Л. Д. Костинюк, В. І. Мороз, Я. С. Паранчук

МОДЕЛЮВАННЯ ЕЛЕКТРОПРИВОДІВ

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Л.Д. Костинюк, В.І. Мороз, Я.С. Паранчук

МОДЕЛЮВАННЯ ЕЛЕКТРОПРИВОДІВ

Рекомендовано Міністерством освіти і науки України як навчальний посібник для студентів вищих навчальних закладів

Львів
Видавництво Національного університету "Львівська політехніка"
2004

УДК 62-83.001.57(075.8)
К 723
ББК 31.291 я-73

Рекомендовано Міністерством освіти і науки України
як навчальний посібник для студентів
вищих навчальних закладів
(лист № 14/18.2–2471 від 27.12.2002 р.)

Рецензенти:

Сопрунюк П.М., доктор технічних наук, професор, завідувач відділу, Фізико-механічний інститут ім. Г. В. Карпенка;

Щур І.З., доктор технічних наук, професор, Національний університет "Львівська політехніка";

Василів К.М., кандидат технічних наук, доцент, Український державний лісотехнічний університет.

Костинюк Л.Д. та ін.

К 723 Моделювання електроприводів: Навч. посібник / Л.Д. Костинюк, В.І. Мороз, Я.С. Паранчук. – Львів: Видавництво Національного університету "Львівська політехніка", 2004. – 404 с.

ISBN 966-553-429-7

Висвітлено математичне, структурне та цифрове моделювання автоматизованих електроприводів. Зроблено короткий огляд і порівняння поширених середовищ програмування Microsoft Quick Basic і Turbo Pascal та широко використовуваних математичних пакетів MathCAD і MATLAB. Розглянуто моделювання елементів автоматизованих електроприводів. Подається огляд аналітичних підходів та числових методів для розв'язування звичайних диференціальних рівнянь, що описують динаміку електроприводів постійного та змінного струмів. Розглянуто моделювання основних структур автоматизованих електроприводів. Викладення матеріалу супроводжується прикладами.

Навчальний посібник призначений для студентів спеціальностей 7.092201 "Електричні системи та комплекси транспортних засобів" і 7.092203 "Електромеханічні системи автоматизації та електропривод", а також інших електротехнічних спеціальностей вищих технічних навчальних закладів та інженерно-технічних працівників.

ББК 31.291 я-73

ISBN 966-553-429-7

ВСТУП

"Оце і є та проблема, Пацю, на розв'язання якої я просила Пуха спрямувати свої Розумові Здібності".

(А. Мілн "Вінні-Пух та його друзі" у перекладі з англ. Л. Солонька)

Розв'язування задач моделювання електроприводів вимагає як розуміння власне електропривода (*об'єкта, що моделюється*), так і достатніх знань у галузі математичного моделювання та числових методів (*для розв'язування систем рівнянь, що подають математичні моделі*) і комп'ютерних технологій (*інструмент для реалізації*). На жаль, у книжкових магазинах полиці не перевантажені такою літературою (за винятком хіба що літератури з комп'ютерних технологій), тому для розв'язання задач такого класу досліднику доводиться здійснювати пошук необхідної інформації за багатьма різними джерелами: в одних описано моделі елементів електроприводів та їх систем, у других – числові методи, а в третіх – мови програмування і математичні пакети. Часто перед інженером чи дослідником постають проблеми: який числовий метод вибрати (*їх багато, а хочеться найкращий*), в якому середовищі виконувати розрахунки (*Turbo Pascal, Quick Basic, MATLAB, а друзі ще й радять Borland C++, а ще інші кажуть, що простіше знайти в Internet готову програму мовою FORTRAN*). Напевно, остаточної, вичерпної та для всіх прийнятної відповіді на усі запитання пропонована книга не дасть. Проте автори сподіваються, що певною мірою їм вдалося досягти задуманого – отримати не тільки посібник, але й до деякої міри довідник, яким можна користуватися не тільки для навчання. Саме тому багато прикладів було доведено до остаточної реалізації в одному чи у декількох середовищах з таких міркувань:

- показати створену комп'ютерну модель на конкретному прикладі;
- відобразити реалізацію комп'ютерної моделі у різних середовищах (мови програмування і математичні пакети), для того, щоби користувач зміг

оцінити недоліки та переваги, простоту і зрозумілість, природність і наочність тієї чи іншої реалізації.

Для поєднання вищезгаданих аспектів моделювання замало однієї книжки, тому що кожен аспект і сьогодні до кінця не досліджений і кожен з них ще розвивається. До того ж ще й об'єкти досліджень постійно вдосконалюються, змінюються тенденції та пріоритети їх розвитку. Тому в посібнику подано доволі повний список літератури, в якому можна глибше познайомитися з тими чи іншими невисвітленими питаннями.

Рівень складності розв'язуваних задач визначається потужністю комп'ютера, а наявне програмне забезпечення дає змогу полегшити розв'язування цієї задачі – тобто рівень складності розв'язуваних задач визначається рівнем розвитку обчислювальної техніки, тому сучасні підходи до моделювання електроприводів докорінно відрізняються від застосовуваних у 70–80-х роках [Б.1–Б.3, Б.8]. З прогресом як у царині розробки апаратної частини комп'ютерів (*hardware*), так і їх програмного забезпечення (*software*), а також з появою нових числових методів та підходів до розв'язання задач зростають можливості розв'язування складних задач моделювання, що потребує нових поглядів на викладання навчальної дисципліни "Моделювання електроприводів".

Поява об'єктно-орієнтованих мов і відповідних середовищ програмування полегшила створення складних швидкодійних моделей та їх подальший розвиток і використання, а поява математичних пакетів (таких, як MathCAD чи MATLAB) дала змогу відчутно спростити створення моделей, зокрема з використанням аналітичних методів, які мають безсумнівну перевагу над числовими, а також дала змогу уникнути необхідності розроблення своїх процедур розв'язування систем диференціальних рівнянь, що описують модель, оскільки вони входять до складу практично кожного математичного пакета. Сучасний персональний комп'ютер завдяки наявному сучасному програмному забезпеченню став незамінним інтелектуальним партнером інженера чи дослідника, з яким працювати – суцільне задоволення, але не можна забувати про заклики Р. Хеммінґа (*R. Hamming*) [В.12]:

- *"Мета розрахунків – розуміння, а не число".*
- *"Перед розв'язуванням задачі подумай, що робити з її розв'язком".*

Загалом моделювання передбачає створення подібної до оригінала копії об'єкта. Моделювати можна об'єкти чи системи в дуже широкому діапазоні – від елементарних ланок до складних електромеханічних систем з інтелектуаль-

ним керуванням. Залежно від об'єкта, а також мети моделювання розрізняють **геометричне, фізичне** і **математичне** моделювання.

Геометричне моделювання передбачає побудову геометричної копії відповідного масштабу і має, як правило, демонстраційне призначення. Таке моделювання найчастіше застосовують в архітектурі, коли створюються демонстраційні моделі будівель або й цілих мікрорайонів. В автомобілебудуванні геометричне моделювання використовується, коли створюються моделі автомобілів спочатку зменшеного масштабу, а потім натурального розміру під час проектування форми кузова.

Фізичне моделювання має за мету дослідження поведінки об'єктів моделювання на моделях в умовах максимального наближення до природних, тобто таких, в яких функціонує (експлуатується) оригінал. Наприклад, дослідження поведінки моделей літальних апаратів в аеродинамічних трубах чи плавальних апаратів у водних басейнах. Тому фізичні моделі повинні відтворювати не тільки геометричні пропорції оригінала, але й ті фізичні властивості, які досліджуються.

Ці два типи моделей потребують матеріальних витрат на їх створення, а останні ще й на виконання досліджень.

Математичне моделювання передбачає створення математичних моделей – систем рівнянь, які описують поведінку об'єкта, який моделюється. Метою моделювання електромеханічних систем є дослідження динамічних і статичних властивостей систем загалом чи окремих їх елементів (ланок). Тому математичним апаратом тут є диференціальне числення, Z-перетворення, перетворення Лапласа, а математичні моделі – це передатні функції та системи диференціальних чи різницевих рівнянь.

Першим етапом математичного моделювання є вибір найвдалішої математичної моделі. В електромеханічних системах електроприводи здебільшого подаються структурними схемами, за якими можна створити математичну модель – систему рівнянь, чи іншими математичними засобами, за якими можна побудувати алгоритм. Тут враховується характер досліджень, який може вплинути на складання математичної моделі.

На **другому етапі** створюється саме математична модель – система рівнянь і зводиться до придатного для її розв'язування вигляду. Якщо це система диференціальних рівнянь, то їх зводять до системи рівнянь у нормальній формі Коші. Якщо використовується Z-перетворення, то отримують систему рекурентних (різницевих) рівнянь.

Третій етап – написання програми для здійснення цифрового (комп'ютерного) моделювання. Програма – це цифрова модель. Найчастіше проблеми виникають під час переходу від другого етапу до третього, коли вибирають метод числового розв'язування диференціальних рівнянь, задання початкових умов, реалізацію нелінійностей тощо. Останнім часом з появою спеціалізованих (наприклад, САПР) і математичних пакетів робота на третьому етапі відчутно спрощується, зокрема з використанням програм, що реалізують візуальне структурне моделювання.

Правильно спроектована математична модель повинна відображати (у певних межах) динамічні властивості досліджуваного процесу. Межі ефективного моделювання визначаються допущеннями, що прийняті під час моделювання. Під час моделювання потрібно мати перелік усіх допущень і визначити їх вплив на результат моделювання, особливо звернути увагу на відсутність чи наявність в електромеханічній системі люфтів, зон нечутливості, пружних зв'язків, нелінійних характеристик елементів, допустимих перевантажень тощо. У зв'язку з цим систему можна розглядати як лінійну чи нелінійну. Найчастіше ми будемо розглядати лінійні системи або лінеаризовані в околі певних (робочих) точок, хоча іноді умови нелінійності будуть враховуватись.

У посібнику поруч з текстом для покращання сприйняття вжито позначення, показані нижче.

Корисне повідомлення

Зауваження, на яке варто звернути увагу

Пропонується текст процедури чи програми, що може бути використана як стандартна

Важлива інформація

Автори вважатимуть своє завдання виконаним, якщо запропонована книга буде корисною не тільки як посібник, але й слугуватиме свого роду довідником, у якому можна знайти необхідне для моделювання досліджуваної системи електропривода, а пропоновані приклади використати як основу для розробки потрібних моделей. Саме тому викладення матеріалу супроводжується прикладами і, за

можливості, автори намагалися подати розв'язання конкретної задачі у всіх середовищах, інколи ж обмежувалися одним-двома, вважаючи у такому разі, що у цьому середовищі процедура розв'язування такої задачі є для користувача найпростішою чи наочнішою. А взагалі програмна "багатомовність" користувача персонального комп'ютера (як і в реальному житті) йде лише йому на користь, даючи змогу розв'язувати кожну задачу в тому програмному пакеті, можливості якого найкраще відповідають особливостям поставленої задачі. Для цього у посібнику наведено цілу низку прикладів розв'язування задач моделювання у різних середовищах (перед кожним прикладом вказано назву програмного пакета):

Quick Basic	*Приклад з використанням Quick Basic*
Turbo Pascal	*Приклад з використанням Turbo Pascal*
MathCAD	*Приклад із застосуванням MathCAD*
MATLAB	*Приклад з використанням MATLAB*
Simulink	*Приклад із застосуванням MATLAB + Simulink*
Power System Toolbox	*Приклад з використанням MATLAB + Simulink і додаткового пакета Power System Toolbox*

Це дає змогу оцінити природність і елегантність реалізації моделі (*автори вважають, що таке поняття існує*) в тому чи іншому середовищі, зручність та оперативність налагодження моделі, витрачений на підготовку до розв'язування задачі час і швидкість її розв'язування.

Для полегшення сприйняття текстів програм застосовано їх шрифтове оформлення, яке, до речі, сьогодні широко використовують сучасні програмні системи (*наприклад, Delphi чи Visual Basic*):

`основний текст`	– звичайний прямий шрифт;
`ключові слова`	– потовщений прямий шрифт;
`текстові константи і змінні`	– потовщений курсив;
`коментарі`	– курсив.

Згідно з наявними сучасними юридичними нормами і вимогами стосовно програмних продуктів та літератури з комп'ютерної тематики, автори і видавництво повідомляють, що не відповідають за можливі помилки в тексті книги, за помилки користувачів під час вивчення і освоєння матеріалу, поданого в посібнику, а також за можливі моральні та економічні збитки, які можуть бути наслідком помилок, неповного розуміння матеріалу книги чи невдалих дій.

Автори готові до дискусій, будуть вдячні за всі зауваження стосовно книги і чекають відгуків на адресу:

Кафедра "Електропривід і автоматизація промислових установок",
Інститут енергетики та систем керування,
Національний університет "Львівська політехніка",
79013, Львів, вул. С. Бандери, 12,
Україна

(032)-258-26-20 (*кафедра*)
(032)-258-25-64 (*лабораторія обчислювальної техніки*)

E-mail: `vmoroz@polynet.lviv.ua, vmoroz58@yahoo.com`
`yparanchuk@yahoo.com`

З повагою, автори

1 ПРОГРАМНІ ЗАСОБИ

У цьому розділі подається короткий опис двох середовищ програмування (Turbo Pascal і Microsoft Quick Basic – починаючи з середніх шкіл ці алгоритмічні мови є основними для вивчення в інформатиці) і двох математичних пакетів (MathCAD і MATLAB), які можуть бути використані для комп'ютерного моделювання електроприводів. Вибираючи їх, автори враховували особисті враження і уподобання, тому їх погляд є певною мірою суб'єктивним. Зокрема, у розгляд не потрапили такі мови програмування, як:

- **C++** – для непідготованого користувача доволі складна в освоєнні та роботі мова програмування; автори після числових експериментів та тестів переконані, що для розв'язування задач моделювання з такою самою ефективністю можна з успіхом використовувати і Turbo Pascal. Крім того, програми з Turbo Pascal і Quick Basic просто переводяться у C++ (але не завжди навпаки).
- **FORTRAN** – застаріла мова програмування, яку не рятують навіть найновіші версії, хоча й досі в Internet є величезні бібліотеки стандартних FORTRAN-процедур, що реалізують різні числові методи.
- **Ada** – повна версія цієї мови дуже складна, а середовища програмування не набули такого поширення, як мови BASIC чи Pascal.

Суперечки про переваги та недоліки тієї чи іншої мови програмування ведуться з моменту виникнення перших засобів автоматизації програмування і тривають досі. Але здебільшого алгоритмічні мови високого рівня мають доволі схожі можливості, а функції, яких немає на операторному рівні, переважно можуть бути реалізовані програмно. Практика показує, що важливішими для ефективного розв'язування задачі є ефективність транслятора з алгоритмічної мови та вибір раціонального числового методу і алгоритму.

Від авторів

Необхідно зауважити, що в більшості середніх шкіл навчальний курс інформатики, зокрема, розділ "алгоритмічні мови та програмування" читають на підставі алгоритмічних мов Quick Basic (QBasic) і Turbo Pascal, що робить їх широковідомими і популярними серед користувачів персональних комп'ютерів. Крім того, програми з цих мов, як вже згадувалося вище, нескладно перевести у C++, FORTRAN та інші мови програмування. Також дається взнаки давня прихильність авторів до середовищ Quick Basic і Turbo Pascal (хто з ними вже працював, той зрозуміє).

Важливим у реалізації комп'ютерної моделі є не тільки раціональний вибір програмного середовища для її розв'язування, а й дотримання певних правил і заходів під час безпосереднього складання програми чи комп'ютерної моделі. Це пов'язано з тим, що, як показує практика, більшість часу (до 80–90 %) йде на пошук помилок у програмі та подальше коригування комп'ютерної моделі, тому відповідна комп'ютерна реалізація (наприклад, програма алгоритмічною мовою чи документ MathCAD) повинна легко читатися і бути зрозумілою. Це проста істина, але більшість користувачів її не дотримуються, і тому створені ними комп'ютерні моделі дуже часто стають загадками не тільки для оточуючих, але через місяць-другий і для самих авторів.

Зарадити цьому може низка простих правил, яких автори наполегливо радять дотримуватися під час складання всіх комп'ютерних моделей (не лише тривалого користування, а й тимчасових, які, **по-перше**, як свідчить практика, застосовуються значно довше, ніж планувалося, і, **по-друге**, для того, щоб постійно дотримуватися належного стилю – "*не втрачати форми*"). Автори вважають, що у пропонованих в посібнику прикладах використано правильний стиль програмування і вони є одним з можливих варіантів для наслідування.

Правила прості, їх сформульовано в літературі, що стосується структурного і модульного програмування [список літератури, розділ А]:

1) Задачу необхідно поділити на окремі логічно завершені підзадачі (алгоритми) – модулі, з яких потім будують цілу задачу (*модульний принцип*).
2) Структура кожного модуля (частини задачі) повинна бути простою, послідовною, без заплутаних переходів.
3) У тексті програми чи документа повинні бути коментарі, що пояснюють модель та хід обчислень.

Коментарів у моделі забагато не буває!

Використання мов програмування для створення комп'ютерної моделі потребує більше часу для користувача, бо вимагає і ґрунтовного пророблення алгоритму (як моделі, так і числового методу), і його реалізації в середовищі алгоритмічної мови, і налагодження отриманої комп'ютерної моделі, але обчислення є швидшими, ніж у середовищі математичного пакета. Більше того, автори можуть навести низку прикладів, у яких комп'ютерна модель, що реалізована за допомогою мови програмування, є простішою та зрозумілішою (і, безумовно, швидшою) за створену, наприклад, у середовищі Simulink. Проте нині, коли наявні потужні персональні комп'ютери і розвинені математичні пакети, які хоч і розв'язують задачу довше, ніж відповідна реалізація за допомогою транслятора з алгоритмічної мови, але зате відчутно економлять особистий час користувача. Тому сьогодні саме використання математичних пакетів є тим раціональним підходом у комп'ютерному моделюванні електроприводів, якого варто дотримуватися.

1.1. Середовище програмування Microsoft Quick Basic

Мова BASIC є зручною для початкового освоєння програмування та роботи з програмами і використовується практично на всіх моделях комп'ютерів, зокрема, на IBM PC-сумісних комп'ютерах. Середовище програмування для BASIC входить у склад операційної системи MS DOS (до версії 6.22 включно). Мовою BASIC можна складати програми для розв'язування різноманітних задач обчислювального характеру, а також інформаційно-пошукових задач, тобто задач опрацювання інформації, заданої, наприклад, у вигляді таблиць тощо. Вона має велике коло палких прихильників (досить глянути на сторінки світової павутини WWW в Internet: (**www.qb45.com**, **www.qbasic.com**). Опоненти мови BASIC, можливо, недостатньо обізнані з можливостями сучасних версій, тому ще й досі висловлюють міркування про недоцільність використання цієї мови для навчання учнів та студентів і застосування в інженерно-технічних розрахунках. Сьогодні серед користувачів мови BASIC найбільшою популярністю користуються програмні продукти фірми Microsoft (**www.microsoft.com**).

Основна перевага Quick Basic (*швидкий Basic*) – це зручність у роботі. Quick Basic фірми Microsoft Corp. – це так назване інтегроване середовище, тобто програма, що містить всі необхідні засоби (транслятор, текстовий редактор, редактор зв'язків, систему налагодження та допомоги) для автоматизації програмування мовою BASIC. Це легка в освоєнні та користуванні багатовіконна система зі зручним меню та розвиненими засобами допомоги, підтримкою різних типів дисплеїв, потужними засобами налагодження програм з контролем синтаксису на стадії введення програми та потужним текстовим редактором.

Quick Basic забезпечує користувачу набагато більші зручності порівняно з попередніми версіями мови завдяки наявності:

- можливості не нумерувати рядки програми;
- блока умовного оператора IF/THEN/ELSE/ENDIF;
- процедурних блоків:
  - SUB – для підпрограми;
  - FUNCTION – для функції;
- конструкції SELECT CASE (вибір одного варіанта з багатьох);
- циклів типу DO ... LOOP;
- різноманітних типів даних: довгих цілих, дійсних різної довжини та діапазону, структурованого типу даних;
- розділу описання констант;
- різноманітних режимів налагодження програми:
  - покрокового і автоматичного;
  - з візуалізацією трасування програми;
  - з встановленням точок переривання програми;
  - редагування з продовженням виконання;
- ефективного текстового редактора середовища Quick Basic:
  - з контролем синтаксису мови і з діагностикою помилок;
  - з корисними для запобігання помилки підказками;
  - з багатовіконною системою відображення;
  - зі зручною системою меню і підтримкою маніпулятора "мишка";
  - сумісність за клавішними командами з текстовим редактором Word Star, що вважається фактичним стандартом для текстового редактора MS DOS;
  - розвиненої інформаційної допомоги (Help), яка доступна у будь-який момент.

Середовище Quick Basic підтримує також й інші режими, що бажані для повноцінної роботи:

- включення до програми додаткових файлів, наприклад, підпрограм і функцій, командою INCLUDE, що зручно під час розробки великих програм з поділом їх на модулі;
- створення бібліотек та виконуваних (EXE) файлів.

Свого часу найпоширенішою стала версія Quick Basic 4.5, яка відрізняється від попередніх версій розширеною допомогою, додатковими бібліотеками, покращеним інтегрованим середовищем і в повній версії займає на диску до 1,5 МБайт. Версія 6.0 виявилась непопулярною. У 1989 році у фірмі Microsoft створили версію 7.0, а наступного року – версію 7.1, інсталяційний варіант якої займає чотири дискети по 1,44 МБайт, а в розгорнутому вигляді на диску – майже 10 МБайт. Загальний обсяг файлів допомоги для цієї системи становить 1,6 МБайт, окрім цього, файли прикладів займають понад 1,5 МБайт. Про якісно новий рівень пакета свідчить його назва – Microsoft BASIC Professional Development System (*система для професійних розробок*).

Зовнішній вигляд екрана після запуску середовища Quick Basic 4.5 показано на рис. 1.1.

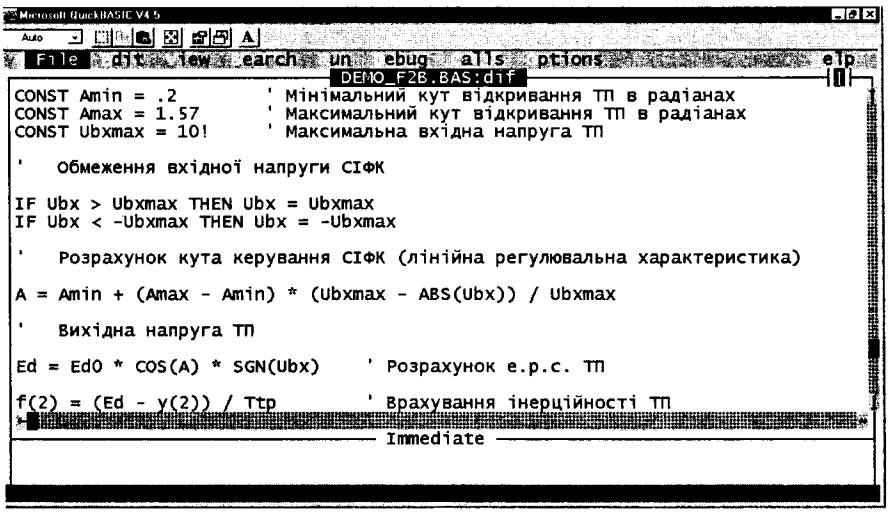


Рис. 1.1. Зовнішній вигляд робочого вікна середовища Quick Basic

Нижче подано основні типи числових даних і основні оператори керування Quick Basic, що допоможе зорієнтуватися в можливостях середовища, а детальнішу інформацію можна отримати з літератури [список літератури, розділ А] чи допомоги середовища.

Основні типи числових даних:

тип **INTEGER** – цілий короткий, займає у пам'яті 2 байти (16 біт). Використовується для подання значень величин з діапазону від – 32768 до 32767;

тип **LONG** – цілий довгий, займає у пам'яті 4 байти. Застосовується для подання значень величин з діапазону від – 2147483648 до 2147483647;

тип **SINGLE** – дійсний звичайної точності, займає у пам'яті 4 байти. Використовується для подання значень величин з діапазону $2.87 \cdot 10^{-45} < |x| \leq \leq 3.47 \cdot 10^{+38}$ у форматі з плаваючою комою $\pm m \cdot E \pm p$, де m – мантиса числа (6–7 значущих цифр), а p є порядком;

тип **DOUBLE** – дійсний подвійної точності, займає у пам'яті 8 байт. Використовується для подання значень величин з діапазону $4.947 \cdot 10^{-324} < |x| \leq 1.797 \cdot 10^{+308}$ у форматі з плаваючою комою $\pm m \cdot D \pm p$. Мантиса m має 15–16 значущих цифр.

Основні керуючі оператори мови:

IF ... THEN ... ELSE – умовний оператор, що може розміщуватись як в одному, так і в багатьох рядках;

SELECT CASE – оператор вибору одного варіанта з багатьох, в якому можна задавати логічні відношення, діапазон чи окреме значення;

FOR ... NEXT – оператор циклу з наперед заданим кроком (зокрема й дробовим) і відомою кількістю повторень;

DO ... LOOP – універсальний оператор циклу, в якому допускається перевірка умови виконання (**WHILE**) чи закінчення (**UNTIL**) циклу як на початку, так і в кінці циклу;

SUB – оператор підпрограми;

FUNCTION – оператор функції.

1.2. Середовище програмування Turbo Pascal

Turbo Pascal фірми Borland, Inc. (адреса в Internet **www.borland.com**) є одним з найпоширеніших і найпопулярніших середовищ програмування у світі. Завдяки фірмі Borland мова Pascal, яку упродовж 1968–1971 рр. створив у Цюріхському інституті інформатики швейцарський вчений Ніклас Вірт, перетворилася зі суто навчальної мови у потужну, але компактну професійну систему, що дає змогу розв'язувати задачі широкого діапазону: від найпростіших навчальних до найскладніших систем проектування та реляційних баз даних.

Популярність Turbo Pascal зумовлена такими чинниками:

- мова Turbo Pascal у природній формі підтримує сучасні концепції розробки програм; структурне і модульне програмування, а починаючи з версії 5.5 введено підтримку об'єктно-орієнтованого програмування (ООП);
- завдяки компактності мова легка в освоєнні;
- незважаючи на відносну простоту, Turbo Pascal придатний для розробки навіть дуже великих і складних систем;
- вдала реалізація фірмою Borland середовища програмування Turbo Pascal:
  - система Turbo Pascal є інтегрованим середовищем, що містить потужний текстовий редактор, компілятор, редактор зв'язків, символьний налагоджувач; підтримується багатовіконний інтерфейс і наявна розвинена система допомоги;
  - компактний компілятор Turbo Pascal інтегрований з інтелектуальним редактором зв'язків, що разом забезпечують широкий набір локальних оптимізацій, і тому користувач отримує компактний ефективний код програми.

Середовище Turbo Pascal вже з перших версій стало улюбленим для багатьох користувачів персональних комп'ютерів і "законодавцем мод" серед середовищ програмування завдяки зазначеним вище властивостям. Вже з перших версій Turbo Pascal вирізнявся компактністю і зручністю: для популярної свого часу версії 3.0 вистачало п'ятидюймової дискети обсягом 360 кБайт; для розміщення інсталяційного варіанта версії 5.0 було достатньо трьох таких дискет, а інсталяція першої об'єктно-орієнтованої версії 5.5 повністю поміщалася на одній дискеті обсягом 1,2 МБайта. Популярна досі версія Turbo Pascal 7.0 встановлюється з двох дискет обсягом 1,2 МБайта кожна.

Також була розроблена версія Turbo Pascal для середовища Windows (Turbo Pascal for Windows). Обидві останні версії (для DOS та Windows) були об'єднані в єдиний пакет для професійних програмістів – Borland Pascal 7.0. Розвиток версій для DOS закінчився на версії 7.1, яка, на відміну від попередніх версій, що постачалися на дискетах (до версії 7.01 включно), вийшла вже у варіанті на компакт-диску.

Сьогодні Turbo Pascal розвинувся у потужне середовище візуального програмування Delphi для Windows, а фірма Borland, Inc. змінила назву на Inprise Corp.

Зовнішній вигляд середовища Turbo Pascal 7.0 (яке, до речі, вже підтримує синтаксичне підсвічування – кольорове виділення фрагментів тексту програми: ключових слів, символів, коментарів тощо) показано на рис. 1.2.

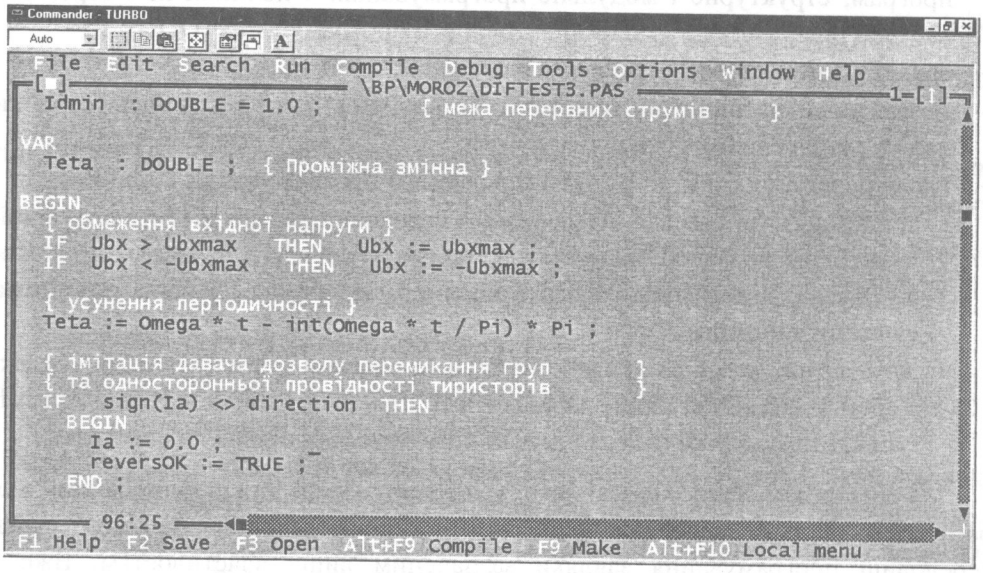


Рис. 1.2. Зовнішній вигляд середовища Turbo Pascal 7.0

Нижче подано основні типи числових даних і основні оператори керування Turbo Pascal, що допоможе читачеві зорієнтуватися в можливостях середовища, а детальнішу інформацію можна отримати з літератури [розділ А], якої сьогодні, на щастя, не бракує.

Основні типи числових даних:

тип **BYTE** – цілий короткий, займає у пам'яті 1 байт (8 біт), подає значення величин без знака з діапазону від 0 до 255;

тип **SHORTINT** – цілий короткий, займає у пам'яті 1 байт (8 біт), подає значення величин зі знаком з діапазону від – 128 до 127;

тип **WORD** – цілий (слово), займає у пам'яті 2 байти (16 біт), подає значення величин без знака з діапазону від 0 до 65535;

тип **INTEGER** – цілий, займає у пам'яті 2 байти, подає значення величин зі знаком з діапазону від – 32768 до 32767;

тип **LONGINT** – цілий довгий, займає у пам'яті 4 байти, подає значення величин зі знаком з діапазону від – 2147483648 до 2147483647;

тип **COMP** – цілий довгий, займає у пам'яті 8 байт, подає значення величин зі знаком з діапазону від $-2^{63}+1$ до $2^{63}-1$;

тип **SINGLE** – дійсний звичайної точності, займає у пам'яті 4 байти. Використовується для подання значень величин з діапазону $1.5\cdot10^{-45} < |x| \le 3.4\cdot10^{+38}$ у форматі з плаваючою комою $\pm m\cdot E\pm p$, де m – мантиса числа (7–8 значущих цифр), а p є порядком;

тип **REAL** – дійсний звичайної точності, займає у пам'яті 6 байт, призначався для комп'ютерів без математичного співпроцесора. Застосовується для подання значень величин з діапазону $2.9\cdot10^{-39} < |x| \le 1.7\cdot10^{+38}$ у форматі з плаваючою комою $\pm m\cdot E\pm p$, де m – мантиса числа (11–12 значущих цифр), а p є порядком;

тип **DOUBLE** – дійсний подвійної точності, займає у пам'яті 8 байт. Використовується для подання значень величин з діапазону $5\cdot10^{-324} < |x| \le 1.7\cdot10^{+308}$ у форматі з плаваючою комою $\pm m\cdot D\pm p$. Мантиса m має 15–16 значущих цифр;

тип **EXTENDED** – дійсний підвищеної точності, займає у пам'яті 10 байт. Використовується для подання значень величин з діапазону $3.4\cdot10^{-4932} < |x| \le 1.1\cdot10^{+4932}$ у форматі з плаваючою комою $\pm m\cdot D\pm p$. Мантиса m має 19–20 значущих цифр.

Основні керуючі оператори мови:

if ... then ... else – умовний оператор, що може розміщуватись як в одному, так і в багатьох рядках;

case – оператор вибору одного варіанта з багатьох;

for ... to – оператор циклу, забезпечує цикл з одиничним кроком для відомої кількості повторень;

while – оператор циклу з перевіркою умови виконання циклу на початку циклу;

repeat ... until – оператор циклу з перевіркою умови закінчення циклу в кінці циклу;

procedure – оператор процедури – підпрограми;

function – оператор процедури – функції.

Великою перевагою Turbo Pascal є можливість створення окремих трансльованих блоків (бібліотек), які називаються **unit**, а також велика бібліотека різноманітних функцій та готових модулів. В Internet можна знайти велику кількість сторінок для користувачів Turbo Pascal, зокрема, `www.pascal.km.ru`, `www.devq.net/pascal`, `www.pascal-central.-com`, `www.dickmann.org` та інші.

Середовище Turbo Pascal є простим і зручним у роботі, а на час написання книги фірма Inprise Inc. зробила доступними для користувачів (*freeware*) його версії 1, 3.01 і 5.5, які після безкоштовної реєстрації можна вільно отримати в Internet на одній зі сторінок фірми: `http://community.borland.com/museum/` або `http://bdn.borland.com/museum`. Як альтернативу можна рекомендувати також один з безкоштовних (*freeware*) компіляторів, що доступні в світовій павутині (Internet), наприклад, 32-розрядний оптимізувальний компілятор Free Pascal, повна версія якого (разом з документацією) наявна в Internet за адресою `www.freepascal.org` і який працює в середовищах DOS, Windows, OS/2, Linux, FreeBSD на комп'ютерах з процесором типу Intel 386 і вище. Для цього компілятора декларується не тільки практично повна сумісність з Turbo Pascal, але й з Borland (Inprise) Delphi.

1.3. Математичний пакет MathCAD

Серед науковців та інженерів доволі популярним є математичний пакет MathCAD (скорочення CAD означає – Computer Aided Design, тобто проектування із застосуванням комп'ютерів), який у звичній математичній формі подання дає змогу розв'язувати різноманітні задачі: визначати корені рівняння, розв'язувати системи лінійних чи нелінійних алгебричних рівнянь, знаходити означені інтеграли та похідні, будувати різноманітні 2- і 3-вимірні графіки, опрацьовувати вектори і матриці, працювати з дійсними та комплексними числами тощо. За свої можливості пакет MathCAD (розроблений фірмою Mathsoft, адреса якої в Internet **www.mathsoft.com**) отримав назву наукового суперкалькулятора. Приклад вікна з відкритим документом у середовищі пакета показано на рис. 1.3.

