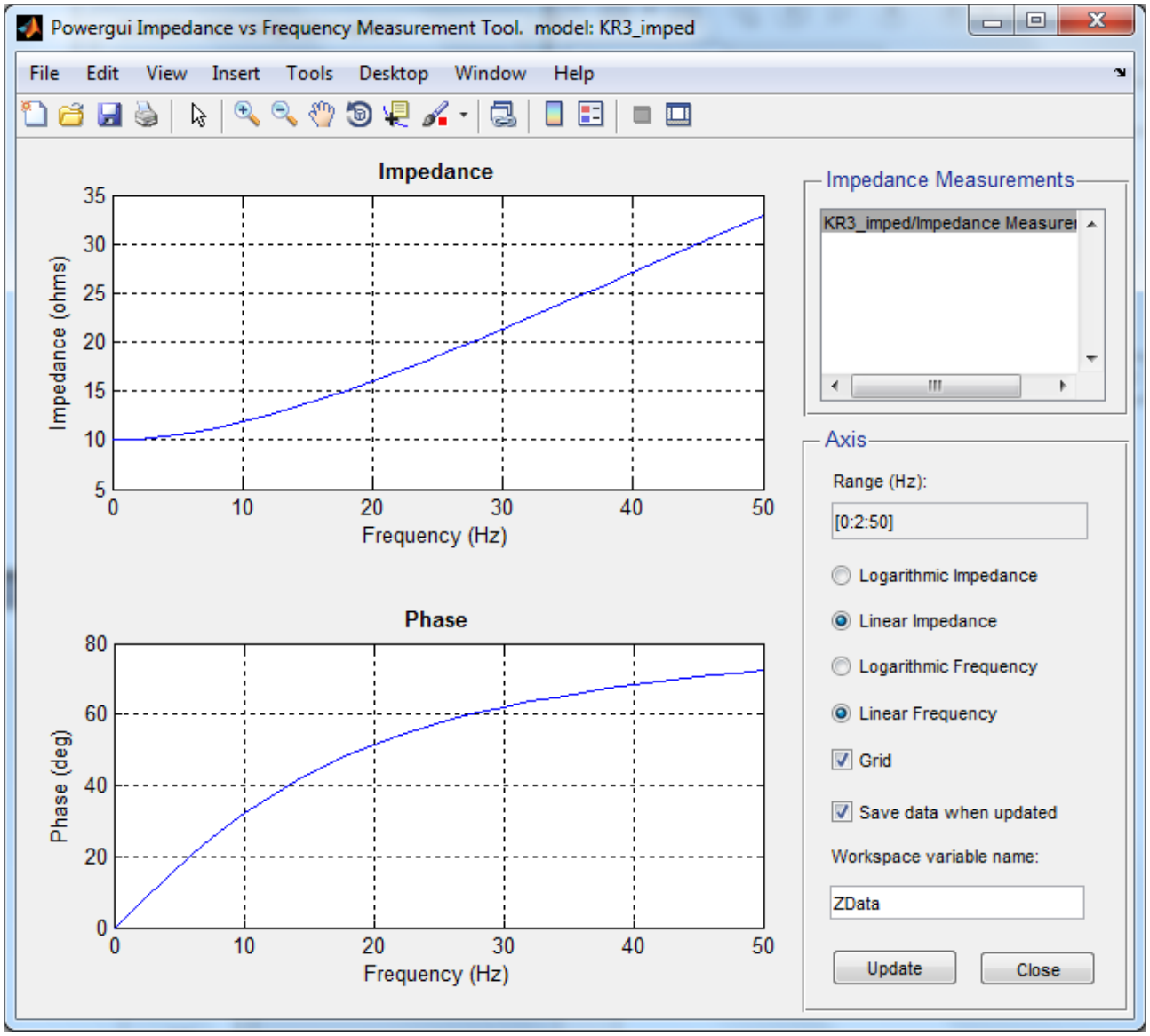


Рис.16.8. Вікно відображення частотних характеристик комплексного опору ділянки електричного кола

16.2.4. Ключові елементи

1. ***Breaker*** – перемикач, що здійснює комутацію між контактами 1 і 2, який може керуватися як у функції зовнішнього сигналу, що подається на керований вхід «*c*» (*External control mode*), так і за допомогою внутрішнього інтервального таймера (*Internal control mode*) (див. рис. 16.9).

Початковий стан ключа задається параметром *Initial State (0 for 'open',*

1 for 'close'). ***Параметр Breaker resistance Ron (Ohm)*** визначає внутрішній опір ключа у замкненому стані, який ***не можна встановлювати нульовим***. У розімкненому стані ключ має опір *Roff* = ∞.

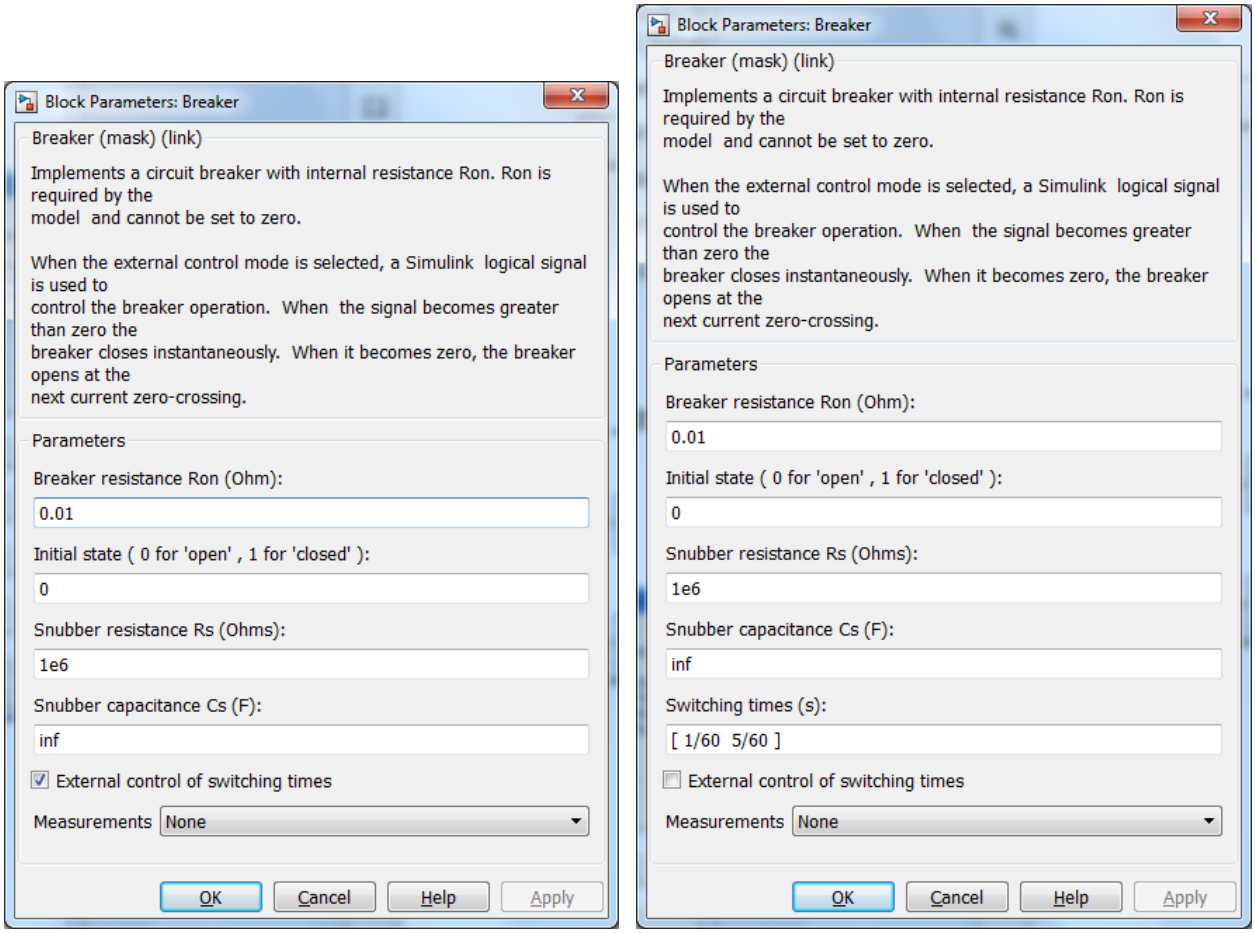


Рис. 16.9. Вікна параметрів блока *Breaker*

У модель «брейкера» включено послідовну *Rs*-*Cs*-гілку (*Snubber*), що здійснює функцію дугогасіння при розмиканні ключа, з параметрами *Snubber resistance Rs (Ohm)* та *Snubber capacitance Cs (F)*. ***За замовчанням*** параметр *Cs* має значення *inf* (∞), тобто ***«снабер» є чисто резистивним***. ***Встановлення Cs = 0 рівнозначно розмиканню «снабера».*** При цьому змінюється «іконка» блока (див. рис. 16.10). «Снабер» необхідно застосовувати, якщо послідовно до «брейкера» приєднані котушка індуктивності або джерело струму.

У режимі *External control mode* на іконці ключа з'являється *S*-вхід «*c*», що дозволяє здійснити керування станом ключа зовнішнім логічним сигналом: 1 подає команду на замикання ключа, а 0 – на розмикання. При відклю-

ченні режиму *External control mode* вхідний порт зникає, а у вікні параметрів з'являється поле *Switching times (s)*, в якому треба задати вектор моментів часу, в які стан ключа змінюється на протилежний.

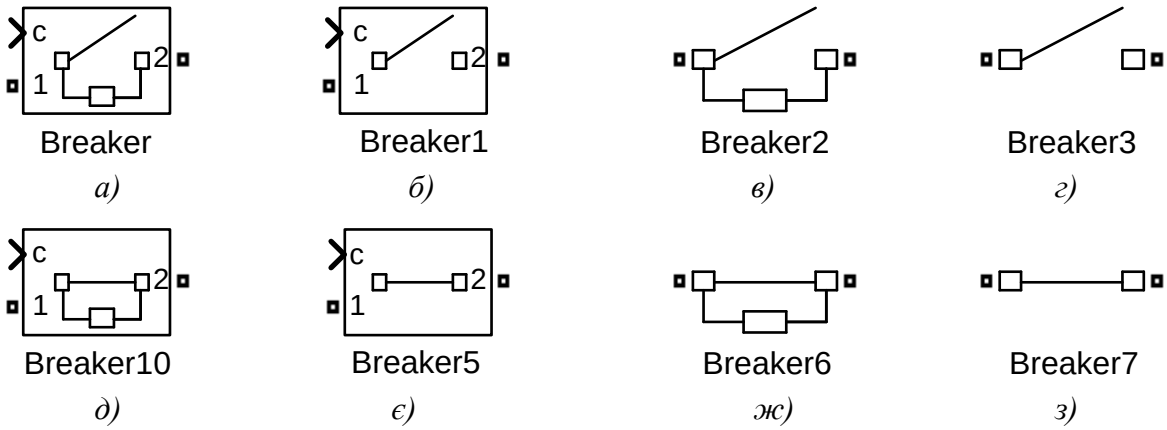


Рис. 16.10. Піктограми блока *Breaker*:
Initial State = 0 – *а, б, в, г*; *Initial State* = 1 – *д, є, ж, з*;
External control mode – а, б, д, є; Internal control mode – в, г, ж, з;
Cs = inf – а, в, д, ж; Cs = 0 – б, г, є, з

Особливістю блока ***Breaker*** є те, що він ***розмикає електричне коло після подачі сигналу на розмикання тільки тоді, коли струм, що протікає через ключ, стає нульовим***. Отже, ***Breaker може замкнути, але не може розімкнути коло постійного струму***. ***Його призначення – комутація у колах змінного струму.***

При наявності у імітаційній моделі блока *Breaker* ***краще за все використовувати для розрахунку перехідних процесів метод ode23tb***.

2. ***Ideal Switch*** – ідеальний ключ, керований *Simulink*-сигналом, що подається на вхідний *S*-порт *gate (g)*: ключ замикається при додатному керуючому сигналі, що перевищує 1, а розмикається при нульовому. Вихідний вимірювальний *S*-порт *m* у включеному стані виводить струм та напругу ключа.

Вікно параметрів цього блока показано на рис. 16.11.

Як бачимо, він має багато спільного із блоком *Breaker*. Це і початковий стан ключа, який задається параметром *Initial State (0 for 'open', 1 for 'close')*, і наявність резистивного *«снабера»*, який можна вимкнути, встановленням *Cs* = 0, що відображається на вигляді піктограми блок (див. рис. 16.10), і на-

явність внутрішнього опору ключа *Internal resistance Ron (Ohm)*. Але, на відміну від блока *Breaker*, блок *Ideal Switch* розмикає електричне коло одразу після того, як на порт приходить нульовий сигнал. Тому ***цей блок може повноцінно застосовуватися для комутації у колах постійного струму***.

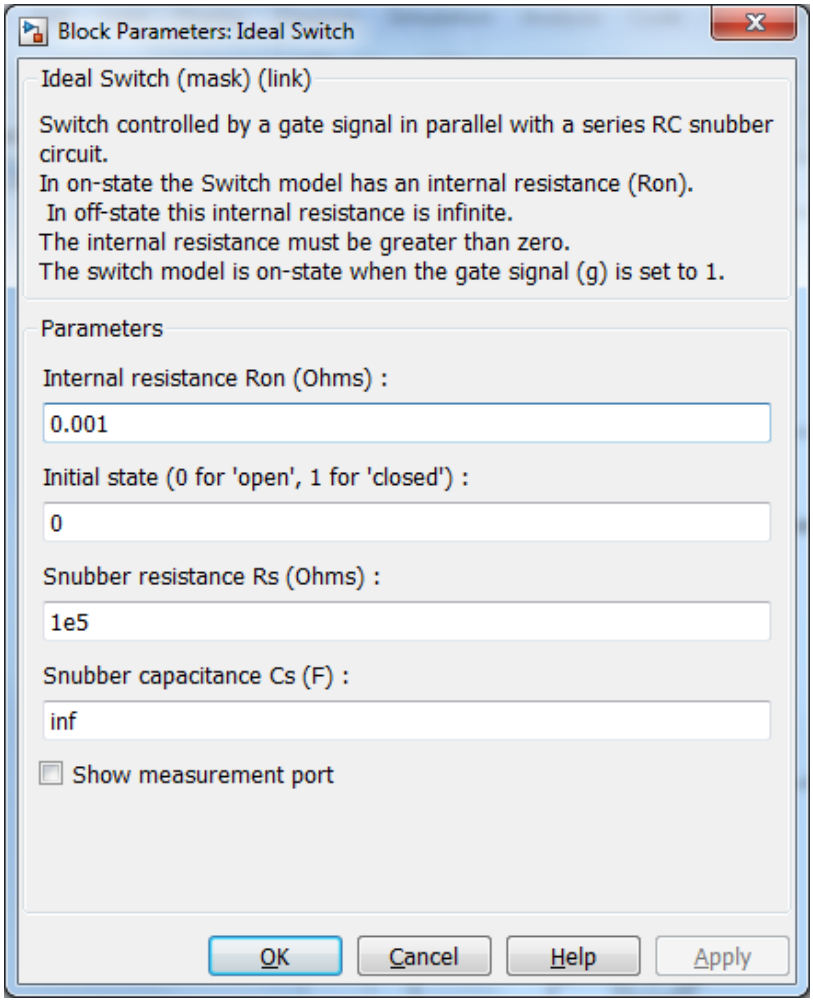


Рис. 16.11. Вікно параметрів блока *Ideal Switch*

Менш суттєвими відмінностями є

- відсутність внутрішнього таймера для керування стану ключа;
- відсутність внутрішнього таймера для керування стану ключа;
- наявність вимірювального порту, який можна робити видимим і невидимим за допомогою параметра *Show measurement port*;
- на порядок менші опори *Ron* (0.001 Ом за замовченням замість 0.01 Ом у «брейкера») і *Rs* (10 кОм у порівнянні зі 100 кОм у «брейкера»);

- не змінює піктограму при замкненому початковому стані (див. рис. 16.10).

16.2.5. Графічний інтерфейс користувача *Powergui*

Powergui (Powerlib Graphical User Interface) призначений для аналізу імітаційних фізичних моделей електричних кіл і систем. Вікно завдання параметрів цього блока подано на рис. 16.12.

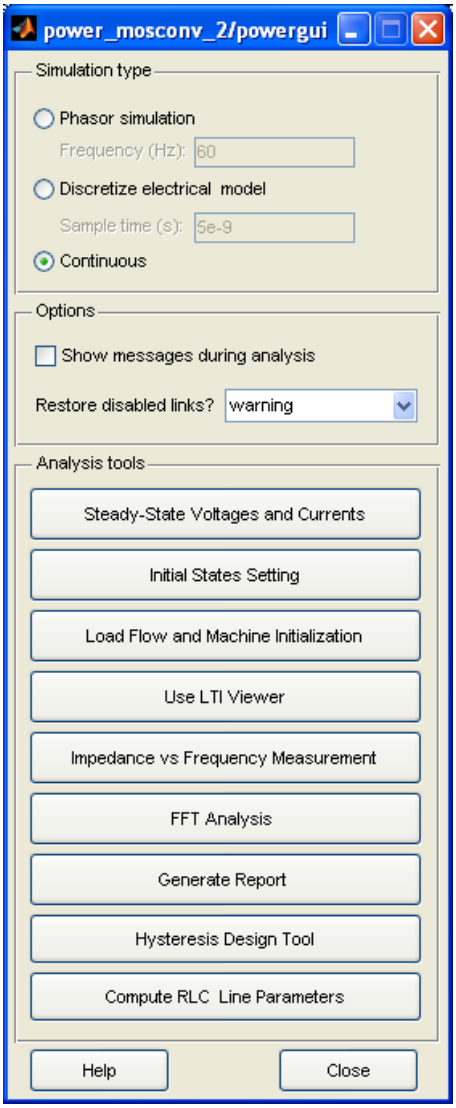


Рис. 12.12. Вікно параметрів блока *Powergui*

Блок обов'язково треба розміщати у вікні моделі, у якій присутній хоча б один з блоків бібліотек *SimPowerSystems* (у версіях MATLAB 7 і вище поміщається у вікно моделі автоматично). Блок має бути тільки один, і його не можна перейменовувати. Обновити список доступних сигналів після внесення змін у імітаційну модель можна кнопкою *Update*.

Блок *Powergui* дозволяє вирішувати такі задачі:

1. ***Simulation type*** – вибір методу розрахунку динамічних та статичних режимі, зазвичай у вигляді вікна вибору однієї з опцій: *Continuous* (неперервні), *Discretize electrical model* (дискретні, треба вказати період дискретності *Sample Time*), *Phasor simulation* (метод узагальнених векторів, треба вказати основну частоту).

2. ***Steady-State Voltages and Currents*** – відображення усталених значень перемінних (вікно *steady state values*).

Вікно визначення параметрів цієї операції зображено на рис. 16.17.

До них належать:

- *Units* (одиниці) – *peak values* (амплітудні значення) / *RMS values* (діючі значення);
- *Frequency* (частота в Гц);
- *Display* – прапорцями слід задати одну/один або декілька змінних стану (*States*), блоків вимірювання (*Measurements*) та джерел (*Sources*);
- *Format* (формат виведення чисел) – з фіксованою точкою або плаваючою;
- *Ordering* – порядок виводу (*Value then name*), *Name then value*

Оновлення інформації досягається натисканням на кнопку *Update steady state values*.

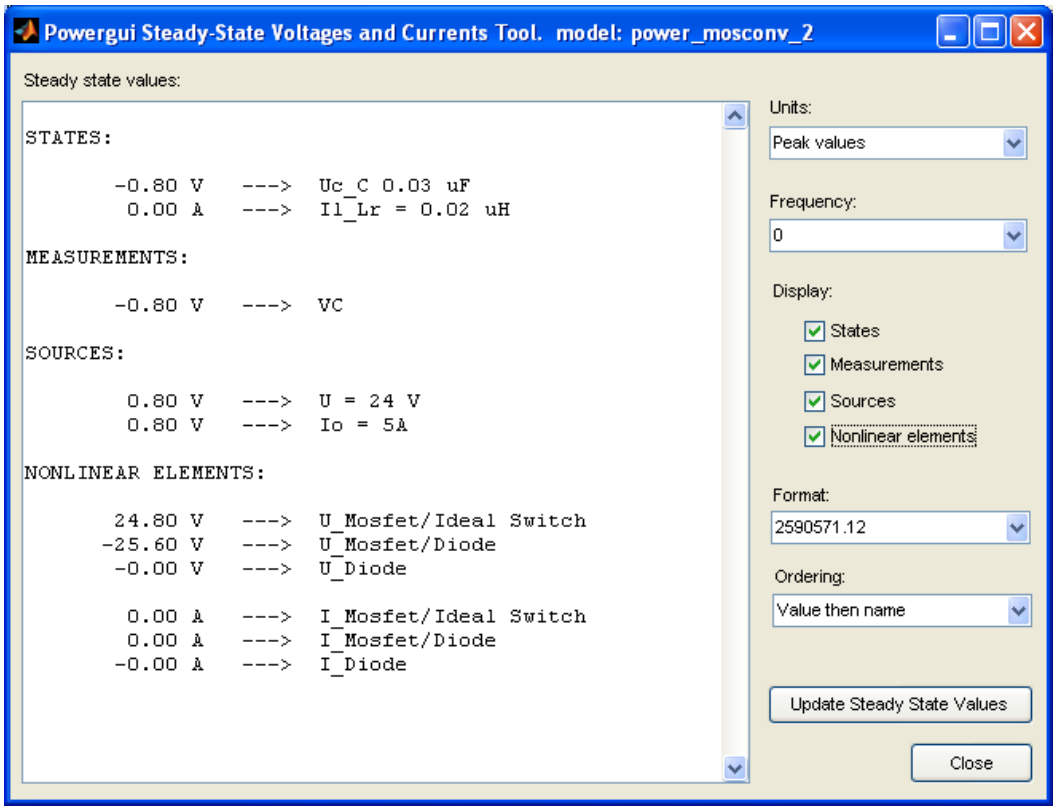


Рис. 16.17. Вікно відображення усталених значень

3. ***Initial States setting*** – редагування або установка початкового стану моделі (завдання початкових умов змінним стану), дозволяє виконати ініціалізацію моделі, визначаючи початок моделювання з усталеного режиму або з нульових початкових умов. Вікно визначення початкових умов подано на

рис. 16.14.

4. ***Load Flow and Machine Initialization*** – ініціалізація трифазних кіл (*Three-Phase Dynamic Load*) та електричних машин (*Machines*). Вікно для виконання цієї операції зображено на рис. 16.15.

Більш детально ознайомитися з можливостями цієї операції можна в [7]. Інші функції інтерфейсу користувача будуть розглянуті пізніше у інших дисциплінах або при необхідності самостійно.

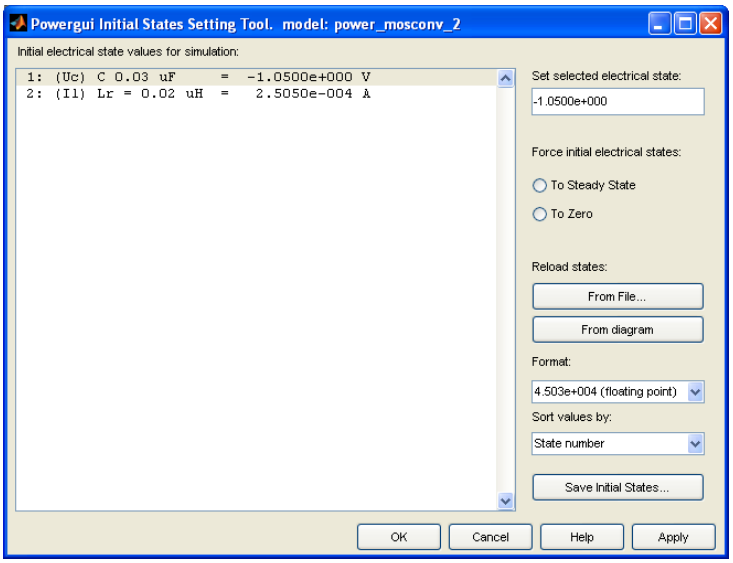


Рис. 16.14. Вікно визначення початкових умов

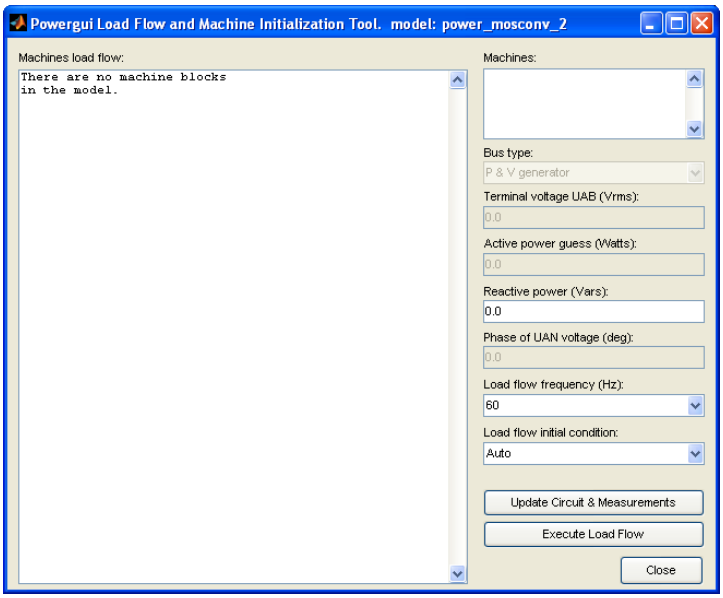


Рис. 16.15. Вікно ініціалізації динамічних трифазних електричних кіл та електричних машин

16.3. Приклад імітаційного фізичного моделювання лінійного розгалуженого електричного кола

Як приклад, розглянемо розгалужену схему, зображену на рис. 16.16, для якої треба розрахувати перехідні процеси, тобто знайти $i_1(t)$, $i_2(t)$, $i_3(t)$, $U_C(t)$ при замиканні ключа при таких параметрах: E = 100 В, J_k = 5 А, C = 200 мкФ, L = 0,3 Гн, r_1 = 20 Ом, r_2 = 50 Ом, r_3 = 70 Ом.

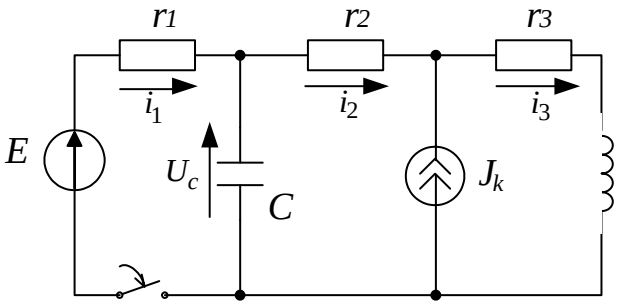


Рис. 12.16. Досліджуване електричне коло

Перевагою імітаційного фізичного моделювання є те, що для його застосування не потрібно складати математичний опис досліджуваної системи. Ця функція покладається на програмне забезпечення. Задачею користувача в даному разі є складання принципової електричної схеми у блоках імітаційної бібліотеки *SimPowerSystems*, що потребує деякого досвіду. *SPS*-модель електричного кола, зображеного на рис. 16.16, подана на рис. 16.15.

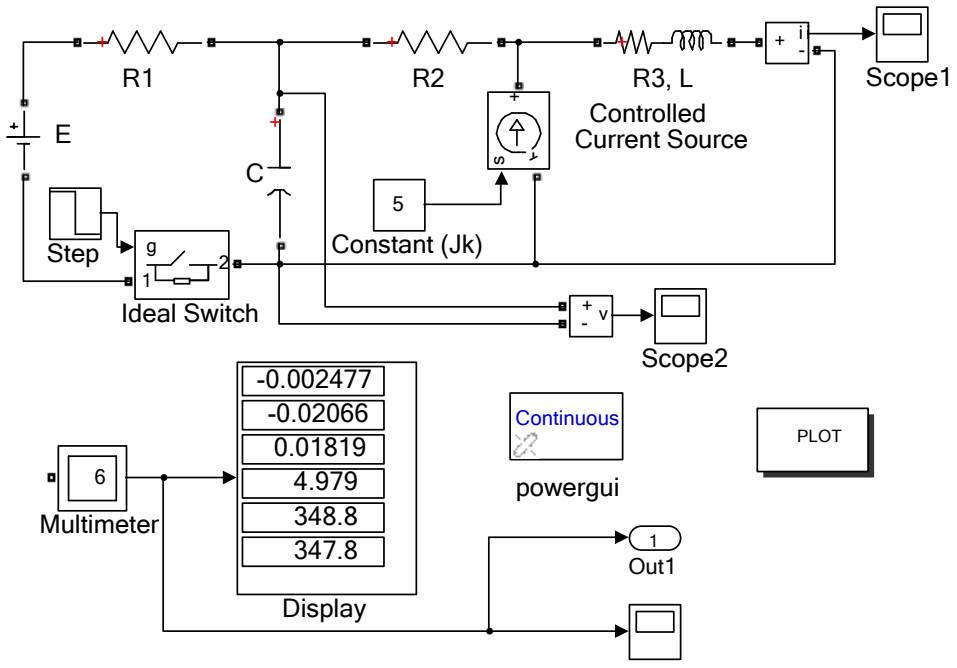


Рис. 16.15. *SPS*-модель електричного кола, зображеного на рис. 16.16

Simulink-блоки у моделі рис. 16.17 використані тільки для фіксації результатів, формування вихідного сигналу джерела струму та керування ключовим елементом *Ideal Switch*. На рисунку продемонстровано можливість вимірювання сигналів різними інструментами.