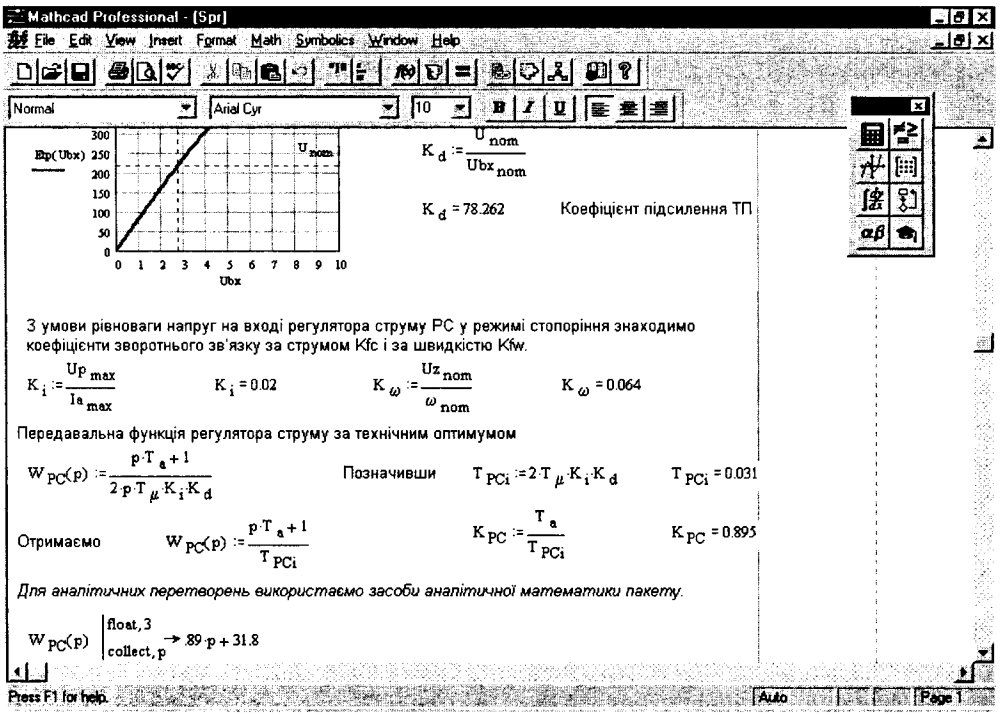


Рис. 1.3. Зовнішній вигляд середовища MathCAD 7 Professional

Починаючи з версії 3.0, MathCAD працює лише в середовищі Windows і тепер містить аналітичне ядро (за ліцензією фірми Maple), яке дає змогу виконувати різноманітні аналітичні перетворення:

- розкриття символьних виразів та їх спрощення;
- символьне інтегрування та диференціювання;
- аналітичне розв'язування рівнянь;
- символьні матричні операції;
- прямі та зворотні перетворення Фур'є і Лапласа тощо.

З кожною наступною версією можливості пакета зростають: додалися функції для розв'язування систем диференціальних рівнянь (починаючи з версії 5 Plus), введені структуровані конструкції для реалізації алгоритмів (у версіях Professional, починаючи з 6.0), додана можливість роботи в Internet тощо. Для використання у навчальних закладах фірма MathSoft свого часу запропонувала безкоштовну версію (*freeware*) MathCAD Explorer, яка підтримує більшість можливостей повного пакета, але не дає змоги зберігати на диску робочі файли.

Дуже корисний MathCAD інженерам-електрикам: до нього додається спеціальний електронний довідник з електро- і радіотехніки з доволі широкою довідковою інформацією та прикладами розв'язування задач. Зокрема, студенти за допомогою програми MathCAD можуть успішно розраховувати лінійні та нелінійні кола постійного та змінного струмів з виведенням результатів на графік, розв'язувати системи лінійних і нелінійних рівнянь, апроксимувати нелінійності, розв'язувати системи диференціальних рівнянь, досліджувати частотні характеристики, будувати кореневі годографи тощо. Фірма MathSoft пропонує також електронні книги (спеціальні додатки до пакета), які можуть бути і підручником, і прикладом, і довідником у різних галузях науки і техніки; зокрема, інженерам-електрикам корисною буде електронна книга "Topics in Electrical Engineering".

У середовищі MathCAD зручно оформляти технічні звіти, пояснювальні записки курсових і дипломних проектів, що містять велику кількість розрахунків та графіків, а також створювати навчально-демонстраційні програми (*"живі документи"*) з різних навчальних дисциплін завдяки унікальній можливості MathCAD відображати інформацію в звичному для користувача вигляді. Детальнішу інформацію про пакет можна отримати з літератури [розділ А] та відповідних сторінок Internet, наприклад, **www.mathcad.com**, **www.mathcad.ru**, **www.exponenta.ru**.

1.4. Математичний пакет MATLAB

Розроблений фірмою MathWorks, Inc. (**www.mathworks.com**) математичний пакет MATLAB є одним з найвідоміших і найпотужніших математичних пакетів для інженерних і наукових розрахунків, що також широко використовується у вищих навчальних закладах світу.

Мова пакета MATLAB – це проста мова програмування, в якій базовим елементом є матриця: так, звичайній змінній у пакеті відповідає матриця 1×1. Перевагою MATLAB є велика кількість наявних функцій (більшість з яких написана власне мовою MATLAB), що реалізують широкий набір числових методів. Отже, користувач пакета не стільки думає над програмуванням, скільки над задачею і способом її описання в пакеті. Вигляд робочого вікна пакета версій 5.X показано на рис. 1.4.

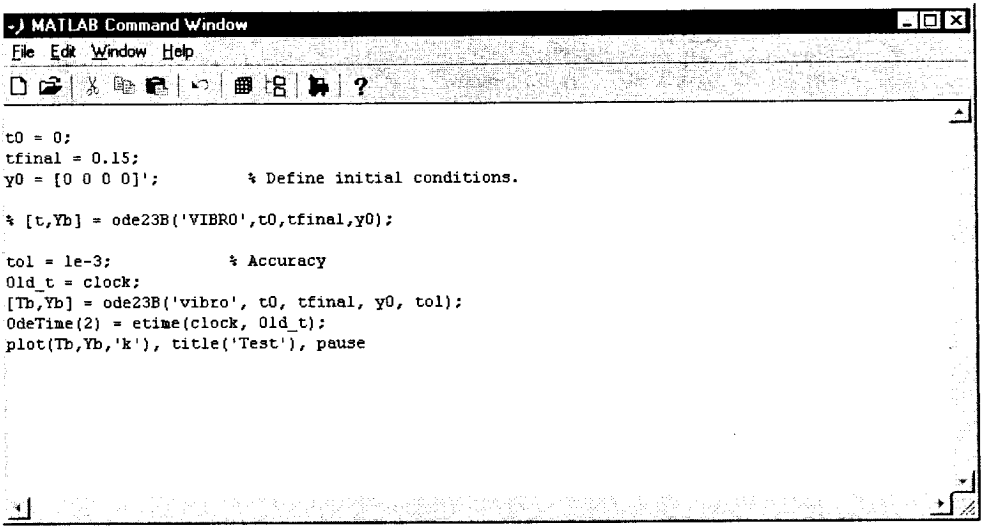


Рис. 1.4. Зовнішній вигляд середовища MATLAB версій 5.X

Полегшує роботу в пакеті MATLAB і вбудований редактор, що інтегрований з налагоджувачем середовища (так само, як і в професійних середовищах програмування). Редактор підтримує кольорове виділення синтаксису, що полегшує сприйняття тексту програми (рис. 1.5).

Середовище MATLAB має деякі проблеми з підтримкою кирилиці, що необхідно враховувати під час роботи з ним:

- *у версіях 5.X вбудований редактор-налагоджувач у текстах програм часто не відображає символів кирилиці;*
- *версія 6 видає повідомлення про помилки, якщо в тексті трапляється мала буква "я" – її треба замінювати на велику;*
- *можливі проблеми з правильним відображенням підписів із символами кирилиці на графіках.*

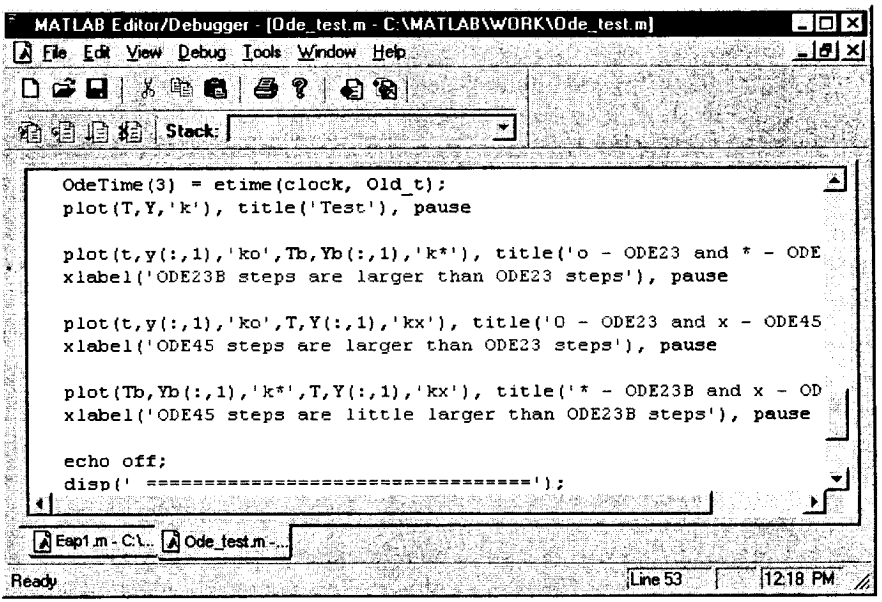


Рис. 1.5. Зовнішній вигляд редактора/налагоджувача MATLAB версії 5.3

Додатково MATLAB містить проблемно-орієнтовані набори функцій – Toolbox, що реалізують типові обчислення та відповідні числові методи для різних галузей науки і техніки: опрацювання сигналів, аналіз та синтез систем керування, оптимізаційні задачі, опрацювання зображень і картографічної інформації, статистика, сучасні методи побудови керуючих систем (зокрема нелінійних) тощо.

Починаючи з версії 4, до MATLAB додано пакет візуального імітаційного моделювання Simulink, який виявився настільки зручним і ефективним для моделювання різноманітних динамічних процесів та систем, що значна частина користувачів застосовує лише його з усіх наявних можливостей MATLAB (рис. 1.6). Дуже зручний Simulink для моделювання автоматизованих електроприводів, додаткової зручності йому надають блоки з Power System Toolbox, які містять уже готові моделі двигунів постійного і змінного струмів (синхронні та асинхронні), моделі силових напівпровідникових приладів і систем керування ними, силових трансформаторів, ліній пересилання тощо (рис. 1.7).

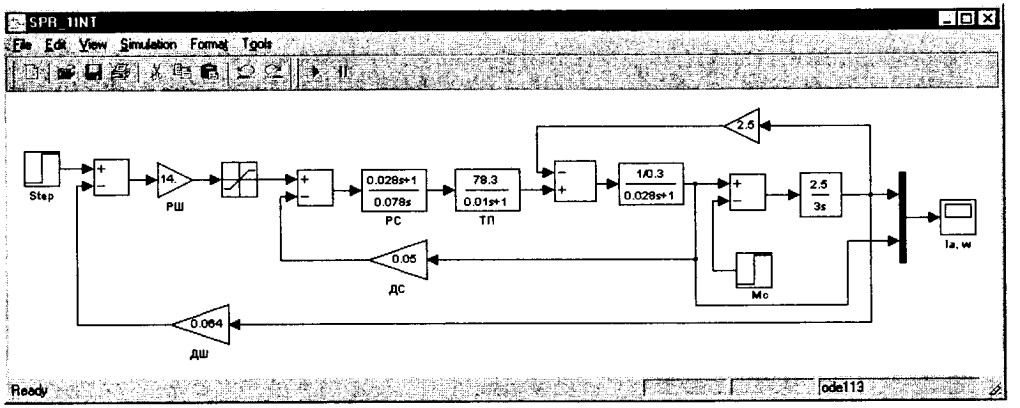


Рис. 1.6. Зовнішній вигляд робочого вікна в середовищі Simulink

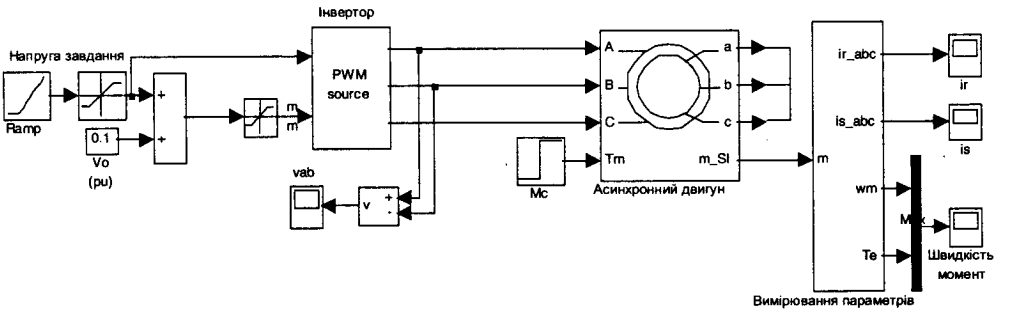


Рис. 1.7. Модель в середовищі Simulink з використанням елементів бібліотеки Power System Toolbox

Однією з вад поєднання MATLAB + Simulink є потреба в достатньо потужному персональному комп'ютері. Наприклад, для нормальної роботи з версією MATLAB 5.2 (випущена у 1997 році) у середовищі Windows 95 потрібно процесор класу Pentium і хоча б 16 (а краще 32) МБайт оперативної пам'яті. Новіші версії ставлять ще вищі вимоги, наприклад, MATLAB 6.1 вимагає мінімум 128 МБайт оперативної пам'яті, а для нормальної роботи рекомендується 256 МБайт і процесор класу Pentium III.

Основний тип числових даних:

тип **DOUBLE** – дійсний подвійної точності, займає у пам'яті 8 байт. Використовується для подання значень величин з діапазону $5\cdot10^{-324} < |x| \le 1.7\cdot10^{+308}$ у форматі з плаваючою комою $\pm m\cdot D\pm p$. Мантиса m має 15–16 значущих цифр.

Основні оператори керування мови:

if ... else – умовний оператор, який може розміщуватись як в одному, так і в багатьох рядках;

case – оператор вибору одного варіанта з багатьох;

for – оператор циклу, забезпечує цикл із заданим кроком та відомою кількістю повторень;

while – оператор циклу з перевіркою умови виконання циклу на початку циклу;

function – оператор процедури-функції.

Детальнішу інформацію про пакет можна отримати з літератури [див. перелік, розділ А] та відповідних сторінок Internet, наприклад, `www.matlab.com`, `www.matlab.com.ua`, `www.exponenta.ru`.

1.5. Порівняння програмних засобів

Часто задачі розрахунку динаміки електроприводів є порівняно нескладними (зокрема, у навчальному курсі з моделювання електроприводів), тому проблем з тривалістю розрахунку найчастіше не виникає. У разі підвищення

складності задачі можливі декілька способів скорочення тривалості обчислень (окрім придбання потужнішого комп'ютера).

На застарілих комп'ютерах збільшити швидкість розрахунку майже на порядок можна використанням математичного співпроцесора (починаючи з процесорів Intel 486-DX, Pentium цей апаратний прискорювач математичних операцій є вмонтованим). Порівняння часу розрахунку для різних середовищ під час виконання тестової задачі, що розраховує пуско-гальмівні процеси для системи з дворазовим інтегруванням, наведено в табл. 1.1. Для розв'язування цієї тестової задачі застосовано метод Рунге – Кутта четвертого порядку (*див. розд. 3, підпрограма* RK4) з фіксованим кроком 0,001 с для інтервалу моделювання 10 с. Порівняння здійснювалося для середовищ програмування Microsoft Quick Basic версії 4.5 (у таблиці подано два варіанти: для IDE (*Integrated Development Environment*) – інтегрованого середовища, а також для зовнішнього оптимізувального компілятора BC (*BASIC Compiler*) – компілятора командного рядка) і Turbo Pascal 7.0 (TP) та математичних пакетів MathCAD 7 Professional (час виконання для старших версій пакета не відрізняється, принаймні для версій 2001 і 11 включно) і MATLAB 5.2 (разом з Simulink) (час виконання для версій пакета до 6.5 включно також не відрізняється).

Таблиця 1.1

Порівняння часу розрахунку тестової задачі для різних середовищ і різних типів процесора ПК

Тип процесора	*MathCAD*	*QB 4.5* *)		*MATLAB*		*TP 7.0* *)
		IDE	*BC*	*m-файл*	*Simulink*	
486 DX/2-66 МГц	3 хв 40 с	23,4 с	9,9 с	24,2 с	165 с	2,8 с
Pentium-200MMX	≈19 с	11,2 с	9,5 с	11,1 с	12 с	0,59 с
Celeron-1400	≈2 с	1,3 с	1,3 с	7,7 с	1,2 с	0,18 с

*) Для режиму DOS

Додаткове пришвидшення розрахунків для Quick Basic (3–3,5 раза) у середовищі Windows 95/98 на порівняно нешвидких машинах (з процесором

Intel 486) можна отримати, якщо розрахунки виконувати в режимі "чистої" DOS, для чого треба вийти з графічного середовища Windows.

Використання мов програмування у багатьох випадках дає змогу використовувати навіть застарілі, здавалось би, антикварні, комп'ютери, наприклад, з процесором Intel 80286. Такий комп'ютер з математичним співпроцесором Intel 80287 має арифметичну швидкодію близько 300 тис. операцій з плаваючою комою за секунду, що часто достатньо для моделювання нескладних систем автоматизованого електропривода. Трохи новіший процесор Intel 486-DX2 з частотою 66 МГц забезпечує вже приблизно 10 млн. операцій з плаваючою комою за секунду (чи, як прийнято казати, мегафлопів), а Pentium-133 досягає швидкодії 70 мегафлопів, що є непоганою швидкодією для моделювання динаміки реальних систем електроприводів. Сучасні ж процесори вже подолали поріг в 1 гігафлоп (1 млрд. операцій з плаваючою комою за секунду) – така швидкодія є достатньою для моделювання систем електропривода будь-якого рівня складності (хоча неграмотний підхід може звести нанівець й таку колосальну швидкодію сучасних комп'ютерів).

2 МОДЕЛІ ЕЛЕМЕНТІВ ЕЛЕКТРОПРИВОДІВ

Динамічні процеси в системах електроприводів можна дослідити за допомогою математичних моделей. Для цього необхідно їх побудувати у вигляді, наприклад, системи диференціальних рівнянь першого порядку в нормальній формі Коші та розв'язати їх за допомогою числового методу. Отримані результати у вигляді таблиць чи графіків перехідних процесів використовують для аналізу динамічних і статичних властивостей систем. У розділі описано моделювання елементів, з яких складаються реальні системи електропривода. Залежно від складності поставленої задачі (навчальна чи практична модель для дослідження реальної системи) розглядають моделі різного рівня деталізації та різної точності. Також запропоновано приклади використання моделей елементів у різних середовищах: Quick Basic, Turbo Pascal, MathCAD, MATLAB, що дає змогу оцінити можливості та простоту використання того чи іншого програмного пакета для розв'язання поставленої задачі та порівняти їх ефективність.

Загальні відомості про лінійні системи

Моделювання лінійних систем застосовують там, де технологічні системи та їх елементи є лінійними загалом або лінійними у визначених межах. У такий клас потрапляють і лінеаризовані системи, тобто системи, у яких поточна область робочої характеристики замінюється їх лінійним наближенням. Використовують лінійні системи ще й тому, що існують аналітично точні розв'язки для лінійних систем рівнянь, спеціальні високоточні методи дослідження і моделювання лінійних систем. Крім того, за допомогою лінійних систем можна оцінити спотворення в моделях нелінійних систем.

У моделюванні і лінійних, і нелінійних систем головним завданням є визначення реакції системи на функцію збурювальної дії. Нехай функція збурення $f_1(t)$, що змінюється в часі, призводить до реакції $r_1(t)$, а інша функція збурення $f_2(t)$ викликає реакцію $r_2(t)$. Це має такий вигляд:

$$f_1(t) \rightarrow r_1(t);$$
$$f_2(t) \rightarrow r_2(t).$$

Для лінійної системи

$$f_1(t) + f_2(t) \rightarrow r_1(t) + r_2(t). \qquad (2.1)$$

Вираз (2.1) описує принцип суперпозиції: суперпозиція індивідуальних збурювальних дій призводить до реакції системи, яка є суперпозицією індивідуальних реакцій.

Принцип суперпозиції є характерним для лінійних систем і призводить до наслідків:

1) дві та більше збурювальні дії не впливають одна на одну;
2) реакції від різних збурень не перетинаються;
3) сукупність збурювальних дій може бути виявлена за сукупністю реакцій: визначається залежність кожної збурювальної дії від реакції, а потім реакції об'єднуються чи накладаються для визначення сумарного збурення.

Іншим висновком з принципу суперпозиції є те, що, якщо на лінійну систему діє N однакових збурень, то реакція від такого впливу визначається як N однакових реакцій, кожна з яких є реакцією системи на одне збурення, тобто,

$$N \cdot f(t) \rightarrow N \cdot r(t). \qquad (2.2)$$

З виразу (2.2) зрозуміло, що лінійні системи зберігають масштабний чинник збурювальних впливів у разі переходу від входу системи до її виходу. Цю властивість лінійних систем називають **принципом однорідності**.

Якщо співвідношення (2.1) і (2.2) справедливі для лінійних систем, то це ще не означає, що виконання одного з них є достатнім для визначення лінійності системи. Система лінійна тільки якщо виконуються співвідношення (2.1) і (2.2).

Іншою характерною рисою лінійних систем є їх **стаціонарність**. Вважають, що лінійна система повинна бути стаціонарною, якщо

$$f(t - T) \rightarrow r(t - T),$$

де T – довільний час запізнення.

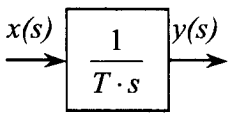
Це реакція лінійної стаціонарної системи на періодичну збурювальну дію. Пояснити це можна ще й так: якщо періодично-змінна збурювальна дія має частоту F, то стаціонарна лінійна система відреагує на неї періодичною

реакцією з такою самою частотою F. Іншими словами, реакція лінійної системи має такі самі спектральні компоненти, як і збурення.

2.1. Математичні моделі елементарних динамічних ланок

Модель інтегратора

Модель інтегратора (інтегральної ланки) є базовою для побудови всіх інших динамічних ланок та інерційних елементів автоматизованих електроприводів. Основним параметром інтегральної ланки є стала часу T.

Диференціальне рівняння	*Структурна модель*
$T\frac{dy}{dt} = x(t)$	

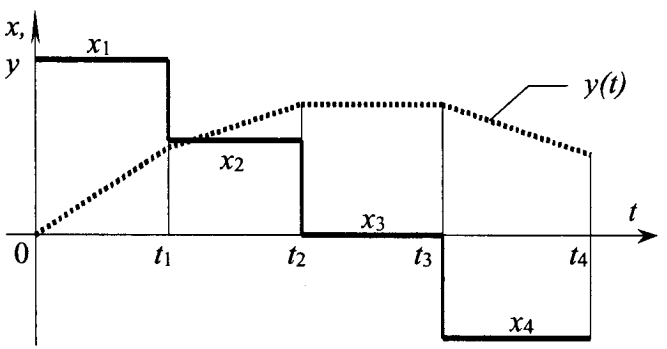


Рис. 2.1. Вихідний сигнал y(t) інтегральної ланки для різних збурень x(t)

На рис. 2.1 зображені перехідні процеси інтегральної ланки для різних значень вхідного сигналу – збурення $x(t)$, що є сталими в заданих інтервалах $0 - t_1$, $t_1 - t_2$, $t_2 - t_3$, $t_3 - t_4$ і постійного значення сталої часу T. Як видно з графіка, темп інтегрування визначається значенням вхідного сигналу інтегратора. Варто звернути увагу на те, що усталеного значення (*статичний режим*) вихідний сигнал набуває

тільки тоді, коли $x(t) = 0$ (на ділянці $t_2 - t_3$), чим забезпечується астатичне регулювання в системах з інтегральними ланками, які детальніше буде розглянуто далі.

Модель аперіодичної ланки

Модель аперіодичної ланки є чи не найвживанішою для моделювання елементів автоматизованих електроприводів. Основними параметрами аперіодичної ланки є стала часу T і коефіцієнт підсилення ланки K.

Диференціальне рівняння	*Структурна модель*
$T\frac{dy}{dt} + y = K \cdot x(t)$	$x(s) \rightarrow \boxed{\frac{K}{T \cdot s + 1}} \rightarrow y(s)$

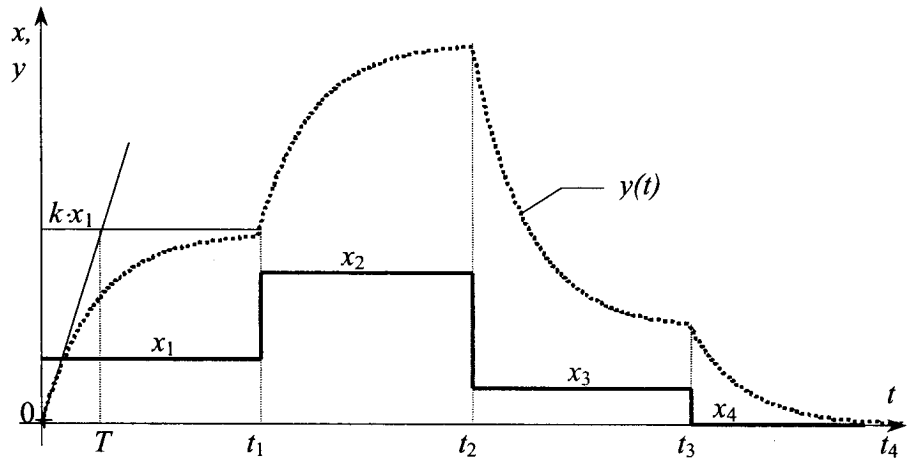


Рис. 2.2. Перехідні процеси y(t) аперіодичної ланки для різних збурень x(t)

На рис. 2.2 зображені графіки перехідних процесів для аперіодичної ланки для різних значень вхідного сигналу $x(t)$ і постійного значення сталої часу T. Як видно з графіка, усталеного значення $k \cdot x$ (статичний режим) вихідний сигнал набуває для будь-яких постійних значень $x(t) = \text{const}$ приблизно через $4T$, що підтверджує очевидну відмінність інтегральної та аперіодичної ланок.

Модель коливної ланки другого порядку

Модель коливної ланки також використовується для моделювання елементів автоматизованих електроприводів. Основними параметрами коливної ланки другого порядку є стала часу T, коефіцієнт підсилення ланки K і коефіцієнт вгамування (демпфування) ξ.

Диференціальне рівняння	*Структурна модель*
$T\frac{d^2y}{dt^2}+2\xi T\frac{dy}{dt}+y=K\cdot x(t)$	$x(s)\rightarrow\boxed{\frac{K}{T^2s^2+2\xi T\cdot s+1}}\rightarrow y(s)$

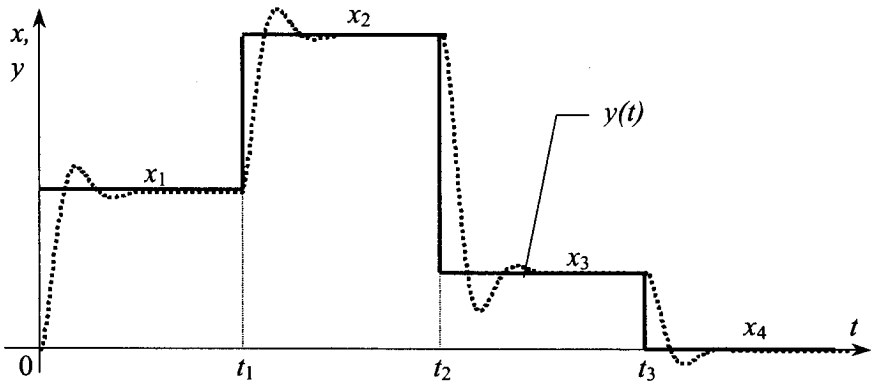


Рис. 2.3. Перехідні процеси y(t) коливної ланки другого порядку з ξ=0,5 для різних збурень x(t)

На рис. 2.3 зображені графіки перехідних процесів для коливної ланки другого порядку для різних значень вхідного сигналу $x(t)$, постійного значення сталої часу T і коефіцієнт вгамування $\xi = 0{,}5$. Як видно з графіка, вихідному сигналові властиві коливання. З теорії автоматичного керування відомо, що частота коливань визначається сталою часу T, а швидкість їх загасання – коефіцієнтом вгамування ξ. Для $\xi = 0$ отримують незагасаючі коливання, для $\xi > 1$ матимемо аперіодичний характер процесів, а $0 < \xi < 1$ визначає область існування загасаючих коливань.

2.2. Математичні моделі електричних машин постійного струму

Модель генератора постійного струму

Машина постійного струму досі залишається одним з найпоширеніших елементів автоматизованих електроприводів, тому її моделі різної складності давно відомі й широко використовуються в дослідженнях. Важливим елементом машини постійного струму є коло збудження. Описання кола збудження є однаковим як для генератора постійного струму (ГПС), так і для двигуна постійного струму (ДПС), що працює зі змінним потоком.

Очевидним є те, що змоделювати, тобто відтворити в моделі динамічні процеси, які відбуваються в оригіналі, наприклад, генераторі постійного струму, абсолютно точно неможливо. Можна говорити лише про наближення, тобто про отримання результатів моделювання з деякою точністю, яка великою мірою визначається допущеннями, які ми приймаємо, створюючи модель. Так, в електроприводі ГПС в неробочому режимі, коли якірне коло розімкнене, з достатньою для багатьох застосувань точністю можна розглядати як аперіодичну ланку першого порядку (рис. 2.4) за таких допущень:

- не враховуємо нелінійність кривої намагнічування;
- нехтуємо вихровими струмами в станині (це цілком справедливо для генераторів зі шихтованою станиною);
- потоки розсіювання обмотки збудження генератора (ОЗГ) пропорційні до струму збудження.

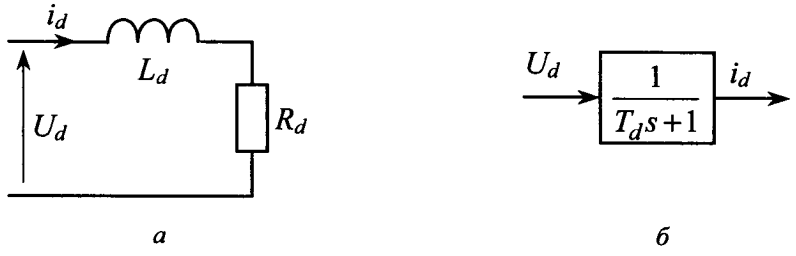


Рис. 2.4. Коло збудження ГПС:
а – електрична схема; б – структурна модель

У випадку врахування навантаження (замкнуте якірне коло) приймають додаткові допущення:

- індуктивність якоря незмінна;
- трансформаторною ЕРС у колі якоря нехтуємо;
- дія реакції якоря залежно від точності моделі або не враховується, або приймається пропорційною до струму якоря.

Найпростіша математична модель ГПС побудована на рівнянні електричної рівноваги кола збудження

$$U_d = i_d R_d + L_d \frac{di_d}{dt} \tag{2.3}$$

та рівнянні

$$e_G = i_d R_d K_G,$$

де U_d – напруга збудження генератора;

i_d – струм кола збудження генератора;

R_d – опір обмотки збудження генератора;

$L_d = L_\mu + L_s$ – індуктивність ОЗГ, яка має дві складові:

L_μ – індуктивність намагнічування;

L_s – індуктивність розсіювання;

e_G – електрорушійна сила якоря генератора;

K_G – коефіцієнт підсилення генератора за напругою.

Виконаємо нескладні перетворення рівняння (2.3), помноживши його на K_G, а останній член – на $\frac{R_d}{R_d}$, після чого отримаємо

$$U_d K_G = i_d R_d K_G + \frac{L_d}{R_d} \frac{d(i_d R_d K_G)}{dt}.$$

Позначивши $\frac{L_d}{R_d} = T_d$ – електромагнітна стала часу ОЗГ, отримаємо

$$\frac{de_G}{dt} = \frac{U_d K_G - e_G}{T_d}. \tag{2.4}$$

Це проста математична модель генератора постійного струму, що описує перехідні процеси генератора без навантаження (режими неробочого ходу).

Для активного навантаження (рис. 2.5) $L_a = 0$, $L_н = 0$ тому значення струм якоря i_a обчислюється за виразом

$$i_a = e_G / R_{a\Sigma},$$

де $R_{a\Sigma} = R_a + R_н$ – сумарний опір якірного кола;

R_a – опір якоря ГПС;

$R_н$ – опір навантаження в якірному колі.

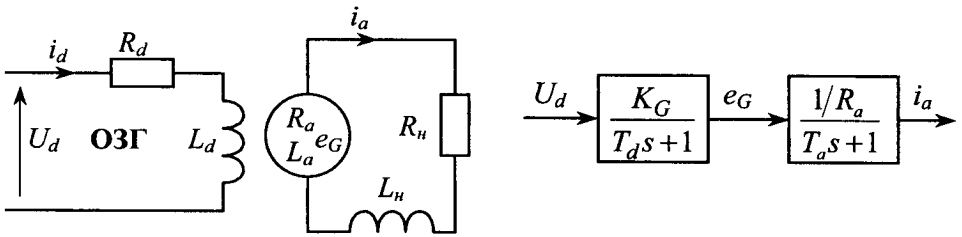


Рис. 2.5. Генератор постійного струму:
а – електрична схема; б – структурна модель

З урахуванням індуктивностей обмотки якоря L_a і навантаження $L_н$ до диференціального рівняння (2.4) описання кола ОЗГ додається ще одне: $e_G = i_a R_{a\Sigma} + L_{a\Sigma} \frac{di_a}{dt}$ (див. рис. 2.5). Поділивши це рівняння на $R_{a\Sigma}$ і позначивши $\frac{L_{a\Sigma}}{R_{a\Sigma}} = T_a$ – електромагнітна стала часу якірного кола, отримаємо диференціальне рівняння якірного кола, яке разом з рівнянням кола збудження є математичною моделлю ГПС:

$$\begin{aligned} \frac{de_G}{dt} &= \frac{U_d K_G - e_G}{T_d}; \\ \frac{di_a}{dt} &= \frac{e_G / R_{a\Sigma} - i_a}{T_a}. \end{aligned} \qquad (2.5)$$

Нижче пропонуються варіанти реалізації у різних середовищах простої моделі генератора постійного струму.

Quick Basic

```
' ------------------------------------------------------------
'     Позначення змінних:
'     y(1) - ЕРС генератора
'     y(2) - струм якоря
' ------------------------------------------------------------
CONST Kg = 2.09          ' Коефіцієнт підсилення генератора
CONST Td = 1.2           ' Стала часу ОЗГ
CONST Ra = .34           ' Сумарний опір якірного кола
CONST Ta = .05           ' Стала часу якірного кола

' Розрахунок напруги збудження генератора
Ud = . . .
' Розрахунок похідної ЕРС
f(1) = (Ud * Kg - y(1)) / Td
' Розрахунок похідної струму
f(2) = (y(1) / Ra - y(2)) / Ta
```

Turbo Pascal

```
{ ------------------------------------------------------------ }
{     Позначення змінних:                                      }
{     y[1] - ЕРС генератора                                    }
{     y[2] - струм якоря                                       }
{ ------------------------------------------------------------ }
const
   Kg = 2.09;      { Коефіцієнт підсилення генератора  }
   Td = 1.1;       { Стала часу ОЗГ                    }
   Ra = 0.34;      { Сумарний опір якірного кола       }
   Ta = 0.05;      { Стала часу якірного кола          }
type
   vector = array [1..N] of double;
```

```
var
   f, y : vector;
   Ud : double;
begin
   Ud :=  . . . ; { розрахунок напруги збудження генератора }
   {  Розрахунок похідної ЕРС  }
   f[1] := (Kg*Ud - y[1])/Td;
   {  Розрахунок похідної струму  }
   f[2] := (y[1]/Ra - y[2])/Ta;
end;
```

MathCAD

$$\text{Diff}(t, y) := \begin{pmatrix} \dfrac{U_d(t) \cdot K_G - y_0}{T_d} \\ \dfrac{\dfrac{y_0}{R_{a\Sigma}} - y_1}{T_a} \end{pmatrix}$$

Функція-вектор правих частин системи диференціальних рівнянь.

y_0 – *ЕРС генератора;*

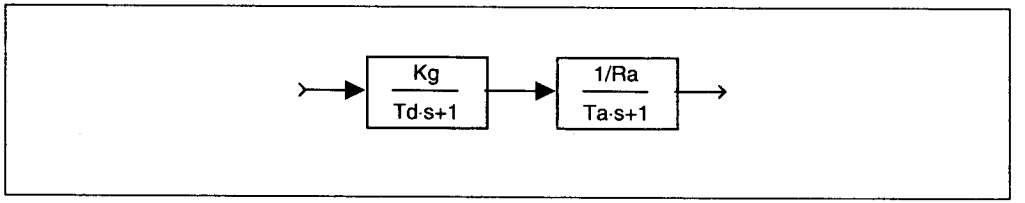
y_1 – *струм якоря.*

MATLAB

```
% ---------------------------------------
%     Позначення змінних:
%     y(1) - ЕРС генератора
%     y(2) - струм якоря
% ---------------------------------------
Kg = 2.09          % Коефіцієнт підсилення генератора
Td = 1.2           % Стала часу ОЗГ
```

```
Ra = 0.34          % Сумарний опір якірного кола
Ta = 0.05          % Стала часу якірного кола

% Розрахунок напруги збудження генератора
Ud = . . .;
% Розрахунок похідної ЕРС
f(1) = (Ud*Kg - y(1))/Td;
% Розрахунок похідної струму
f(2) = (y(1)/Ra - y(2))/Ta;
```

Simulink



Значення індуктивності намагнічування ОЗГ L_μ та індуктивності обмотки якоря L_a машини постійного струму (якщо ці відомості відсутні в каталозі) можна розрахувати за формулами:

$$L_\mu = 2p_n W_p^2 \frac{d\Phi}{dF} \approx 2p_n W_p^2 \frac{\Phi_н}{F_н} \quad \text{і} \quad L_a \approx \frac{kU_{Gн}}{p_n \omega_н I_{ан}},$$

де p_n – кількість пар полюсів;

W_p – кількість витків одного полюса ОЗГ;

Φ – магнітний потік одного полюса ОЗГ;

F – ампер-витки ОЗГ;

k – емпіричний коефіцієнт: $k = 0.6$ для некомпенсованих машин, $k = 0{,}15$ для компенсованих машин;

$\omega_н$ – номінальна частота обертання якоря генератора.