Графіки перехідних процесів, отриманих за результатами симуляції досліджуваної моделі показані на рис. 16.18**.**

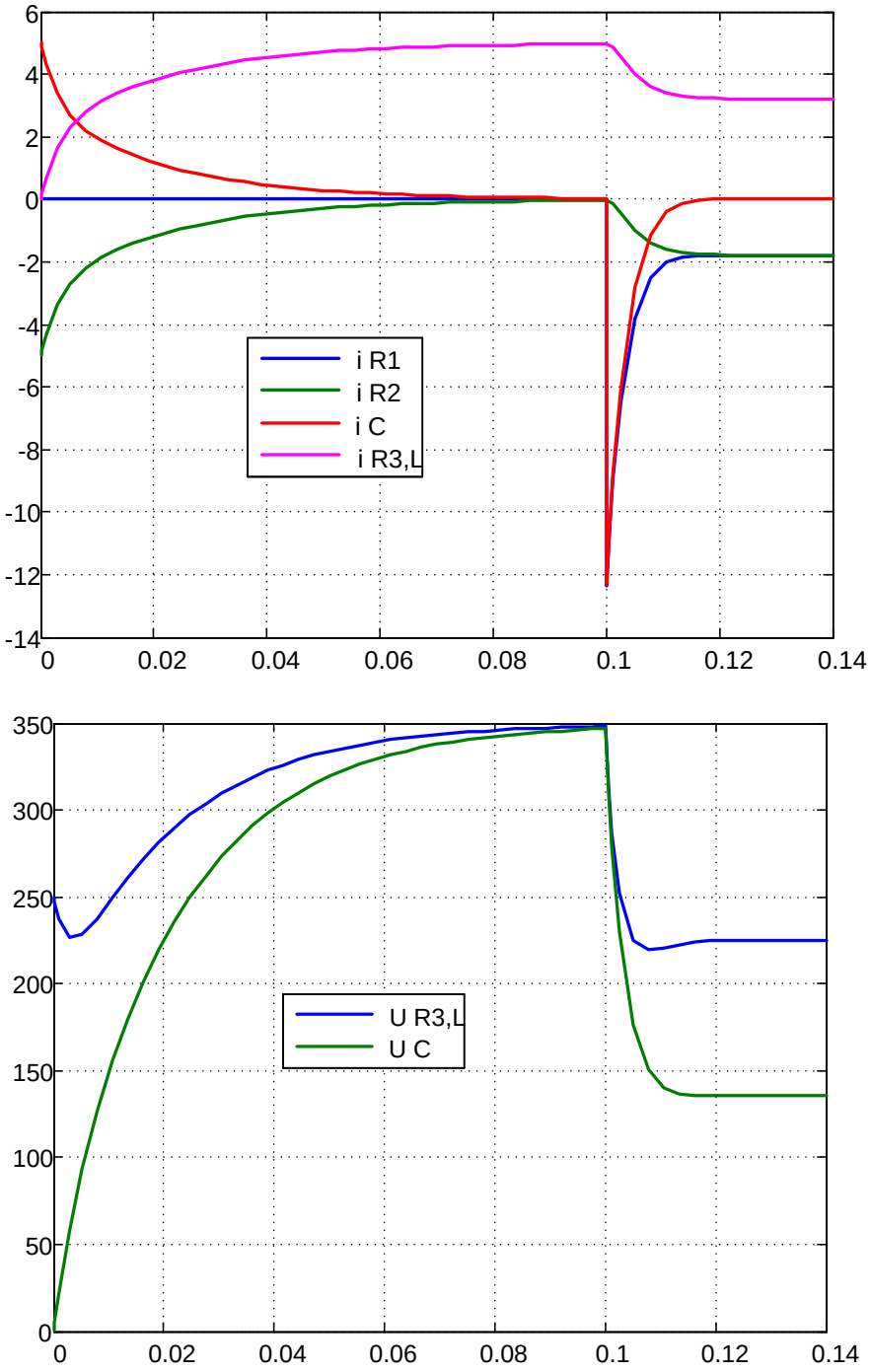


Рис. 16.18. Результати симуляції моделі рис. 16.17

Контрольні питання та завдання

1. Чим відрізняються блоки бібліотек *SimPowerSystems* від *Simulink*-блоків?
2. Які порти мають *SPS*-блоки?
3. Як поєднати між собою *SPS*-блоки і блоки *Simulink*?
4. Для чого застосовують блок *Powergui*?
5. Як скористатися блоком *Powergui* для встановлення початкових умов?
6. Як працюватиме *SPS*-модель електричного кола без ключових елементів, якщо їй не задати нульові початкові умови?
7. Порівняйте між собою переваги та недоліки імітаційного фізичного та структурного математичного моделювання.

17. ІМІТАЦІЙНЕ ФІЗИЧНЕ МОДЕЛЮВАННЯ ЕЛЕКТРИЧНИХ ДВИГУНІВ

Імітаційні фізичні моделі електричних машин знаходяться в бібліотеці *Machines* додатка *SimPowerSystems*. До складу її входять машини постійного струму, крокові двигуни, асинхронні, синхронні та реактивні машини. Електричні машини можуть працювати як генератори і як двигуни. В них передбачені різні варіанти включення обмоток, різні способи збудження синхронних машин, можливість створення систем «електричний вал» та «механічний вал». В багатьох машинах параметри можуть задаватися як в абсолютних одиницях системи СІ (*SI Units*), так і у відносних одиницях (*pu Units*). В даному підручнику розглянемо тільки моделі двигуна постійного струму (ДПС) з незалежним збудженням та асинхронного двигуна (АД).

17.1. Імітаційне фізичне моделювання двигуна постійного струму з незалежним збудженням

На рис. 17.1 подано зображення блока *DC Machine* – машина постійного струму, який знаходиться в розділі *Machines* бібліотеки *SimPowerSystems*. Як бачимо, воно складається з якоря та обмотки збудження. Отже, на основі цього блока можна створити машину постійного струму з послідовним, паралельним та незалежним збудженням [7].

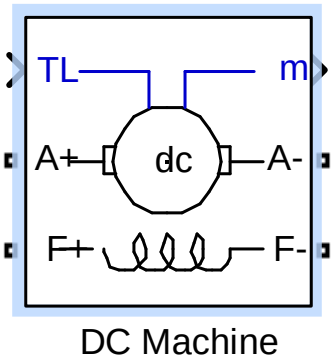


Рис. 17.1. *SPS*-блок *DC Machine*

Блок має «електричні» порти A+, A- та F+, F-, що відповідають затискачам якоря та обмотки збудження відповідно. Крім «електричних» портів, блок *DC Machine* має один механічний вхід та векторний «інформаційний» вихід m (від *measurements*), який передбачає подальшу обробку або фіксацію сигналів засобами базових блоків *Simulink*. На рис. 17.2 та 17.3 зображені вкладки вікна введення параметрів цього

блока.

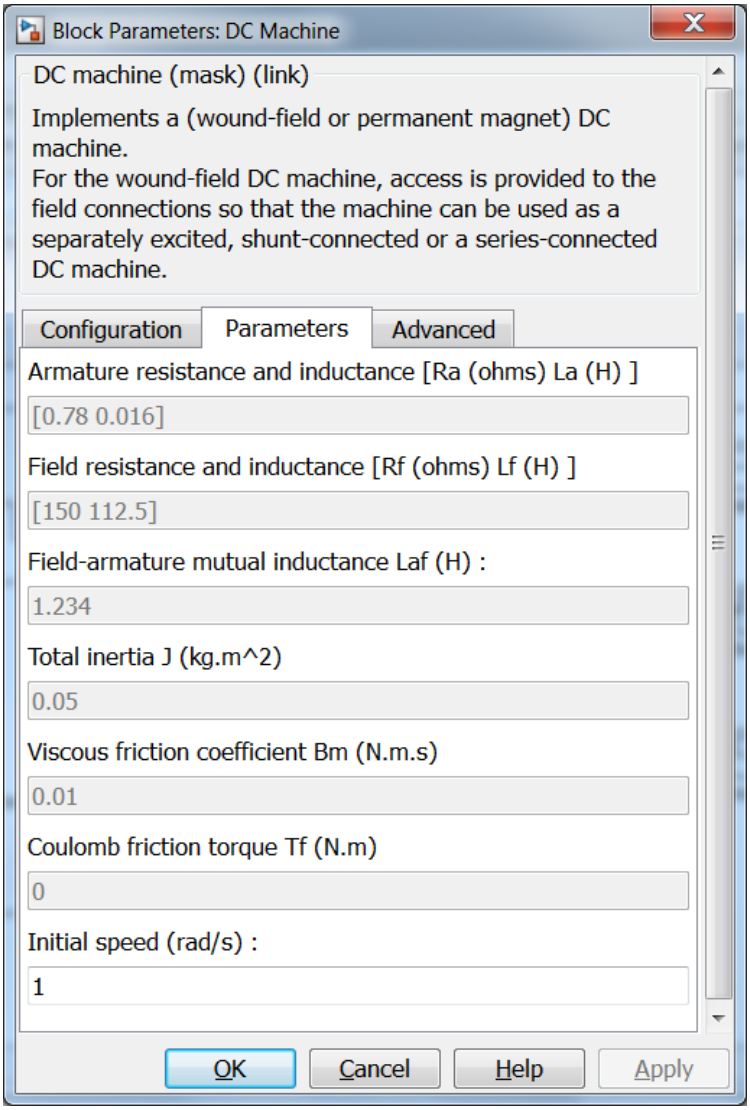


Рис. 17.2. Вкладка *Parameters* вікна встановлення параметрів блока *DC Machine*

Вкладка *Parameters* (рис. 17.2) діалогового вікна установки параметрів блока дозволяє встановити значення таких параметрів ДПС:

- *Armature resistance* Ra (ohms) – активний опір якоря $R_я$ (Ом);
- *Armature inductance* La (H) – індуктивність якоря $L_я$ (Гн);
- *Field resistance* Rf (ohms) – активний опір обмотки збудження $R_з$ (Ом);
- *Field inductance* Lf (H) – індуктивність обмотки збудження $L_з$ (Гн);
- *Field-Armature mutual inductance* Laf (H) – взаємна індуктивність, L_{af} (Гн);

- *Inertia* J (kg.m^2) – момент інерції двигуна, J $(\text{кг} \cdot \text{м}^2)$;
- *Viscous friction coefficient* Bm (N.m.s) – коефіцієнт в'язкого тертя $k_{f\omega}$ $(\text{Нм}/(\text{рад/с}))$;
- *Coulomb friction torque* Tf (N.m) – момент сухого тертя $M_{\text{т}}$ (Нм);
- *Initial speed* (rad/s) – початкова кутова швидкість.

Параметри ДПС та можливість їх встановлення або коригування залежать від стану опції *Preset Model* (попередня ініціалізація моделі даними деякого двигуна) вкладки *Configuration* (рис. 17.3).

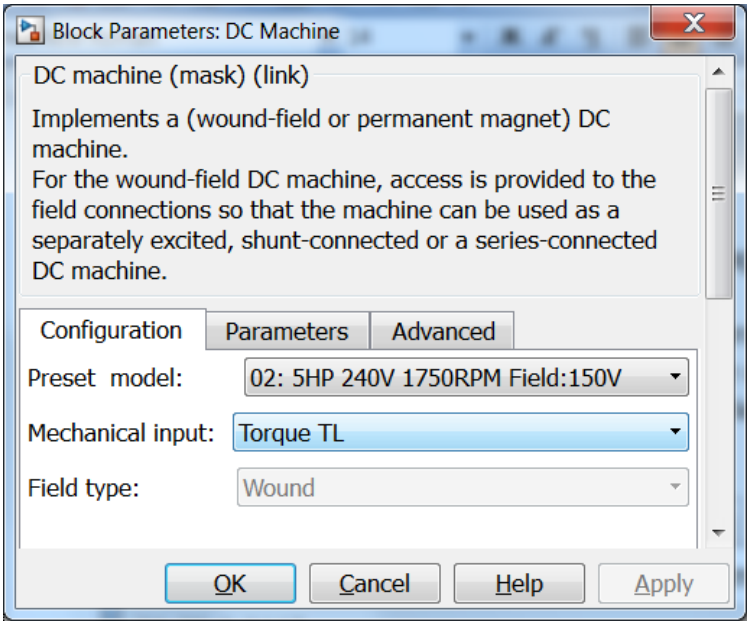


Рис. 17.3. Вкладка *Configuration* вікна встановлення параметрів блока *DC Machine*

У початковому стані ця функція має значення *No*, параметри вкладки *Parameters* мають певні значення, які після переміщення блока у вікно моделі можна змінювати. Це зручно робити, якщо Ви обрали зі стороннього джерела якийсь двигун та, крім тих даних, що вводяться у вікно вкладки *Parameters* знаєте потужність двигуна та його номінальні дані, які необхідні для моделювання.

Якщо користувач не має даних двигуна для дослідження, то він може його обрати з випадного меню опції *Preset Model*, в якому для кожного з

двигунів наводиться його потужність у кінських силах HP (1HP = 746 Вт), напруга якоря та напруга збудження (*Field*:) у вольтах (V) і номінальна швидкість в обертах за хвилину (RPM –*Revolutions Per Minute*).

Вибір певного двигуна, підтверджений натисканням кнопки *Apply*, призводить до автоматичного встановлення відповідних параметрів у вкладці *Parameters*, які тепер не можуть бути скориговані користувачем.

Для того, щоб зробити дані попередньо обраного двигуна доступними до коригування, треба після вибору цього двигуна віртуальною кнопкою *Apply* знову встановити функцію *Preset Model* у значення *No*.

Якщо порівняти параметри структурної моделі ДПС, розглянутої в лабораторній роботі №7 з параметрами *SPS*-блока *DC Machine*, то можна побачити, що у структурній моделі існує параметр c, відсутній в *SPS*-моделі, але не існує параметру L_{af}. Зв'язок між цими параметрами встановлюється формулою

$$c = L_{af} i_{fn} = L_{af} \frac{U_{fn}}{R_f} = L_{af} \frac{U_{зн}}{R_з}. \qquad (17.1)$$

Модель для дослідження ДПС з незалежним збудженням, у якій використано розглянути вище блок *DC Machine* показано на рис. 17.4.

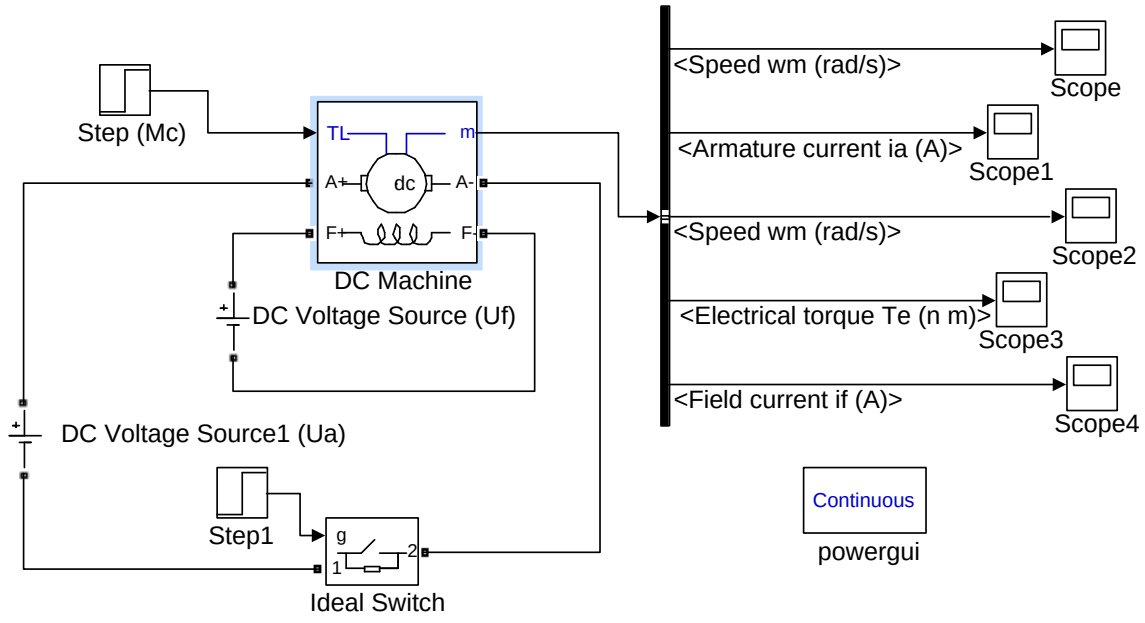
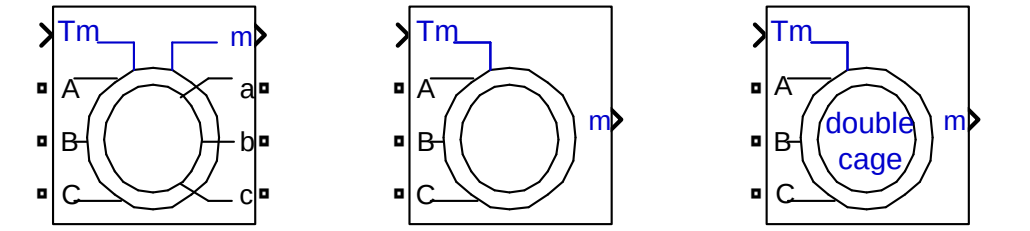


Рис. 17.4. Імітаційна фізична модель двигуна постійного струму з незалежним збудженням

17.2. Імітаційне фізичне моделювання короткозамкненого асинхронного двигуна

В розділі *Machines* бібліотеки *SimPowerSystems* представлені два блоки трифазної асинхронної машини: *Asynchronous Machine SI Units* (асинхронна машина в абсолютних одиницях системи СІ) та *Asynchronous Machine pu Units* (асинхронна машина у відносних одиницях).

Блоки мають порти A, B, C та a, b, c, що відповідають «електричним» затискачам статора та ротора відповідно. Крім «електричних» портів, моделі мають один механічний вхід та векторний «інформаційний» вихід m (від *measurements*), який передбачає подальшу обробку або фіксацію сигналів засобами базових блоків *Simulink*. Зовнішній вигляд блоків *Asynchronous Machine* (рис. 17.5) визначається типом ротора та типом механічного входу, які встановлюються за допомогою меню параметрів *Rotor type* і *Mechanical input* вкладки *Configuration* (рис. 17.6а).



а) *б)* *в)*

Рис. 17.5. Блоки *Asynchronous Machine* з різними типами ротора (*Rotor Type*): а) з фазним ротором (*Wound*), б) з білячою кліткою на роторі (*Squirrel-cage*), в) з двома білячими клітками на роторі (*Double squirrel-cage*)

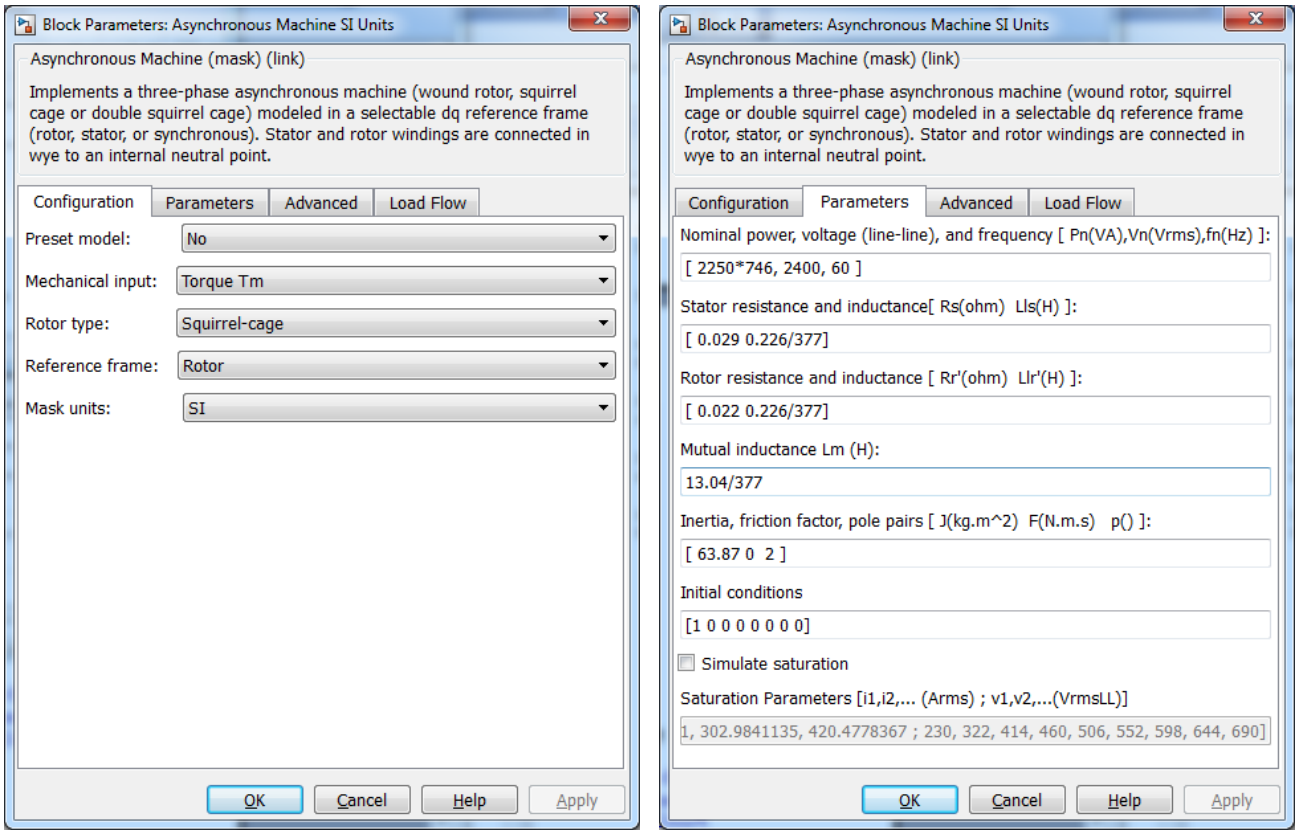
Вкладка *Parameters* (рис. 17.6*б*) діалогового вікна установки параметрів блока дозволяє встановити значення таких параметрів асинхронного двигуна [7-9]:

- *Nominal power* Pn (VA) – номінальна потужність P_n (ВА);
- *Voltage (line-line)* Vn (Vrms) – діюче, тобто ефективне, або середньоквадратичне (*rms – root mean square*) значення номінальної лінійної напруги

статора U_{len} (В);

- *Frequency* fn (Hz) – номінальна частота напруги живлення f_n (Гц);
- *Stator resistance* Rs (Ohm) – активний опір фази статора, R_s (Ом);
- *Stator inductance* Lls – індуктивність розсіяння фази статора (l – leakage – розсіювання), $L_{s\sigma}$ (Гн);
- *Rotor resistance* Rr' (Ohm) – активний опір фази ротора, приведений до статора R_r (Ом);
- *Rotor inductance* Llr' – індуктивність розсіяння фази ротора, приведена до статора $L_{r\sigma}$ (Гн);
- *Mutual inductance* Lm (H) – головна взаємна індуктивність, L_m (Гн);
- *Inertia* J (kg.m^2) – момент інерції ротора, J (кг $\cdot$ м^2);
- *Friction factor* F (N.m.s) – коефіцієнт в'язкого тертя k_{fv} (Н $\cdot$ м/(рад/с));
- *Pole pairs* p () – кількість пар полюсів Z_p;
- *Initial conditions* – початкові умови за координатами: ковзання, кутове положення ротора, амплітуди та фазові кути струмів фаз статора та (якщо треба) ротора;
- *Simulate saturation* – при наявності в полі цього параметру прапорця буде враховуватись ефект насичення магнітного кола за табличними даними кривої намагнічування $U_s = f(I_s)$, що вводяться у вигляді дворядкової матриці параметру *Saturation Parameters;* перший стовпець цієї матриці отримує координати тієї точки кривої намагнічування, з якої починає проявлятися ефект насичення, тобто він має бути ненульовим.

Параметри АД та можливість їх встановлення або коригування залежать від стану опції *Preset Model* (попередня ініціалізація моделі даними деякого двигуна) вкладки *Configuration* (рис. 17.6*а*). У початковому стані ця функція має значення *No*, параметри вкладки *Parameters* мають певні значення, які після переміщення блока у вікно моделі можна змінювати.



а) *б)*

Рис. 17.6. Вкладки діалогового вікна встановлення параметрів блока *Asynchronous Machine SI Units*: *а*) *Configuration*; *б*) *Parameters*

В меню функції *Preset Model* наводиться перелік, з якого можна обрати конкретний двигун за його потужністю у кінських силах HP (1HP = 746 Вт), діючим значенням лінійної напруги статора у Vrms, номінальною частотою в Hz та номінальною швидкістю в RPM (*Revolutions Per Minute* = об/хв). Вибір певного двигуна, підтверджений натисканням кнопки *Apply*, призводить до автоматичного встановлення відповідних параметрів у вкладці *Parameters*, які тепер не можуть бути скориговані користувачем.

Слід зазначити, що до параметрів попередньо обраних двигунів не входять координати кривої намагнічування, тобто, ці координати задані тільки для одного двигуна, параметри якого встановлюються за замовченням (початковий стан моделі).

Для того, щоб зробити дані попередньо обраного двигуна доступними до коригування, треба після вибору цього двигуна віртуальною кнопкою

Apply знову встановити функцію *Preset Model* у значення *No*. Підкреслимо, що можливість ініціалізації моделі через встановлення певного набору параметрів АД передбачена тільки для двигунів з однією білячою кліткою на роторі.

Параметри вкладок *Reference frame* будуть пояснені у подальших дисциплінах, а параметри вкладок *Advanced* та *Load Flow* (рис. 17.6а) зазвичай можна не змінювати.

17.2.1. Моделювання асинхронного двигуна в режимі прямого пуску

На рис. 17.7 зображена *SPS*-модель, призначена для дослідження прямого пуску АД [7-9].

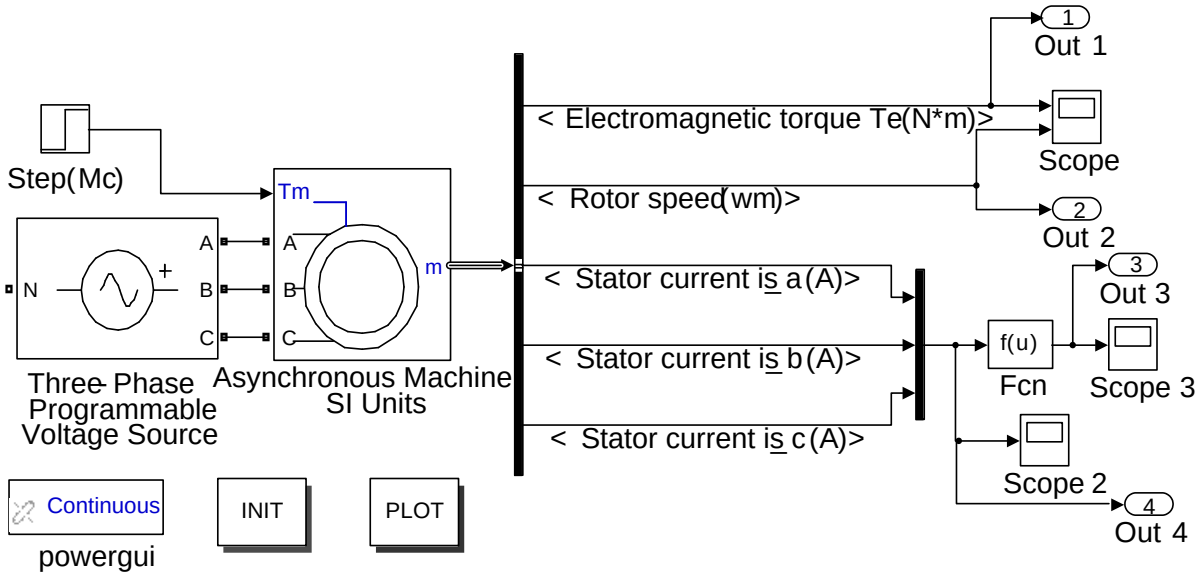


Рис. 17.7. *SPS*-модель прямого пуску асинхронного двигуна

У наведеній моделі вибір сигналів для реєстрації здійснюється з загального вектору вихідних *Simulink*-сигналів (порт m) *SPS*-блока *Asynchronous Machine SI Units S*-блоком *Bus Selector* бібліотеки *Signal Routing*. Вікно блока *Bus Selector* показане на рис. 17.8.

Обчислення діючого (ефективного) значення струму статора виконує блок *Fcn* за формулою:

$$I_{se} = \sqrt{\left(I_A^2 + I_B^2 + I_C^2\right)/3}. \qquad (17.2)$$

Фазні напруги формуються джерелом *Tree-Phase Programmable Voltage*

Source, вікно параметрів якого відображено на рис. 17.9.

Замість одного джерела *Tree-Phase Programmable Voltage Source*, можна використати три блока *AC Voltage Source*, з'єднані зіркою.

При визначенні параметрів джерел необхідно звернути увагу на те, що в параметрах двигуна задано діюче значення лінійної напруги [Voltage (line-line) (Vrms)], а блоки *AC Voltage Source* потребують амплітудних значень фазних напруг [Peek amplitude (V)], а також, що SPS-джерела *AC Voltage Source* використовують частоту не в рад/с, а в герцах, а фазовий зсув синусоїдальних сигналів – не в радіанах, а в градусах [Phase (deg)].

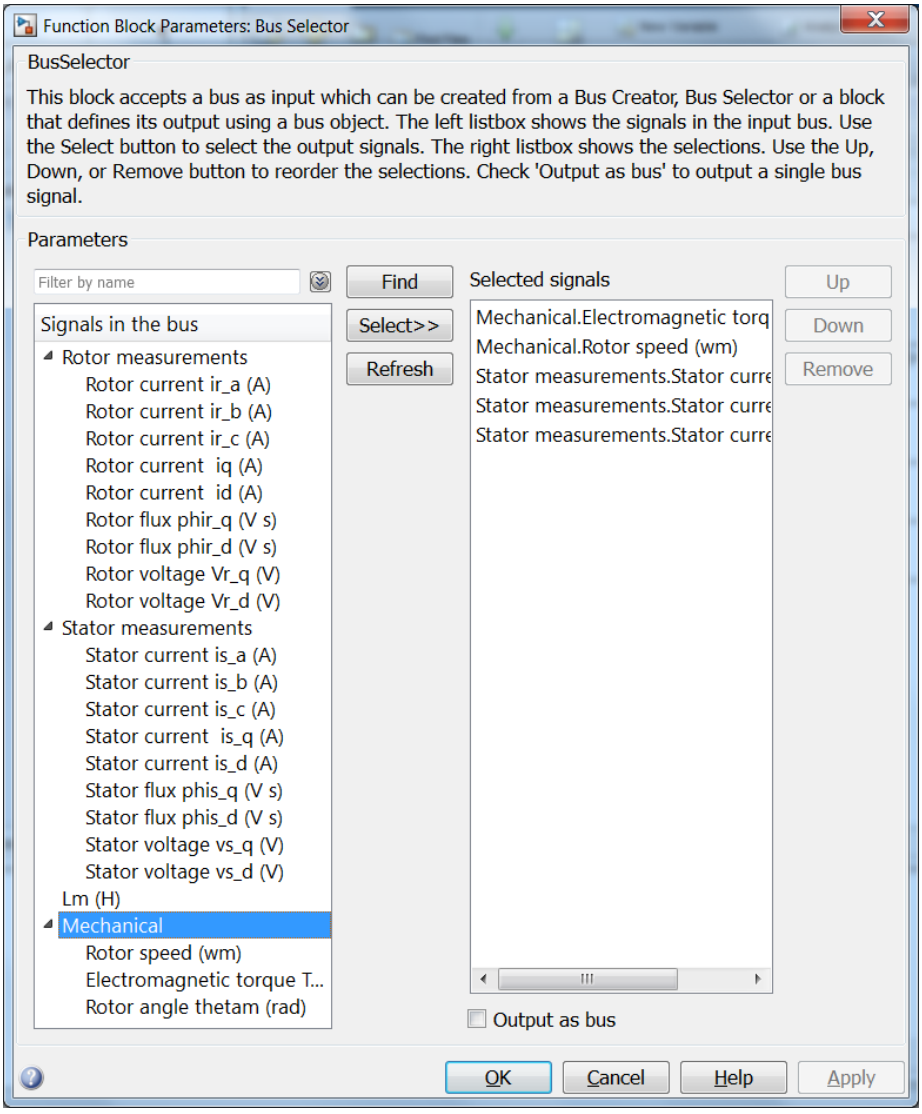


Рис. 17.8. Діалогове вікно блока *Bus Selector*,
підключеного до вихідного інформаційного порту двигуна

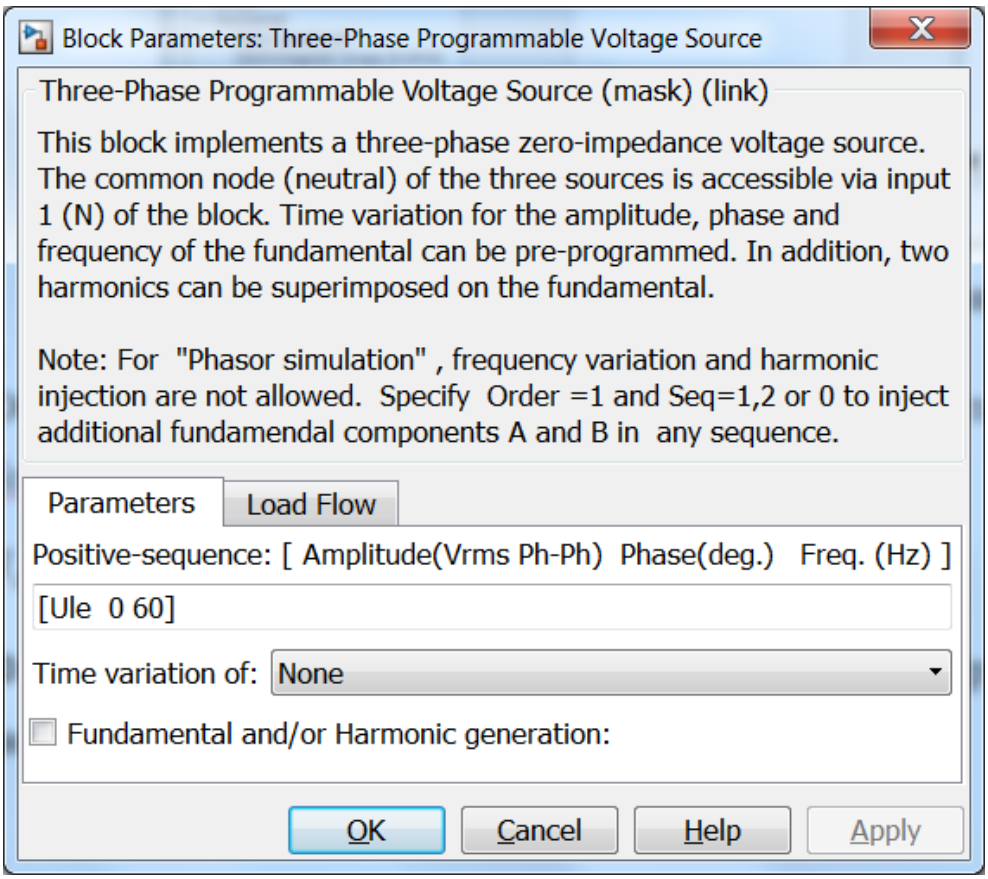


Рис. 17.9. Діалогове вікно блока *Tree-Phase Programmable Voltage Source*

Перехідні процеси, отримані за результатами симуляції моделі рис. 17.7 подані на рис. 17.10.