Значення індуктивності розсіювання обмотки збудження L_s машини постійного струму можна знайти за наближеною формулою $L_s \cong \sigma \cdot L_{\mu\,ном}$, де σ –

коефіцієнт розсіювання, приблизно дорівнює 0.1...0.2, а точніше обчислюється за формулою

$$\sigma = \delta \left(4.5 \frac{p_n}{D} + \frac{2}{l} \right),$$

де δ – повітряний проміжок під полюсом, см;

D – діаметр якоря, см;

l – довжина якоря, см.

Точнішу модель генератора, що враховує нелінійність кривої намагнічування, можна отримати за умови апроксимації характеристики намагнічування функцією арктангенса [Д.2]:

$$\begin{cases} \dfrac{di_d}{dt} = \dfrac{U_d / R_d - i_d}{T_d} \; ; \\ e_G = A \cdot arctg\left(B \cdot i_d\right) \; ; \\ \dfrac{di_a}{dt} = \dfrac{e_G / R_{a\Sigma} - i_a}{T_a} \; . \end{cases}$$

У такому разі для отримання вищої точності в моделі генератора залежність $e_G(i_d)$ апроксимується залежністю $e_G = A \cdot \mathrm{arctg}\left(B \cdot i_d\right)$, де A, B – коефіцієнти апроксимації. Коефіцієнти апроксимації для першого наближення вибирають так: $A = E_{Gн}$, тоді $BI_{dн} = \mathrm{tg}(1)$, звідки $B = \mathrm{tg}(1)/I_{dн}$, після чого можливе їх уточнення ітераційним чи якимось іншим методом (для цього дуже зручно використовувати математичний пакет MathCAD). Така апроксимація є простою і, водночас, забезпечує похибку не більше ніж 1–2 %, а за плавністю та точністю відтворення похідної dE_G/dI_d має переваги перед іншими, наприклад, поліноміальними наближеннями.

Реалізований в пакеті MathCAD приклад знаходження апроксимаційної залежності для напруги і ампер-витків обмотки збудження генератора подано нижче.

MathCAD

$N := 6$ *Кількість точок робочої характеристики* $i := 0 .. N$

Робоча характеристика:

Номінальна точка № 3

$$Eg := \begin{bmatrix} 0 \\ 230 \\ 345 \\ 460 \\ 477.7 \\ 506 \\ 575 \end{bmatrix} \qquad AW := \begin{bmatrix} 0 \\ 1048 \\ 1623 \\ 2618 \\ 2955 \\ 3780 \\ 6628 \end{bmatrix}$$

Розрахунок коефіцієнтів апроксимації

$A := Eg_3$ $\qquad B := \dfrac{\tan(1)}{AW_3}$

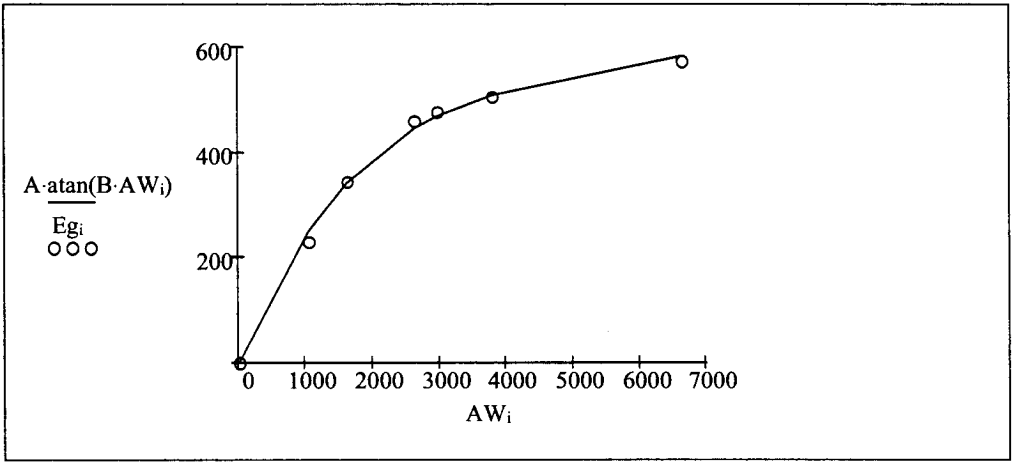
Given *Рівнянь повинно бути не менше ніж змінних*

$$\sum_i \left(Eg_i - A \cdot \operatorname{atan}\left(B \cdot AW_i\right)\right)^2 = 0 \qquad A > 0$$

$\begin{pmatrix} A \\ B \end{pmatrix} := \text{Minerr}(A, B)$ $\qquad A = 440.177$

$B = 6.072 \times 10^{-4}$

Побудова графіка апроксимаційної кривої та експериментальних точок



За відсутності даних про криву намагнічування у технічному паспорті генератора можна скористатися універсальною кривою намагнічування [Д.6] (рис. 2.6), якій відповідають коефіцієнти апроксимації A = 0.858 і B = 2.351, тобто вихідна напруга генератора буде описуватися залежністю

$$E_G = 0{,}858 E_{Gном} \cdot \operatorname{arctg}\left(2{,}351 \frac{i_d}{I_{dном}}\right).$$

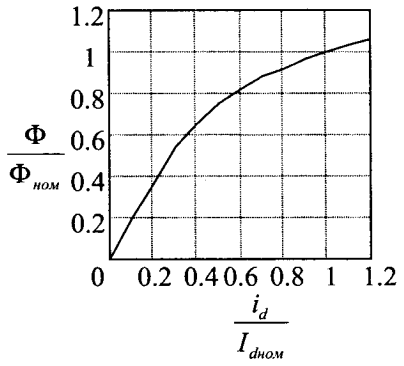


Рис. 2.6. Універсальна крива намагнічування

Практичне застосування знайшла й інтерполяція кривої намагнічування сплайнами [В.10, Д.14], що також відповідає вимогам точності та гладкості, але потребує відповідного програмного забезпечення. Обидва способи наближення кривої намагнічування практично рівноцінні щодо точності та витрат комп'ютерного часу.

Підвищити точність відтворення реальних процесів у колі збудження генератора постійного струму з масивною станиною можна, якщо врахувати розмагнічувальну дію вихрових струмів у станині [Д.3, Д.20]. Для цього в електричну схему моделі кола збудження вводиться додатковий контур $R_k - L_\mu$, що імітує дію вихрових струмів (рис. 2.7, *а*). Ввівши позначення $T_k = L_\mu / R_k$ (стала часу контуру вихрових струмів), $T_\mu = L_\mu / R_d$, $T_s = L_s / R_d$, можна побудувати структурну схему, показану на рис. 2.7, *б*.

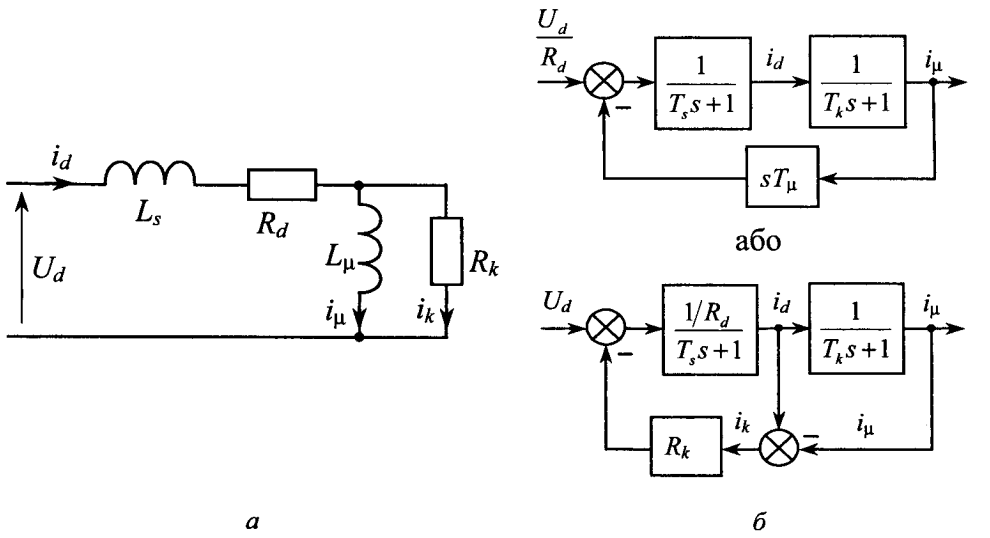


Рис. 2.7. Схема і структурна модель генератора постійного струму з урахуванням контуру вихрових струмів: а – електрична схема; б – структурна модель

Така модель описується системою диференціальних і алгебричних рівнянь:

$$\begin{cases} L_s \dfrac{di_d}{dt} + i_d R_d + L_\mu \dfrac{di_\mu}{dt} = U_d \ ; \\ L_\mu \dfrac{di_\mu}{dt} = i_k R_k \ ; \\ i_\mu + i_k - i_d = 0 \ , \end{cases} \quad \text{або} \quad \begin{cases} T_s \dfrac{di_d}{dt} + i_d + T_\mu \dfrac{di_\mu}{dt} = \dfrac{U_d}{R_d} \ ; \\ T_k \dfrac{di_\mu}{dt} = i_k ; \\ i_\mu + i_k - i_d = 0 \ , \end{cases} \tag{2.6}$$

де i_μ – струм намагнічування;

i_k – струм контуру імітації вихрових струмів,

звідки можна отримати вирази в операторній формі для струмів збудження та намагнічування:

$$i_d = \frac{T_k s + 1}{s^2 T_s T_k + \left(T_s + T_k + T_\mu\right) s + 1} \cdot \frac{U_d}{R_d};$$

$$i_\mu = \frac{1}{s^2 T_s T_k + \left(T_s + T_k + T_\mu\right) s + 1} \cdot \frac{U_d}{R_d}.$$

Для більшості генераторів постійного струму з масивною станиною стала часу еквівалентного контуру вихрових струмів T_k становить 15–30 % від сталої часу обмотки збудження T_d. Точніше значення сталої часу T_k знаходять за емпіричною формулою [Д.20]:

$$T_k \cong \frac{4 l_j \ a^2 b^2}{\pi k_\rho \delta \left(a^2 + b^2\right)};$$

де l_j – довжина силової лінії у спинці між полюсами, см;

a, b – товщина і довжина спинки станини, см;

$k_\rho = 2 \cdot 10^4$ (для сталі).

Недоліком цієї моделі є наявність паразитного кореня характеристичного рівняння, який для реальних співвідношень T_μ, T_k, T_s на два порядки менший від сталої часу T_d. В електричному колі моделі цьому кореневі (сталій часу)

відповідає ланка $L_s - R_d - R_k$. Якщо взяти типові співвідношення для електричних машин постійного струму $L_s = 0{,}1L_\mu$, а $R_k = 5R_d$, тоді

$$\frac{L_s}{R_d + R_k} = \frac{0{,}1L_\mu}{R_d + 5R_d} = \frac{0{,}1L_\mu}{6R_d} \approx 0{,}017 \quad T_\mu,$$

тобто, характеристичні корені знаменника для типових співвідношень $T_s = 0{,}1T_\mu$, $T_k = 0{,}2T_\mu$ становлять $T_1 = 0{,}017T_\mu$; $T_2 = 1{,}28T_\mu$. Поява такої малої сталої часу (кореня характеристичного рівняння) спричиняє необхідність істотно зменшувати крок числового розв'язання під час моделювання, наслідком чого є зростання тривалості моделювання (у 4–5 разів порівняно з моделлю кола збудження як аперіодичної ланки). Крім того, у цій моделі характер перехідного процесу за дії стрибкоподібного збурення (прикладеної напруги) не відповідає реальним процесам в обмотці збудження на початковій ділянці: у моделі наростання струму збудження відбувається майже стрибкоподібно, тобто набагато швидше ніж у реальних електричних машинах внаслідок наявності "паразитної" малої сталої часу.

Це пояснюється тим, що у такій моделі кола збудження електричної машини (до речі, вона найпоширеніша) не було враховано чинник витіснення магнітного поля в станині (поверхневий або скін-ефект) [Д.15, Д.17]. Наявність цього явища змінює у перехідних процесах параметри контуру вихрових струмів, що можна доволі просто врахувати за допущення, що станина не насичується. Аналітичний вираз розподілу магнітного поля в металі відомий з електродинаміки [Д.24], так само, як і вираз розподілу струму в металі скінченної товщини:

$$j = j_0\left(e^{-\alpha d} + e^{-\alpha(b-d)}\right),$$

де j_0 – значення густини струму на краю металу;
d – відстань від краю;
b – товщина металу;

$$\alpha = \sqrt{\frac{\omega\mu\sigma}{2}},$$

де ω – кутова частота змінного струму;
μ – магнітна проникність металу;
σ – питома провідність, для сталі $\sigma \cong 10^7$ См/м.

Врахувати поверхневий ефект можна введенням коефіцієнта зміни опору з частотою, тобто

$$K_\sigma = \frac{R_0}{R_\omega},$$

де R_ω – опір еквівалентного контуру вихрових струмів на частоті ω;

R_0 – опір еквівалентного контуру вихрових струмів постійному струму.

Коефіцієнт K_σ знаходять інтегруванням розподілу струму внаслідок поверхневого ефекту:

$$K_\sigma = \frac{1}{2b}\int_0^b \left(e^{-\alpha x} + e^{-\alpha(b-x)}\right)\, dx = \frac{1-e^{-\alpha b}}{\alpha b}.$$

Імітувати збільшення опору з частотою можна з деяким наближенням, увівши додаткову індуктивність (позначимо її L_k) послідовно з опором R_k контуру вихрових струмів. Така заміна є наближеною з таких причин:

- коефіцієнт K_σ зростає пропорційно до квадратного кореня з частоти, на відміну від індуктивного опору;
- не враховується насичення станини у місцях концентрації магнітного поля.

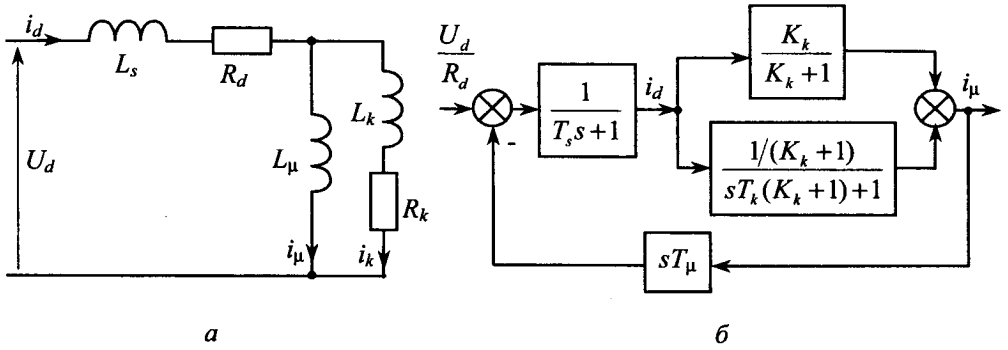
Значення додаткової індуктивності L_k вибирають за максимальним наближенням перехідної характеристики цієї моделі до експериментальної перехідної характеристики реальної машини постійного струму в режимі генератора або за зближенням амплітудно-частотних характеристик коефіцієнта K_σ і ланки, утвореної додатковою індуктивністю L_k і опором R_k. Як показали експериментальні дослідження, для більшості електричних машин постійного струму з масивною станиною значення L_k міститься в межах (1–2) L_μ, за відсутності експериментальних даних чи для першого наближення у побудові моделі можна брати значення додаткової індуктивності L_k таким самим, як індуктивність обмотки збудження L_d. Заступна електрична схема кола збудження зображена на рис. 2.8, *а*, і описується системою рівнянь:

$$\begin{cases} L_s \dfrac{di_d}{dt} + i_d R_d + L_\mu \dfrac{di_\mu}{dt} = U_d \ ; \\ L_\mu \dfrac{di_\mu}{dt} = L_k \dfrac{di_k}{dt} + i_k R_k \ ; \\ i_\mu + i_k - i_d = 0 \ . \end{cases} \tag{2.7}$$

За цією системою рівнянь, ввівши позначення $K_k = L_k / L_\mu$, можна побудувати структурну схему, показану на рис. 2.8, б. Зі структурної схеми отримуємо вирази для струмів збудження та намагнічування:

$$i_d = \frac{sT_k(1+K_k)+1}{T_k\left(T_s(1+K_k)+T_\mu K_k\right)s^2 + \left(T_s + T_k(1+K_k)+T_\mu\right)s+1} \cdot \frac{U_d}{R_d};$$

$$i_\mu = \frac{sT_k K_k + 1}{T_k\left(T_s(1+K_k)+T_\mu K_k\right)s^2 + \left(T_s + T_k(1+K_k)+T_\mu\right)s+1} \cdot \frac{U_d}{R_d}.$$



а *б*

Рис. 2.8. Схема і структурна модель генератора постійного струму з уточненим урахуванням контуру вихрових струмів: а – електрична схема; б – структурна модель

Характеристичні корені знаменника (для типових співвідношень $T_s = 0.1T_\mu$, $T_k = 0.2T_\mu$, $K_k = 1.5$) дорівнюють: $T_1 = 0.26T_\mu$; $T_2 = 1.34T_\mu$. Оскільки корінь T_1 пропонованої моделі на порядок перевищує аналогічний корінь у

традиційній моделі, потрібно менше звертань до функції обчислень правих частин системи диференціальних рівнянь, що забезпечує більший середній крок розв'язання, а загалом – вищу достовірність відтворення динамічних режимів зміни струму збудження генератора постійного струму.

Моделювання двигуна постійного струму незалежного збудження

Найчастіше для розрахунків динаміки електроприводів використовують спрощену модель двигуна постійного струму (ДПС) [Д.10, Д.21] (рис. 2.9), що справедлива за таких допущень:

- індуктивність якірного кола незмінна (включає індуктивності якоря L_a, обмотки додаткових полюсів – ОДП і компенсаційної обмотки – КО);
- реакція якоря відсутня.

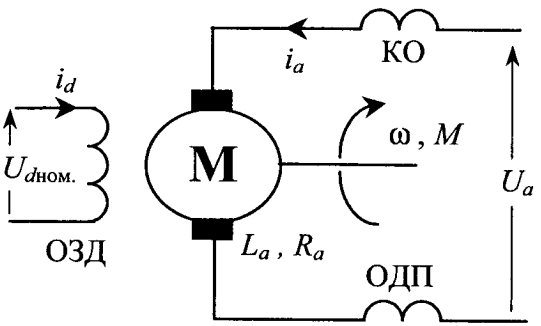


Рис. 2.9. Двигун постійного струму незалежного збудження з номінальним магнітним потоком

Для постійного і номінального збудження математичною моделлю двигуна є система диференціальних рівнянь:

$$\begin{cases} U_a = C\omega + i_a R_a + L_a \dfrac{di_a}{dt}; \\ M = M_c + J\dfrac{d\omega}{dt}, \end{cases} \tag{2.8}$$

де $C = k\Phi_н$; $k = \dfrac{p_n N}{2\pi a}$ – конструктивний коефіцієнт двигуна;

U_a – напруга на якорі двигуна;

ω – кутова швидкість вала двигуна;

M – момент на валі двигуна;

M_c – момент статичного навантаження на валі двигуна;

J – момент інерції якоря двигуна.

За відсутності даних для обчислення $\Phi_н$ і k можна скористатися виразом

$$C = \frac{U_н - I_н R_a}{\omega_н}.$$

Можлива інша форма запису математичної моделі ДПС. Поділимо рівняння (2.8) на R_a та позначимо $\dfrac{L_a}{R_a} = T_a$ – електромагнітна стала часу якірного кола. Тоді математичною моделлю ДПС будуть рівняння:

$$\begin{cases} \dfrac{di_a}{dt} = \dfrac{(U_a - C\omega)/R_a - i_a}{T_a}; \\ \dfrac{d\omega}{dt} = \dfrac{M - M_c}{J}. \end{cases} \tag{2.9}$$

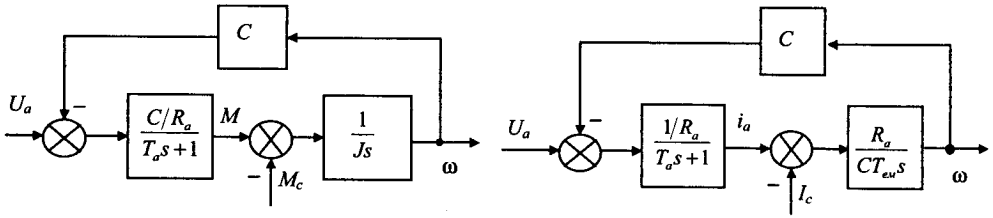
Позначимо

$$J\frac{d\omega}{dt}\cdot\frac{R_a}{C^2}\frac{C^2}{R_a} = T_{ем}\frac{d\omega}{dt}\frac{C^2}{R_a},$$

де $T_{ем} = J\dfrac{R_a}{C^2}$ – електромеханічна стала часу двигуна. Враховуючи, що $M = i_a C$ і $M_c = I_c C$, отримаємо таку математичну модель:

$$\begin{cases} \dfrac{di_a}{dt} = \dfrac{(U_a - C\omega)/R_a - i_a}{T_a}; \\ \dfrac{d\omega}{dt} = \dfrac{(i_a - I_c)R_a}{CT_{ем}}. \end{cases} \tag{2.10}$$

Математичним моделям (2.9) і (2.10) відповідають структурні моделі, які показані на рис. 2.10, *а, б* відповідно.



а *б*

Рис. 2.10. Структурні моделі ДПС для $\Phi = \Phi_н$

Нижче пропонуються варіанти реалізації у різних середовищах моделі двигуна постійного струму незалежного збудження.

Quick Basic

```
' ------------------------------------------------------------
'     Позначення змінних:
'     y(2) - напруга на якорі двигуна
'     y(3) - струм якоря
'     y(4) - швидкість двигуна
' ------------------------------------------------------------
CONST Ce = 2.34          ' Електромеханічна стала двигуна
CONST Ra = .34           ' Сумарний опір якірного кола
CONST Ta = .05           ' Стала часу якірного кола
CONST J = 9.2            ' Сумарний момент інерції привода
CONST Mc = 64.5          ' Момент опору на валу
     . . .  (розрахунок напруги на якорі y(2))   . . .

' Розрахунок похідної струму
f(3) = ((y(2) - Ce * y(4)) / Ra - y(3)) / Ta
' Розрахунок похідної швидкості
f(4) = (Ce * y(3) - Mc) / J
```

Turbo Pascal

```
{ ------------------------------------------------- }
{     Позначення змінних:                           }
{     y[2] - напруга на якорі двигуна               }
{     y[3] - струм якоря                            }
{     y[4] - швидкість двигуна                      }
{ ------------------------------------------------- }
const
   Ce = 2.34 ;       { Електромеханічна стала двигуна  }
   Ra = 0.34  ;      { Сумарний опір якірного кола     }
   Ta = 0.05  ;      { Стала часу якірного кола        }
   Mc = 64.5 ;       { Момент опору на валу            }
   J = 1.2 ;         { Сумарний момент інерції привода }
type
   vector = array [1..N] of double ;
var
   f, y : vector ;
begin
     . . .  (розрахунок напруги на якорі  y[2])   . . .

   { Розрахунок похідної струму   }
   f[3] := ((y[2] - Ce*y[4])/Ra - y[3])/Ta;
   { Розрахунок похідної швидкості }
   f[4] := (Ce*y[3] - Mc)/J;
end;
```

MathCAD

$$\text{Diff}(t, y) := \begin{bmatrix} \dfrac{\text{Ua}(t) - C \cdot y_1}{R_a} - y_0 \\ \dfrac{}{T_a} \\ \dfrac{C \cdot y_0 - M_C}{J} \end{bmatrix}$$

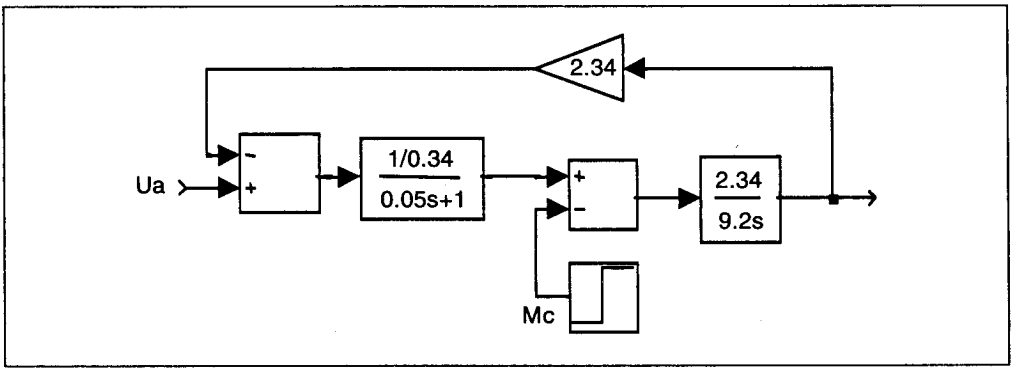
Вектор-функція правих частин системи диференціальних рівнянь, що описують двигун:

y_0 – *струм якоря;*
y_1 – *швидкість.*

MATLAB

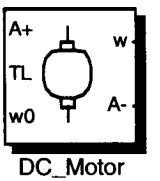
```
% ------------------------------------------------------
%     Позначення змінних:
%     y(2) - напруга на якорі двигуна
%     y(3) - струм якоря
%     y(4) - швидкість двигуна
% ------------------------------------------------------
Ce = 2.34 ;         % Електромеханічна стала двигуна
Ra = 0.34 ;         % Сумарний опір якірного кола
Ta = 0.05 ;         % Стала часу якірного кола
J = 9.2 ;           % Сумарний момент інерції привода
Mc = 64.5 ;         % Момент опору на валу

%    . . .  (розрахунок напруги на якорі  y(2) )  . . .

%  Розрахунок похідної струму
f(3) = ((y(2) - Ce * y(4)) / Ra - y(3))/Ta;
%  Розрахунок похідної швидкості
f(4) = (Ce * y(3) - Mc)/J;
end
```

Simulink

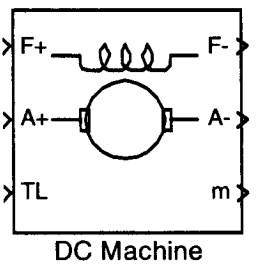


MATLAB

У версії 1.0 (MATLAB 5.2 R10)



У версії 1.1 (MATLAB 5.3 R11) і сучасніших



Моделювання двигуна постійного струму з регулюванням струму збудження

Модель двигуна у режимі двозонного регулювання (регулювання з ослабленням поля) є складнішою за попередню (рис. 2.11).

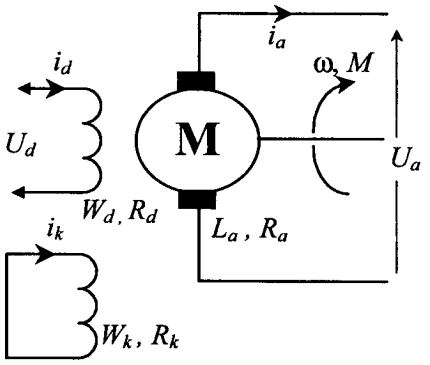


Рис. 2.11. Схема ДПС для двозонного регулювання швидкості

Приймемо такі допущення: реакцією якоря нехтуємо, а індуктивність якірного кола незмінна. У великих машинах (100 кВт і вище) помітна дія вихрових струмів, які виникають у масивних частинах магнітної системи під час зміни магнітного потоку Ф і протидіють зміні потоку полюсів. Традиційно їх дію розглядають як дію додаткової короткозамкненої фіктивної обмотки ($W_k - R_k$), яка розміщена на полюсах (див. модель кола збудження ГПС).

Запишемо систему диференціальних і алгебричних рівнянь механічної, електричної та магнітної рівноваги для такої моделі:

$$U_a = i_a R_a + k\Phi\omega + L_a \frac{di_a}{dt}; \tag{2.11}$$

$$M = M_C + J \cdot \frac{d\omega}{dt}; \tag{2.12}$$

$$U_d = i_d R_d + 2 p_n \xi W_d \frac{d\Phi}{dt}; \tag{2.13}$$

$$i_k R_k = 2 p_n \xi W_k \frac{d\Phi}{dt}; \tag{2.14}$$

$$F = i_d W_d - i_k W_k; \tag{2.15}$$

$$\Phi = \mathrm{f}(F). \tag{2.16}$$

Рівняння (2.11) і (2.12) описують динаміку електромагнітних і електромеханічних процесів двигуна і розглянуті вище. Рівняння (2.13)–(2.16) описують процеси у колі збудження. У наведених вище рівняннях використано такі позначення:

Φ – корисний магнітний потік одного полюса;

p_n – кількість пар полюсів;

W_d – кількість витків одного полюса обмотки збудження;

ξ = 1+(0.5...0.7)(σ–1) – коефіцієнт, який залежить від коефіцієнта розсіювання магнітного потоку;

σ = 1.12–1.18 – коефіцієнт розсіювання, тоді ξ = 1,06–1,13 – це означає, що є додаткове потокозчеплення з потоком, який замикається через повітря, тому $\xi > 1$;

F – сила намагнічування;

залежність $\Phi(F)$ нелінійна (рис. 2.12), тому в системі рівнянь вона подана виразом (2.16).

Для реалізації цифрової моделі можна використати або лінеаризовану модель кола збудження, або нелінійну з урахуванням кривої намагнічування (див. розділ про ГПС).

Рівняння (2.11) і (2.12) описують динаміку електромеханічних процесів у якірному колі двигуна, які вже розглядалися у попередньому розділі. Рівняння (2.13)–(2.16) описують процеси в колі збудження двигуна, над якими будуть виконані перетворення, щоб отримати диференціальні рівняння, зручні для розв'язування та побудови структурної моделі.

Рівняння (2.13) перетворимо так:

$$\frac{U_d}{R_d} = i_d + \frac{2 p_n \xi W_d^2 k_\Phi}{R_d} \cdot \frac{s\Phi}{k_\Phi W_d},$$

де $k_\Phi = \Phi_н / F_н = \Phi_н / (I_{dн} W_d)$ визначають з кривої намагнічування.

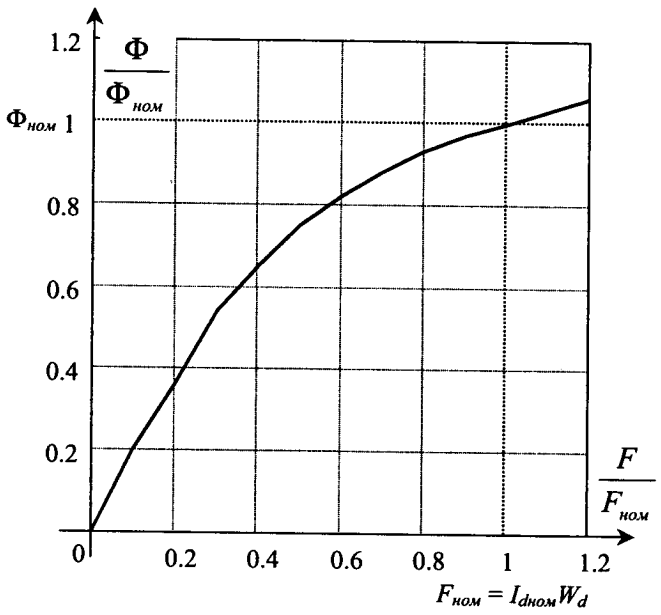


Рис. 2.12. Графік кривої намагнічування типового ДПС

Позначимо $T_d = \dfrac{2p_n \xi W_d^2 k_\Phi}{R_d}$ – електромагнітна стала часу обмотки збудження і в операторній формі запишемо

$$\Phi = \frac{k_\Phi W_d}{T_d s}\left(\frac{U_d}{R_d} - i_d\right). \tag{2.17}$$

З (2.14)

$$i_k = \frac{T_k s \Phi}{k_\Phi W_k}, \tag{2.18}$$

де $T_k = \dfrac{2p_n \xi W_k^2 k_\Phi}{R_k}$ – стала часу фіктивної короткозамкненої обмотки.

Вираз (2.16) подамо як $\Phi = k_\Phi F$ за умови лінійності характеристики намагнічування. Тоді з рівняння (2.15) отримаємо

$$i_d = \frac{\Phi}{k_\Phi W_d} + \frac{i_k W_k}{W_d}. \tag{2.19}$$

За (2.17)–(2.19) побудуємо структурну модель, яка зображена на рис. 2.13.

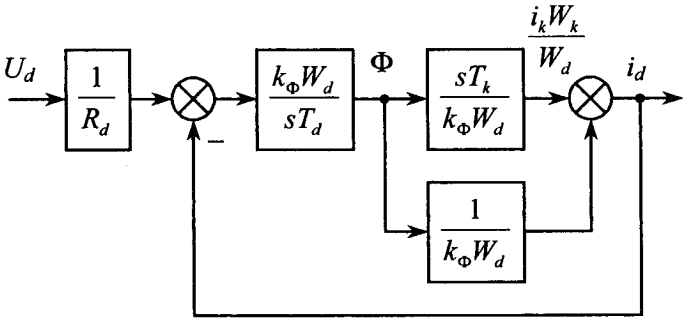


Рис. 2.13. Лінеаризована структурна схема моделі контуру збудження двигуна

Структурну модель зведемо до вигляду одноконтурної так:

$$W_{роз}(s) = \frac{k_\Phi W_d}{T_d s}\left(\frac{T_k s}{k_\Phi W_d} + \frac{1}{k_\Phi W_d}\right) = \frac{T_k s + 1}{T_d s};$$

$$W_{замк}(s) = \frac{T_k s + 1}{T_d s} \Bigg/ \left(1 + \frac{T_k s + 1}{T_d s}\right) = \frac{T_k s + 1}{(T_k + T_d)s + 1}. \tag{2.20}$$

Виділимо контур $i_d(s)$–$\Phi(s)$ і запишемо:

$$\frac{i_d(s)}{\Phi(s)} = \frac{T_k s + 1}{k_\Phi W_d};$$

а в зворотному напрямі

$$\frac{\Phi(s)}{i_d(s)} = \frac{k_\Phi W_d}{T_k s + 1}. \tag{2.21}$$

За (2.20) і (2.21) побудуємо структурну модель контуру збудження, яка зображена на рис. 2.14.

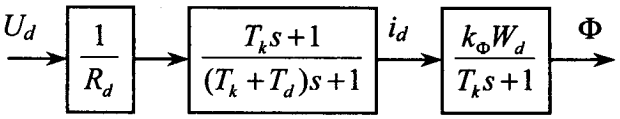


Рис. 2.14. Структурна схема моделі контуру збудження двигуна

З урахуванням рис. 2.14 повна структурна модель ДПС незалежного збудження з регулюванням магнітного потоку зображена на рис. 2.15. Щоб виключити операцію диференціювання, виконано нескладні структурні перетворення.

Варто зауважити, що у пропонованій на рис. 2.14 структурній моделі кола збудження ДПС використано спрощений підхід до урахування вихрових струмів у станині. Для їх точнішого описання треба застосувати спосіб, ще запропонований для кола збудження генератора постійного струму.

$$
\begin{aligned}
&\frac{di_a}{dt}=\frac{(U_a-k\Phi\omega)/R_a-i_a}{T_a}\ ;\\
&\frac{d\omega}{dt}=\frac{i_a k\Phi-M_c}{J}\ ;\\
&u_1=\left(U_d-(u_1+u_2)\right)\frac{T_k}{T_d}\ ;\\
&\frac{du_2}{dt}=\frac{U_d-(u_1+u_2)}{T_d}\ ;\\
&i_d=(u_1+u_2)/R_d\ ;\\
&\frac{d\Phi}{dt}=\frac{i_d W_d K_\Phi-\Phi}{T_k}\ .
\end{aligned}
\qquad (2.22)
$$

Для електричних машин постійного струму з масивною станиною можна прийняти $T_k = (0.2-0.3)T_d$. Значення електромагнітної сталої часу обмотки

збудження обчислено за виразом $T_d = 2p_n \xi W_d k_\Phi / R_d$, де R_d – опір кола збудження двигуна.

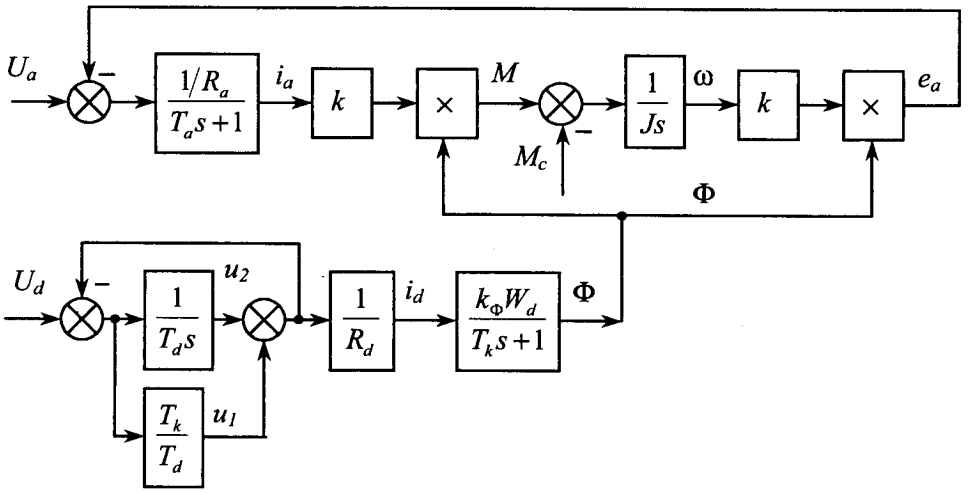


Рис. 2.15. Структурна схема моделі ДПС незалежного збудження для двозонного регулювання швидкості з урахуванням дії вихрових струмів

Якщо знехтувати дією вихрових струмів, що справедливо для машин з шихтованою станиною, то моделі значно спрощуються. Тоді $T_k = 0$ і з системи (2.22) отримаємо математичну модель:

$$\begin{aligned}
\frac{di_a}{dt} &= \frac{(U_a - k\Phi\omega)/R_a - i_a}{T_a}; \\
\frac{d\omega}{dt} &= \frac{i_a k\Phi - M_c}{J}; \\
\frac{di_d}{dt} &= \frac{U_d/R_d - i_d}{T_d}; \\
\Phi &= i_d w_d K_\Phi .
\end{aligned} \qquad (2.23)$$

Цій математичній моделі відповідає структурна схема моделі, яка зображена на рис. 2.16.

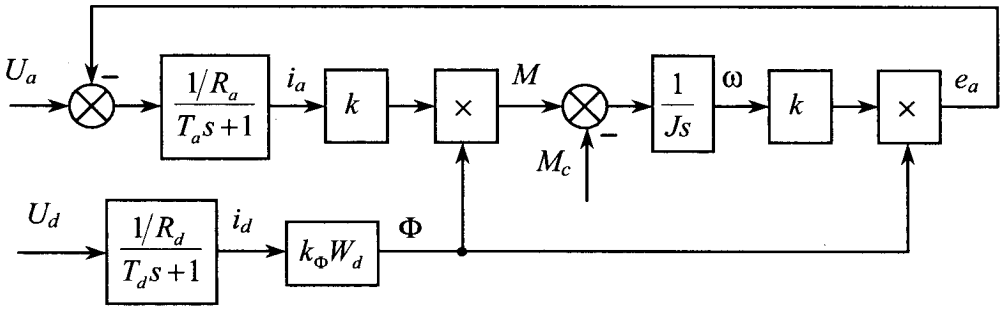


Рис. 2.16. Структурна схема моделі ДПС незалежного збудження з двозонним регулюванням швидкості без урахування дії вихрових струмів

Якщо необхідно в моделі врахувати нелінійність характеристики намагнічування, то це можна виконати аналогічно як для моделі генератора постійного струму.

Моделювання двигуна постійного струму послідовного збудження

Двигуни постійного струму послідовного збудження (серієсні двигуни) донині широко використовують в тягових електроприводах, зокрема, в електричному транспорті, завдяки їх механічним характеристикам.

Обмотка збудження ввімкнена в якірне коло послідовно з обмоткою якоря, потужність якої значно більша (порядків на два), ніж послідовної. Це призводить до форсування збудження на початкових стадіях пускогальмівних режимів, внаслідок чого магнітний потік змінюється в широкому діапазоні. Така швидка зміна магнітного потоку зумовлює появу вихрових струмів в полюсах і станині аналогічно, як і в двигунів незалежного збудження, якщо швидкість регулювати збудженням двигуна. Розмагнічувальну дію вихрових струмів врахуємо включенням на головних полюсах фіктивної короткозамкненої обмотки (рис. 2.17), яка має умовні витки W_k, опір R_k та струм i_k (див. модель генератора). Індуктивність якірного кола приймемо незмінною, реакцією якоря знехтуємо, а залежність $\Phi = k_\Phi F$ вважатимемо лінійною в околі робочої точки кривої $\Phi = f(F)$. Такі допущення дають змогу вважати модель лінійною.

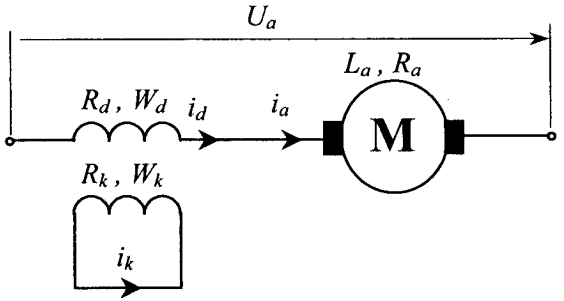


Рис. 2.17. Двигун постійного струму послідовного збудження в перехідних режимах

Запишемо систему рівнянь електричної, магнітної та механічної рівноваг, які описують динамічні процеси в двигуні, так:

$$\begin{aligned} &U_a = i_d\left(R_a + R_d\right) + L_a s i_a + k\Phi\omega + 2p_n\xi W_d s\Phi\,; \\ &i_k R_k = 2p_n\xi W_k s\Phi\,; \\ &F = i_a W_d - i_k W_k\,; \\ &\Phi = Fk_\phi\ ; \\ &M = i_a k\Phi\ ; \\ &Js\omega = M - M_c\,. \end{aligned} \tag{2.24}$$

Перше рівняння системи (2.24) поділимо на $R_\Sigma = R_a + R_d$, а останній його член домножимо на $W_d k_\Phi / W_d k_\Phi$ і, позначивши $L_a/R_\Sigma = T_a$, $T_d = \dfrac{2p_n\xi W_d^2 k_\Phi}{R_\Sigma}$, отримаємо в операторній формі

$$i_a = \frac{1}{sT_a}\left(\frac{U_a - k\Phi\omega}{R_\Sigma} - i_a - \frac{T_d s\Phi}{k_\Phi W_d}\right), \tag{2.25}$$

Друге рівняння поділимо на R_k, а праву його частину домножимо на $W_d k_\Phi / W_d k_\Phi$, у результаті чого отримаємо

$$i_k = \frac{T_k s\Phi}{k_\Phi W_k}.$$

З третього рівняння системи (2.24) виключимо i_k і, об'єднавши його з четвертим, визначимо $s\Phi$ у такій послідовності:

$$\frac{\Phi}{k_\Phi} = i_a W_d - \frac{T_k s\Phi}{k_\Phi W_k} W_k;$$

$$s\Phi = \frac{i_a k_\Phi W_d - \Phi}{T_k}. \qquad (2.26)$$

За виразами (2.25), (2.26) та системою (2.24) побудуємо структурну модель, яка показана на рис. 2.18.

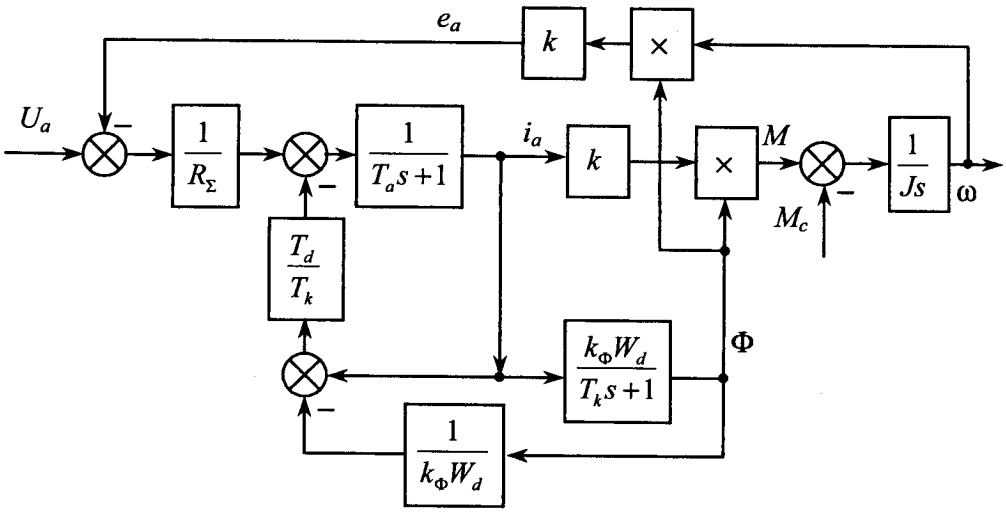


Рис. 2.18. Структурна схема моделі ДПС послідовного збудження з урахуванням вихрових струмів

Для серієсних двигунів, як і для двигунів незалежного збудження з регулюванням потоку середньої та великої потужності (більшої за 50 кВт), можна прийняти $T_k \approx (0{,}1 \ldots 0{,}2) T_d$, і чим більша потужність, тим більше значення T_k.

Математичну модель, до складу якої входять диференціальні рівняння першого порядку, що подані у нормальній формі Коші, зручно записувати,

користуючись структурною моделлю (рис. 2.18), до складу якої входять три динамічні ланки – дві аперіодичні та одна інтегральна, так:

$$\begin{cases} \dfrac{di_a}{dt} = \dfrac{1}{T_a}\left(\dfrac{U_a - k\Phi\omega}{R_\Sigma} - \dfrac{T_d}{T_k}\left(i_a - \dfrac{\Phi}{k_\Phi W_d}\right) - i_a\right); \\ \dfrac{d\Phi}{dt} = \dfrac{i_d k_\phi W_d - \Phi}{T_k}; \\ \dfrac{d\omega}{dt} = \dfrac{i_a k\Phi - M_c}{J}. \end{cases} \tag{2.27}$$

Якщо знехтувати ефектом дії вихрових струмів, то модель істотно спрощується. Виходимо з таких рівнянь:

$$\begin{aligned} U_a &= i_a\left(R_a + R_d\right) + s i_a L_a + k\Phi\omega + 2p_n\xi W_d s\Phi \ ; \\ F &= i_a W_d \ ; \\ \Phi &= F k_\phi \ . \end{aligned} \tag{2.28}$$

Нескладні перетворення приводять до такої структурної (рис. 2.19) моделі.

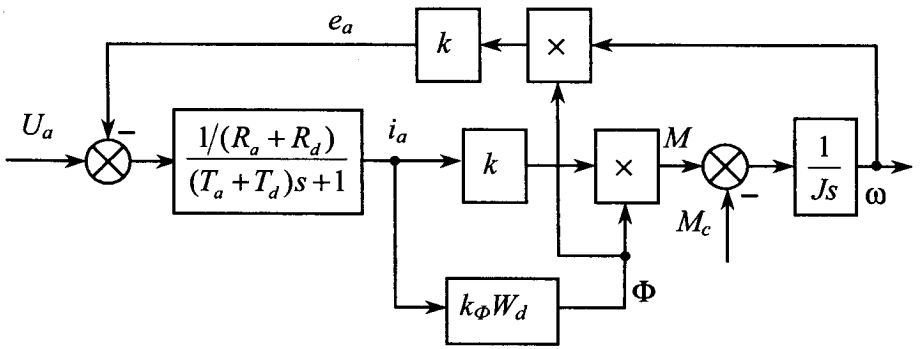


Рис. 2.19. Структурна модель ДПС послідовного збудження без урахування вихрових струмів

Двигуни послідовного збудження можуть також працювати в режимі зашунтованого якірного кола (рис. 2.20). Розглянемо таку модель.

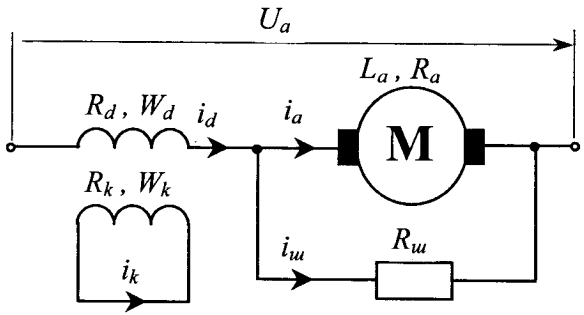


Рис. 2.20. Двигун постійного струму послідовного збудження з шунтуванням якоря

Динамічні режими двигуна опишемо системою таких диференціальних і функціональних рівнянь електричної та механічної рівноваги і рівняннями перетворення енергії:

$$\begin{aligned}
&U_a = i_d R_d + i_a R_a + k\Phi\omega + 2p_n\xi W_d s\Phi\,;\\
&U_a = i_d R_d + 2p_n\xi W_d s\Phi + i_ш R_ш\,;\\
&i_d = i_a + i_ш\,;\\
&F = i_d W_d - i_k W_k\,;\\
&i_k R_k = 2p_n\xi W_k s\Phi\,;\\
&\Phi = Fk_ф\;;\\
&M = i_a k\Phi\,;\\
&Js\omega = M - M_c\,.
\end{aligned} \tag{2.29}$$

Після нескладних перетворень перше рівняння системи (2.29) зводимо до вигляду:

$$i_a = \frac{1}{sT_a}\left(\frac{U_a - k\Phi\omega}{R_{ad}} - i_a\frac{R_a}{R_{ad}} - i_d\frac{R_d}{R_{ad}} - T_{ad}\frac{s\Phi}{k_\Phi W_d}\right), \tag{2.30}$$

де

$$R_{ad} = R_a + R_d\,; \quad T_{ad} = \frac{2p_n\xi W_d^2 k_\Phi}{R_{ad}}\,; \quad T_a = \frac{L_a}{R_{ad}}.$$

З другого і третього рівнянь отримуємо

$$i_d = \frac{U_a}{R_{dш}} - i_a \frac{R_ш}{R_{dш}} - T_{dш} \frac{s\Phi}{k_\Phi W_d}, \tag{2.31}$$

де

$$R_{dш} = R_d + R_ш; \qquad T_{dш} = \frac{2 p_n \xi W_d^2 k_\Phi}{R_{dш}}.$$

Перетворивши четверте, п'яте і шосте рівняння системи (2.29) і, позначивши,

$$T_k = \frac{2 p_n \xi W_d^2 k_\Phi}{R_k}$$

маємо

$$\frac{s\Phi}{k_\Phi W_d} = \frac{1}{T_k}\left(i_d - \frac{\Phi}{k_\Phi W_d} \right). \tag{2.32}$$

Структурна модель двигуна, яка зображена на рис. 2.21, побудована за рівняннями (2.30)–(2.32) і двома останніми рівняннями системи (2.29).

Позначимо $i_\Phi = \Phi / k_\Phi W_d$ і $A = i_d - i_\Phi$, тоді математична модель матиме вигляд:

$$\begin{aligned}
&\frac{di_a}{dt} = \frac{1}{T_a}\left(\frac{U_a - k\Phi\omega}{R_{ad}} - i_a \frac{R_a}{R_{ad}} - i_d \frac{R_d}{R_{ad}} - A\frac{T_d}{T_k} \right); \\
&i_\Phi = \frac{\Phi}{k_\Phi W_d}; \\
&i_d = \frac{U_a}{R_{dш}} + i_a \frac{1}{1 + R_d / R_ш} - A\frac{T_{dш}}{T_k}; \\
&A = i_d - i_\Phi; \\
&\frac{di_\Phi}{dt} = \frac{A}{T_k}; \\
&\Phi = i_\Phi k_\phi W_d \\
&M = i_a k\Phi; \\
&\frac{d\omega}{dt} = \frac{M - M_c}{J}.
\end{aligned} \tag{2.33}$$

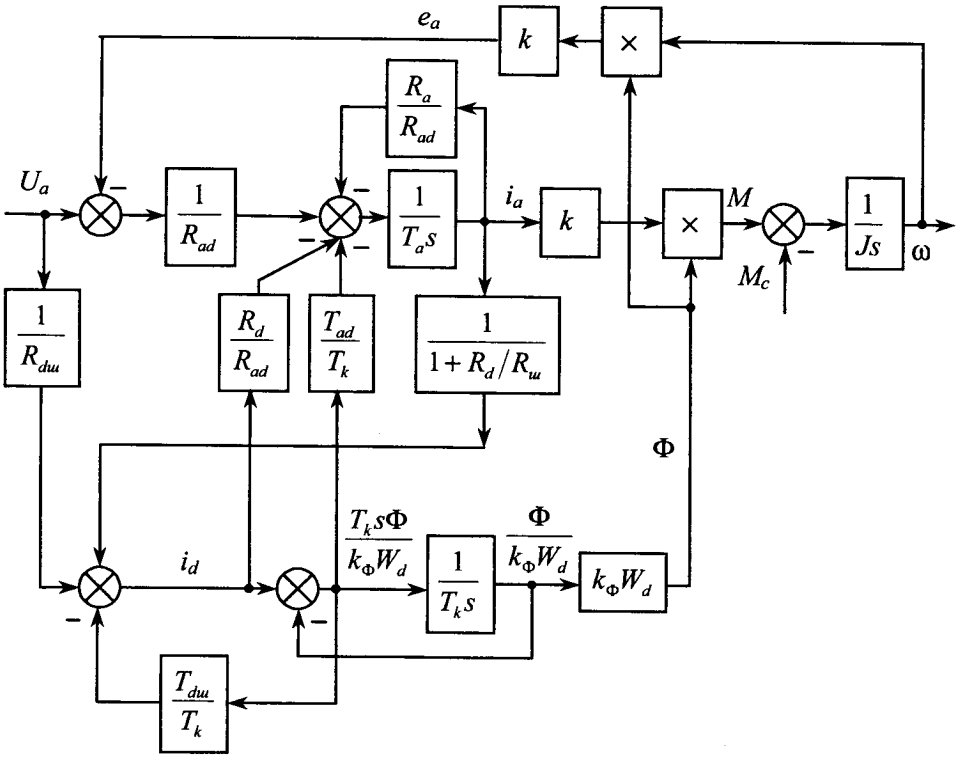


Рис. 2.21. Структурна схема моделі ДПС послідовного збудження з шунтуванням якоря і урахуванням дії вихрових струмів

Якщо не враховувати дію вихрових струмів, то в системі рівнянь (2.29) матимемо: $i_k W_k = 0$; $F = i_d W_d$. Тоді $\Phi = k_\Phi i_d W_d$, і система рівнянь, які описують процеси в електромагнітних колах, буде такою:

$$\frac{di_a}{dt} = \frac{1}{T_a}\left(\frac{U_a - k\Phi\omega}{R_{ad}} - i_a\frac{R_a}{R_{ad}} - i_d\frac{R_d}{R_{ad}} - si_dT_d\right);$$

$$\frac{di_d}{dt} = \frac{1}{T_{dш}}\left(\frac{U_a}{R_{dш}} + i_a\frac{R_ш}{R_{dш}} - i_d\right); \qquad (2.34)$$

$$\Phi = i_d k_\Phi W_d \ .$$

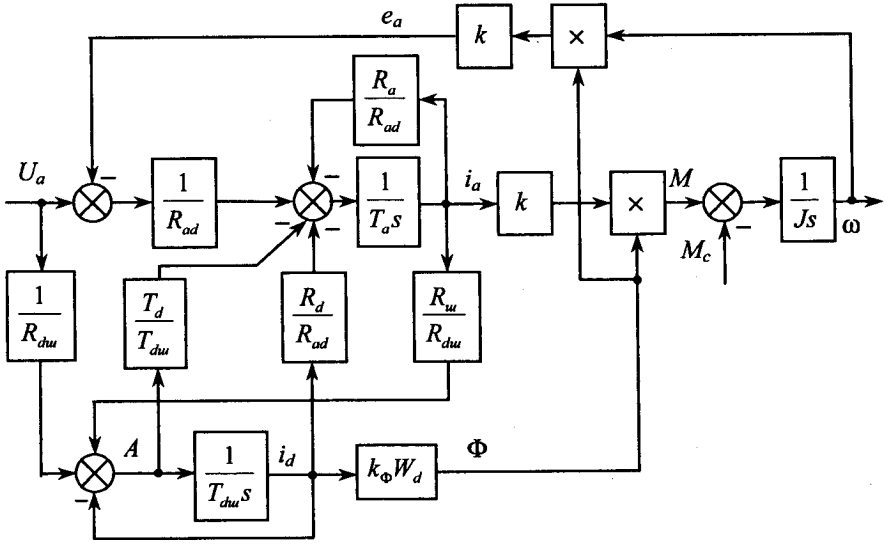


Рис. 2.22. Структурна схема моделі ДПС послідовного збудження з шунтуванням якоря без урахування дії вихрових струмів

Для обчислень зручно позначити:

$$A = s\ \ i_a T_{dш} = \frac{U_a}{R_{dш}} + i_a \frac{R_ш}{R_{dш}} - i_d\,. \tag{2.35}$$

З урахуванням (2.29) і (2.30) матимемо таку математичну модель ДПС послідовного збудження без урахування розмагнічувальної дії вихрових струмів:

$$\begin{aligned}
&\frac{di_a}{dt} = \frac{1}{T_{ad}}\left(\frac{U_a - k\Phi\omega}{R_{ad}} - i_a\frac{R_a}{R_{ad}} - i_d\frac{R_d}{R_{ad}} - A\frac{T_{ad}}{T_{dш}}\right);\\
&A = \frac{U_a}{R_{dш}} + i_a\frac{R_ш}{R_{dш}} - i_d;\\
&\frac{di_d}{dt} = \frac{A}{T_{dш}};\\
&\Phi = i_d k_\phi W_d;\\
&\frac{d\omega}{dt} = \frac{1}{J}\left(i_a k\Phi - M_c\right).
\end{aligned} \tag{2.36}$$

Моделювання двигуна постійного струму змішаного збудження

Двигуни постійного струму змішаного збудження (їх ще називають компаундними – від англ. *compound – змішувати*) також використовують у тягових електроприводах. Магнітний потік таких двигунів створюється двома обмотками – незалежною з опором $R_{dн}$ і кількістю витків $W_{dн}$ та послідовною (R_{dn} , W_{dn}) (рис. 2.23), тому $\Phi = k_{\Phi}(i_{dn}W_{dn} + i_{dн}W_{dн})$. Ролі послідовної та незалежної обмоток у створенні магнітного потоку для різних режимів роботи двигуна неоднакові і це варто враховувати під час створення моделі та вибору допущень для лінеаризації моделі. На рис. 2.24 показано графік залежності потоку збудження від сумарних ампер-витків, які створюються струмами послідовної та незалежної обмоток.

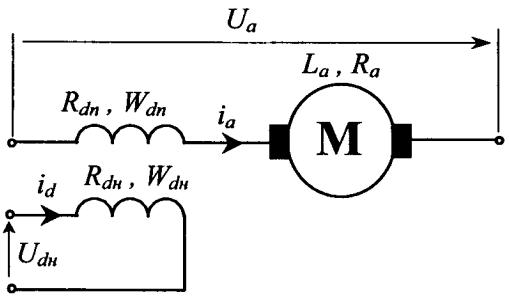


Рис. 2.23. Двигун постійного струму змішаного збудження

Складова потоку, утвореного незалежною обмоткою $I_{dн}^{ном}W_{dн}$, де $I_{dн}^{ном}$ – номінальний струм збудження незалежної обмотки, є незмінною, як і в двигунів незалежного збудження. Тому для i_a = 0 (режим ідеального неробочого ходу) потік двигуна визначається незалежною обмоткою – $\Phi_н$, і для режиму двигуна становить (0.7–0.8)$\Phi_н$. Отже, частка потоку послідовної обмотки становить (0.2–0.3) від загального, тому дії вихрових струмів, які зумовлені зміною магнітного потоку, можна не враховувати.

У генераторному режимі ефективність розмагнічення послідовної обмотки збільшується, а це призводить до зменшення ефективності рекуперативного гальмування. Тому в таких приводах застосовують шунтування послідовної

обмотки в генераторних режимах, чим покращується гальмування та зростає ефект рекуперації енергії.

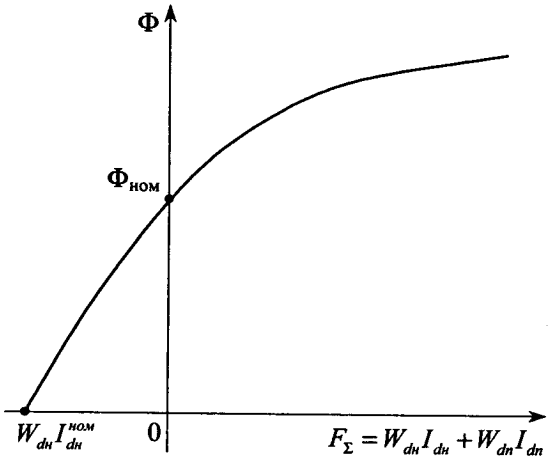


Рис. 2.24. Характеристика $\Phi = \mathrm{f}(F_\Sigma)$ *двигуна змішаного збудження*

Зважаючи на це, для спрощення моделі можна не враховувати розмагнічення вихровими струмами. Вихідні рівняння для двигуна змішаного збудження запишемо за аналогією з (2.29):

$$U_a = i_a(R_a + R_{dn}) + sL_a i_a + 2p_n \xi W_n s\Phi + k\Phi\omega;$$
$$U_{dн} = i_{dн}R_{dн} + 2p_n \xi W_{dн} s\Phi;$$
$$F = i_{dn}W_{dn} + i_{dн}W_{dн};$$
$$\Phi = F_\Sigma k_\phi \quad .$$

Для зручності побудови структурної моделі зробимо деякі перетворення.

$$\frac{U_a - k\Phi\omega}{R_{ad}} = i_a + si_a T_a + T_{dn}\frac{s\Phi}{k_\Phi W_{dn}},$$

де

$$R_{ad} = R_a + R_{dn} \;;\; T_a = L_a / R_{ad} \;;\; T_{dn} = \frac{2p_n \xi W_{dn}^2 k_\Phi}{R_{ad}};$$

$$i_a = \frac{1}{T_a s + 1}\left(\frac{U_a - k\Phi\omega}{R_\Sigma} - T_{dn}\frac{s\Phi}{k_\Phi W_{dn}}\right); \tag{2.37}$$

$$\frac{U_{dн}}{R_{dн}} = i_{dн} + T_{dн}\frac{s\Phi}{k_\Phi W_{dн}},$$

де

$$T_{dн} = \frac{2p_n\xi W_{dн}^2 k_\Phi}{R_{dн}},\ i_{dн} = \frac{\Phi}{k_\Phi W_{dн}} - i_a\frac{W_{dn}}{W_{dн}}.$$

Після підстановки отримаємо

$$T_{dн}\frac{s\Phi}{k_\Phi W_{dн}} = \frac{U_{dн}}{R_{dн}} - \frac{\Phi}{k_\Phi W_{dн}} + i_a\frac{W_{dn}}{W_{dн}}. \tag{2.38}$$

З використанням виразів (2.37), (2.38) побудована структурна схема математичної моделі, яка показана на рис. 2.25.

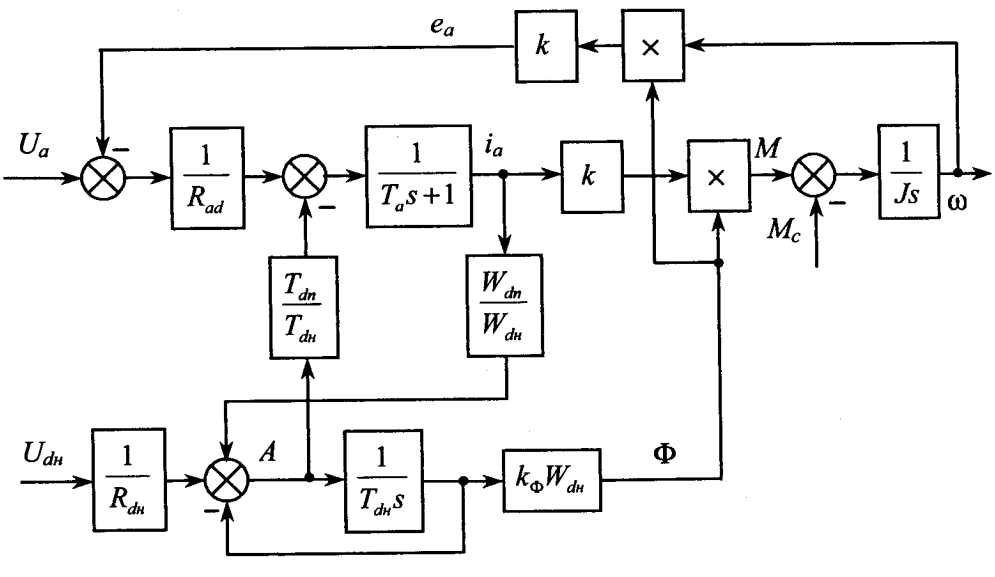


Рис. 2.25. Структурна схема моделі ДПС змішаного збудження

Математична модель має такий вигляд:

$$\begin{aligned}
&\frac{di_a}{dt} = \frac{1}{T_a}\left(\frac{U_a - k\Phi\omega}{R_{ad}} - i_a - A\frac{T_{dn}}{T_{dн}}\right);\\
&A = \frac{U_a}{R_{dн}} + i_a\frac{W_{dn}}{W_{dн}} - \frac{\Phi}{k_\Phi W_{dн}};\\
&\frac{d\Phi}{dt} = \frac{A}{T_{dн}};\\
&M = i_a k\Phi;\\
&\frac{d\omega}{dt} = \frac{M - M_c}{J}.
\end{aligned} \qquad (2.39)$$

2.3. Математичні моделі електричних машин змінного струму

Існує декілька рівнів деталізації моделей електричних машин змінного струму, але всі вони ґрунтуються на низці допущень, що визначаються поняттям "ідеалізована машина" [Д.13, Д.27]:

1) електрорушійні сили, струми і магнітні потоки двигуна змінюються синусоїдно в часі;
2) магнітне поле у повітряному проміжку і в сталі статора та ротора розподіляється синусоїдно;
3) обмотки статора і ротора симетричні, а повітряний проміжок навколо ротора рівномірний, внаслідок чого процеси у трьох фазах відбуваються одночасно;
4) гістерезис і вихрові струми, а, отже, і втрати у сталі відсутні;
5) механічні втрати у двигуні відсутні, тому момент на валі двигуна дорівнює електромагнітному моменту;
6) реактивні опори статора і ротора не змінюються зі зміною магнітного потоку двигуна, тобто машина ненасичена.

Проста модель асинхронного двигуна

Для робочої лінійної ділянки механічної характеристики асинхронного двигуна (АД) (в межах $-0.8M_{кр} \le M \le 0.8M_{кр}$) можна використати таке рівняння, що пов'язує момент і швидкість АД [Д.13, Д.27]:

$$\frac{M}{\omega_0 - \omega} = \frac{2M_{кр}/\omega_0 s_{кр}}{T_e s + 1}, \qquad (2.40)$$

де M – електромагнітний момент двигуна;

$s_{кр}$ – критичне ковзання асинхронного двигуна;

ω_0 – синхронна кутова швидкість;

ω – кутова швидкість обертання ротора асинхронного двигуна;

$M_{кр} = \dfrac{3U_ф^2}{2\omega_0\left(r_1 + \sqrt{r_1^2 + (x_1 + x_2')^2}\right)}$ – критичний момент асинхронного двигуна;

$T_e = \dfrac{1}{\omega_{ел} s_{кр}}$ – електромагнітна стала часу,

де $\omega_{ел} = 2\pi f_1$ – кутова частота напруги статора;

f_1 – частота напруги електричної мережі живлення.

Значення $s_{кр}$ обчислюють за формулою

$$s_{кр} = \frac{r_2'}{\sqrt{r_1^2 + (x_1 + x_2')^2}},$$

де r_1 – активний опір обмотки статора;

x_1 – індуктивний опір розсіювання обмотки статора;

r_2' – зведений до статора сумарний активний опір кола ротора;

x_2' – зведений до статора індуктивний опір розсіювання обмотки ротора.

Згідно з цими виразами маємо таку структурну (рис. 2.26) і математичну моделі АД:

$$\frac{dM}{dt} = \left((\omega_0 - \omega)\cdot 2M_{кр}/\omega_0 s_{кр} - M\right)/T_e \ ;$$
$$\frac{d\omega}{dt} = \left(M - M_c\right)/J \ . \qquad (2.41)$$

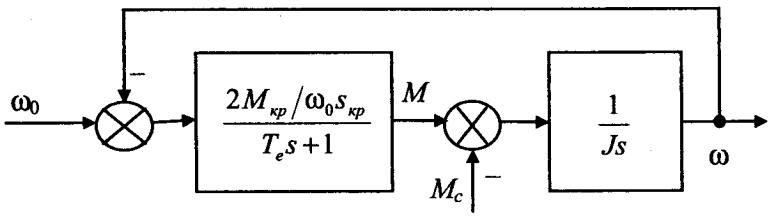


Рис. 2.26. Структурна схема лінеаризованої моделі асинхронного двигуна

Врахувати істотну нелінійність механічної характеристики АД можна, якщо момент розраховувати за простою

$$M = \frac{2M_{кр}}{\frac{s}{s_{кр}} + \frac{s_{кр}}{s}}$$

чи уточненою

$$M = \frac{2M_{кр}(1+a)}{\frac{s}{s_{кр}} + \frac{s_{кр}}{s} + 2a}$$

формулами Клоса,

де $s = \dfrac{\omega_0 - \omega}{\omega_0}$ – ковзання АД (*в інших формулах літерою* **s** *позначено оператор Лапласа*);

$a = r_1/r_2'$ – співвідношення активних опорів статора і ротора.

Тоді математична модель АД буде описуватися такою системою рівнянь:

$$\begin{aligned} s &= \frac{\omega_0 - \omega}{\omega_0}\ ; \\ \frac{dM}{dt} &= \frac{\dfrac{2M_{кр}(1+a)}{\dfrac{s}{s_{кр}} + \dfrac{s_{кр}}{s} + 2a} - M}{T_e}\ ; \\ \frac{d\omega}{dt} &= (M - M_c)/J\ . \end{aligned} \qquad (2.42)$$

Для точнішого порівняно з формулою Клоса відтворення основних координат моделі можна скористатися залежностями [Д.3, Д.13]:

$$M = \frac{3U_\phi^2}{\omega_0} \cdot \frac{r_2' \cdot s_a}{A(\nu, s_a)}$$

– момент двигуна;

$$I_1 = U_\phi \sqrt{\frac{C(s_a)}{A(\nu, s_a)}}$$

– струм статора;

де $A(\nu, s_a) = (b^2 + c^2\nu^2)s_a^2 + 2r_1 r_2' \nu s_a + (d^2 + e^2\nu^2)r_2'^2$;

$C(s_a) = \dfrac{r_2'^2}{x_m^2} + (1 + k_{\sigma 2})^2 s_a^2$;

$b = r_1(1 + k_{\sigma 2})$; $c = x_m k_\sigma$; $d = \dfrac{r_1}{x_m}$; $e = 1 + k_{\sigma 1}$;

$k_{\sigma 1} = x_1 / x_m$; $k_{\sigma 2} = x_2' / x_m$; $k_\sigma = k_{\sigma 1} + k_{\sigma 2} + k_{\sigma 1} k_{\sigma 2}$ – коефіцієнти розсіювання;

x_m – зведений до статора індуктивний опір взаємної індуктивності між статором і ротором на номінальній частоті живлення;

U_ϕ – фазна напруга на статорі двигуна;

ω_1 – частота напруги живлення статора двигуна;

$s_a = \dfrac{\omega_1 - \omega}{\omega_0}$ – абсолютне ковзання;

$\nu = \dfrac{\omega_1}{\omega_0}$ – відносна частота напруги статора.