За допомогою цієї ж моделі можна побудувати і статичні характеристики двигуна, якщо змінювати момент статичного опору за лінійним законом настільки повільно, щоб практично не призводити до збудження електромагнітного перехідного процесу. Завершення сеансу моделювання у цьому експерименті можна здійснити блоком *Stop Simulation*, який здійснює зупинку моделювання тоді, коли швидкість АД після перекидання явно перейде в область від'ємних значень, що досягається застосуванням блока *Relational Operator*.

Недоліком прямого пуску є значні стрибки і коливання електромагнітного моменту та струмів АД.

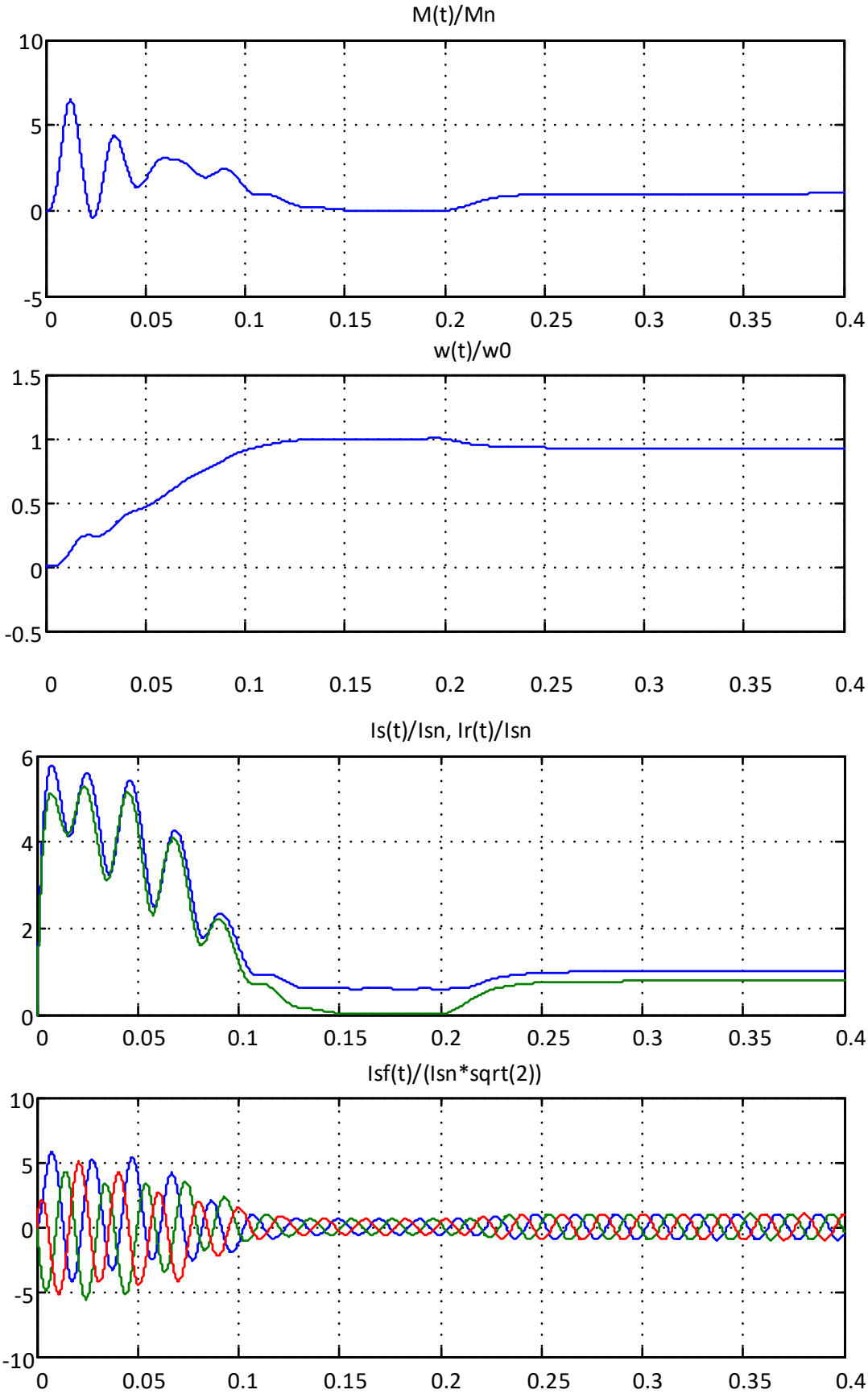


Рис. 17.10. Перехідні процеси при прямому пуску асинхронного двигуна

17.2.2. Моделювання асинхронного двигуна в режимі плавного пуску

Плавний пуску (Soft Start) полягає у плавній зміні амплітуди напруги статора з постійною частотою. Найпростішим варіантом поступової зміни амплітуди є зміна її до номінального значення за лінійним законом [16-17]. На рис. 17.11 зображена модель [3], призначена для формування такої напруги за допомогою *Simulink*-блоків за рівняннями

$$\begin{cases} U_{sA}(t)=U_{sfm}\sin(\varphi(t)), \\ U_{sB}(t)=U_{sfm}\sin(\varphi(t)-2\pi/3), \\ U_{sC}(t)=U_{sfm}\sin(\varphi(t)-4\pi/3), \end{cases}$$

де

$$\varphi(t)=\int_0^t \omega_s(t)dt=2\pi\int_0^t f_s(t)dt$$

та подальшого перетворення отриманих фазних напруг в *SPS*-сигнали за допомогою керованих джерел напруги (*Controller Voltage Sourse*).

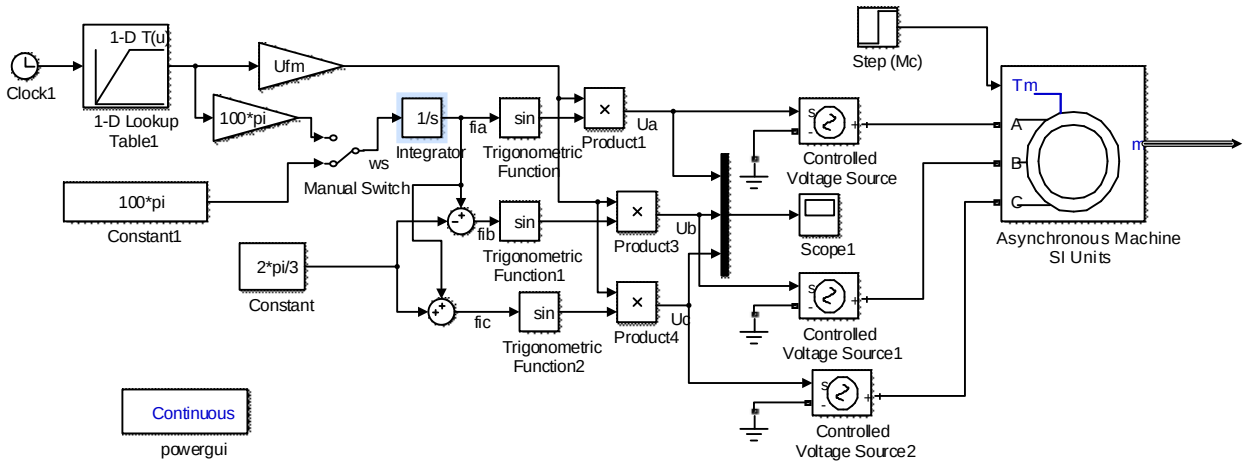


Рис. 17.11. Модель формування напруг статора для плавного пуску

Звісно, що при $f_s=50\text{Гц}=const$, $\omega_s=2\pi f_s=100\pi=const$ фазовий кут $\varphi(t)$ змінюється за лінійним законом $\varphi(t)=\omega_s t$. Перехідні процеси в АД при плавному пуску показані на рис. 17.12, а напруги статора – на рис. 17.17.

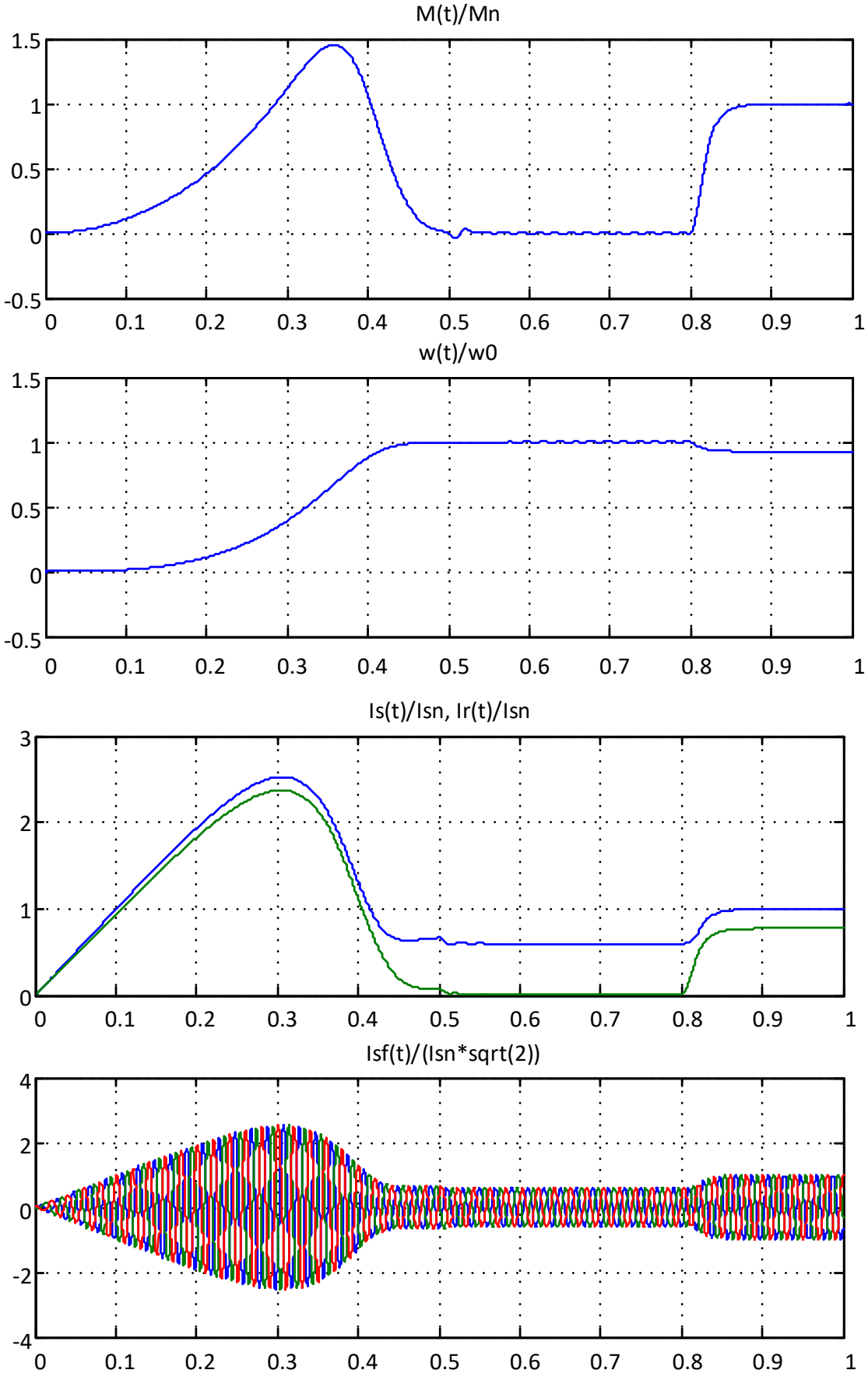


Рис. 17.16. Перехідні процеси при плавному пуску асинхронного двигуна

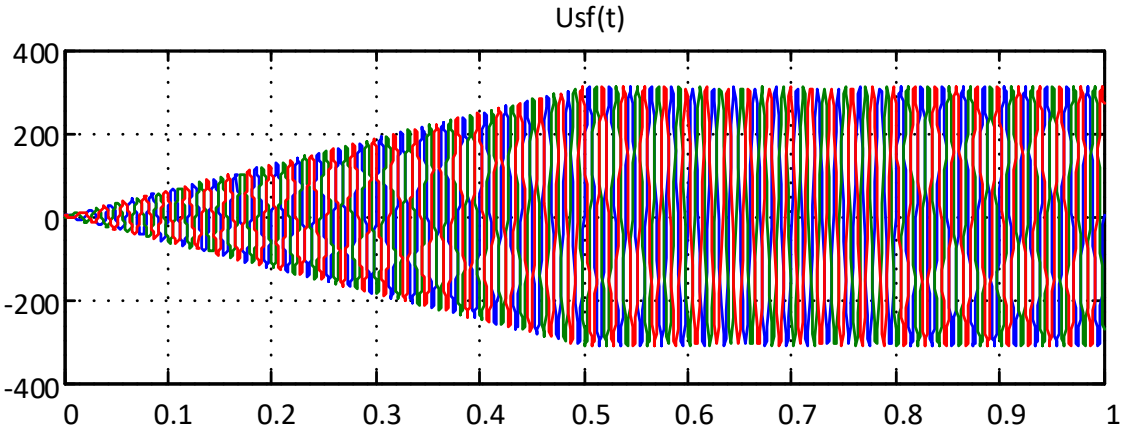


Рис. 17.17. Фазні напруги статора при плавному пуску АД

Плавний пуск АД та регулювання швидкості в невеликому діапазоні здійснюється тиристорними регуляторами напруги (системи ТРН-АД), які у сучасній технічній літературі називають пристроями плавного пуску (*Soft Starter*).