У такому разі модель АД є такою (зі збереженням прийнятих вище позначень):

$$\begin{cases} s_a = \dfrac{\omega_1 - \omega}{\omega_0}; \\ \dfrac{dM}{dt} = \dfrac{\dfrac{3U_\phi^2}{\omega_0} \cdot \dfrac{r_2' \cdot s_a}{A(\nu, s_a)} - M}{T_e}; \\ \dfrac{d\omega}{dt} = \dfrac{M - M_c}{J}. \end{cases} \qquad (2.43)$$

Варто відзначити, що ця модель дає змогу також отримувати значення струму статора АД.

Статичні моделі асинхронного двигуна, побудовані на основі однофазних заступних схем

Середнє значення моменту асинхронного двигуна визначається подібно, як і для двигуна постійного струму, за таким виразом:

$$M_{сер} = \frac{1}{\sqrt{2}} m_2 p w_2 k_{об2} \Phi_m I_2 \cos \varphi_2 = k_m \Phi_m I_2 \cos \varphi_2 .$$

Відсутність даних для обчислення значення коефіцієнта k_m – (кількість витків w_2 фази ротора, коефіцієнт $k_{об2}$ та інші) ускладнює практичне використання цієї залежності.

Інший вираз для електромагнітного моменту $M_{сер}$ можна отримати через електромагнітну потужність, яка передається зі статора в ротор:

$$M_{сер} = \frac{P_{ем}}{\omega_0} = \frac{m_2 \cdot E_2 \cdot I_2 \cdot \cos\ \varphi_2}{\omega_0},$$

де m_2 – фазність ротора; ω_0 – синхронна швидкість поля статора;

E_2, I_2 – ЕРС і струм фази ротора;

$\cos \varphi_2 = r_2 / z_2$ ротора.

Для визначення цих складових потужності використовують заступні схеми фази двигуна і простішою є залежність:

$$P_{ем} = 3I_2'^2 \cdot r_2'/s$$

де $I'_2 = I_2\, k_i$, $r'_2 = r_2\, k_e\, k_i$ – зведені до обмотки статора струм та активний опір фази ротора;

k_e, k_i – коефіцієнти трансформації відповідно за напругою та струмом.

Відомі три заступні схеми АД: Т-подібна, Г-подібна і Г-подібна уточнена, які забезпечують різну точність або різну наближеність до реальних процесів у двигуні, оскільки також є наближеними моделями оригіналу.

У Т-подібній заступній схемі (рис. 2.27) контур намагнічування r_μ, x_μ під'єднаний послідовно після первинного (статорного) кола. Тому струм намагнічування i_μ, який створює основний магнітний потік Φ_m, залежить від навантаження первинного кола і його параметрів r_1 та x_1, зумовлюючи разом з $\cos \varphi_2$ непропорційність між моментом і струмом статора двигуна.

Обчислити значення параметрів схеми нескладно за відомостями, які подаються у звичайних каталогах чи довідникових матеріалах.

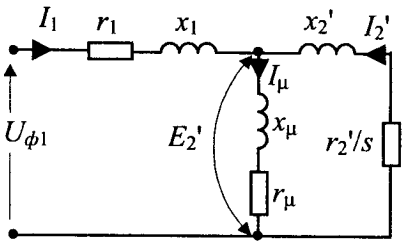


Рис. 2.27. Т-подібна заступна схема фази асинхронного двигуна

Приклад 1. Обчислити значення параметрів заступної схеми асинхронного двигуна з фазним ротором МТВ 412–6, $P_н = 30$ кВт, $n_н = 970$ об/хв, $I_{c0} = 42$ А, $r_1 = 0{,}125$ Ом, $x_1 = 0.23$ Ом, $r_2 = 0.055$ Ом, $x_2 = 0.225$ Ом, $k_e = 1.4$, $U = 380$ В, $\cos\varphi_0 = 0.06$.

Повний опір неробочого ходу

$$z_0 = \frac{U_{ф1}}{I_{c0}} = \frac{220}{42} = 5.238 \text{ Ом}.$$

Індуктивний та активний опори неробочого ходу:

$$x_0 = x_\mu + x_1 = z_0\sqrt{1-\cos^2\varphi} = 5.238\sqrt{1-0.06^2} = 5.23 \text{ Ом};$$

$$r_0 = r_\mu + r_1 = z_0 \cdot \cos\varphi = 5.238 \cdot 0.06 = 0.314 \text{ Ом}.$$

Індуктивний та активний опори контуру намагнічування:

$$x_\mu = x_0 - x_1 = 5.23 - 0.23 = 5 \text{ Ом}; \; r_\mu = r_0 - r_1 = 0.314 - 0.125 = 0.189 \text{ Ом}.$$

Індуктивність головного потоку (контуру намагнічування):

$$L_m = \frac{x_\mu}{\omega_{0e}} = \frac{5}{2\pi \cdot 50} = 0.0159 \text{ Гн}.$$

Зведені до статора опори нерухомого ротора:

$$r_2' = r_2 \cdot k_e^2 = 0.055 \cdot 1.4^2 = 0.108 \text{ Ом};$$

$$x_2' = x_2 \cdot k_e^2 = 0.225 \cdot 1.4^2 = 0.441 \text{ Ом}.$$

Індуктивності полів розсіювання:

$$L_{1\sigma} = \frac{x_1}{\omega_{0e}} = \frac{0.23}{314.16} = 0.73 \cdot 10^{-3} \text{ Гн};$$

$$L_{2\sigma}' = \frac{x_2'}{\omega_{0e}} = \frac{0.441}{314.16} = 1.4 \cdot 10^{-3} \text{ Гн}.$$

Повні еквівалентні індуктивності фаз статора і ротора:

$$L_1 = L_m + L_{1\sigma} = 0.0159 + 0.73 \cdot 10^{-3} = 0.0166 \text{ Гн};$$

$$L_2 = L_m + L_{2\sigma}' = 0.0159 + 1.4 \cdot 10^{-3} = 0.0173 \text{ Гн}.$$

Повні опори:

$$z_1 = \sqrt{r_1^2 + x_1^2} = \sqrt{0.125^2 + 0.23^2} = 0.262 \text{ Ом};$$

$$z_2' = \sqrt{r_2'^2 + x_2'^2} = \sqrt{0.108^2 + 0.441^2} = 0.453 \text{ Ом};$$

$$z_\mu = \sqrt{r_\mu^2 + x_\mu^2} = \sqrt{0.189^2 + 5^2} = 5.003 \text{ Ом}.$$

За цими отриманими даними статичну механічну характеристику $M(s)$ можна побудувати, послідовно обчисливши:

$$z_2' = \sqrt{(r_2'/s)^2 + x_2'^2}; \quad E_1 = U_{\phi 1} - I_1 \cdot z_1; \quad I_2' = \frac{E_1}{z_2'}; \quad I_1 = \frac{U_{\phi 1}}{z_1 + \frac{z_2' \cdot z_\mu}{z_2' + z_\mu}};$$

$$M = \frac{3I_2'^2 \cdot r_2'}{\omega_0 \cdot s}. \tag{2.44}$$

Таку саму залежність $M(s)$ можна отримати з виразів:

$$M = \frac{3E_1 I_2' \cdot \cos\varphi_2}{\omega_0}, \tag{2.45}$$

де $\cos\varphi_2 = r_2/z_2$.

Широко використовується для побудови механічної характеристики асинхронного двигуна вже згадувана раніше відома формула Клоса

$$M = \frac{M_{кр}(2+q)}{\frac{s_{кр}}{s} + \frac{s}{s_{кр}} + q}, \tag{2.46}$$

де $M_{кр} = \frac{3U_\phi^2}{2\omega_0(r_1 + \sqrt{r_1^2 + (x_1 + x_2')^2})}$; $s_{кр} = \frac{r_2'}{\sqrt{r_1^2 + (x_1 + x_2')^2}}$; $q = 2r_1 \cdot s_{кр}/r_2'$.

Отримання виразів (2.46) ґрунтується на Г-подібній заступній схемі (рис. 2.28) і не враховує залежність струму намагнічування I_μ, а, отже, і магнітного потоку Φ_m від струму статора, тому що контур намагнічування винесений на затискачі первинного контуру, що зрозуміло з таких виразів:

$$I_2' = \frac{U_{\phi 1}}{\sqrt{(r_1 + r_2'/s)^2 + (x_1 + x_2')^2}}; \tag{2.47}$$

$$M = \frac{3I_2'^2 \cdot r_2'/s}{\omega_0} = \frac{3U_{\phi 1}^2 \cdot r_2'}{\omega_0 s\left((r_1 + r_2'/s)^2 + (x_1 + x_2')^2\right)}, \tag{2.48}$$

що далі приводить до виразу (2.46).

На рис. 2.29 зображені характеристики $M(s)$ асинхронного двигуна для різних заступних схем. Характеристика 1 побудована за розрахунками виразів (2.44) і (2.45), а характеристика 2 – виразів (2.46) і (2.48).

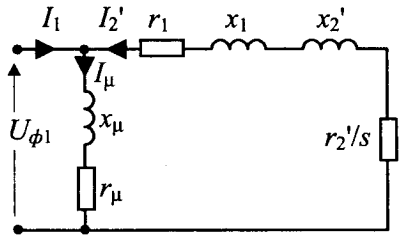


Рис. 2.28. Г-подібна заступна схема фази асинхронного двигуна

Замість Г-подібної можна користуватись Г-подібною уточненою схемою (рис. 2.30). Така схема завдяки застосуванню поправкового коефіцієнта $\sigma_1 = z_1/z_m$ наближає її до Т-подібної та забезпечує достовірніше моделювання, ніж Г-подібна.

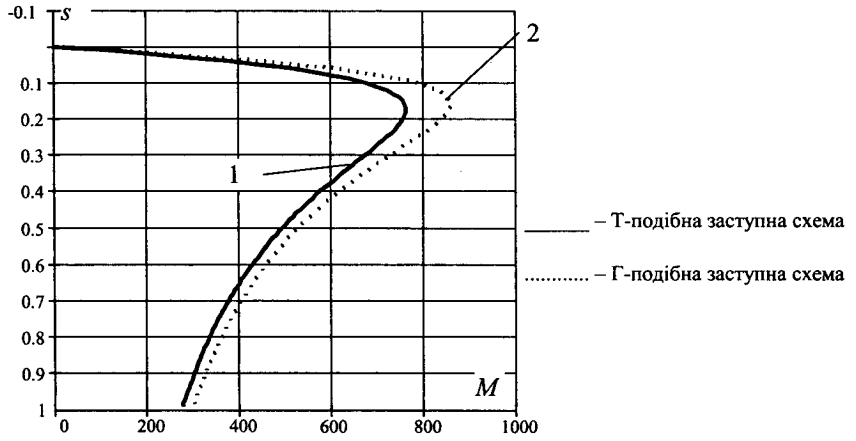


Рис. 2.29. Механічні характеристики моделей асинхронного двигуна з прикладу 1, побудованих за Т-подібною (1) та Г-подібною (2) заступними схемами

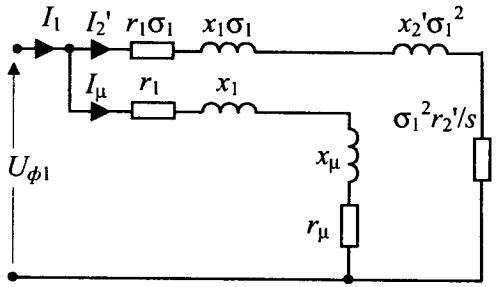


Рис. 2.30. Уточнена Г-подібна заступна схема фази асинхронного двигуна

Динамічні моделі асинхронного двигуна, побудовані на основі однофазних заступних схем

У разі під'єднання кола R, L до джерела синусоїдної напруги $U_m \cdot \sin \omega t$ в колі буде протікати перехідний струм, який складається з вимушеної та вільної складових. Амплітуда вимушеної складової струму

$$i_{вим} = \frac{U_m}{z},$$

де

$$z = \sqrt{R^2 + (\omega \cdot L)^2},$$

а вільна складова

$$i_{вс} = A \cdot e^{-\frac{t}{\tau}},$$

де

$$\tau = \frac{L}{R}, \qquad A = -\frac{U_m}{z} \quad \text{для } t = 0.$$

На підставі цього однофазну заступну схему R-L кола змінного струму будемо розглядати як еквівалентне коло постійного струму, в якому $U_= = U_{\phi 1}$, $z = R$, а $T = \tau = \frac{L}{R}$ – електромагнітна стала часу. Це дає змогу доволі просто,

застосувавши закон Кірхгофа для кола постійного струму, створювати динамічні моделі асинхронних і синхронних двигунів.

Математична та структурна моделі асинхронного двигуна, побудовані за Г-подібною заступною схемою

Значення моменту будемо обчислювати за виразом

$$M_{ем} = \frac{P_{ем}}{\omega_0},$$

де $\omega_0 = 2\pi f_1 / p_n$ – синхронна частота поля статора для f_1 частоти струму статора;

$P_{ем} = 3i_2'^2 r_2' / s$ – електромагнітна потужність, яка передається в роторне коло двигуна;

$s = (\omega_0 - \omega)/\omega_0$ – ковзання.

$$M = \frac{3i_2'^2 r_2'}{\omega_0 s} = \frac{3i_2'^2 r_2'}{\omega_0 - \omega}. \qquad (2.49)$$

Зведене значення струму ротора i_2' визначимо з такого диференціального рівняння, складеного для кола постійного струму, еквівалентного колові змінного струму за рис. 2.28:

$$U_{\phi 1} = i_2'(z_1 + z_2') + (L_{1\sigma} + L_{2\sigma}')\frac{di_2'}{dt};$$

де

$$(L_{1\sigma} + L_{2\sigma}') = \frac{x_1 + x_2'}{\omega_1} = \frac{x_k}{\omega_1};\ \omega_1 = 2\pi f_1;\ z_2' = \sqrt{(r_2'/s)^2 + x_2^2};\ z_1 = \sqrt{r_1^2 + x_1^2}.$$

Разом з відомим рівнянням механічної частини математичну модель асинхронного двигуна складають такі рівняння:

$$\frac{di_2'}{dt} = \frac{\omega_1}{x_k}(U_{\phi 1} - i_2'(z_1 + z_2'));$$

$$M = \frac{3 \cdot i_2'^2 \cdot r_2'}{\omega_0 - \omega}; \qquad (2.50)$$

$$\frac{d\omega}{dt} = \frac{M - M_c}{J_\Sigma}.$$

Структурна модель згідно з рівняннями (2.50) зображена на рис. 2.31.

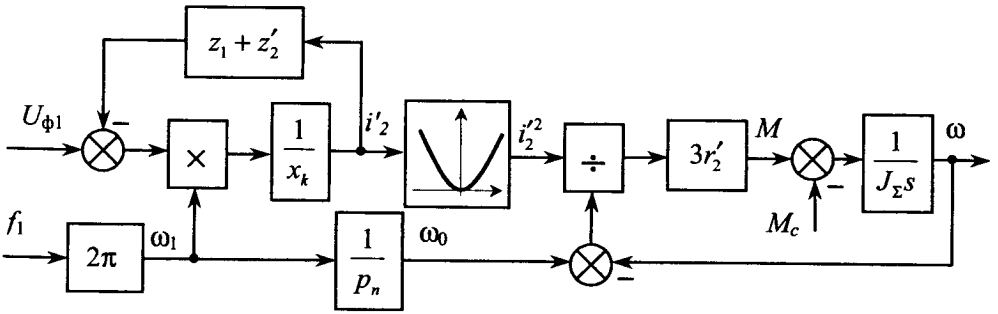


Рис. 2.31. Структурна модель асинхронного двигуна, яка побудована на основі Г-подібної заступної схеми фази двигуна

Така модель дає змогу моделювати динамічні режими, наприклад, запуск на природній характеристиці двигуна з фазним чи короткозамкненим ротором.

На рис. 2.32 показано графіки $\omega(t)$, $M(t)$ та $i_2'(t)$ для режимів запуску та зміни навантаження $0{,}1M_{\text{ном}} \rightarrow M_{\text{ном}} \rightarrow 0{,}1M_{\text{ном}}$ для моделі асинхронного двигуна, побудованої за моделлю (2.50) і параметри якого отримані в прикладі 1.

Для відтворення моделі двигуна, який працює в системі регулювання з використанням зворотного зв'язку за струмом статора, до рівнянь (2.50) треба додати такі рівняння:

$$\begin{cases} \dfrac{di_\mu}{dt} = \dfrac{\omega_1}{x_\mu}(U_{\phi 1} - i_\mu z_\mu) \ ; \\ i_1 = i_2' + i_\mu \ , \end{cases} \qquad (2.51)$$

де $z_\mu = \sqrt{r_\mu^2 + x_\mu^2}$ (див. вище).

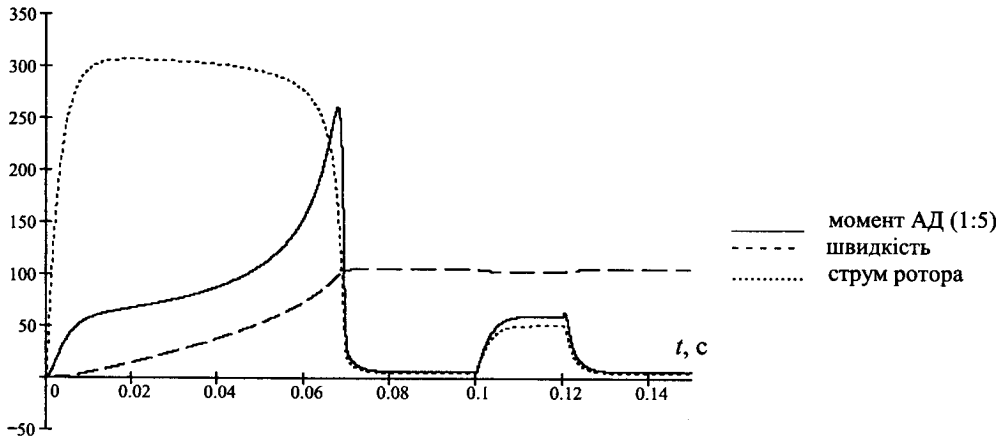


Рис. 2.32. Графіки динамічних режимів моделі АД на основі Г-подібної заступної схеми

Структурна схема математичної моделі АД, яка побудована за виразами (2.50) і (2.51), зображена на рис. 2.33.

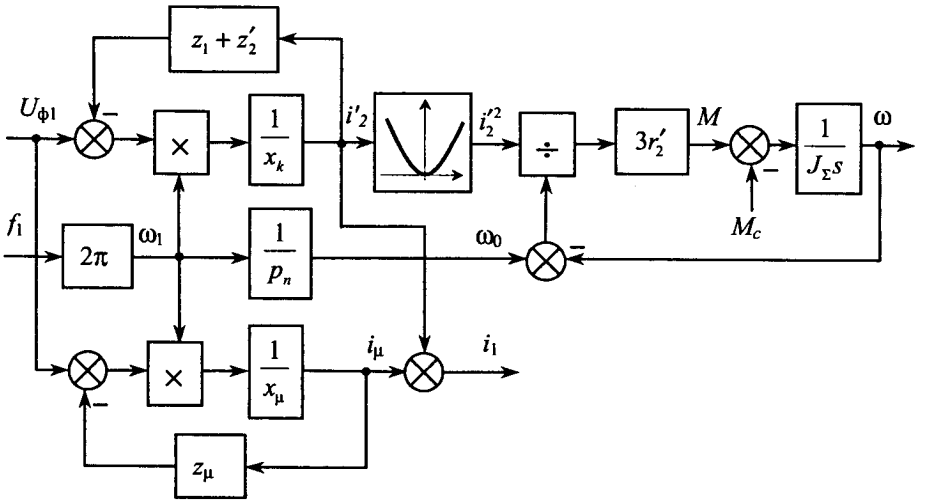


Рис. 2.33. Структурна схема математичної моделі асинхронного двигуна з виділенням струму статора, яка побудована на основі Г-подібної заступної схеми фази двигуна

Математична та структурна моделі асинхронного двигуна, побудовані за Т-подібною заступною схемою

Для створення динамічної моделі за Т-подібною заступною схемою, яка забезпечує кращу точність статичних характеристик, а тому і динамічних, зробимо деякі перетворення, а саме: замінимо паралельний контур $z_\mu - z_2'$ еквівалентним послідовним за правилами перетворень провідностей синусоїдних кіл змінного струму. Загалом комплексна провідність:

$$Y = \frac{1}{z} = \frac{1}{r + j \cdot x} = \frac{r - j \cdot x}{r^2 + x^2} = \frac{r}{z^2} - j\frac{x}{z^2} = a - j \cdot b,$$

звідки $a = \frac{r}{z^2}$; $b = \frac{x}{z^2}$, і навпаки, $r = a \cdot z^2 = \frac{a}{y^2}$; $x = b \cdot z^2 = \frac{b}{y^2}$.

Приклад 2. Розрахувати параметри заступної схеми із заміною паралельного контуру $z_\mu - z_2'$ на послідовний з $r_e - x_e$ еквівалентної Т-подібної заступної схеми для асинхронного двигуна з прикладу 1, якщо ковзання $s = 1$.

$$a = \frac{r_\mu \cdot z_2'^2 + r_2' \cdot z_\mu^2}{z_\mu^2 \cdot z_2'^2} = \frac{0.189 \cdot 0.453^2 + 0.1078 \cdot 5^2}{5^2 \cdot 0.456^2} = 0.531;$$

$$b = \frac{x_\mu \cdot z_2'^2 + x_2' \cdot z_\mu^2}{z_\mu^2 \cdot z_2'^2} = \frac{5 \cdot 0.453^2 + 0.441 \cdot 5^2}{5^2 \cdot 0.456^2} = 2.35;$$

$$y^2 = a^2 + b^2 = 0.531^2 + 2.35^2 = 5.803;$$

$$r_e = \frac{a}{y^2} = \frac{0.531}{5.803} = 0.091 \text{ Ом}; \; x_e = \frac{b}{y^2} = \frac{2.35}{5.803} = 0.404 \text{ Ом};$$

$$z_e = \sqrt{0.091^2 + 0.404^2} = 0.4135 \text{ Ом}.$$

Еквівалентна заступна схема, що отримана з рис. 2.27, зображена на рис. 2.34. З рівнянь електричної рівноваги для динамічних режимів створимо математичну модель (з урахуванням у виразах r_2'/s):

$$U_{\phi 1} = i_1(z_1 + z_e) + \frac{x_1 + x_0}{\omega_1}\frac{di_1}{dt},$$

звідки

$$\frac{di_1}{dt} = \frac{\omega_1}{x_1 + x_e}(U_{\phi 1} - i_1(z_1 + z_e)); \quad e_1 = U_{\phi 1} - i_1 \cdot z_1; \quad e_1 = i_2' \cdot z_2' + \frac{x_2'}{\omega_1}\frac{di_2'}{dt}.$$

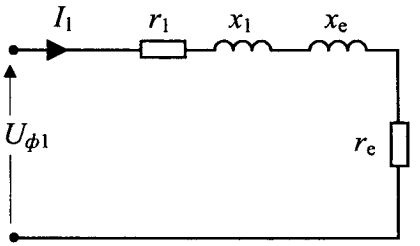


Рис. 2.34. Еквівалентна заступна схема Т-подібної заступної схеми фази асинхронного двигуна

Тоді:

$$\frac{di_1}{dt} = \frac{\omega_1}{x_1 + x_e}(U_{\phi 1} - i_1(z_1 + z_e)); \tag{2.52}$$

$$\frac{di_2'}{dt} = \frac{\omega_1}{x_2'}(e_1 - i_2' z_2') = \frac{\omega_1}{x_2'}(U_{\phi 1} - i_1 z_1 - i_2' z_2'); \tag{2.53}$$

$$M = 3i_2'^2 r_2' / (\omega - \omega_0); \tag{2.54}$$

$$\frac{d\omega}{dt} = \frac{M - M_c}{J_\Sigma}. \tag{2.55}$$

За виразами (2.52)–(2.55) математичної моделі побудована структурна схема моделі (рис. 2.35).

Така модель застосована для моделювання різноманітних статичних та динамічних режимів асинхронного двигуна, за отриманими в прикладі 1 параметрами. На рис. 2.36 зображено статичні механічні природна та реостатні характеристики (а), а також характеристики для різних фазних напруг $U_{1\phi}$ (б) та

її частоти f_1 (в), які побудовані за результатами, отриманими із залежностей (2.52)–(2.54) та рис. 2.32 для статичного режиму.

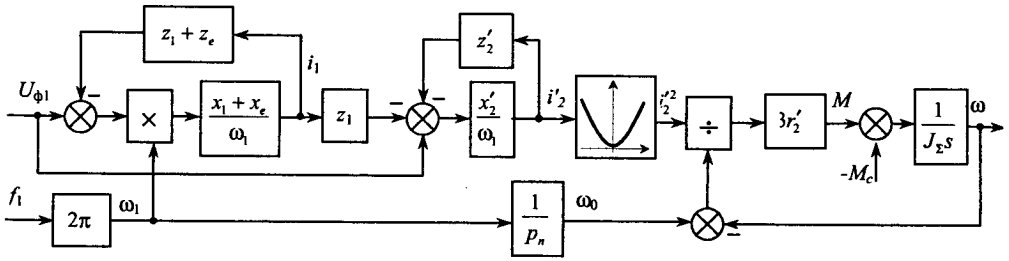
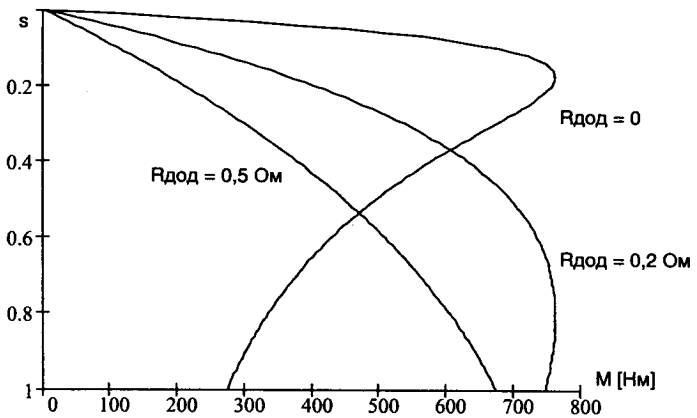


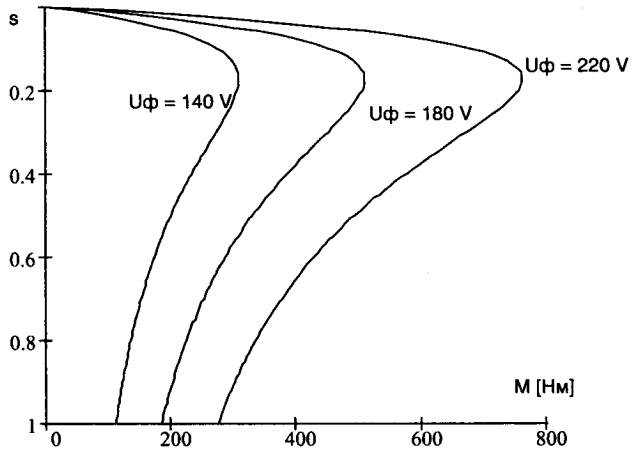
Рис. 2.35. Структурна схема моделі асинхронного двигуна , побудована на основі Т-подібної заступної схеми фази двигуна

Характеристики рис. 2.36, *в* розраховані за виразом $M = 3i'^2_2 r'_2 / \omega_0 s$, в якому $x_1 = x_{1н} \cdot f_1 / f_{1н} = x_{1н} \cdot k_f$, $x_2 = x_{2н} \cdot k_f$, $x_\mu = x_{\mu н} \cdot k_f$, де x_{1H}, x_{2H} і $x_{1\mu H}$ – значення індуктивних опорів для номінальної частоти напруги статора.

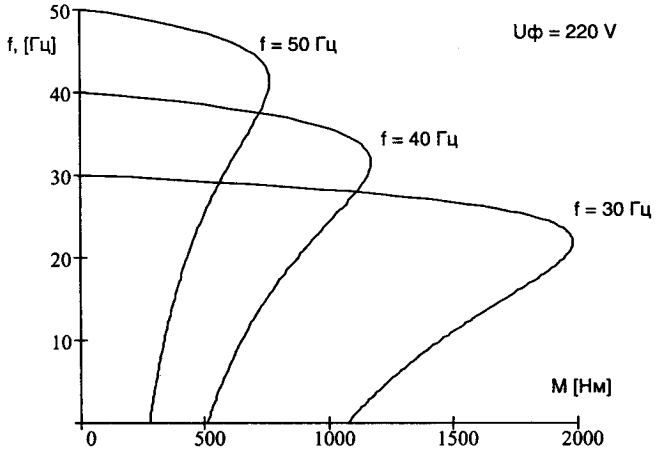


а

Рис. 2.36. Механічні характеристики моделі асинхронного двигуна, побудованої на основі Т-подібної заступної схеми



б



в

Рис. 2.36. Механічні характеристики моделі асинхронного двигуна, побудованої на основі Т-подібної заступної схеми (продовження)

На рис. 2.37 зображено графіки $\omega(t)$, $M(t)$ та $i_2'(t)$ для режимів запуску та зміни навантаження $0{,}1M_{\text{ном}} \rightarrow M_{\text{ном}} \rightarrow 0{,}1M_{\text{ном}}$ для моделі асинхронного

двигуна, побудованої за виразами (2.52)–(2.55) і дані якого використані в прикладі 1.

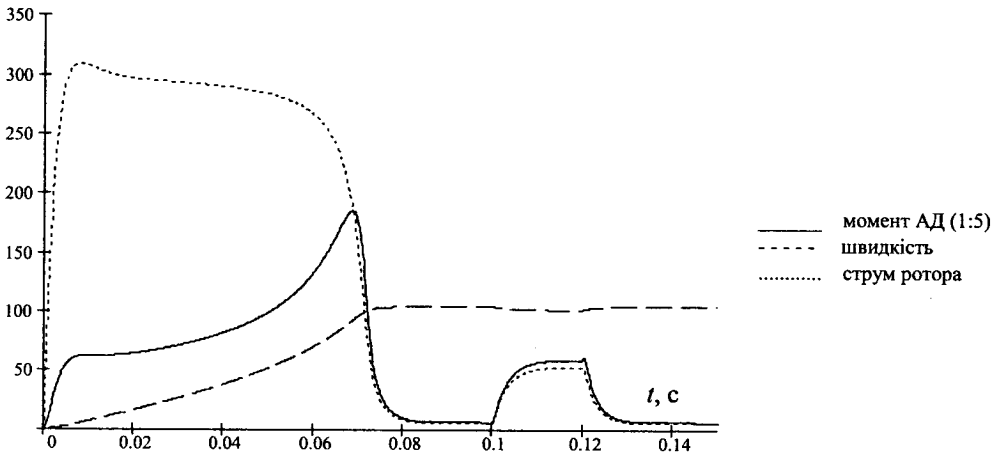


Рис. 2.37. Графіки динамічних режимів моделі АД на основі Т-подібної заступної схеми

Модель асинхронного двигуна з короткозамкненим ротором

Точніша модель асинхронного двигуна з короткозамкненим ротором описується у стаціонарній системі координат статора (α, β) системою диференціальних рівнянь, яка пов'язує складові потокозчеплення ротора (Ψ_α, Ψ_β) і струму статора (i_α, i_β) [Б.10, Д.21, Д.26]:

$$\frac{d\Psi_\alpha}{dt} = -\frac{R_2'}{L_r}\cdot\Psi_\alpha - p_n\cdot\omega\cdot\Psi_\beta + \frac{R_2'}{L_r}\cdot L_m\cdot i_\alpha;$$

$$\frac{d\Psi_\beta}{dt} = -\frac{R_2'}{L_r}\cdot\Psi_\beta + p_n\cdot\omega\cdot\Psi_\alpha + \frac{R_2'}{L_r}\cdot L_m\cdot i_\beta;$$

$$\frac{di_\alpha}{dt} = \frac{L_r}{L_s\cdot L_r - L_m^2}\left(U_\alpha - R_2\cdot i_\alpha - \frac{L_m}{L_r}\cdot\frac{d\Psi_\alpha}{dt}\right); \qquad (2.56)$$

$$\frac{di_\beta}{dt} = \frac{L_r}{L_s \cdot L_r - L_m^2}\left(U_\beta - R_1 \cdot i_\beta - \frac{L_m}{L_r} \cdot \frac{d\Psi_\beta}{dt}\right);$$

$$M = \frac{3}{2} \cdot p_n \cdot \frac{L_m}{L_r}\left(\Psi_\alpha \cdot i_\beta - \Psi_\beta \cdot i_\alpha\right);$$

$$J\frac{d\omega}{dt} = M - M_c,$$

де $L_{\sigma 1} = x_1 / 2\pi f_1$ – індуктивність розсіювання обмотки статора;

$L'_{\sigma 2} = x'_2 / 2\pi f_1$ – зведена до статора індуктивність розсіювання обмотки ротора;

L_m – взаємна індуктивність між обмотками статора і ротора; за відсутності паспортних даних її значення можна обчислити за наближеним співвідношенням: $L_m = (2.5...3.5)\frac{Z_б}{2\pi f_1}$, де $Z_б = U_{фн}/I_{фн}$ – базовий опір;

$L_s = L_{\sigma 1} + L_m$ – повна індуктивність обмотки статора;

$L_r = L'_{\sigma 2} + L_m$ – зведена до статора повна індуктивність обмотки ротора;

f_1 – частота напруги мережі живлення;

p_n – кількість пар полюсів.

Якщо вісь фази А збігається з віссю α, зв'язок між компонентами напруги статора і відомими фазними напругами U_A, U_B, U_C описується співвідношенням:

$$\begin{bmatrix} U_\alpha \\ U_\beta \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & -1/2 & -1/2 \\ 0 & \sqrt{3}/2 & -\sqrt{3}/2 \end{bmatrix} \cdot \begin{bmatrix} U_A \\ U_B \\ U_C \end{bmatrix}$$

і навпаки, для струмів статора матимемо

$$\begin{cases} i_A = i_\alpha \cos(\omega\ t) - i_\beta \sin(\omega\ t)\ ; \\ i_B = i_\alpha \cos(\omega\ t - 2\pi/3) - i_\beta \sin(\omega\ t - 2\pi/3)\ ; \\ i_C = i_\alpha \cos(\omega\ t + 2\pi/3) - i_\beta \sin(\omega\ t + 2\pi/3)\ . \end{cases}$$

Приклад використання такої моделі в різних програмних середовищах подано у розд. 4.2.

Модель асинхронного двигуна у фазних координатах

Теоретичною основою для побудови математичної моделі асинхронних двигунів є рівняння електричної та механічної рівноваг, перетворення електромагнітної енергії в електричну. Створюючи модель, приймаємо такі допущення:

1) втрати в сталі ротора і статора відсутні;
2) обмотки статора і ротора зсунуті на 120°;
3) намагнічувальні сили в повітряному проміжку синусоїдні;
4) магнітні кола ненасичені.

Візьмемо за вихідну систему рівняння миттєвих значень напруг для обмоток трьох фаз статора і ротора:

$$\begin{cases} u_{1A} = i_{1A}R_1 + \dfrac{d\Psi_{1A}}{dt}; \\ u_{1B} = i_{1B}R_1 + \dfrac{d\Psi_{1B}}{dt}; \\ u_{1C} = i_{1C}R_1 + \dfrac{d\Psi_{1C}}{dt}; \\ u'_{2A} = i'_{2A}R'_2 + \dfrac{d\Psi_{2A}}{dt}; \\ u'_{2B} = i'_{2B}R'_2 + \dfrac{d\Psi_{2B}}{dt}; \\ u'_{2C} = i'_{2C}R'_2 + \dfrac{d\Psi_{2C}}{dt}, \end{cases} \tag{2.57}$$

де $u_{1A}...u'_{2C}$ – миттєві значення фазних напруг статора і ротора, зведених до статора;

$i_{1A}...i'_{2C}$ – миттєві значення фазних струмів статора і ротора, зведених до статора;

$R_1...R'_2$ – активні опори обмоток, відповідно, статора і ротора, зведені до статора;

$\Psi_{1A}...\Psi_{2C}$ – повні потокозчеплення фазних обмоток.

У загальному випадку $\psi = i \cdot L$.

Для подальших перетворень при створенні математичної моделі, перейдемо від рівняння миттєвих значень до рівняння результівних векторів двополюсної ($p_n = 1$) трифазної машини. Перехід здійснимо так. Результівний вектор миттєвих значень, наприклад, струмів фаз i_A, i_B, i_B описується так:

$$\bar{i} = \frac{2}{3}\left(i_A e^{j0} + i_B e^{\frac{2}{3}\pi \cdot j} + i_C e^{\frac{4}{3}\pi \cdot j} \right).$$

Позначимо $e^{j0} = a^0 = 1$; $e^{\frac{2}{3}\pi \cdot j} = a^1$; $e^{\frac{4}{3}\pi \cdot j} = a^2$, тоді $\bar{i} = \frac{2}{3}\left(i_A + i_B a + i_C a^2 \right)$.

Аналогічно запишемо функції для напруг і потокозчеплень:

$$\begin{aligned} \bar{U} &= \frac{2}{3}\left(u_A + u_B a + u_C a^2 \right); \\ \bar{\Psi} &= \frac{2}{3}\left(\Psi_A + \Psi_B a + \Psi_C a^2 \right). \end{aligned} \tag{2.58}$$

Використаємо ці вирази для запису системи рівнянь (2.57), перемноживши перше рівняння на $\frac{2}{3}a^0$, друге – на $\frac{2}{3}a^1$, третє – на $\frac{2}{3}a^2$, і додамо їх.

$$\begin{aligned} &\frac{2}{3}a^0 u_{1A} = \frac{2}{3}a^0 i_{1A} R_1 + \frac{2}{3}a^0 \frac{d\psi_{1A}}{dt}; \\ +\ &\frac{2}{3}a^1 u_{1B} = \frac{2}{3}a^1 i_{1B} R_1 + \frac{2}{3}a^1 \frac{d\psi_{1B}}{dt}; \\ &\frac{2}{3}a^2 u_{1C} = \frac{2}{3}a^2 i_{1C} R_1 + \frac{2}{3}a^2 \frac{d\psi_{1C}}{dt}; \end{aligned} \tag{2.59}$$

$$\frac{2}{3}\left(u_{1A} + a u_{1B} + a^2 u_{1C} \right) = \frac{2}{3} R_1 \left(i_{1A} + a i_{1B} + a^2 i_{1C} \right) + \frac{2}{3}\frac{d\left(\psi_{1A} + a\psi_{1B} + a^2 \psi_{1C} \right)}{dt}.$$

Або у векторній формі для статора і ротора:

$$\bar{U}_1 = \bar{i}_1 R_1 + \frac{d\bar{\Psi}_1}{dt}; \tag{2.60}$$

$$\bar{U}_2' = \bar{i}_2' R_2' + \frac{d\bar{\Psi}_2}{dt}. \tag{2.61}$$

Рівняння (2.60) і (2.61) записані в різних координатах: рівняння (2.60) в координатах нерухомого статора, а рівняння (2.61) – в координатах ротора, які, як і ротор, обертаються з кутовою швидкістю ω.

Для сумісних перетворень цих рівнянь їх необхідно звести до одних координат, які б оберталися з однаковою кутовою швидкістю – довільною ω_k, нерухомих відносно статора чи ротора, який обертається. Зручно користуватись системою координат, що обертається з кутовою швидкістю ω_1 поля статора двополюсної машини, для якої $p_n = 1$, а $\omega_1 = 2\pi f_1$, де f_1 – частота напруги статора. У такому разі рівняння (2.60) доповнюється вектором ЕРС, зумовленої різницею кутових швидкостей статора і його поля. Це буде похідна від вектора $\frac{d\bar{\psi}_1}{dt}$, який обертається зі швидкістю ω_1. Нагадаємо, що похідна вектора, незмінного за модулем, який обертається з кутовою швидкістю ω, дорівнює добутку цього вектора на $j\omega$. Тому похідна від вектора $\frac{d\bar{\psi}_1}{dt}$ записується виразом $j\omega_1\bar{\psi}_1$.

Зведення рівнянь до синхронно обертових координат застосовується в дослідженні частотно керованих електроприводів, тому розглянемо такий спосіб перетворення.

Рівняння (2.61) для ротора з вектором потокозчеплення $\bar{\psi}_2$, який обертається зі швидкістю $\omega_1 - p_n\omega$, тобто різниці швидкостей поля статора двополюсної машини ω_1 і ротора реальної полюсної машини ω, зведеної до двополюсної – $p_n\omega$, доповнюється такою ЕРС – $j\bar{\psi}_2(\omega_1 - p_n\omega)$.

Тепер рівняння (2.60) і (2.61) в одних координатах мають такий вигляд:

$$\begin{cases} \bar{U}_1 = \bar{i}_1 R_1 + \dfrac{d\bar{\psi}_1}{dt} + j\bar{\psi}_1\omega_1; \\ \bar{U}_2' = \bar{i}_2' R_2' + \dfrac{d\bar{\psi}_2}{dt} + j\bar{\psi}_2(\omega_1 - p_n\omega). \end{cases} \tag{2.62}$$

У другому рівнянні зробимо незначні перетворення:

$$(\omega_1 - p_n\omega)\frac{\omega_1}{\omega_1} = \frac{\omega_1 - p_n\omega}{\omega_1}\omega_1 = \frac{\omega_1 / p_n - \omega}{\omega_1 / p_n}\omega_1 = \frac{\omega_0 - \omega}{\omega_0}\omega_1 = s\omega_1,$$

де ω_1/p_1 – синхронна швидкість (поля статора) машини з p_1 парами полюсів.

Тоді

$$\bar{U}_2' = \vec{i}_2 R_2' + \frac{d\bar{\Psi}_2}{dt} + j\bar{\Psi}_2 s\omega_1. \tag{2.63}$$

Потокозчеплення пов'язані зі струмами через індуктивності і векторна форма запису є такою:

$$\begin{cases} \bar{\Psi}_1 = \bar{i}_1 L_1 + \vec{i}_2 L_m; \\ \bar{\Psi}_2 = \vec{i}_2 L_2' + \bar{i}_1 L_m, \end{cases} \tag{2.64}$$

де L_1, L_2', L_m – індуктивності фази статора, зведеної ротора до статора і взаємна індуктивність фаз статора і ротора.

Електромагнітний момент – це векторний добуток

$$\bar{M} = \frac{3}{2} p_n \left(\bar{\Psi}_1 \times \bar{i}_1\right) \text{ або } M = -\frac{3}{2} p_n \left(\bar{\Psi}_2 \times \vec{i}_2\right). \tag{2.65}$$

З рівнянь (2.62) та (2.63) виділимо похідні і разом з (2.64) та (2.65) запишемо систему рівнянь у векторній формі, що утворюють математичну модель асинхронного двигуна:

$$\left.\begin{aligned} &\frac{d\bar{\Psi}_1}{dt} = \bar{U}_1 - \bar{i}_1 R_1 - j\omega_1 \bar{\Psi}_1, \\ &\frac{d\bar{\Psi}_2}{dt} = \bar{U}_2' - \vec{i}_2 R_2' - js\omega_1 \bar{\Psi}_2, \\ &\bar{\Psi}_1 = \bar{i}_1 L_1 + \vec{i}_2 L_m, \\ &\bar{\Psi}_2 = \vec{i}_2 L_2' + \bar{i}_1 L_m, \\ &\bar{M} = \frac{3}{2} p_n \left(\bar{\Psi}_1 \times \bar{i}_1\right), \\ &\bar{M} = -\frac{3}{2} p_n \left(\bar{\Psi}_2 \times \vec{i}_2\right). \end{aligned}\right\} \tag{2.66}$$

Перехід до векторної форми запису дає змогу аналізувати сумісну дію обмоток статора і ротора у вигляді еквівалентних векторів.

Система рівнянь (2.66) є нелінійною і розв'язати її можна, застосувавши відомі пакети програм, наприклад, MathCAD. Таке описання динаміки процесів у асинхронних двигунах відображає реальні процеси, які проходять в них. Але

для синтезу систем керування асинхронними приводами бажано мати простіші та наочніші динамічні моделі асинхронних двигунів у вигляді передатних функцій або структурних схем.

Перед тим, як перейти до перетворення рівнянь системи (2.66) векторної форми до скалярної, варто пригадати поняття узагальненої двофазної електричної машини.

Математичне описання багатофазних двигунів складається з $n+m$ рівнянь, які описують процеси в n обмотках статора та m обмотках ротора. За відомих допущень [Д.21, Д.27] реальну багатофазну машину подають еквівалентною двофазною, в якої на статорі і роторі розміщені по дві взаємно перпендикулярні обмотки – α і β на статорі, d і q на роторі. У такій спрощеній моделі реальної машини магніторушійна сила (МРС) будь-якої багатофазної машини створюється двома еквівалентними взаємно перпендикулярними обмотками.

Узагальнена машина вважається двополюсною, тобто $p_n = 1$. Багатополюсні машини зводять до двополюсної перерахунком синхронної швидкості поля і моменту двигуна. Параметри обмоток ротора зводяться до статора.

У машинах змінного струму статорними обмотками створюється обертове магнітне поле системою трифазного струму, що підводиться до них. Таке поле зображають вектором постійного модуля, який обертається в нерухомих відносно статора ортогональних осях з кутовою швидкістю поля $\omega_0 = 2\pi f_1 / p_n$. Стан вектора в будь-який момент часу визначається його проекціями на ці осі, наприклад, для струму

$$\bar{i}_1 = \bar{i}_{1\alpha} + j\bar{i}_{1\beta},$$

якщо вважати вісь α (абсцису) дійсною, а вісь β (ординату) – уявною.

Такого самого ефекту можна досягти, якщо до двох обмоток, розміщених на нерухомих відносно статора (жорстко з'єднаних з ним) осях, підвести симетричну двофазну систему напруг

$$U_{1\alpha} = U_m \cos\omega_0 t \;; \qquad U_{1\beta} = U_m \sin\omega_0 t \,,$$

або до однієї обмотки, яка розміщена на дійсній осі і яка обертається зі швидкістю поля, підвести постійну напругу, яка дорівнює амплітудному значенню U_{m1}. Отже, в синхронно обертових координатах u і v реальні змінні напруги, які підведені до обмоток статора, перетворюються в постійну напругу значення U_{m1}, яка підводиться до обмотки, що розміщена на дійсній осі u, і це має реальний фізичний

зміст. Така заміна синусоїдних змінних постійними з перетворенням координат істотно спрощує моделювання, чим ми скористаємось.

Запишемо систему рівнянь – проекцій на ортогональні синхронно обертові осі u і v векторів рівнянь (2.66) за умови, що $\bar{U}_2' = 0$, а також $U_{1v} = 0$. Керування здійснюється поданням на статор постійної напруги $U_{1m} = U_{1\phi}$. Диференціальні рівняння запишемо формою потокозчеплень. Виразимо струми через потокозчеплення з виразів (2.64):

$$\begin{cases} \bar{i}_1 = \left(L_2'\bar{\Psi}_1 - L_m\bar{\Psi}_2\right)/\left(L_1L_2' - L_m^2\right), \\ \bar{i}_2 = \left(L_1\bar{\Psi}_2 - L_m\bar{\Psi}_1\right)/\left(L_1L_2' - L_m^2\right). \end{cases} \tag{2.67}$$

Як зазначено вище, для сумісного перетворення змінних статора і ротора здійснено перехід до системи нових координат з осями u і v, які обертаються синхронно зі швидкістю поля. Це означає, що в такій узагальненій машині обмотки статора і ротора розміщені на цих осях і взаємно нерухомі. Перехід до моделі із взаємно нерухомими обмотками істотно спрощує математичне описання динамічних процесів електромеханічного перетворення енергії. Коефіцієнти взаємоіндукції і потокозчеплення взаємно нерухомих обмоток стають незалежними від механічної координати, а рух реальних обмоток і обертання координатних осей враховується введенням у рівняння електричної рівноваги додаткових ЕРС.

Тоді матимемо остаточну модель асинхронного двигуна:

$$\left.\begin{aligned} \frac{d\Psi_{1u}}{dt} &= U_{1m} - \frac{R_1}{L_1\sigma}\Psi_{1u} + \frac{R_1L_m}{L_1L_2'\sigma}\Psi_{2u} + \omega_1\Psi_{1v}; \\ \frac{d\Psi_{1v}}{dt} &= \frac{R_1L_m}{L_1L_2'\sigma}\Psi_{2v} - \frac{R_1}{L_1\sigma}\Psi_{1v} - \omega_1\Psi_{1u}; \\ \frac{d\Psi_{2u}}{dt} &= \frac{R_2'L_m}{L_1L_2'\sigma}\Psi_{1u} - \frac{R_2'}{L_2'\sigma}\Psi_{2v} + \omega_1\Psi_{2v}; \\ \frac{d\Psi_{2v}}{dt} &= \frac{R_2'L_m}{L_1L_2'\sigma}\Psi_{1v} - \frac{R_2'}{L_2'\sigma}\Psi_{2u} + \omega_1\Psi_{2u}; \\ M &= \frac{3}{2}p_n\frac{L_m}{L_1L_2'\sigma}\left(\Psi_{1v}\Psi_{2u} - \Psi_{1u}\Psi_{2v}\right). \end{aligned}\right\} \tag{2.68}$$

2.4. Математичні моделі тиристорних перетворювачів

Тиристори та симістори – це напівпровідникові силові керовані вентилі, увімкнення яких реалізується через подання сигналу на керуючий електрод при додатній напрузі на аноді, а вимкнення (закривання) відбувається за умови зменшення анодного струму до певного мінімального значення – струму утримання. Після увімкнення сигнал керування може бути знятий. На відміну від тиристора стан транзистора у ключовому режимі визначається лише наявністю імпульсу керування в базовому колі.

Тиристорні перетворювачі складаються з двох основних складових частин: системи імпульсно-фазового керування (СІФК) та силового кола (СК) тиристорного перетворювача (ТП). Загалом тиристорний перетворювач як елемент електромеханічної системи залежно від характеру та особливостей розв'язуваної задачі може моделюватися у миттєвих чи усереднених координатах. Під час моделювання тиристорних перетворювачів найчастіше складають дві математичні моделі: для СІФК та силового кола ТП, які математично пов'язуються в загальній моделі досліджуваної системи, а також моделі навантаження.

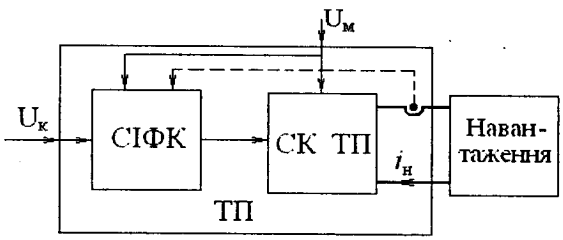


Рис. 2.38. Функціональна блок-схема тиристорного перетворювача

Отже, тиристорні перетворювачі можна моделювати на рівні миттєвих або середніх значень ЕРС. Це зумовлено характером і метою досліджень. Критерій вибору типу моделі визначається як

$$K = \frac{t_{пп}}{t_{прт}},$$

де $t_{пп}$ – тривалість перехідного процесу;

$t_{прт} = 1/mf_1$ – тривалість періоду провідності тиристора;

m – фазність перетворювача;
f_1 – частота мережі живлення.

Тоді значення K визначають такі типи моделей:

$K < 5$ – імітаційні моделі;
$5 < K < 10$ – імпульсні моделі;
$10 < K < 30$ – нелінійні неперервні моделі;
$K > 30$ – спрощені неперервні моделі;

Тривалість перехідних процесів визначається інерційністю СК ТП і кола навантаження. Чим більше значення K, тим "грубша" модель.

Математичне моделювання ТП в усереднених координатах

Під час моделювання режимів тиристорних перетворювачів в усереднених координатах необхідно враховувати низку чинників, що пов'язані з проходженням процесів на миттєвому рівні, які ускладнюють створення такої моделі:

- дискретний характер роботи ТП, який проявляється в тому, що вхідний сигнал керування визначає моменти відкривання вентилів;
- неповна керованість, що проявляється під час наростання чи спадання вхідного сигналу – вихідна напруга не може змінюватися швидше ніж напруга фази мережі;
- наявність пульсацій вихідної напруги.

Під час досліджень систем електроприводів постійного струму ТП часто моделюється як аперіодична ланка з передавальною функцією

$$W_{mn}(s) = \frac{E_{mn}(s)}{U_к(s)} = \frac{K_{mn}}{T_{mn}s + 1}, \tag{2.69}$$

якій відповідає таке диференціальне рівняння:

$$\frac{de_{mn}}{dt} = \frac{U_к K_{mn} - e_{mn}}{T_{mn}}, \tag{2.70}$$

де T_{mn} – еквівалентна стала часу моделі тиристорного перетворювача, що враховує чисте дискретне запізнення τ керування тиристорами (визначається фазністю m силової схеми ТП чи кількістю імпульсів, що формуються на періоді

напруги) $\tau = T/m$, де T – період напруги мережі, а також стала часу самої СІФК ТП $T_{СІФК}$ та фільтра нижніх частот $T_{фнч}$, що часто вмикається на вході СІФК $T_{тп} = \tau + T_{фнч} + T_{СІФК}$. Найчастіше сталу часу моделі ТП приймають такою T_{mn} = = (0.01–0.015) с. За такого підходу імпульсний характер роботи ТП імітується моделлю аперіодичної ланки з еквівалентною сталою часу T_{mn}.

Можливий інший підхід – стале дискретне запізнення вилучають з еквівалентної сталої часу T_{mn} і використовують варіанти моделей ТП, яким відповідають такі передавальні функції:

$$W_{mn}(s) = \frac{E_{mn}(s)}{U_к(s)} = \frac{K_{mn} \cdot e^{-\tau \cdot s}}{T_{mn}s + 1}; \tag{2.71}$$

$$W_{mn}(s) = \frac{E_{mn}(s)}{U_к(s)} = K_{mn} \cdot e^{-\tau \cdot s} \tag{2.72}$$

для доволі малого значення $T_{СІФК}$ та відсутності вхідного ФНЧ

$$W_{mn}(s) = \frac{E_{mn}(s)}{U_к(s)} = \frac{K(U_к) \cdot e^{-\tau \cdot s}}{T_{mn}s + 1} \tag{2.73}$$

з урахуванням нелінійності регулювальної характеристики тиристорного перетворювача.

У разі навантаження тиристорного перетворювача на велику індуктивність (обмотку збудження) інерційністю ТП нехтують і тоді

$$W_{mn}(s) = \frac{E_{mn}(s)}{U_к(s)} = K_{mn}(U_к). \tag{2.74}$$

Врахувати імпульсний характер поведінки тиристорного перетворювача також можна, використовуючи фіксований крок розв'язання замість аперіодичної ланки. У такому разі крок розв'язування вибирають залежно від пульсності m ТП. Наприклад, для однофазної місткової схеми крок становитиме 10 мс. Такий підхід зручний для моделювання процесів з використанням Z-перетворення.

Часто з достатньою для інженерних розрахунків точністю можна вважати, що перетворювач працює на лінійній (початковій) ділянці характеристики керування, у такому разі коефіцієнт підсилення K_{mn} задають сталим. Якщо ж необхідним є врахування нелінійності характеристики керування, то з певним наближенням (без врахування обмеження кута регулювання реального ТП,

комутаційних спадів напруги тощо), задати її можна функційними залежностями. Нелінійність характеристики керування ТП і його статичний коефіцієнт передачі за лінійного сигналу розгортки СІФК в моделі подають відповідно такими залежностями:

$$E_d(U_к) = E_{d0} \sin\left(\frac{\pi}{2} \cdot \frac{U_к}{|U_{\max}|}\right), \tag{2.75}$$

$$K_{mn}(U_к) = \frac{E_d(U_к)}{U_к}, \tag{2.76}$$

де E_d – усереднене на періоді (діюче чи середньовипрямлене) значення вихідної ЕРС тиристорного перетворювача;

E_{d0} – напруга неробочого ходу ТП;

$U_к$ – вхідний сигнал керування ТП ($|U_к| \le |U_{\max}|$);

$U_{\max}$ – найбільша вхідна напруга ТП (звичайно становить ±10 В для більшості СІФК).

На рис. 2.39 показано приклад характеристики керування та залежність статичного коефіцієнта тиристорного перетворювача, що розраховані за наведеними виразами (2.75) та (2.76) відповідно. Як видно з цих залежностей, ТП є нелінійною ланкою.

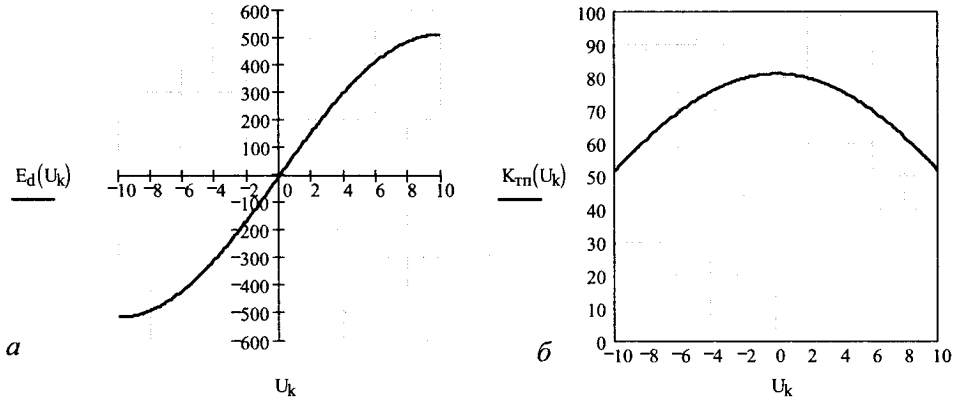


Рис. 2.39. Регулювальна характеристика (а) та залежність статичного коефіцієнта передачі від вхідного сигналу керування (б) реверсивного тиристорного перетворювача

Нижче пропонуються варіанти реалізації моделі тиристорного перетворювача як аперіодичної ланки з урахуванням нелінійності регулювальної характеристики керування і записом у нормальній формі Коші.

Quick Basic

```
' ----------------------------------------
'    Модель тиристорного перетворювача  ТП
' ----------------------------------------
'     Позначення змінних:
'     y(1) - напруга ТП
'     Ubx  - вхідна напруга системи імпульсно-фазного керування
' ----------------------------------------
CONST Pi = 3.141593      ' Число Пі
CONST Ubxmax = 10!       ' Максимальна вхідна напруга ТП
CONST Ed0 = 514.8        ' Напруга неробочого ходу ТП
CONST Ttp = .01          ' Стала часу ТП

' . . . Обчислення вхідної напруги СІФК  Ubx  . . .

IF Ubx > Ubxmax THEN Ubx = Ubxmax    ' Обмеження вхідної напруги
IF Ubx < -Ubxmax THEN Ubx = -Ubxmax  ' СІФК
Ed = Ed0*SIN(Pi*Ubx/(2*Ubxmax))      ' Розрахунок ЕРС ТП
f(1) = (Ed - y(1)) / Ttp             ' Врахування інерційності ТП
```

Turbo Pascal

```
{ ---------------------------------------- }
{    Модель тиристорного перетворювача  ТП  }
{ ---------------------------------------- }
{     Позначення змінних:                   }
{     y[1] - напруга ТП                     }
{     Ubx  - вхідна напруга СІФК            }
{ ---------------------------------------- }
const
   Ubxmax = 10.0 ;  { Максимальна вхідна напруга ТП }
```

```
   Ed0 = 514.8 ;      { Напруга неробочого ходу ТП     }
   Ttp = 0.01 ;       { Стала часу ТП                  }
var
   Ubx, Ed   : double ;

. . .  { Обчислення вхідної напруги СІФК  Ubx }  . . .

if Ubx > Ubxmax then Ubx := Ubxmax ; { Обмеження вхідної }
if Ubx < -Ubxmax then Ubx := -Ubxmax ;   { напруги СІФК     }

Ed := Ed0*sin(Pi*Ubx/(2*Ubxmax)) ;     { Розрахунок ЕРС ТП   }
f[1] := (Ed - y[1]) / Ttp ;      { Врахування інерційності ТП}
. . .
```

MathCAD

$U_{bxmax} := 10$ *Максимальна вхідна напруга ТП*

$E_{d0} := 514.8$ *Напруга неробочого ходу ТП*

$T_{\mu} := 0.01$ *Стала часу ТП*

. . . Обчислення вхідної напруги СІФК U_{bx} *. . .*

Розрахунок ЕРС ТП з обмеженням вихідної напруги

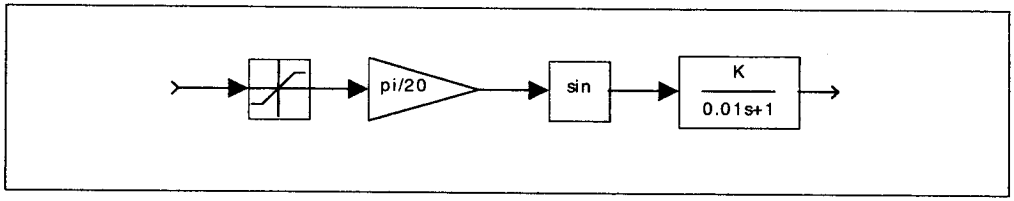
$$E_d(U_{вх}) := \begin{cases} E_{d0} \cdot \sin\left(\dfrac{\pi \cdot U_{вх}}{2 \cdot U_{вх\max}}\right) & \text{if} \quad |U_{вх}| < U_{вх\max} \\ E_{d0} \cdot \dfrac{U_{вх}}{|U_{вх}|} & \text{otherwise} \end{cases}$$

MATLAB

```
% ----------------------------------------
%    Модель тиристорного перетворювача  ТП
% ----------------------------------------
```

```
%      Позначення змінних:
%      y(1) - напруга ТП
%      Ubx  - вхідна напруга системи імпульсно-фазного керування
% ----------------------------------------
Ubxmax = 10.0 ;       % Максимальна вхідна напруга ТП
Ed0 = 514.8 ;         % Напруга неробочого ходу ТП
Ttp = 0.01  ;         % Стала часу ТП

% . . . Обчислення вхідної напруги СІФК  Ubx  . . .

% Обмеження вхідної напруги СІФК
if abs(Ubx) > Ubxmax then
   Ubx = Ubxmax * sign(Ubx) ;
end

Ed = Ed0*sin(Pi*Ubx/(2*Ubxmax)) ;    % Розрахунок ЕРС ТП
f(1) = (Ed - y(1)) / Ttp ;           % Врахування інерційності ТП
```

Simulink



Приклад використання такої спрощеної моделі ТП показано далі в розд. 4 у розрахунках динаміки систем автоматизованих електроприводів постійного струму. Підвищити точність моделей тиристорного перетворювача можна застосуванням імітаційних моделей [Д.10, Д.23].

Нескладний варіант імітаційної моделі тиристорного перетворювача з однофазною двопівперіодною силовою схемою випростування з нульовим діодом можна реалізувати за таким алгоритмом:

"*Усунення періодичності синусоїди напруги живлення*"

$$\varphi := \omega \cdot t - \pi \cdot \mathrm{trunc}\left(\frac{\omega \cdot t}{\pi}\right)$$

"*Обмеження вхідної керуючої напруги*"

якщо $|U_{вх}| \geq U_{max}$ **тоді** $U_{вх} := U_{max} \cdot \mathrm{sign}(U_{вх})$

"*Знаходимо кут керування:*" $\alpha := \pi \cdot \left(1 - \frac{|U_{вх}|}{U_{max}}\right)$

"*Знаходимо вихідну напругу перетворювача*"

якщо $\alpha > \varphi$ **тоді** $E_d := 0$ **інакше** $E_d := E_{d0} \cdot \sin(\varphi) \cdot \mathrm{sign}(U_{вх})$

де ω – кутова частота напруги мережі живлення;

trunc – операція виділення цілої частини числа (операція "усічення" числа);

$U_{вх}$ – вхідна (керуюча) напруга ТП ($|U_{вх}| \leq |U_{\max}|$);

U_{max} – найбільша вхідна напруга ТП (звичайно становить ±10 В для більшості сучасних СІФК);

E_d – вихідна ЕРС тиристорного перетворювача;

E_{d0} – напруга неробочого ходу ТП.

Нижче наведено варіанти реалізації такої імітаційної моделі тиристорного перетворювача.

Quick Basic

```
' ------------------------------------------
' Імітаційна модель тиристорного перетворювача
' ------------------------------------------
'     Позначення змінних:
'  wt   - зведене до 0...Pi значення кута напруги живлення
'  Ubx  - вхідна напруга СІФК
' ------------------------------------------
```

```
CONST Pi = 3.141593      ' Число Пі
CONST Ubxmax = 10!       ' Максимальна вхідна напруга ТП
CONST Ed0 = 514.8        ' Напруга неробочого ходу ТП
CONST omega = 100 * Pi   ' Кутова частота напруги живлення

' . . . Обчислення вхідної напруги СІФК  Ubx  . . .

IF Ubx > Ubxmax THEN Ubx = Ubxmax     ' Обмеження вхідної напруги
IF Ubx < -Ubxmax THEN Ubx = -Ubxmax   ' СІФК

' Усунення періодичності напруги живлення
wt = omega * t - Pi * FIX(omega * t / Pi)

alpha = Pi * (1 - ABS(Ubx) / Ubxmax)   ' Розрахунок кута керування

' Розрахунок вихідної напруги
IF alpha > wt THEN Ed = 0 ELSE Ed = Ed0 * SGN(Ubx) * SIN(wt)
```

Turbo Pascal

```
{ ------------------------------------------ }
{ Імітаційна модель тиристорного перетворювача }
{ ------------------------------------------ }
{     Позначення змінних:                    }
{ wt   - зведене до 0...Pi значення кута напруги живлення }
{ Ubx  - вхідна напруга СІФК                 }
{ ------------------------------------------ }
const
   Ubxmax = 10.0;  { Максимальна вхідна напруга ТП   }
   Ed0 = 514.8;    { Напруга неробочого ходу ТП      }
   w = 100*Pi;     { Кутова частота напруги живлення }
var
   Ubx, wt, alpha, Ed  : double;

. . .  { Обчислення вхідної напруги СІФК  Ubx }  . . .

if Ubx > Ubxmax then Ubx := Ubxmax;  { Обмеження вхідної напруги }
if Ubx < -Ubxmax then Ubx := -Ubxmax;      { СІФК                 }
```

```
{ Усунення періодичності напруги живлення }
wt := w*t - Pi*trunc(w*t/Pi);

alpha := Pi*(1.0 - abs(Ubx)/Ubxmax); { Розрахунок кута керування }

{ Розрахунок вихідної напруги }
if alpha > wt then Ed := 0.0 else Ed := Ed0*Ubx/abs(Ubx)*sin(wt);
```

MathCAD

$U_{bxmax} := 10$ *Максимальна вхідна напруга ТП*

$E_{d0} := 514.8$ *Напруга неробочого ходу ТП*

$\omega := 100 \cdot \pi$ *Кутова частота напруги живлення*

Опис функції обчислення ЕРС ТП:

$$Em(\omega t, U_{in}) := \left| \begin{array}{l} \text{"Усуваємо періодичність синусоїди"} \\ \omega t \leftarrow \omega t - \text{trunc}\left(\frac{\omega t}{\pi}\right) \cdot \pi \\ \text{"Обмеження напруги керування"} \\ U_{in} \leftarrow 10 \cdot \text{sign}(U_{in}) \quad \text{if} \quad |U_{in}| > 10 \\ \text{"Знаходимо кут керування"} \\ \alpha \leftarrow \pi \cdot \left(1 - \frac{|U_{in}|}{10}\right) \\ \text{"Знаходимо вихідну напругу перетворювача"} \\ 0 \quad \text{if} \quad \alpha > \omega t \\ E_{d0} \cdot \text{sign}(U_{in}) \cdot \sin(\omega t) \quad \text{otherwise} \end{array} \right.$$

MATLAB

```
% ------------------------------------------
% Імітаційна модель тиристорного перетворювача
% ------------------------------------------
%     Позначення змінних:
%  wt   - зведене до 0...Pi значення кута напруги живлення
%  Ubx  - вхідна напруга СІФК
% ------------------------------------------
Ubxmax = 10.0;      % Максимальна вхідна напруга ТП
Ed0 = 514.8;        % Напруга неробочого ходу ТП
w = 100*pi;         % Кутова частота напруги живлення

% . . . Обчислення вхідної напруги СІФК  Ubx  . . .

% Обмеження вхідної напруги СІФК
if abs(Ubx) > Ubxmax then
   Ubx = Ubxmax * sign(Ubx) ;
end

% Усунення періодичності напруги живлення
wt = w*t - pi*fix(w*t/pi);

alpha = pi*(1 - abs(Ubx)/Ubxmax);  % Обчислення кута керування

% Розрахунок вихідної напруги
if alpha > wt
   Ed = 0;
else
   Ed = Ed0*sign(Ubx)*sin(wt);
End
```

На рис. 2.40 показано графік вихідної напруги пропонованої імітаційної моделі однофазного двопівперіодного тиристорного перетворювача з нульовим діодом у силовій схемі для вхідної керуючої напруги, що лінійно наростає.

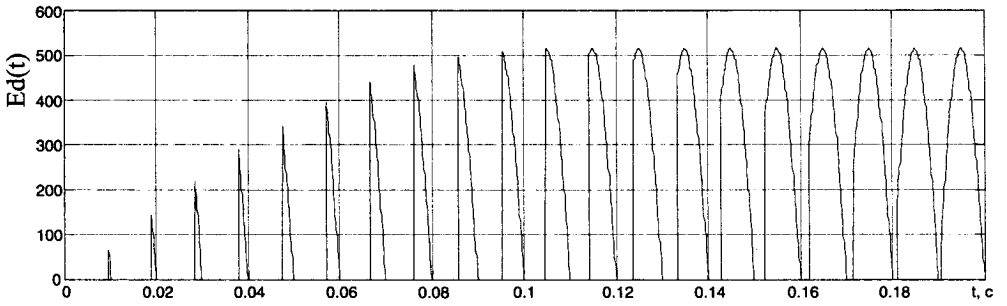
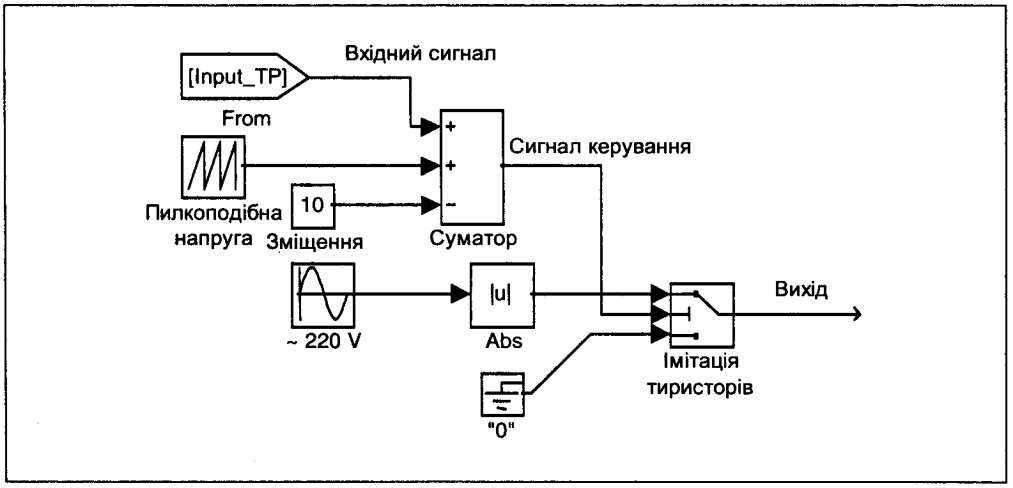


Рис. 2.40. Вихідна напруга імітаційної моделі однофазного ТП

Нижче наведено ще один варіант побудови імітаційної моделі однофазного нереверсивного тиристорного перетворювача з нульовим діодом у силовій схемі в середовищі пакета MATLAB+Simulink.