17.2.3. Моделювання асинхронного двигуна при скалярному частотному регулювання швидкості

При частотному регулюванні швидкості двигунів змінного стану застосовують плавну зміну як амплітуди, так і частоти фазних напруг статора [16]. Зазвичай частоту регулюють за лінійним законом, а відношення амплітуди до частоти змінюють за законами $U/f=const$, $U/f=const$ та $\sqrt{U}/f=const$. Для реалізації першого із цих законів достатньо в моделі рис. 17.11 перевести контакт ключа *Manual Switch* у нижню позицію. При цьому перехідні процеси координат АД набудуть вигляду рис. 3.14, 17.15.

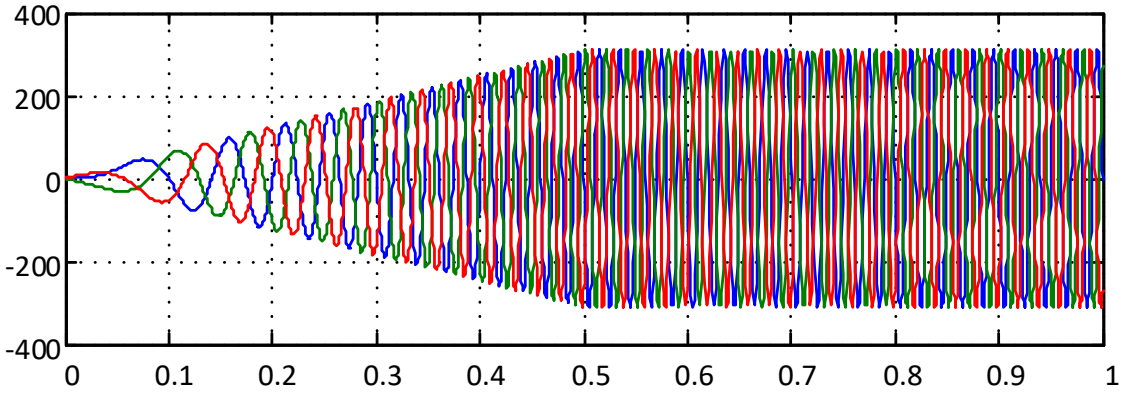


Рис. 17.14. Фазні напруги статора при частотному керуванні $U/f=const$

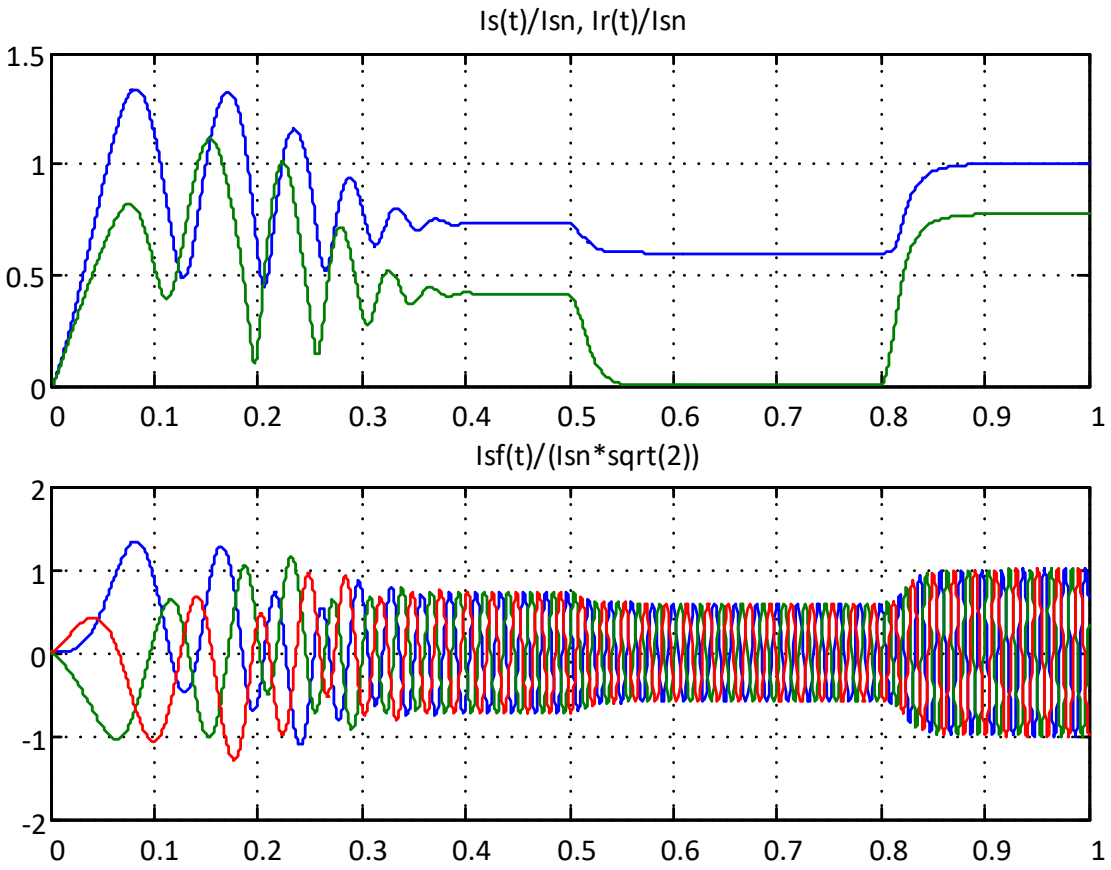


Рис. 17.15. Перехідні процеси в АД при частотному пуску $U / f = const$

17.2.4. Побудова статичних характеристик АД методом симуляції

За результатами симуляції розроблених моделей можна побудувати не тільки графіки перехідних процесів, а й статичні механічну $\omega(M)$ та швидкісну $I_s(M)$ характеристики АД (див. рис. 17.16, 17.17).

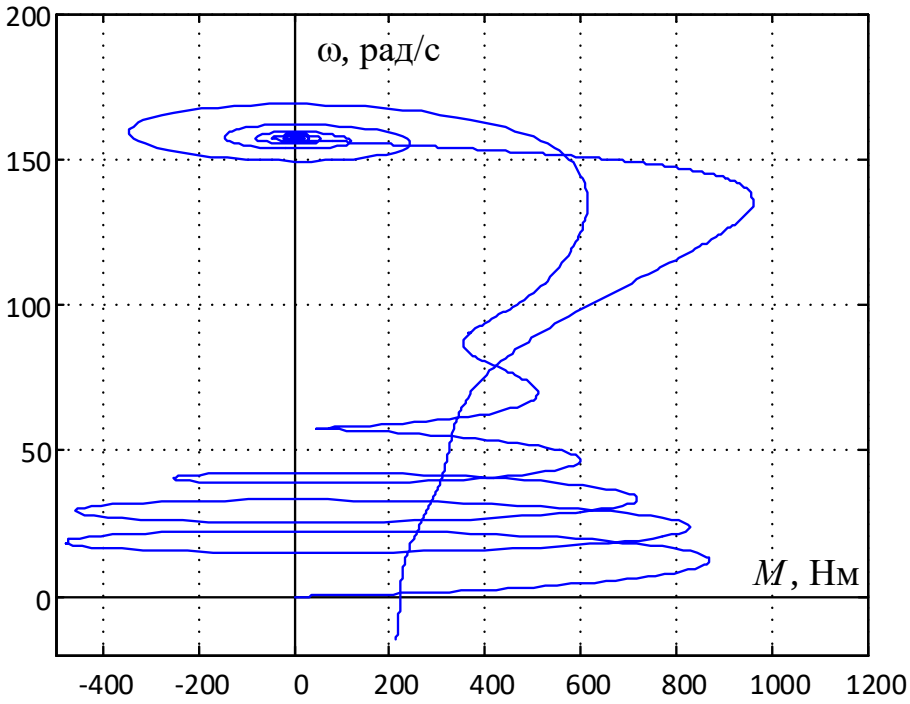


Рис. 17.16. Статична та динамічна механічні характеристики АД

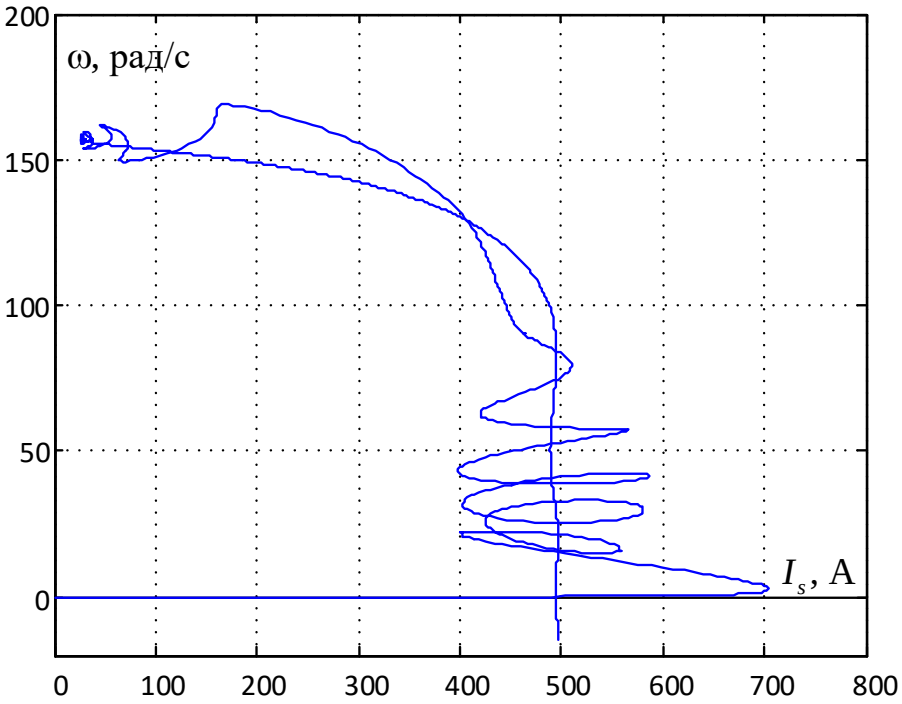


Рис. 17.15. Статична та динамічна швидкісні характеристики АД

Для побудови статичних характеристик достатньо змінювати момент навантаження не стрибком, а поступово, наприклад, лінійно [9]. Динамічні характеристики, які ще називають фазовими портретами отримані в процесі прямого пуску двигуна, а статичні – при повільній зміні моменту статичного опору за лінійним законом.

17.3. Використання *SPS*-моделей двигунів для дослідження системи «механічний вал»

Дослідження системи «механічний вал» за допомогою SPS-моделей електричних двигунів стало можливим після відокремлення моделей електромагнітної та механічної частин двигуна, та оснащення блоків механічним портом «швидкість» [5, 9]. Модель дводвигунної системи електромеханічної системи «механічний вал» на базі асинхронних двигунів подана на рис. 17.18 [5, 9].

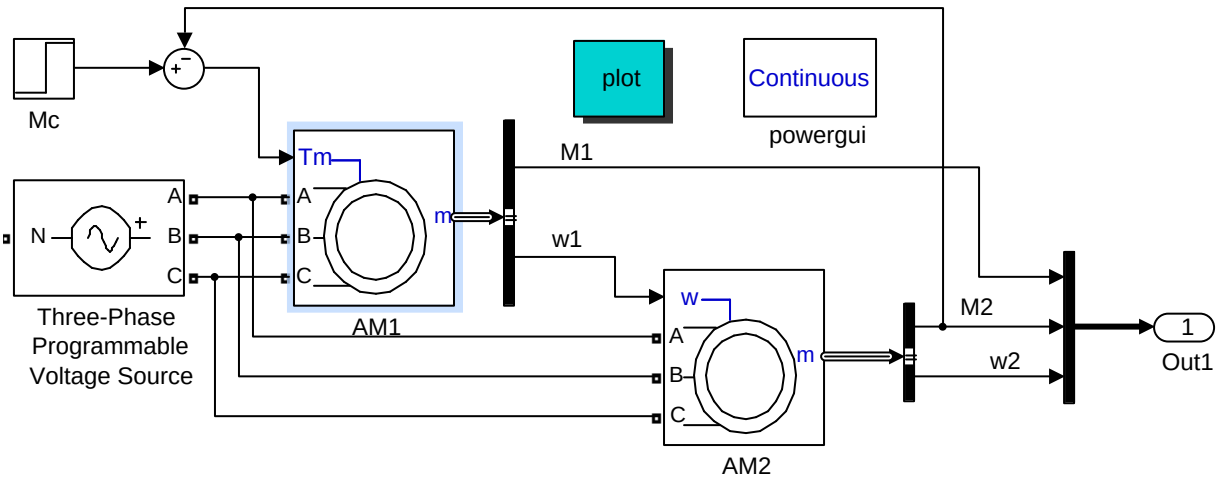


Рис. 17.18. *SPS*-модель системи «механічний вал» на базі асинхронних двигунів

Вона складається з двох асинхронних двигунів AM1 та AM2 з однаковими значеннями номінальної напруги та швидкості, але з різними номінальними моментами. Тому вони живляться від спільного трифазного SPS-джерела *Three-Phase Programmable Voltage Source*. Механічний зв'язок між двигунами здійснюється подачею швидкості w1 першого двигуна AM1, на механічний порт w другого двигуна AM2.

Спільний момент статичного опору Mc формується блоком *Step*, але перед подачею його на механічний порт Tm першого двигуна від нього віднімається електромагнітний момент M2, створений другим двигуном. Момент інерції машини зі швидкісним вхідним портом ігнорується і відноситься до першого двигуна. При чому перша частина цієї операції виконується автоматично, а друга частина має бути виконана власноруч користувачем. У такий спосіб моменти обох двигунів підсумовуються і разом діють на спільну механічну масу з моментом інерції $J_\Sigma = J_1 + J_2$ згідно з рівнянням руху:

$$M_1 + M_2 - M_c = M_j = J_\Sigma \frac{d\omega}{dt}, \tag{17.3}$$

де M_1, M_2 – електромагнітні моменти, створені першим AM1 та другим AM2 двигунами відповідно.

Дослідження системи «механічний вал» за допомогою моделі рис. 17.18 виконано для двигунів, обраних з використанням функції *Preset Model:* AM1 – 100 HP, 460 V, 60 Hz, 1780 RPM; AM2 – 50 HP, 460 V, 60 Hz, 1780 RPM. Перехідні процеси в досліджуваній системі при прямому пуску та при стрибкоподібному накиді навантаження $M_c = 1000$ Нм в момент часу 0,7 с зображені на рис. 17.19.

Аналогічний експеримент можна виконати і для двигунів постійного струму з незалежним збудженням. Модель для цього випадку подана на рис. 17.20.

Вона складається з двох двигунів постійного струму DCM1 та DCM2. Якорі машин з'єднані паралельно і живляться від спільного джерела Ua, а обмотки збудження мають індивідуальні джерела Uf1 та Uf2. Механічний зв'язок між двигунами здійснюється, як і в моделі рис. 17.18, подачею швидкості першого двигуна на механічний вхідний порт w другого двигуна. В якості досліджуваних двигунів за допомогою функції *Preset Model* обрано 2 однакових двигуна: 200 HP, 500 V, 1750 RPM з напругою збудження150 V.