Simulink

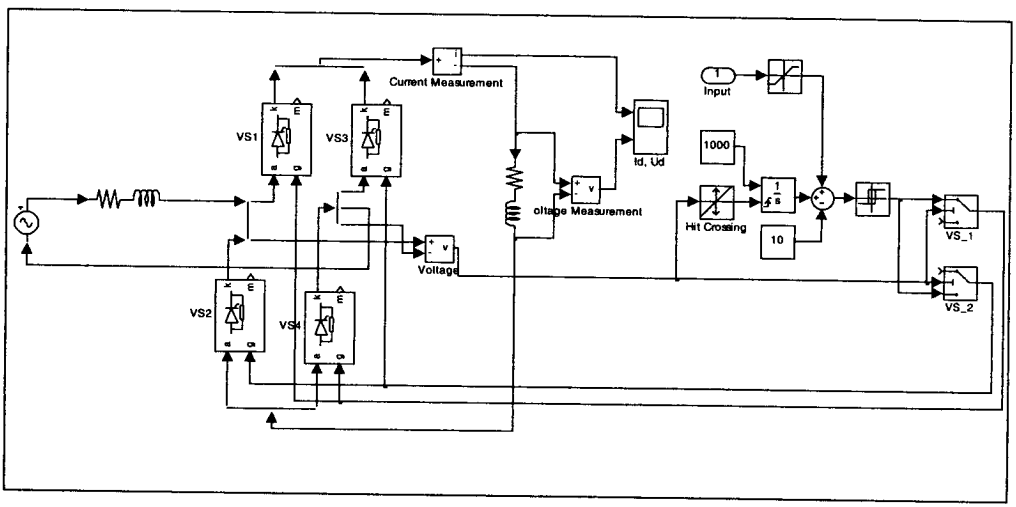


Підвищення рівня деталізації перехідних процесів, що відбуваються у напівпровідникових перетворювачах, є доволі складною задачею, оскільки

вимагає урахування структури силової схеми, поведінки силових вентилів та системи керування вентилями. Дуже часто для інженерного моделювання автоматизованих електроприводів не потрібні такі точні моделі, для яких час розрахунку різко зростає, – у кожному випадку рівень деталізації моделі вибирає дослідник залежно від поставлених задач.

Точніші моделі тиристорних перетворювачів, які враховують комутаційні процеси у силових вентилях, виходять за межі пропонованої книги, оскільки вони є значно складнішими і вимагають інших підходів [Б.5, Д.1, Д.31]. Для спрощення дослідження таких моделей можна рекомендувати пакет MATLAB+Simulink з використанням Power System Blockset. Як приклад його застосування нижче наведена проста модель керованого однофазного двопівперіодного випрямувача з активно-індуктивним навантаженням та імітаційною моделлю системи імпульсно-фазного керування. Докладніше про моделювання систем такого рівня деталізації можна дізнатися з демонстраційних прикладів Power System Blockset чи відповідної літератури [Б.5, Б.8, Б.10].

Power System Blockset



2.5. Математичні моделі елементів систем керування

Математична модель ПІ-регулятора

Принципова схема ПІ-регулятора без обмеження вихідного сигналу і реалізація його моделі за допомогою паралельно з'єднаних ланок показані на рис. 2.41. Описується вона такою передатною функцією:

$$W_p(p) = \frac{T_i s + 1}{T_i s} = K_p + \frac{1}{T_i s}.$$

Фрагмент можливого варіанта реалізації моделі ПІ-регулятора струму подано нижче.

Quick Basic

```
'     Позначення змінних:
'     y(1)    - інтегральна складова ПІ-регулятора струму
'     y(3)    - струм якоря
'     UPW     - вихідна напруга регулятора швидкості

CONST KPC = .34      ' Коефіцієнт підсилення регулятора струму
CONST Ti = .147      ' Стала часу регулятора струму
CONST KDC = .045     ' Коефіцієнт зворотного зв'язку за струмом

UPCbx = (UPW - KDC * y(3))   ' Вхідна напруга регулятора
UPCp = UPCbx * KPC           ' Пропорційна складова регулятора
f(1) = UPCbx / Ti            ' Інтегральна складова регулятора
Upc = UPCp + y(1)            ' Вихід ПІ-регулятора струму
```

Turbo Pascal

```
{     Позначення змінних:                                    }
{     y[1]    - інтегральна складова ПІ-регулятора струму    }
{     y[3]    - струм якоря                                  }
```

```
{      UPW      - вихідна напруга регулятора швидкості      }

const
   KPC = 0.34;     { Коефіцієнт підсилення регулятора струму }
   Ti = 0.147;     { Стала часу регулятора струму            }
   KDC = 0.045;    { Коефіцієнт зворотного зв'язку за струмом }

   . . .

UPCbx := (UPW - KDC * y[3]); { Вхідна напруга регулятора    }
UPCp := UPCbx * KPC;         { Пропорційна складова регулятора }
f[1] := UPCbx / Ti;          { Інтегральна складова регулятора }
Upc := UPCp + y[1];          { Вихід ПІ-регулятора струму    }
```

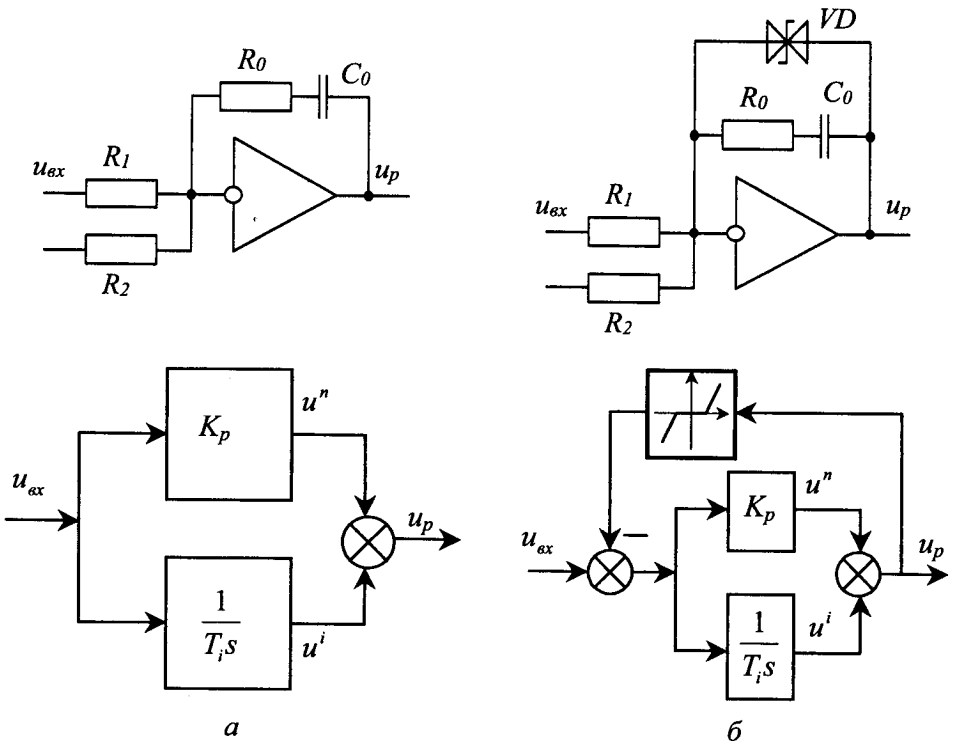


Рис. 2.41. Електрична схема і структурна модель ПІ-регулятора: а – без обмеження вихідної напруги; б – з обмеженням вихідної напруги

MathCAD

$$f_1 \leftarrow \frac{U_{bx}}{T_P} \quad \text{"інтегральна складова регулятора"}$$

$$U_P \leftarrow y_1 + K_P \cdot U_{bx}$$

MATLAB

```
%     Позначення змінних:
%     y(1)     - інтегральна складова ПІ-регулятора струму
%     y(3)     - струм якоря
%     UPW      - вихідна напруга регулятора швидкості

KPC = 0.34;       % Коефіцієнт підсилення регулятора струму
Ti = 0.147;       % Стала часу регулятора струму
KDC = 0.045;      % Коефіцієнт зворотного зв'язку за струмом

   . . .

UPCbx = (UPW - KDC * y(3)); % Вхідна напруга регулятора
UPCp = UPCbx * KPC;         % Пропорційна складова регулятора
f(1) = UPCbx / Ti;          % Інтегральна складова регулятора
Upc = UPCp + y(1);          % Вихід ПІ-регулятора струму
```

Важливим моментом під час моделювання ПІ-регулятора з обмеженням вихідної координати (наприклад, регулятора швидкості) є правильна реалізація в моделі власне обмеження: після досягнення максимального значення вихідної напруги регулятора треба не просто обмежити умовним оператором рівень виходу регулятора, а й зупинити інтегрування. Цей процес відображено у пропонованому алгоритмі.

"$U_{вх}$ – *вхідна напруга регулятора*; $U_{\max}$ – *максимальна вихідна напруга регулятора*"

$$\frac{dU_P^i}{dt} := \frac{U_{вх}}{T_i};$$ "*розрахунок похідної напруги інтегратора*"

$$U_P := K_P \cdot U_{вх} + U_P^i;$$ "*розрахунок напруги регулятора*"

якщо $|U_P| \geq U_{\max}$ **тоді** $\begin{pmatrix} U_P := U_{\max} \operatorname{sign}(U_P); \\ \frac{dU_P^i}{dt} := 0 \end{pmatrix}$.

З принципової схеми регулятора з обмеженням (рис. 2.41, *б*) зрозуміло, що такий режим реалізується стабілітроном у колі зворотного зв'язку. Часто кращі результати дає імітація такого процесу різними способами. Деякі приклади реалізації такого регулятора пропонуються нижче.

Quick Basic

```
' ------------------------------------------------------------
'     Позначення змінних:
'     y(1) - вихідна напруга задавача інтенсивності
'     y(2) - інтегральна складова ПІ-регулятора швидкості
'     y(6) - швидкість двигуна
' ------------------------------------------------------------
CONST Kpw = 23.5    ' Коефіцієнт підсилення РШ
CONST Tpw = .0034   ' Стала часу інтегратора РШ
CONST KDW = .045    ' Коефіцієнт зворотного зв'язку за швидкістю
CONST UPWmax = 10!  ' Напруга обмеження РШ

  . . .

' ============================================================
'      Регулятор швидкості (РШ)
' ============================================================
```

```
UPWbx = y(1) - KDW * y(6)     ' Вхідна напруга регулятора
UPWp = UPWbx * Kpw            ' Пропорційна складова регулятора
f(2) = UPWbx / Tpw            ' Інтегральна складова регулятора
Upw = UPWp + y(2)               ' Вихід ПІ-регулятора швидкості

'   Тепер обмежимо, за потреби, його вихідну напругу
IF ABS(Upw) > UPWmax THEN
  Upw = UPWmax * SGN(Upw)
  f(2) = 0                      ' Зупинка інтегрування
END IF
```

Turbo Pascal

```
{ ------------------------------------------------------------ }
{     Позначення змінних:                                      }
{     y[1] - вихідна напруга задавача інтенсивності            }
{     y[2] - інтегральна складова ПІ-регулятора швидкості      }
{     y[6] - швидкість двигуна                                 }
{ ------------------------------------------------------------ }
const
   Kpw = 23.5;    { Коефіцієнт підсилення РШ                  }
   Tpw = 0.0034; { Стала часу інтегратора РШ                  }
   KDW = 0.045;  { Коефіцієнт зворотного зв'язку за швидкістю }
   UPWmax = 10.0; { Напруга обмеження РШ                      }

  . . .

{ ================================= }
{     Регулятор швидкості (РШ)      }
{ ================================= }
UPWbx := y[1] - KDW * y[6]; { Вхідна напруга регулятора    }
UPWp := UPWbx * Kpw;        { Пропорційна складова регулятора }
f[2] := UPWbx / Tpw;        { Інтегральна складова регулятора }
Upw := UPWp + y[2];         { Вихід ПІ-регулятора швидкості }

{  Тепер обмежимо, за потреби, його вихідну напругу  }
if abs(Upw) > UPWmax then
   begin
      Upw := UPWmax * Upw / abs(Upw);
      f[2] := 0.0;             { Зупинка інтегрування }
   end;
```

MathCAD

$$U_P \leftarrow \begin{cases} \text{if } |U_P| > Up_{max} \\ \quad f_1 \leftarrow 0 \quad \text{"Обчислення похідної інтегратора"} \\ \quad Up_{max} \dfrac{U_P}{|U_P|} \\ U_P \text{ otherwise} \end{cases}$$

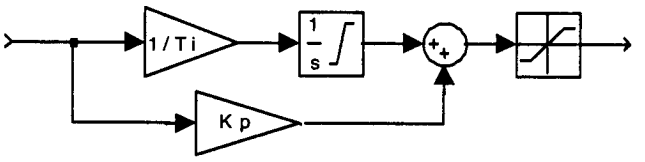
MATLAB

```
% ------------------------------------------------------------
%      f    - вектор-стовпець похідних
%      Позначення змінних:
%      y(1) - вихідна напруга задавача інтенсивності
%      y(2) - інтегральна складова ПІ-регулятора швидкості
%      y(6) - швидкість двигуна
% ------------------------------------------------------------
Kpw = 23.5;       % Коефіцієнт підсилення РШ
Tpw = 0.0034;     % Стала часу інтегратора РШ
KDW = 0.045;      % Коефіцієнт зворотного зв'язку за швидкістю
UPWmax = 10.0;    % Напруга обмеження РШ

  . . .

% ============================================================
%       Регулятор швидкості (РШ)
% ============================================================
UPWbx = y(1) - KDW * y(6); % Вхідна напруга регулятора
UPWp = UPWbx * Kpw;        % Пропорційна складова регулятора
f(2) = UPWbx / Tpw;        % Інтегральна складова регулятора
Upw = UPWp + y(2);         % Вихід ПІ-регулятора швидкості
```

```
%   Тепер обмежимо, за потреби, його вихідну напругу
if abs(Upw) > UPWmax
  Upw = UPWmax * sign(Upw);
  f(2) = 0;                          % Зупинка інтегрування
end
```

Simulink

У середовищі Simulink, яке входить у пакет MATLAB, можна використовувати інтегратор з обмеженням вихідної координати ("насиченням"), що дає змогу просто реалізувати ПІ-регулятор з обмеженням вихідної координати. У такій реалізації треба забезпечити однаковий рівень обмеження вихідного сигналу як для регулятора загалом (блок обмеження на виході регулятора), так і для інтегратора зокрема (встановленням відповідного параметра обмеження під час його налагодження).



2.6. Математичні моделі механічної частини привода

Двомасова модель механічної частини

Найпростішим варіантом урахування механічної частини привода є використання двомасової моделі [Д.3, Д.4, Д.10, Д.21, Д.27] (рис. 2.42), що описується такою системою рівнянь:

$$\begin{cases} \dfrac{d\omega_1}{dt} = \dfrac{M_1 - M_{12}}{J_1}; \\ M_{12} = C_{12} \cdot \varphi; \\ \dfrac{d\varphi}{dt} = \omega_1 - \omega_2; \\ \dfrac{d\omega_2}{dt} = \dfrac{M_{12} - M_{c2}}{J_2}, \end{cases} \tag{2.77}$$

де ω_1, ω_2 – кутові швидкості відповідно першої та другої мас;
J_1, J_2 – моменти інерції першої та другої мас;
M_1 – рушійний момент;
M_{12} – пружний момент;
M_{c2} – статичний момент навантаження;
C_{12} – пружність;
φ – кут скручування вала.

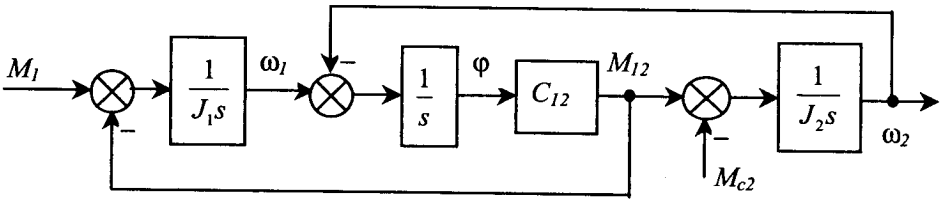


Рис. 2.42. Структурна схема моделі двомасової системи

З урахуванням в'язкого тертя (природного демпфування) пружний момент розраховують за формулою:

$$M_{12} = C_{12} \cdot \varphi + \beta \cdot (\omega_1 - \omega_2),$$

де $\beta = 2d\sqrt{\dfrac{C_{12} J_1 J_2}{J_1 + J_2}}$ – коефіцієнт в'язкого тертя;

$d = 0.1$–0.3 – коефіцієнт природного демпфування (вгамовування).

Дещо складнішою є модель механіки з урахуванням люфтів у передачах (рис. 2.43); позначивши $\Delta\varphi$ –люфт у передачі, отримаємо систему рівнянь:

$$\begin{cases} \dfrac{d\omega_1}{dt} = \dfrac{M_1 - M_{12} - \beta(\omega_1 - \omega_2)}{J_1} \ ; \\ \varphi_{12} = \begin{cases} 0 \ , & \text{якщо} \ \ |\varphi| \le \Delta\varphi/2 \ ; \\ \varphi - \Delta\varphi/2 \cdot \text{sign}(\varphi) \ , & \text{якщо} \ \ |\varphi| > \Delta\varphi/2 \ ; \end{cases} \\ M_{12} = C_{12} \cdot \varphi_{12} \ ; \\ \dfrac{d\varphi}{dt} = \omega_1 - \omega_2 \ ; \\ \dfrac{d\omega_2}{dt} = \dfrac{M_{12} - M_{c2}}{J_2} \ , \end{cases} \tag{2.78}$$

де φ – кутове розузгодження між вхідним і вихідним валами;

φ_{12} – кут скручування вала.

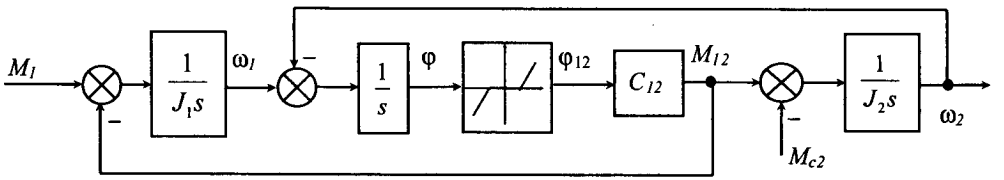


Рис. 2.43. Структурна схема моделі двомасової системи з люфтом

У розділі 4 подано приклад використання моделі механічної частини.

2.7. Математичні моделі нелінійних ланок

Типові безінерційні нелінійності

Під час досліджень динамічних режимів систем автоматизованих електроприводів у "великому", тобто у режимах пуску, реверсу, гальмування, в яких координати системи змінюються у широких (близьких до максимальних) межах, необхідно враховувати нелінійності, що властиві окремим її елементам. До них належать, наприклад, регулювальні характеристики тиристорних пере-

творювачів чи характеристики вільного ходу генераторів постійного струму, обмеження координат чи зони нечутливості тощо. Нелінійності бувають істотними, менш істотними чи практично невідчутними, тобто такими, що проявляються на усьому діапазоні зміни вхідної координати чи лише на певному відтинку її зміни. Нелінійні елементи бувають інерційними та безінерційними. Зрозуміло, що для створення точнішої математичної моделі системи необхідно враховувати і менш істотні нелінійності окремих елементів.

Нелінійності елементів систем електроприводів подаються статичною характеристикою "вхід–вихід", яка зображується на структурних схемах графічною залежністю, а у математичному описанні – відповідною аналітичною залежністю.

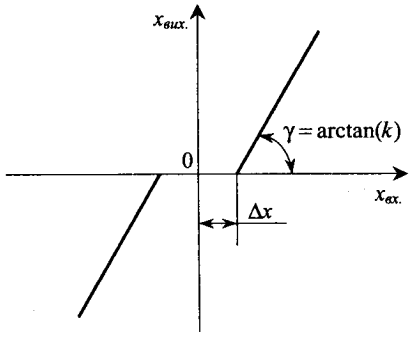
Далі наведено типові нелінійності, що найчастіше трапляються в системах електроприводів, та приклади їх реалізацій. Оскільки в стандарті мови Pascal відсутня реалізація математичної функції знака числа **sign (signum)**, нижче пропонується один з варіантів програмування такої функції в середовищі Turbo Pascal для використання у поданих далі прикладах.

Turbo Pascal

```
{ ------------------------------ }
{ Реалізація функції знака sign  }
{ ------------------------------ }
function sign(x : double): double;
begin
   if  x = 0.0  then
      sign := 0.0
   else
      sign := x/abs(x);
end.
```

1. Зона нечутливості

Такий елемент, як зона нечутливості, використовується для описання люфтів у механічних передачах і зачепленнях, для реалізації затриманих зворотних зв'язків (наприклад, струмового відсікання) тощо.

Статична характеристика	*Аналітичний вираз*
	$x_{вих.} = \begin{cases} 0 & \text{для } \lvert x_{вх.} \rvert < \Delta x; \\ k \cdot (x_{вх.} - \Delta x) & \text{для } x_{вх.} > \Delta x; \\ k \cdot (x_{вх.} + \Delta x) & \text{для } x_{вх.} < -\Delta x, \end{cases}$ де $x_{вх.}$ – вхідний сигнал; $x_{вих.}$ – вихідний сигнал; k – коефіцієнт передачі; Δx – ширина зони нечутливості.

Quick Basic

```
' ------------------------------------------------
'      Реалізація зони нечутливості
' ------------------------------------------------
CONST K = 1.5              ' Коефіцієнт передачі
CONST Dx = 0.5             ' Ширина зони нечутливості

'  Розрахунок вихідного сигналу
SELECT CASE  x
   CASE IS < -Dx
      y = K * (x + Dx)
   CASE IS > Dx
      y = K * (x - Dx)
   CASE ELSE
      y = 0
END SELECT
```

Turbo Pascal

```
{ ------------------------------------ }
{ Реалізація зони нечутливості         }
{ ------------------------------------ }
```

```
const
   K = 1.5;          { Коефіцієнт передачі       }
   Dx = 0.5;         { Ширина зони нечутливості }
  . . .
{  Розрахунок вихідного сигналу    }
if x > Dx then
   y := K*(x - Dx)
else
   if x < -Dx then
      y := K*(x + Dx)
   else
      y := 0;
```

MathCAD

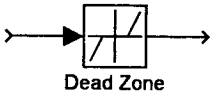
Розрахунок виходу ланки "зона нечутливості":

$$y := \begin{cases} (x - \Delta x)\cdot K & \text{if } x > \Delta x \\ (x + \Delta x)\cdot K & \text{if } x < -\Delta x \\ 0 & \text{otherwise} \end{cases}$$

MATLAB

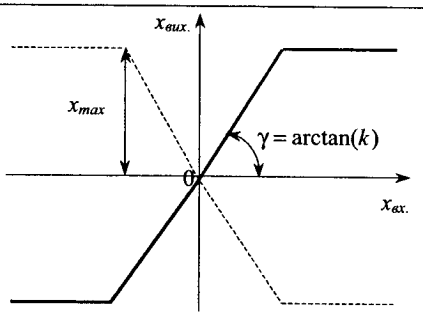
```
% ----------------------------------------
%      Реалізація зони нечутливості
% ----------------------------------------
K = 1.5;            % Коефіцієнт передачі
Dx = 0.5;           % Ширина зони нечутливості

%  Розрахунок вихідного сигналу
if x > Dx
   y = K*(x - Dx);
elseif x < -Dx
   y = K*(x + Dx);
else
   y = 0;
end
```

Simulink



2. Насичення

Насичення (обмеження вихідного сигналу) використовується для описання елементів електромеханічних систем з обмеженням вихідного сигналу, наприклад, регулятора з обмеженням вихідної напруги.

Статична характеристика	*Аналітичний вираз*
	$x_{вих} = \begin{cases} k \cdot x_{вх.} & \text{для } \lvert x_{вх.} \rvert < x_{\max}/k \\ x_{\max} \cdot \operatorname{sign}(x_{вх.}) & \text{для } x_{вх.} > x_{\max}/k \end{cases}$. де $x_{вх}$ – вхідний сигнал; $x_{вих}$ – вихідний сигнал; k – коефіцієнт передачі; x_{max} – максимальне значення вихідного сигналу.

Quick Basic

```
' ------------------------------------------------
'      Реалізація насичення
' ------------------------------------------------
CONST K = 1.5              ' Коефіцієнт передачі
CONST Ymax = 10.0          ' Величина обмеження координати

'  Розрахунок вихідного сигналу
IF ABS(x) > Ymax / K THEN
```

```
   y = Ymax * SGN(x)
ELSE
   y = x * K
END IF
```

Turbo Pascal

```
{ ------------------------------- }
{ Реалізація насичення            }
{ ------------------------------- }
const
   K = 1.5;          { Коефіцієнт передачі }
   Ymax = 10.0;      { Величина обмеження  }
var
   x, y : double;
begin
   . . .

   {  Розрахунок вихідного сигналу   }
   if abs(x) > Ymax/K then
      y := Ymax*sign(x)
   else
      y := K*x;
end.
```

MathCAD

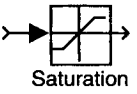
Розрахунок виходу ланки "насичення":

$$y(x) := \begin{cases} Y_{max} \cdot \text{sign}(x) & \text{if } |x| > \dfrac{Y_{max}}{K} \\ K \cdot x & \text{otherwise} \end{cases}$$

MATLAB

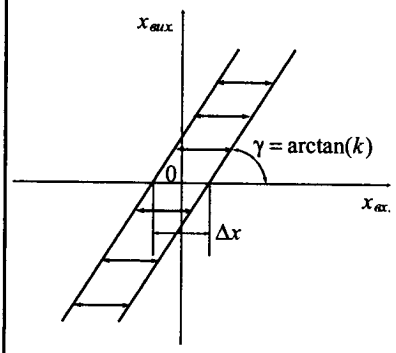
```
% ----------------------------------------
%     Реалізація насичення
% ----------------------------------------
K = 1.5;           % Коефіцієнт передачі
Ymax = 10.0;       % Величина обмеження координати

%  Розрахунок вихідного сигналу
if abs(x) > Ymax/K
   y = Ymax*sign(x);
else
   y = K*x;
end
```

Simulink



3. Люфт

Люфт зустрічається в елементах механічних передач (див. розд. 2.6). Для його правильної реалізації необхідне значення похідної вихідного (вхідного) сигналу, яке може бути знайдене як аналітично, так і за допомогою числового диференціювання.

Статична характеристика	*Аналітичний вираз*
	$x_{вих.} = \begin{cases} x_{вих.} = \text{const} & \text{для } \lvert x_{вих.} - kx_{вх.} \rvert \le \Delta x \ ; \\ k\left(x_{вх.} - \Delta x \cdot \text{sign}\left(\frac{dx_{вих.}}{dt} \right) \right) & \text{для } \frac{dx_{вих.}}{dt} \ne 0 \ , \end{cases}$ де $x_{вх.}$ – вхідний сигнал; $x_{вих.}$ – вихідний сигнал; k – коефіцієнт передачі; Δx – величина люфту.

Quick Basic

```
' ------------------------------------------------
'      Реалізація люфту
' ------------------------------------------------
CONST K = 1.5          ' Коефіцієнт передачі
CONST dX = 0.2         ' Величина люфту

' Розрахунок вихідного сигналу, fx - похідна вихідного сигналу
IF ABS(Xout - Xin * K) <= dX THEN Xout = k * (Xin - dX* SGN(fx))
```

Turbo Pascal

```
{ ---------------------------- }
{ Реалізація люфту             }
{ ---------------------------- }
const
   K = 1.5;          { Коефіцієнт передачі }
   dX = 0.2;         { Величина люфту      }
```

```
var
   Xin, Xout, fx : double;
begin
   . . .

 { Розрахунок вихідного сигналу, fx -похідна вихідного сигналу }
   if (abs(Xout - Xin*K) > dX) and (abs(fx)<>0) then
      Xout := K*(Xin - dX*sign(fx));
end.
```

MathCAD

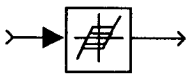
Розрахунок виходу ланки "люфт":

$$X_{out} := K \cdot \left(X_{in} - \Delta x \cdot \mathrm{sign}\left(\frac{d}{dt} X_{out} \right) \right) \quad \text{if} \quad \left| X_{out} - K \cdot X_{in} \right| \le \Delta x$$

MATLAB

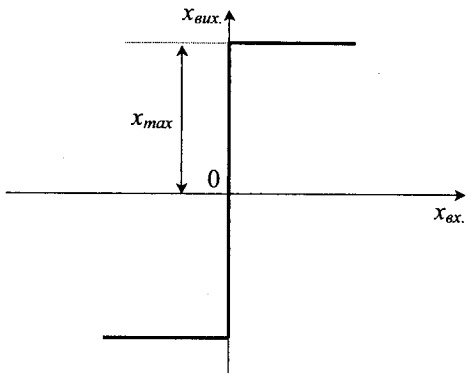
```
% ----------------------------------------
%     Реалізація люфту
% ----------------------------------------
K = 1.5;          % Коефіцієнт передачі
dX = 0.2;         % Величина обмеження координати

% Розрахунок вихідного сигналу, fx - похідна вихідного сигналу
if abs(Xout - Xin * K) <= dX
   Xout = k * (Xin - dX*sign(fx));
End
```

Simulink



4. Момент сухого тертя

Така нелінійна залежність відповідає зміні координат у разі моделювання реактивного моменту навантаження електропривода, релейним регуляторам, компараторам напруги тощо.

Статична характеристика	*Аналітичний вираз*
	$x_{вих} = x_{max} \cdot \text{sign}(x_{вх})$ де $x_{вх.}$ – вхідний сигнал; $x_{вих.}$ – вихідний сигнал

Quick Basic

```
' ------------------------------------------------
'     Реалізація моменту сухого тертя
' ------------------------------------------------
CONST Xmax = 10.0       ' Величина моменту сухого тертя
y = Xmax * SGN(x)
```

Turbo Pascal

```
{ ------------------------------ }
{ Реалізація моменту сухого тертя }
{ ------------------------------ }
const
   Xmax = 10.0;     { Величина моменту сухого тертя }
var
   x, y : double;
begin
   . . .

   {  Розрахунок вихідного сигналу    }
   y := Xmax*sign(x);
end.
```

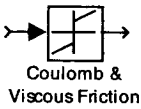
MathCAD

Розрахунок моменту сухого тертя: $y := X_{max} \cdot \text{sign}(x)$

MATLAB

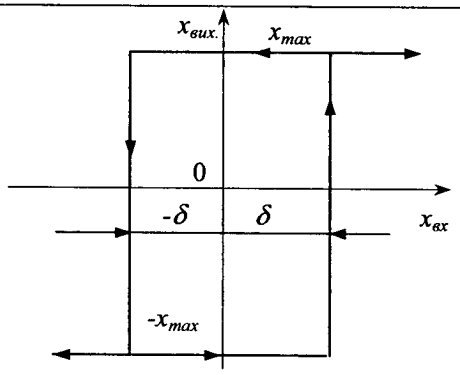
```
% ----------------------------------------
%     Реалізація моменту сухого тертя
% ----------------------------------------
Xmax = 10.0;       % Величина моменту сухого тертя
%  Розрахунок вихідного сигналу
y = Xmax*sign(x);
```

Simulink



5. Релейна характеристика

Релейна характеристика зустрічається в деяких імпульсних системах, релейних регуляторах струму тощо. Для її правильної реалізації необхідне значення похідної вихідного (вхідного) сигналу, яке може бути знайдене як аналітично, так і за допомогою числового диференціювання.

Статична характеристика	*Аналітичний вираз*
	$x_{вих.} = x_{\max} \cdot \text{sign}\left(x_{вх} + \delta \cdot \text{sign}\left(\frac{dx_{вих.}}{dt} \right) \right)$ де $x_{вх}$ – вхідний сигнал; $x_{вих}$ – вихідний сигнал; δ – половинна ширина релейної зони.

Quick Basic

```
' ------------------------------------------------
'     Реалізація релейної характеристики
' ------------------------------------------------
CONST Xmax = 10.0       ' Вихідна напруга
CONST Delta = 0.2       ' Ширина зони перемикання
```

```
'  Розрахунок вихідного сигналу Y залежно від вхідного  X
'  fx - похідна вихідного сигналу
y = Xmax * SGN(x + Delta * SGN(fx))
```

Turbo Pascal

```
{ ---------------------------------- }
{ Реалізація релейної характеристики }
{ ---------------------------------- }
const
   Xmax = 1.0;      { Величина вихідної напруги }
   Delta = 0.2;     { Ширина зони перемикання   }
var
   x, fx, y : double;
begin
   . . .

{ Розрахунок вихідного сигналу  Y залежно від вхідного  X}
{ fx - похідна вихідного сигналу}
   y := Xmax * sign(x + Delta*sign(fx))
end.
```

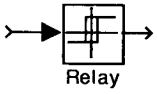
MathCAD

Реалізація релейної характеристики: $y = X_{\max} \cdot \operatorname{sign}\left(x + \delta \cdot \operatorname{sign}\left(\frac{dy}{dt}\right)\right)$

MATLAB

```
% ----------------------------------------
%     Реалізація релейної характеристики
% ----------------------------------------
Xmax = 10.0;      % Вихідна напруга
Delta = 0.2;      % Ширина зони перемикання

% Розрахунок вихідного сигналу  Y залежно від вхідного  X
% fx - похідна вихідного сигналу
y = Xmax * sign(x + Delta * sign(fx));
```

Simulink



Зрозуміло, що наведені вище нелінійності не охоплюють всієї гами статичних нелінійних залежностей, властивих елементам систем автоматизованих електроприводів. Так, в системах керування часто трапляються нелінійні характеристики 1, 2 та 3 з протилежним знаком зміни вихідного сигналу. Відповідні їм аналітичні вирази отримаємо, змінивши на протилежний знак у наведених відповідних їм виразам. Також можлива комбінація вказаних нелінійних залежностей, наприклад, якщо в елементі типу "зона нечутливості" обмежується значення вихідної координати, то аналітичний вираз цієї нелінійності можна записати так:

$$x_{вих}(x_{вх}) = \begin{cases} 0 & \text{для } |x_{вх}| \le \delta; \\ (x_{вх} - \delta \cdot \text{sign}(x_{вх})) \cdot k & \text{для } X_{\max}/k > |x_{вх}| < \delta; \\ \left(\dfrac{X_{\max}}{k} \cdot \text{sign}(x_{вх})\right) & \text{для } |x_{вх}| \ge X_{\max}/k. \end{cases}$$

Складаючи математичні моделі електромеханічних систем з нелінійностями, останні подають наведеними вище аналітичними виразами, а в цифрових моделях для цього доцільно утворити відповідні процедури (функції користувача чи підпрограми) з використанням умовних конструкцій середовища програмування.

На жаль, не завжди вдається з бажаною точністю наблизити реальні нелійності наведеними вище кусково-лінійними аналітичними виразами, доволі часто використовують гладкі нелінійності різних видів, наприклад, криві намагнічування чи характеристики керування тиристорних перетворювачів. У такому разі для встановлення аналітичного виразу, за яким в моделі обчислюватимуться значення вихідної координати нелінійного елемента для довільних значень його вхідного сигналу з області його зміни із необхідною точністю, необхідно попередньо виконати наближення характеристики "вхід–вихід" нелінійного елемента.

У задачах електромеханіки такі характеристики найчастіше задають у вигляді таблиць значень – таблично задані функції (ТЗФ), у яких задають значення функції на деякій скінченній множині значень аргументу. У вигляді таблиць такі функції подають у довідковій літературі, заводських формулярах електротехнічних матеріалів чи окремих елементів систем електроприводів. У такому вигляді подають характеристики намагнічування магнітом'якого матеріалу – залежність напруженості магнітного поля від магнітної індукції для одновимірного намагнічування цього матеріалу; крива намагнічування електричної машини; механічна характеристика двигуна постійного струму послідовного збудження у відносних одиницях для певних серій машин чи отримують експериментально, залежність динамічної вольт-амперної характеристики дуги змінного струму чи напівпровідникового діода тощо.

Залежно від способу отримання ТЗФ поділяють на детерміновані та імовірнісні (недетерміновані). До детермінованих, як правило, належать таблично задані функції, які отримують безпосередньо з математичного експерименту чи за допомогою статистичного опрацювання результатів різних експериментів, а до імовірнісних – таблично задані функції, які отримані безпосередньо у фізичному експерименті, бо результати кожного вимірювання значень як вхідного, так і вихідного сигналів нелінійного елемента, крім корисної інформації, містять ще й похибки, спричинені діями випадкових чинників: похибки вимі-

рювання, неоднакові умови упродовж експерименту (коливання напруги мережі, зміна температури обмоток машин тощо).

Отримання аналітичного виразу детермінованої таблично заданої функції нелінійного елементу виконують із залученням процедури інтерполяції, а для недетермінованої (імовірнісної) – розв'язанням задачі апроксимації.

Сутність процедури інтерполяції полягає у знаходженні аналітичного виразу – інтерполянти, що містить n невідомих коефіцієнтів за таблично заданою у n вузлах функцією. За встановленим інтерполяційною процедурою аналітичним виразом можна обчислити значення функції між вузлами інтерполяції, а у вузлах отримують абсолютно точне наближення (яке дорівнює заданому таблицею).

Для виконання алгоритму інтерполяції необхідно:

- задати вираз інтерполянти, що містить n невідомих коефіцієнтів (зрозуміло, що характер зміни інтерполянти на відрізку наближення повинен якомога краще відповідати характерові зміни таблично заданої функції);
- застосувати його почергово до кожного з вузлів інтерпольованої таблично заданої функції;
- розв'язати отриману систему алгебричних рівнянь стосовно n невідомих коефіцієнтів інтерполянти;
- підставити отримані значення коефіцієнтів інтерполянти у вираз інтерполянти.