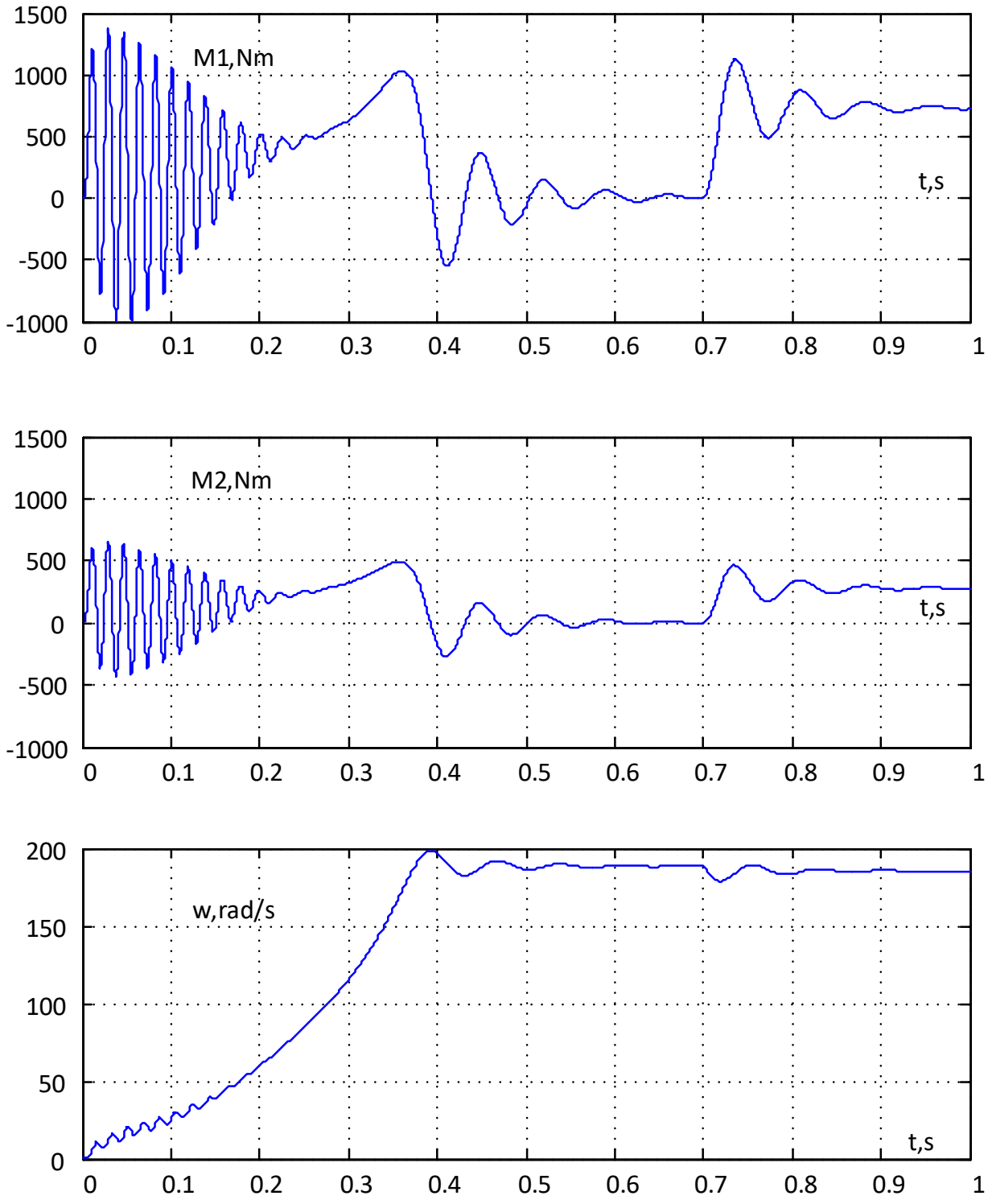


Рис. 17.19. Результати імітаційного моделювання
дводвигунної системи «механічний вал» на базі асинхронних двигунів

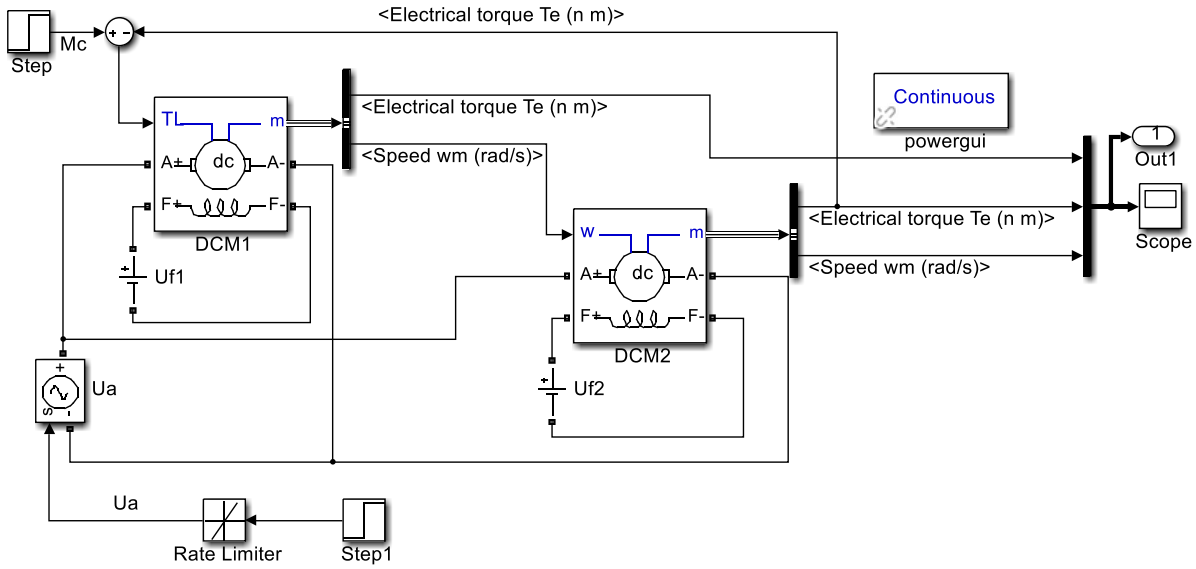


Рис. 17.20. *SPS*-модель системи «механічний вал» на базі двигунів постійного струму

Через те, що параметри навіть однакових двигунів можуть виявитися неузгодженими із-за неточностей виготовлення, різних умов експлуатації, в результаті виконання ремонтних робіт тощо, у другому двигуні навмисно змінили активний опір якоря. Отже, експеримент проводили при $R_{я1} = 0{,}058$ Ом, $R_{я2} = 0{,}08$ Ом [9]. Момент інерції першого двигуна згідно з методикою, викладеною для складання попередньої моделі, подвоїли. Графіки перехідних процесів для описаних вище умов зображені на рис. 17.21.

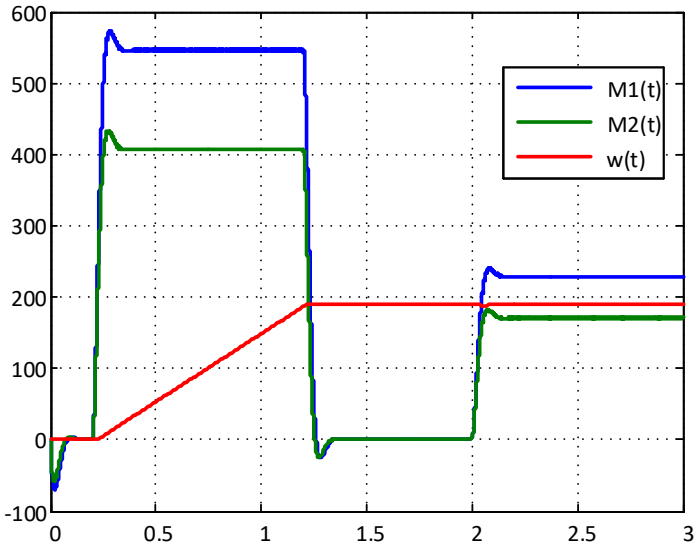


Рис. 17.21. Результати імітаційного моделювання системи «механічний вал» на базі двигунів постійного струму з різними активними опорами якорів

Ще більший вплив на розподілення навантажень чинить розузгодження потоків намагнічування [13]. Наприклад, на рис.11.5 наведені перехідні процеси в досліджуваній системі при зменшенні напруги збудження першого двигуна лише на 1В.

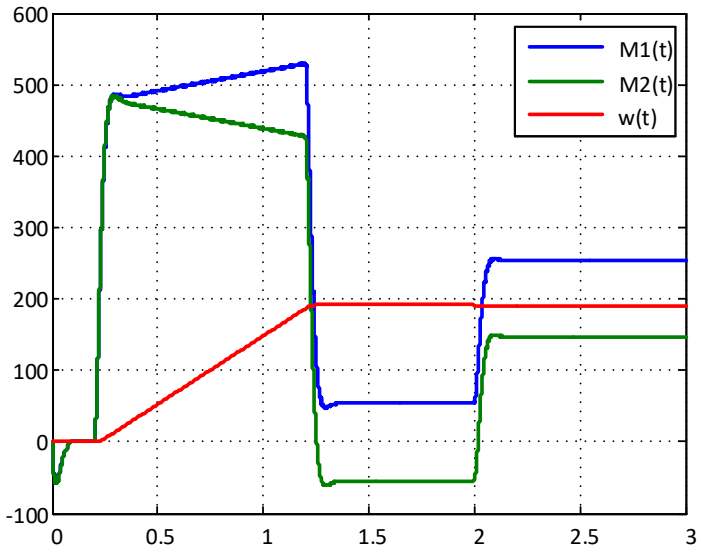


Рис. 17.22. Результати імітаційного моделювання системи «механічний вал» на базі двигунів постійного струму з різними напругами збудження

17.4. Використання SPS-моделі АД з *SimScape*-портом S для дослідження двомасової електромеханічної системи

Механічний *Simscape*-порт призначений для створення механічних навантажень за допомогою інших блоків бібліотеки *Simscape*, що мають механічні обертові порти.

На рис. 17.23 продемонстрована модель асинхронної машини з використанням механічного *Simscape*-порту для моделювання двомасової електромеханічної системи з урахуванням моменту тертя, що діє на другу масу (робочий орган виконавчого механізму) [9, 13].

Математичний опис механічної частини досліджуваної системи наведено у розділі 12.

Механічна частина моделі реалізована за допомогою блоків розділу *Mechanical* фундаментальної бібліотеки (*Foundational library*) інструментів

SimScape.

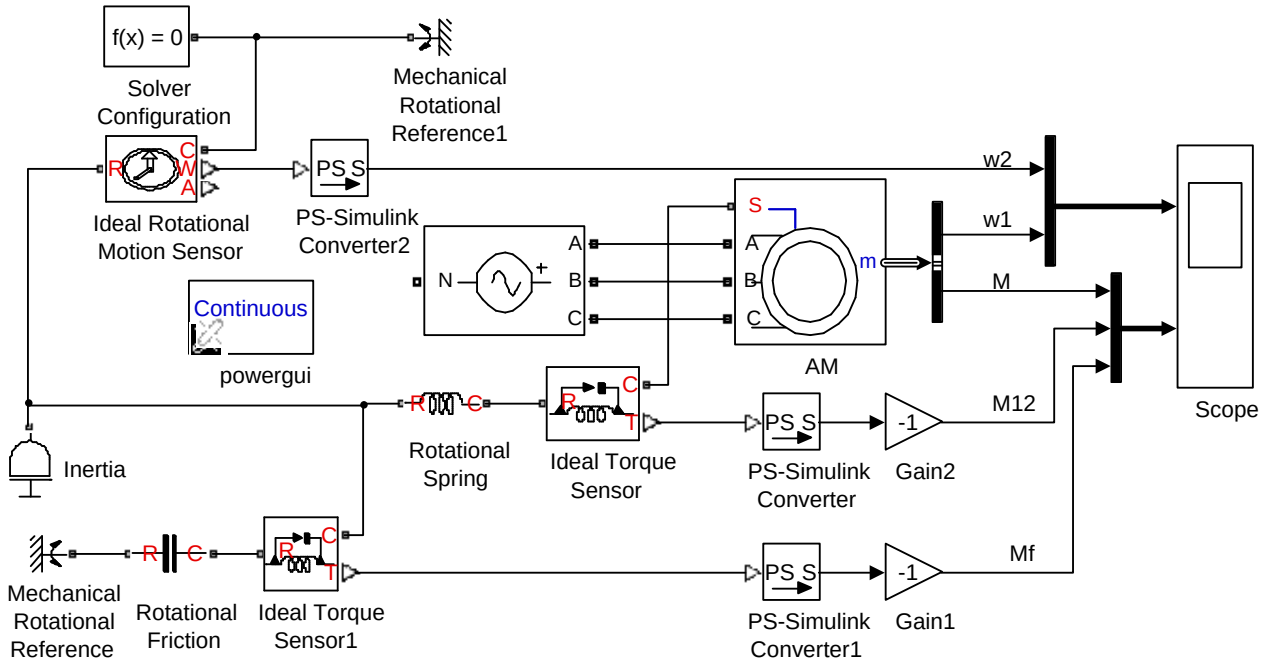


Рис. 15.23. Модель двомасової електромеханічної системи на базі АД, що реалізована за допомогою «електричних» та «механічних» блоків

В моделі рис. 15.23 друга маса представлена блоком *Inertia*, пружний вал – блоком *Rotational Spring*, а тертя – блоком *Rotational Friction* бібліотеки *Rotational Elements*. Блок *Mechanical Rotational Reference* прив'язує конкретну точку механічної системи до каркасу або до землі.

Блок *Rotational Friction* формує момент тертя за рівняннями (10.).

Перехідні процеси у двомасовій системі, отримані за допомогою моделі рис. 15.20 для двигуна потужністю 3,7 кВт з моментом інерції $J_1 = 0{,}02$ кг·м^2 і приєднаної до нього через пружний вал з коефіцієнтом жорсткості $c_{12} = 100$ Н·м/рад другої зосередженої обертової маси з моментом інерції $J_2 = 0{,}005$ кг·м^2 при врахуванні кулонівського тертя з ефектом Штрібека, приведені на рис. 15.24.

Для фіксації вихідних сигналів механічних *SimScape*-блоків у досліджувану модель підключені ідеальні датчики моменту (*Ideal Torque Sensor*) та руху (*Ideal Rotational Motion Sensor*). Усі механічні блоки мають механічні порти, позначені латинськими буквами R і C, причому передача моменту або

зусилля здійснюється в напрямку від порту R до порту C. Механічні датчики, крім цих портів, мають ще порти фізичних сигналів, позначені трикутниками та відповідними буквами: T – *torque* (момент), W – *angular velocity* (кутова швидкість), A – *angular displacement* (кутове положення).

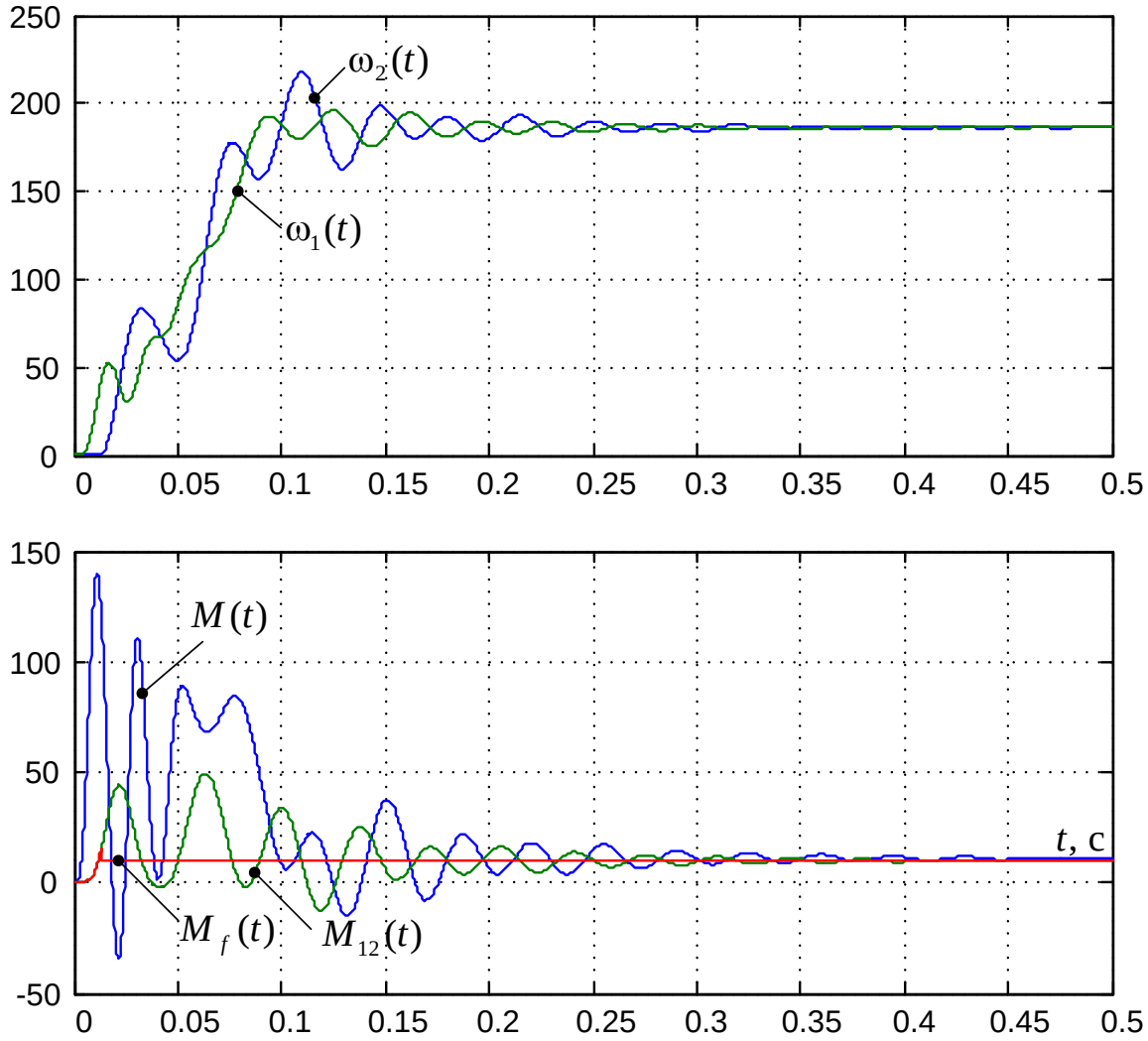


Рис. 17.24. Графіки перехідних процесів у моделі
двомасового електромеханічного об’єкта рис. 17.23

Фізичні сигнали після перетворення їх блоками *PS Simulink Converter* бібліотеки *Utilites* додатку *SimPowerSystem* можна використовувати для фіксації результатів засобами *Simulink*.

Ще однією з особливостей використання механічних *SimScape*-блоків є необхідність приєднання хоча б до одного з механічних портів блоку *Solver Configuration* бібліотеки *Utilites*.

17.5. Моделювання асинхронного двигуна при обриві фази статора

При експлуатації асинхронного двигуна можливе виникнення різноманітних нештатних ситуацій: обриви фаз, короткі замикання, тощо. Щоб розібратися, наскільки небезпечними є такі ситуації, до яких негативних наслідків і за який час вони можуть 9, як їх діагностувати та які міри треба застосувати до їх усунення, треба виконати їх дослідження. Враховуючи, що кожна з цих ситуацій може виявитися аварійною, основним методом дослідження таких явищ слід вважати математичне моделювання.

Розглянемо методику моделювання описаних вище процесів на прикладі обриву на ходу однієї з фаз статора.

Першим спонуканням при розв'язанні цієї задачі є думка про використання SPS-моделі АД, в якій одна з фаз ідеального джерела змінної напруги приєднується до фази АД через блок *Breaker*, керований блоком *Step*, як це показано на рис. 17.25 [9, 13].

Як видно, ця модель імітує живлення обмоток статора від трифазного джерела напруги за схемою «зірка без нульового проводу». При чому врахувати наявність нульового проводу у поданій моделі неможливо, тому що обмотки статора «приховані» в моделі асинхронної машини.

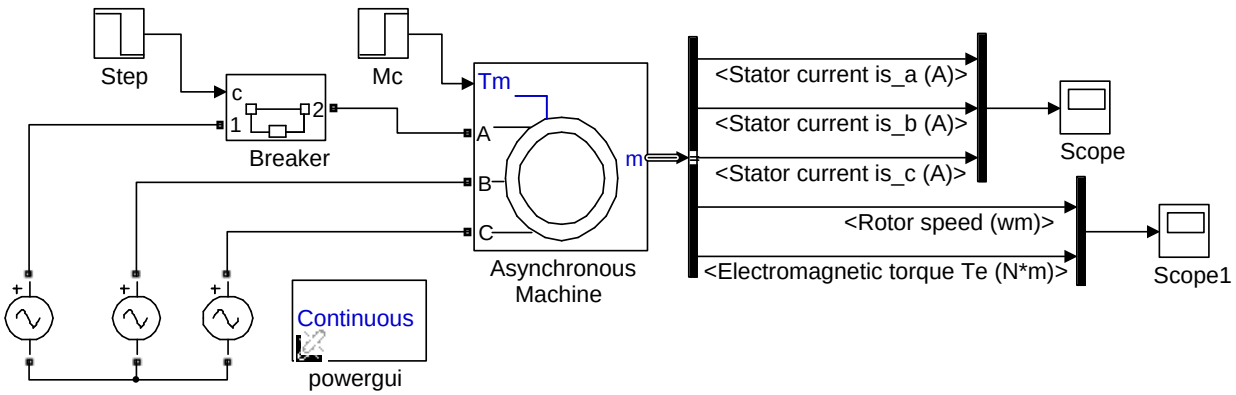


Рис. 17.25. *SPS*-модель процесу обриву фази АД при з'єднанні джерела напруги з обмотками статора за схемою «зірка без нульового проводу»

Графіки перехідних процесів, отримані за допомогою представленої

моделі зображені на рис. 17.26.

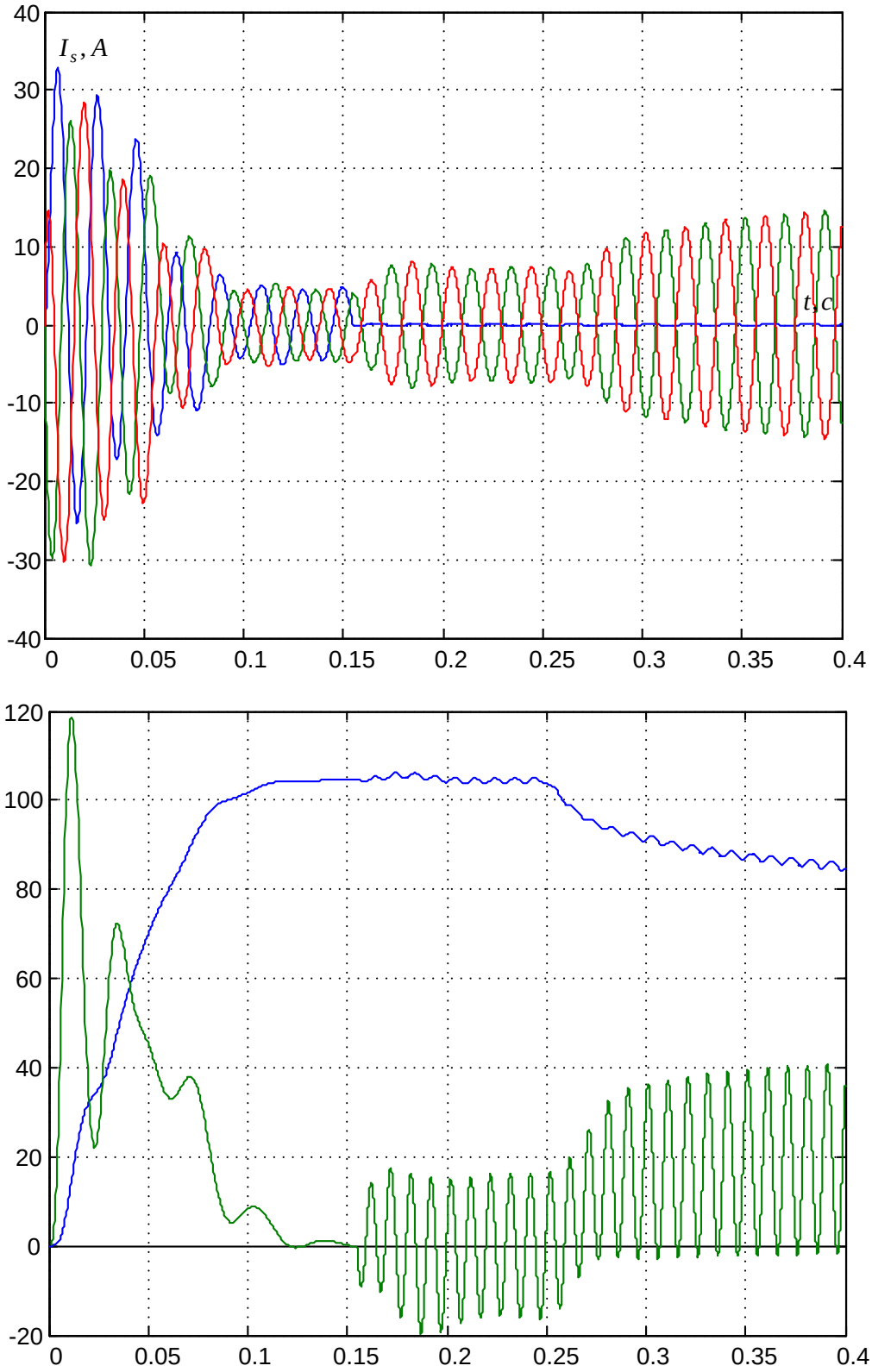


Рис. 17.26. Графіки перехідних процесів в АД при обриві фази А статора при живленні АД за схемою «зірка без нульового проводу»

Віртуальний експеримент організовано так, що спочатку двигун запус-

кається без навантаження у повнофазному режимі. При досягненні ним синхронної швидкості (в момент часу 0.15 с) відбувається обрив фази.

Як видно із приведених графіків, обрив фази статора приводить до того, що струми у двох інших фазах зростають за амплітудою, а фазовій зсув між ними стає рівним 180º, тобто струми знаходяться у проти фазі. Це зумовлено відсутністю нульового проводу.

Електромагнітний момент стає синусоїдальним з частотою 100 Гц, що удвічі перевищує частоту напруги живлення. Це пояснюється тим, що при переході до будь-якого несиметричного режиму в АД виникають напруги, струми і потокозчеплення прямої і зворотної послідовності, які при взаємодії подвоюють частоту електромагнітного моменту. Таку ж ситуацію можна спостерігати на графіках рис. 2.9, отриманих при асиметрії параметрів електромагнітного кола статора внаслідок міжвиткового короткого замикання у фазі А статора. Амплітуда коливань моменту близька до номінального значення, а амплітуда коливань струмів статора зростає приблизно на 60%. Це приводить до марних втрат електроенергії. Можна також показати, що в досліджуваному режимі зменшується критичний момент, і двигун може перекинутися при незначному навантаженні. Слід також відзначити, що при такому способі живлення двигуна обрив фази унеможливлює його запуск із зупиненого стану. Тому при можливості виникнення в АД неповнофазних режимів або асиметрії трифазного навантаження краще використовувати схему живлення з нульовим проводом.

Контрольні питання та завдання

1. Які параметри ДПС треба знати, щоб скористатися блоком *DC Machine*?
2. Як визначити параметр Laf двигуна?
3. Чи передбачає блок *DC Machine* врахування ефекту насичення сталі?

4. Які параметри АД треба знати, щоб скористатися блоком *Asynchronous Machine SI Units*?
5. Для чого і як використовують блок *Bus Selector*?
6. Як розрахувати діючі (середньо-квадратичні) та амплітудні значення фазових величин?
7. Як відрізнити на графіках перехідних процесів сигнали статора від сигналів ротора?
8. Які недоліки прямого пуску АД?
9. Як здійснюється плавний пуск АД? Як його промоделювати?
10. Як промоделювати скалярне частотне регулювання швидкості за законом $U / f = const$?

ЛІТЕРАТУРА

1. Островерхов М.Я., Пежов В.М. Моделювання електромеханічних систем в Simulink: навч. посібник [для студентів ВНЗ] / М.Я. Островерхов, В.М. Пежов. – К.: ВД «Стилос», 2008. – 528 с.

2. Лазарев Ю.Ф. Начала програмування в середовищі MatLAB: навч. посібник [для студентів ВНЗ] / – Ю.Ф. Лазарев. – К.: НТУУ КПІ, 2003. – 424 с.

3. Моделювання систем автоматичного керування. Методичні вказівки до лабораторних робіт: навч. посібник [для студентів ВНЗ] / О. І. Толочко. – Київ: КПІ ім. Ігоря Сікорського, 2023. – 250 с.

4. Solving control engineering problems with MATLAB (MATLAB curriculum series) / K. Ogata. – Englewood Cliffs, NJ: Prentice Hall, 1994. – 359 p.

5. Моделювання електромеханічних систем: навч. посібник [для студентів ВНЗ] / О.П. Чорний [та ін.]. – Кременчук: КДПІ, 1999. – 204 с.

6. Лозинський А., Мороз В., Паранчук Я. Розв'язування задач електромеханіки в середовищі пакетів MathCAD і MATLAB: навч. посібник [для студентів ВНЗ] / А. Лозинський, В. Мороз, Я. Паранчук. – Львів: видавництво Державного університету «Львівська політехніка», 2000. – 166 с.

7. Толочко О.І. Розробка складних електромеханічних систем в середовищі пакета MATLAB з використанням блоків додатка віртуального фізичного моделювання SimScape / О.І. Толочко // Вісник НТУ «ХПІ». Проблеми автоматизованого електроприводу. – Харків: НТУ «ХПІ», 2015, 12. – С. 118-123.

8. Математичні методи та особливості чисельних розрахунків динаміки електроприводів з асинхронними двигунами: монографія / О.П. Чорний, О.І. Толочко, В.К. Титюк. – Кременчук: ПП Щербатих О.В., 2016. – 300 с.

9. Толочко О.І. Моделювання електромеханічних систем. Математичне моделювання систем асинхронного електроприводу: навч. посібник [для

студентів ВНЗ] / О.І. Толочко; КПІ ім. Ігоря Сікорського. – Київ: НТУ «КПІ», 2016. – 150 с.

10. Проектування систем керування / Г.К. Гудвін, С.Ф. Гребе, М.Е. Сальгадо, 2004. – 911 с.

11. Dingyü Xue, YangQuan Chen, Derek P. Atherton. Linear Feedback Control Analysis and Design with MATLAB / Xue Dingyü та ін. – Philadelphia: Siam. – 2007. – 356 p.

12. Математичні методи в електромеханіці: навч. посібник [для студентів ВНЗ] / О. І. Толочко; КПІ ім. Ігоря Сікорського. – Київ: КПІ ім. Ігоря Сікорського, 2020. – 212 с.

13. Зеленов О.Б. Теорія електроприводу. Частина 1: навч. посібник [для студентів ВНЗ] / О.Б. Зеленов. – Алчевськ: ДонДТУ, 2005. – 394 с.

14. Божко С.В., Пересада С.М., Печеник М.В., Толочко О.І. Цифрове керування електромеханічними системами: підручник для студентів, які навчаються за спеціальністю 141 «Електроенергетика, електротехніка та електромеханіка» / С.В. Божко [та ін.]; КПІ ім. Ігоря Сікорського. – 2022. – 149 с.

15. Tolochko O., Palis S., Burmelov O., Kaluhin D. Discrete approximation of continuous objects with MATLAB / O. Tolochko, S. Palis, O. Burmelov, D. Kaluhin // International Journal of Science «Applied Aspects of Information Technology», Vol. 4, № 2, 2021. – Pp. 178-191.

16. Частотно-керовані асинхронні та синхронні електроприводи: навч. посібник [для студентів ВНЗ] / О.Г. Плахтина [та ін.]. – Львів: видавництво Національного університету «Львівська політехніка», 2002. – 228 с.

17. Шевченко І.С., Морозов Д.І. Електромеханічні системи в асинхронному електроприводі: навч. посібник [для студентів ВНЗ] / І.С. Шевченко, Д.І. Морозов. – Алчевськ: ДонДТУ, 2009. – 349 с.

ЗМІСТ