Сутність задачі апроксимації полягає у визначенні аналітичного виразу – апроксиманти, що містить n невідомих коефіцієнтів за таблично заданою в m експериментальних точках ($n \le m$). Встановлений за процедурою апроксимації аналітичний вираз проходить поміж точками таблично заданої функції з найменшою середньоквадратичною похибкою (на множині точок плану експерименту сума квадратів відхилень значень ординат експериментальних точок від розрахункових за отриманою апроксимантою мінімальна).

Як приклад розглянемо розв'язання задачі апроксимації на прикладі лінійної апроксиманти (задача лінійної регресії), тобто $y(x) = a \cdot x + b$. Для k-ї точки плану експерименту можна записати:

$$y_k = a \cdot x_k + b,$$

де y_k, x_k – координати k-ї точки плану експерименту.

Відхилення Δ_k експериментального y_k значення функції у k-й точці плану від отриманого за апроксимантою y_{kt} у цій самій точці x_k обчислюється як

$$\Delta_k = y_{kt} - y_k = y_{kt} - (a \cdot x_k + b),$$

а квадрат означеного вище відхилення – це

$$\Delta_k^2 = (y_{kt} - (a \cdot x_k + b))^2 .$$

Згідно з методом найменших квадратів параметри (коефіцієнти) a та b апроксиманти повинні бути такими, щоб сума квадратів відхилень в усіх точках плану експерименту була мінімальною, тобто

$$\Delta(a,b) = \sum_{k=1}^{m} \Delta_k^2 = \sum_{k=1}^{m} (y_{kt} - (ax_k - b))^2 \to \min .$$

У точці аналітичного екстремуму такої функції двох незалежних змінних $\Delta(a, b)$ частинні похідні за кожною з її незалежних змінних дорівнюють нулеві, тобто

$$\frac{\partial \Delta(a, b)}{\partial a} = 0; \qquad \frac{\partial \Delta(a,b)}{\partial b} = 0 .$$

З урахуванням цього і (2) отримаємо:

$$\sum_{k=1}^{m} 2(y_{km} - (ax_k + b))x_k = 0;$$

$$\sum_{k=1}^{m} 2(y_{km} - (ax_k + b)) = 0.$$

Виконавши прості арифметичні перетворення у разі використання лінійної апроксиманти, отримаємо лінійну систему алгебричних рівнянь:

$$\left.\begin{aligned} m \cdot b + a \cdot \sum_{k=1}^{m} x_k &= \sum_{k=1}^{m} y_{mk}; \\ b \cdot \sum_{k=1}^{m} x_k + a \cdot \sum_{k=1}^{m} x_k^2 &= \sum_{k=1}^{m} x_k \cdot y_{mk} \cdot \end{aligned}\right\}$$

Розв'язання цієї системи лінійних рівнянь дасть шуканий вектор коефіцієнтів лінійної апроксиманти, що наближає таблично задану функцію за методом найменших квадратів. Функціонал $\Delta(a, b)$ за такого алгоритму розрахунку значень коефіцієнтів лінійної апроксиманти b та a набуде найменшого значення.

Узагальнивши наведений приклад лінійної регресії, запишемо алгоритм визначення коефіцієнтів (параметрів) будь-якої апроксиманти, який ґрунтується на аналітичному методі дослідження екстремуму функції багатьох незалежних змінних як поетапну послідовність виконання таких дій:

- задати вираз апроксиманти з невідомими коефіцієнтами (за математичними властивостями вибраний вираз повинен бути якнайближчим до таблично заданої функції, що підлягає апроксимації);
- скласти систему алгебричних рівнянь n-го порядку з невідомими коефіцієнтами апроксиманти, прирівнюючи до нуля частинні похідні виразу для суми квадратів відхилень за невідомими коефіцієнтами;
- розв'язати отриману (загалом нелінійну) систему алгебричних рівнянь;
- одержати конкретний вираз апроксиманти, підставивши отримані числові значення коефіцієнтів у вибраний загальний вираз апроксиманти.

Для апроксимації ТЗФ використовують степеневі чи тригонометричні многочлени, елементарні чи спеціальні функції та їх комбінації. Так, якщо ТЗФ функція є періодичною, то аналітичну залежність апроксиманти треба вибирати з класу тригонометричних многочленів.

Одним з перспективних і водночас простим шляхом інтерполяції ТЗФ є застосування сплайнів – відрізків поліномів невисокого порядку, за допомогою яких з високою точністю виконують інтерполяцію потрібної залежності в заданому діапазоні [В.10, Д.14].

Для розв'язання задач інтерполяції, апроксимації та екстраполяції в середовищі універсальних математичних пакетів є низка стандартних функцій, зокрема, пакети MathCAD і MATLAB мають широкий їх набір, що достатньо повно відображено як в документації та у допомозі до пакетів, так і у відповідній літературі [див. список літератури, част. А].

2.8. Моделювання випадкових процесів

Під час моделювання, розв'язання задач параметричної чи структурної оптимізації електромеханічних систем, що піддані дії випадкових збурень, виникає потреба у відтворенні у моделі збурень зі статичними характеристиками, які адекватні до реальних.

Для цього необхідно апріорно чи під час моделювання формувати штучні реалізації випадкових функцій. Найчастіше для відтворення в моделях реальних випадкових процесів, що відбуваються в тій чи іншій стохастичній системі, в універсальних середовищах програмування чи математичних пакетах використовують стандартну функцію генерування послідовності рівномірно розподілених на певному інтервалі випадкових (псевдовипадкових) чисел, на підставі чого далі формують реалізації випадкових функцій із заданим (відповідним реальному) законом розподілу.

Отримані у такий спосіб випадкові таблиці (масиви, матриці) дискретизованих випадкових процесів можуть використовуватися під час симулювання режимів стохастичних електромеханічних систем для відтворення в них реальних неперервних випадкових процесів.

Дослідження динамічних і статичних властивостей багатьох електромеханічних систем, що піддаються дії випадкових збурень, виконують методом статистичного моделювання. Для отримання достовірних результатів досліджень необхідно, окрім інших умов, адекватно відтворити випадкові функції, що описують збурення реальної системи.

Відтворити реальні процеси, які відбуваються в тій чи іншій системі, можна на підставі використання джерела (таблиць, генератора тощо) випадкових чисел, що відповідають випадковим величинам з різними законами розподілу.

Основою більшості алгоритмів синтезу випадкових чисел, дискретних чи неперервних випадкових функцій або системи випадкових величин і випадкових функцій є отримання послідовності випадкових чисел, які рівномірно розподілені на інтервалі [0, 1]. Надалі на підставі такої послідовності можна отримати випадкові величини із заданим законом розподілу і формувати реалізації випадкових функцій.

Розрізняють три способи отримання рівномірно розподілених випадкових чисел:

1) використання таблиць випадкових чисел;
2) генераторів випадкових чисел;
3) метод псевдовипадкових чисел – послідовності чисел, що з достатньою для інженерних розрахунків точністю близька до властивостей рівномірно розподіленої на інтервалі [0, 1] випадкової послідовності.

У цифровому моделюванні із зазначених найчастіше використовують метод псевдовипадкових чисел.

Одним з перших вважається алгоритм для отримання псевдовипадкових чисел, запропонований Дж. фон Нейманом. Згідно з цим алгоритмом для отримання послідовності випадкових чисел необхідно виконувати таку послідовність дій:

a) довільне число з діапазону [0, 1] підносять до квадрата;
b) в отриманому результаті відкидають цифри з обох боків;
c) отриману послідовність цифр приймають як дробову частину i-го випадкового числа.

Пункти **a-c** виконують багаторазово, кожного разу підставляючи отримане в п. **c** число у п. **a**; першим числом з діапазону (0, 1) задаються довільно.

Наприклад,

$$r_1 = 0.84789 \rightarrow r_1^2 = 0.71\,|\,89174\,|\,52;$$
$$r_2 = 0.89174 \rightarrow r_2^2 = 0.79\,|\,52002\,|\,27;$$
$$r_3 = 0.52002 \rightarrow r_3^2 = 0.27\,|\,04208\,|\,00\ldots$$

Інший метод генерування послідовності псевдовипадкових чисел з рівномірним розподілом передбачає виконання таких дій:

$$x_{i+1} = (M \cdot x_i + A) \bmod D,$$

де x_i, x_{i+1} – попереднє і наступне випадкові числа;
mod – операція залишку від цілочислового ділення;
M, A, D – цілі числа, які необхідно задати.

Для виконання цього алгоритму необхідно задатися також і початковим значенням (першим числом) x_0 псевдовипадкової послідовності чисел. За цим методом генерують послідовність псевдовипадкових цілих чисел з діапазону

[0; $D-1$]. Для того, щоб послідовність мала достатньо великий період повторення, числа M, A, D повинні бути достатньо великими. Добрий результат отримується, наприклад, для $M = 40$, $A = 3641$, $D = 729$. Щоб отримати послідовність дійсних чисел з діапазону [0, 1] із зазначеними властивостями, необхідно отримувані випадкові числа ділити на $D-1$.

Так, для $M = 40$, $A = 3641$, $D = 729$, $x_0 = 13$ початок послідовності псевдовипадкових чисел буде таким: 13, 516, 224, 208, 297, 212, 457, 51, 578, 517, 264, 350, 145, 693, 14, 556, а нормовані до $D-1$, вони стануть початком нової послідовності псевдовипадкових чисел: 0.018; 0.709; 0.308; 0.286; 0.408; 0.291; 0.628; 0.07; 0.794; 0.71; 0.363; 0.481; 0.199; 0.952; 0.019; 0.764...

Отримати рівномірно розподілену послідовність псевдовипадкових чисел на відрізку [0, 1] можна також, виділяючи дробову частину значення арифметичного виразу:

$$V_{i+1} = \mathrm{frac}(k \cdot V_i),$$

де $k = 8t \pm 3$ і t – непарне ціле число (наприклад, $t = 5$, $k = 37$; початковим значенням V_0 задаються).

Для отримання рівномірно розподілених на довільному відрізку [a, b] випадкових чисел на основі отриманого на відрізку [0, 1] розподілу можна за такою формулою:

$$x_{i+1} = a + (b-a)V_{i+1}.$$

Випадкові числа з різними законами розподілу отримують за допомогою відповідних формул перетворення. Так, наприклад, послідовність чисел з нормальним розподілом можна отримати за такою формулою:

$$r_i' = \sqrt{2\ln(1/V_{i+1})}\cos(2\pi V_i), \qquad r_i' = \sqrt{2\ln(1/V_{i+1})}\sin(2\pi V_i),$$

де V_i – послідовність рівномірно розподілених чисел.

За такою обчислювальною процедурою для кожного i отримують спряжену пару чисел, що має середнє значення $\overline{R} = 0$, а середньоквадратичне відхилення $\sigma = 1$. Для генерування послідовності випадкових чисел з іншими значеннями $\overline{R}$ та σ необхідно виконати перерахунок за формулою:

$$r_i = \overline{R} + r_i \cdot \sigma.$$

Часто для потреб моделювання вистачає стандартних засобів, які пропонують розробники мов програмування та математичних пакетів. Нижче подано реалізації відомих генераторів випадкових чисел.

Quick Basic

Для генерування випадкового числа використовують стандартну функцію **RND**, що записується так:

```
RND [(n)]
```

де n – необов'язковий аргумент, значення якого визначає спосіб отримання наступного випадкового числа:

n < 0 – залежно від значення отримують одне і те саме випадкове число, встановлюючи тим самим генератор у новий початковий стан;

n > 0 або випущене – з послідовності читають наступне випадкове число;

n = 0 – з послідовності повторно читають останнє отримане випадкове число.

Функція **RND** генерує дійсне число подвійної точності з послідовності випадкових чисел з рівномірним законом розподілу, що належать діапазону [0, 1]. Виходом генератора є одна і та сама послідовність випадкових чисел (фактично псевдовипадкових, бо послідовність є скінченною, хоча й дуже довгою). Під час запуску програми генератор ініціалізується завжди одним і тим самим числом, тому послідовність буде починатися завжди однаково. Для зміни такої ситуації (тобто отримання для різних запусків програми різних послідовностей випадкових чисел) застосовують оператор **RANDOMIZE**, який записують так:

```
RANDOMIZE [m%]
```

де m% – число, що використовується для ініціалізації (перевстановлення) бази генератора випадкових чисел. Якщо аргумент m% випущено, то система виводить повідомлення, у якому запрошує ввести його значення з проміжку [–32768, 32767].

Цей оператор змінює число, яким ініціалізується генератор випадкових чисел, на задане. Замість аргументу m% можна використовувати, наприклад, значення функції **TIMER**, яка застосовується без аргументів. Ця функція визначає кількість секунд, що пройшли від початку доби до поточного моменту часу. Значення функції подають дійсним числом подвійної точності.

Turbo Pascal

Для генерування випадкового числа використовують стандартну функцію **Random**, що записується так:

```
Random[(n: word)]
```

де n – необов'язковий аргумент, якщо він не заданий, то функція генерує дійсне випадкове число типу **real** у діапазоні [0, 1]; якщо параметр **n** задано – то у діапазоні [0; n];

n = 0 – у такому разі функція повертає значення нуля.

Для ініціалізації генератора випадкових чисел застосовують процедуру **Randomize**, яка використовує покази внутрішнього годинника комп'ютера (аналогічно до реалізації в Quick Basic).

MathCAD

Для генерування рівномірно розподілених випадкових чисел використовують стандартну функцію **rnd**, що записується так:

rnd(x)

і генерує дійсне випадкове число у діапазоні $[0, x]$.

Крім вказаної функції, у пакеті наявний широкий набір різноманітних функцій для генерації та роботи з випадковими числами з різними законами розподілу.

MATLAB

Для генерування випадкових чисел і матриць з рівномірним розподілом використовують стандартну функцію **rand**. Для генерування випадкових чисел і матриць з нормальним розподілом застосована стандартна функція **randn**.

2.9. Математичні моделі систем електроприводів

Моделі систем створюються з окремих вузлів та ланок, що розглянуті у попередніх розділах. У цьому розділі немає прикладів реалізації моделей – такі приклади наведено у розділі 4.

Математична модель системи Г–Д з паралельним коригуванням

Математичну модель системи Г–Д з паралельним коригуванням (рис. 2.44) подають системою диференціальних рівнянь:

$$\begin{aligned}
\frac{du_{mn}}{dt} &= \frac{u_{вх}K_{mn} - u_{mn}}{T_{mn}};\\
\frac{de_G}{dt} &= \frac{u_{mn}K_G - e_G}{T_G};\\
\frac{di_a}{dt} &= \frac{(e_G - C\omega)/R_a - i_a}{T_a};\\
\frac{d\omega}{dt} &= \frac{C(i_a - I_c)}{J}.
\end{aligned} \qquad (2.79)$$

де $u_{вх} = U_з - (K_U \cdot U_G + i_a \cdot K_i + \omega \cdot K_\omega)$ – вхідна напруга ТП;

$U_з$ – напруга завдання;

$U_G = e_G - i_a R_G$ – напруга генератора;

K_U, K_i, K_ω – коефіцієнти зворотного зв'язку відповідно за напругою генератора, струмом якоря, швидкістю;

u_{mn} – вихідна напруга ТП.

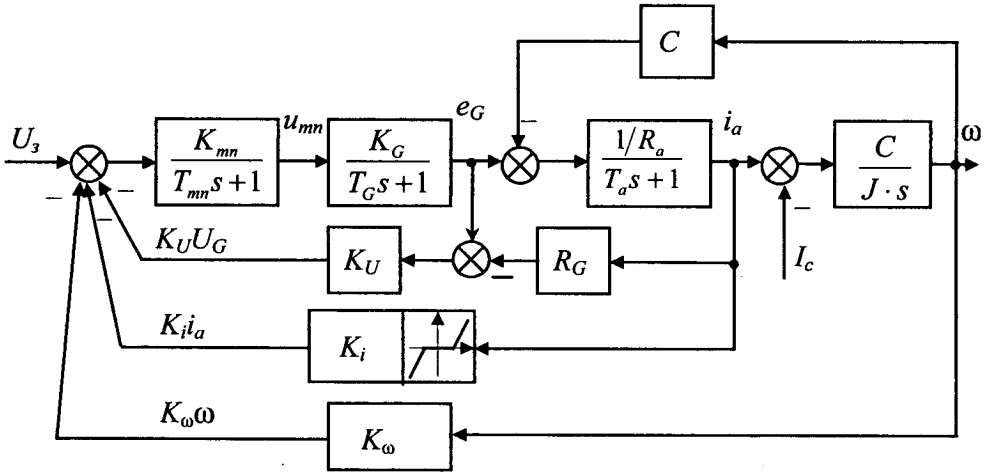


Рис. 2.44. Структурна схема системи Г–Д з паралельним коригуванням

Моделювання динамічних режимів у системах електропривода з підпорядкованим регулюванням координат (СПР)

Структурна схема одноразово інтегрувальної СПР швидкості двигуна постійного струму показана на рис. 2.45 [Д.3, Д.13, Д.21].

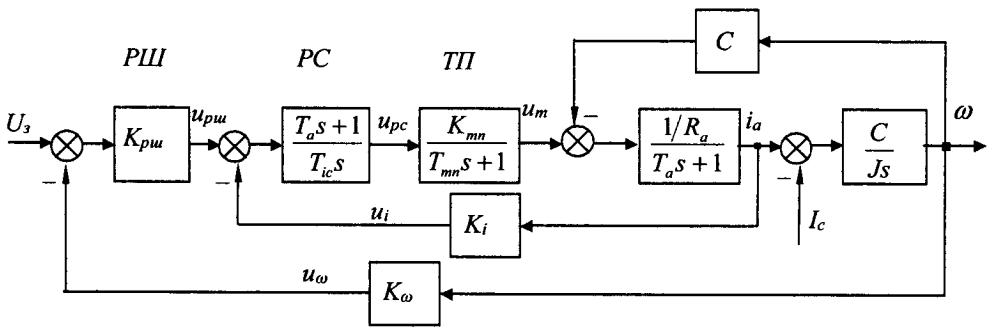


Рис. 2.45. Структурна схема СПР швидкості ДПС з одноразовим інтегруванням

Налагоджувальні параметри системи регулювання обчислюють за відомими виразами [Д.3, Д.21, Д.27]: $T_{ic} = 2T_{mn}K_{mn}K_i/R_a$; $K_{pc} = T_a/T_{ic}$;

$K_{рш} = T_{ем} C K_i / 4 T_{тп} K_\omega R_a$; $K_i = U_i^{\max} / I_a^{\max}$; $K_\omega = U_\omega^{\max} / \omega_{\max}$. Значення $U_i^{\max}$, $U_\omega^{\max}$ – максимальні значення вихідних напруг відповідно давача струму і швидкості приймають залежно від типу елементної бази регуляторів і найчастіше становлять ±10 В. Значення напруги завдання $U_з$ співмірне зі значенням $U_\omega^{\max}$ і залежить від усталеного значення кутової швидкості ω.

Математична модель такої системи, записана системою алгебричного і диференціальних рівнянь першого порядку в нормальній формі Коші, має вигляд:

$$u_{рш} = (U_з - K_\omega \omega) K_{рш};$$

необхідно передбачити обмеження $|u_{рш}| \le K_i I_a^{\max}$;

$$\begin{aligned}
\frac{du_{ic}}{dt} &= \frac{u_{рш} - K_i i_a}{T_{ic}};\\
\frac{du_{тп}}{dt} &= \frac{\left(u_{ic} + \left(u_{рш} - K_i i_a\right) K_{рс}\right) K_{тп} - u_{тп}}{T_{тп}};\\
\frac{di_a}{dt} &= \frac{(u_{тп} - C\omega)/R_a - i_a}{T_a};\\
\frac{d\omega}{dt} &= \frac{(i_a - I_c) C}{J}.
\end{aligned} \qquad (2.80)$$

Структурна модель дворазово інтегрувальної СПР електропривода постійного струму показана на рис. 2.46. Математична модель подана системою рівнянь:

$$\begin{aligned}
\frac{du_{зі}}{dt} &= \frac{U_з - u_{зі}}{T_{зі}};\\
\frac{du_{іш}}{dt} &= \frac{u_{зі} - K_\omega \omega}{T_{іш}};\\
u_{рш} &= u_{іш} + (u_{зі} - K_\omega \omega) K_{рш};
\end{aligned}$$

треба передбачити обмеження $|u_{рш}| \le u_{рш}^{\max}$ та наростання її інтегральної складової;

$$\frac{du_{ic}}{dt} = \frac{u_{рш} - K_i i_a}{T_{ic}};$$

$$u_{рс} = u_{ic} + \left(u_{рш} - K_i i_a\right) K_{рс}; \qquad (2.81)$$

$$\frac{du_{mn}}{dt} = \frac{u_{рс} K_{mn} - u_{mn}}{T_{mn}};$$

$$\frac{di_a}{dt} = \frac{\left(u_{mn} - C\omega\right)/R_a - i_a}{T_a};$$

$$\frac{d\omega}{dt} = \frac{\left(i_a - I_c\right)C}{J}.$$

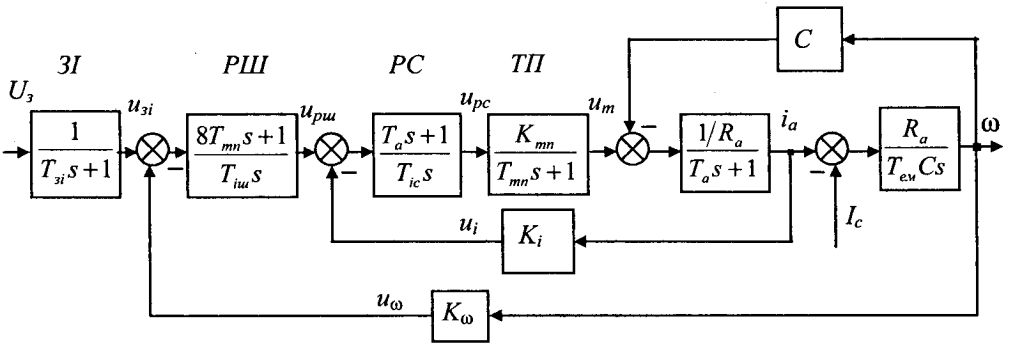


Рис. 2.46. Структурна схема дворазово інтегрувальної СПР

Важливим моментом під час моделювання ПІ-регулятора з обмеженням вихідної координати (наприклад, регулятора швидкості) є правильна реалізація в моделі власне обмеження: після досягнення максимального значення треба не просто обмежити умовним оператором рівень виходу регулятора, а й зупинити інтегрування (див. розд. 2.5).

Параметри налагодження регулятора контуру швидкості за "технічним оптимумом" у дворазово інтегрувальній СПР розраховують за формулами:

$$T_{зі} = 8T_\mu; \quad T_{іш} = \frac{32T_\mu^2 K_\omega R_a}{T_{ем} K_i C}; \quad K_{рш} = 8T_\mu / T_{іш}; \quad T_\mu = T_{mn} + T_ф.$$

3 МЕТОДИ РОЗВ'ЯЗУВАННЯ ДИФЕРЕНЦІАЛЬНИХ РІВНЯНЬ, ЩО ОПИСУЮТЬ ДИНАМІЧНІ РЕЖИМИ ЕЛЕКТРОПРИВОДІВ

Перехідні процеси в системах електроприводів можуть бути описані системами диференціальних рівнянь. Для аналізу динамічних властивостей електроприводів у різноманітних режимах – запуску, гальмування, реверсування, зміни навантаження, регулювання параметрів необхідно розв'язати диференціальні рівняння. Розрізняють аналітичне, числове і графічне розв'язування. Аналітичне передбачає отримання інтегральних функцій, які є розв'язками диференціальних рівнянь. Числові методи забезпечують обчислення значень інтегральних функцій у заданих точках, а графічні – тільки їх графіки (див. таблицю нижче).

Розв'язування диференціальних рівнянь	Вигляд отриманих результатів		
1) аналітичне	інтегральні функції	таблиці значень	графіки
2) числове	–	таблиці значень	графіки
3) графічне	–	–	графіки

Аналітичне розв'язування диференціальних рівнянь забезпечує отримання точних результатів, числові методи – наближених. Графічні методи, наприклад, метод А. В. Башаріна [Д.3] чи метод приростів (графоаналітичний), мають невелику точність і поступаються за точністю числовим, тому можуть використовуватися іноді як ілюстративні.

3.1. Аналітичні методи

Загальні відомості про диференціальні рівняння

Диференціальні рівняння – це співвідношення, які пов'язують незалежну змінну x, невідому функцію y та її похідні. У диференціальних рівняннях невідомою є функція, яка входить у рівняння під знаком похідних того чи іншого порядку.

Наприклад, у диференціальному рівнянні

$$y' = x^2 - ax$$

невідомою є функція $y(x)$, яку можна отримати, проінтегрувавши її похідну:

$$\int dy = \int \left(x^2 - ax\right)\ dx; \qquad y = \int \left(x^2 - ax\right)\ dx.$$

Будь-яку функцію, яка задовольняє на деякому проміжку це диференціальне рівняння, називають його розв'язком.

Приклад: Розв'язати диференціальне рівняння $y' = 3x$.

Проінтегруємо:

$$\int dy = \int 3x\ dx \Rightarrow y = \frac{3}{2}x^2 + C.$$

Отриманий вираз – це загальний розв'язок, C – довільна стала. Загальний розв'язок містить нескінченну множину часткових для конкретних значень сталої C. Це ілюструє рис. 3.1, для якого наведено графіки часткових розв'язків для $C = 0; 1; 2$. Графік кожного часткового розв'язку називається інтегральною кривою, а його функція – інтегральною функцією.

Частковий розв'язок задається значенням $y(x_0) = y_0$, де x_0, y_0 – числа, які називають початковими значеннями (умовами). Це означає, що інтегральна крива повинна проходити через точку (x_0, y_0). З цією проблемою ми зіткнемося далі у ході визначення початкових значень координат систем електроприводів під час моделювання їх динамічних режимів.

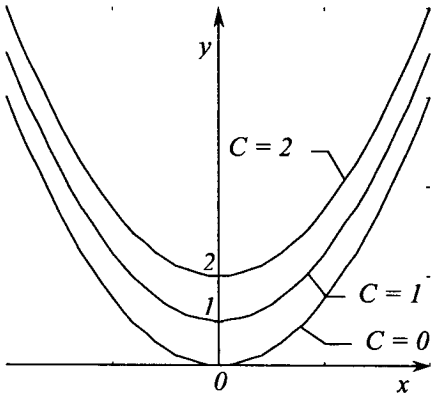


Рис. 3.1. Інтегральні криві часткових розв'язків диференціального рівняння

Приклад: Знайти частковий розв'язок диференціального рівняння $y' = 3x$ для початкової умови $y(0) = 2$, загальний розв'язок якого $y = \frac{3}{2}x^2 + C$.

Із загального розв'язку отримуємо:

$$2 = \frac{3}{2} \cdot 0 + C \Rightarrow C = 2.$$

Диференціальне рівняння першого порядку

Диференціальне рівняння першого порядку

$$y' + ay = f(x), \tag{3.1}$$

де a і $f(x)$ є відомими функціями від x, які задані та неперервні на деякому проміжку, називають неоднорідним лінійним диференціальним рівнянням. Всі розв'язки такого рівняння задають формулою

$$y = y_{одн} + y_{част},$$

де $y_{одн}$ – будь-який частковий розв'язок відповідного однорідного рівняння (диференціальне рівняння (3.1) без правої частини);

$y_{част}$ – будь-який частковий розв'язок рівняння (3.1).

Розглянемо на прикладі аналітичне розв'язування диференціального рівняння, яке описує динамічні режими двигуна постійного струму незалежного збудження за допущення, що $L_a = 0$.

З рівняння руху електропривода

$$M - M_c = J \cdot \omega'; \qquad M = i_a k\Phi,$$

визначаємо струм якоря двигуна

$$i_a = \frac{M_c}{k\Phi} + \frac{J}{k\Phi} \cdot \omega'.$$

Тоді підставимо отриманий вираз у рівняння балансу напруг якірного кола двигуна:

$$U = i_a R_a + k\Phi\omega.$$

Як результат отримаємо диференціальне рівняння

$$\omega' + \frac{\omega}{T_M} = \frac{(\omega_0 - \Delta\omega_c)}{T_M}, \tag{3.2}$$

де

$$\omega_0 = \frac{U}{k\Phi}; \qquad \Delta\omega_c = \frac{R_a M_c}{(k\Phi)^2}; \qquad T_M = J\frac{R_a}{(k\Phi)^2}.$$

Спочатку знайдемо загальний розв'язок однорідного диференціального рівняння

$$\omega' + \frac{\omega}{T_M} = 0,$$

подаючи його як

$$\frac{d\omega}{dt} = -\frac{\omega}{T_M} \qquad \text{або} \qquad \frac{d\omega}{\omega} = -\frac{dt}{T_M}.$$

Проінтегрувавши його ліву і праву частини, отримаємо:

$$\int\frac{d\omega}{\omega} = -\int\frac{dt}{T_M}; \qquad \ln\omega = -\frac{t}{T_M} + C,$$

$$\omega = e^{-\frac{t}{T_M} + C}; \qquad \omega = e^{-\frac{t}{T_M}} \cdot e^C,$$

або

$$\omega = e^{-\frac{t}{T_M}} C. \tag{3.3}$$

Частковий розв'язок отримаємо як частковий випадок усталеного (статичного) режиму, коли $\omega' = 0$. Тоді з (3.2) маємо

$$\frac{\omega}{T_M} = \frac{\omega_0 - \Delta\omega_c}{T_M},$$

звідки

$$\omega = \omega_0 - \Delta\omega_c = \omega_c. \tag{3.4}$$

Розв'язок рівняння (3.2) з урахуванням (3.3) і (3.4) має вигляд

$$\omega = Ce^{-\frac{t}{T_M}} + \omega_c.$$

Значення C знайдемо за умови $t = 0$; $\omega = \omega_{поч}$, тобто

$$\omega_{поч} = C + \omega_c;$$

звідки

$$C = \omega_{поч} - \omega_c,$$

Остаточно

$$\omega = (\omega_{поч} - \omega_c) \cdot e^{-\frac{t}{T_M}} + \omega_c,$$

або

$$\omega = \omega_{поч} \cdot e^{-\frac{t}{T_M}} + \left(1 - e^{-\frac{t}{T_M}}\right) \cdot \omega_c. \tag{3.5}$$

Функція (3.5) $\omega(t)$ є аналітичним розв'язком диференціального рівняння (3.2) і є його точним розв'язком. Маючи аналітично отриману інтегральну функцію, можна заповнити таблицю значень функції та побудувати її графік.

Аналітичне розв'язування диференціальних рівнянь вищих порядків

Аналітичний метод розв'язування звичайних диференціальних рівнянь має незаперечні переваги перед числовими:

1) на відміну від числових методів для **будь-якого** кроку:
   - отриманий розв'язок є точним;
   - відсутня проблема числової нестійкості;
2) програма, що ґрунтується на аналітичному розв'язанні, є простішою, зрозумілішою і швидше виконується.

Загалом лінійна система описується рівнянням в операторній формі

$$a_n s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0 = f(s),$$

де $f(s)$ – зовнішня збурювальна дія;
a_i – коефіцієнти рівняння.

Для часткового розв'язання лінійної системи, яка описується лінійним однорідним рівнянням зі сталими коефіцієнтами $a_n s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0 = 0$, кожному дійсному кореню α_i відповідає частковий розв'язок $A_i e^{\alpha_i t}$ (*у частковому випадку може бути і* $\alpha_i = 0$); кожному дійсному кореню кратності m відповідає m часткових розв'язків: $e^{\alpha_k t}\left(A_k + A_{k+1} t + \cdots + A_{k+m-1} t^{m-1}\right)$. Кожній парі комплексно спряжених коренів $\alpha_i \pm \beta_i$ відповідає частковий розв'язок

$$e^{\alpha_i t}\left(A_i \sin(\beta_i t) + B_i \cos(\beta_i t)\right) = C_i e^{\alpha_i t} \sin(\beta_i t + \phi_i),$$

а кожній парі комплексно спряжених коренів кратності m відповідає m часткових розв'язків:

$$e^{\alpha_i t}\left(A_1 \sin(\beta_i t) + B_1 \cos(\beta_i t) + \cdots + A_m t^{m-1} \sin(\beta_i t) + B_m t^{m-1} \cos(\beta_i t)\right) =$$

$$= e^{\alpha_i t}\left(C_1 \sin(\beta_i t + \phi_1) + \cdots + C_m t^{m-1} \sin(\beta_i t + \phi_m)\right).$$

Такий підхід придатний і для систем, в яких коефіцієнти є поліномами від незалежної змінної t. Такі системи називають параметричними.

Знаходження аналітичного розв'язання для диференціальних рівнянь вищих порядків є складною процедурою навіть із застосуванням комп'ютерних програм аналітичної математики. Полегшити цей процес можна, використовуючи перетворення Лапласа.

Запис диференціальних рівнянь за передатними функціями Лапласа

Системи електроприводів зручно зображати з'єднанням окремих ланок, кожна з яких описана передатною функцією Лапласа. Такі структури зручні для аналізу режимів роботи системи, синтезу регуляторів, є наочними. Передатні функції простих типових ланок відомі. Це дає змогу створювати математичні моделі систем як системи диференціальних рівнянь за відомими передатними функціями за допомогою переходу від зображення до оригіналів нескладними перетвореннями. Проілюструємо цю процедуру на простому прикладі.

Як відомо, зображення за Лапласом оригіналу функції $x(t)$ з використанням інтеграла Лапласа таке

$$L(x(t)) = X(s) = \int_0^\infty x(t)e^{-st}\,dt, \tag{3.6}$$

де s – комплексна змінна.

Нехай маємо функцію диференціювання

$$y(t) = dx/dt = \frac{d}{dt}x. \tag{3.7}$$

Згідно з (3.6) знайдемо її зображення:

$$Y(s) = L\left(\frac{dx}{dt}\right) = \int_0^\infty y(t)e^{-st}\,dt = \int_0^\infty e^{-st}\,dx.$$

За відомою формулою інтегрування частинами $\int u dv = u \cdot v - \int v du$ обчислимо значення інтеграла, прийнявши $u = e^{-st}$, $dv = dx$. Тоді

$$Y(s) = \left(x(t)e^{-st}\right)\Big|_0^\infty - \int_0^\infty (-s)e^{-st}x(t)\;\;dt = s\int_0^\infty x(t)e^{-st}\;\;dt - x(0).$$

Якщо початкові умови вважати нульовими, тобто, $x(0) = 0$, тоді з урахуванням (3.6) отримаємо

$$Y(s) = sX(s). \tag{3.8}$$

Порівнюючи вирази (3.7) і (3.8), приходимо до висновку, що перейти від передатних функцій до диференціальних рівнянь можна формальною заміною комплексної змінної s оператором диференціювання, тобто, $s \to \frac{d}{dt}$. Це правило можна визначити по-іншому: операція диференціювання оригіналу відповідає операції перемноження зображення цього оригіналу на комплексне число s (оператор Лапласа). Відповідно, операція інтегрування – ділення зображення на s.

Покажемо на прикладі аперіодичної ланки (генератор постійного струму, якірне коло ДПС), передатна функція якої $W(s) = \frac{K}{Ts+1}$, якщо $X(s)$, $Y(s)$ – зображення вхідного і вихідного сигналів, відповідно,

$$\frac{Y(s)}{X(s)} = \frac{K}{Ts+1},$$

$$s \cdot T \cdot Y(s) + Y(s) = K \cdot X(s).$$

Переходимо до оригіналів:

$$T\frac{d}{dt}y + y = K \cdot x \quad \to \quad \frac{dy}{dt} = \frac{Kx - y}{T}.$$

Останній запис диференціального рівняння можна подати в операторній формі:

$$sy = \frac{Kx - y}{T}.$$

Розв'язування диференціальних рівнянь за допомогою перетворення Лапласа

Аналітичне розв'язування лінійних диференціальних рівнянь можна спростити, застосувавши перетворення Лапласа [Б.4, В.3]. Таке перетворення дає змогу отримати аналітичний розв'язок диференціального рівняння лінійної

системи за умови наявності прямого перетворення Лапласа $X(s)=L\big(x(t)\big)$ вхідного сигналу $x(t)$. Прикладом такого розв'язування є знаходження перехідної характеристики лінійної системи (реакції системи на одиничний стрибкоподібний сигнал, для якого перетворення Лапласа відоме: $L\big(1(t)\big)=1/s$). У такому разі переваги перетворення Лапласа безсумнівні, особливо з огляду на можливість широкого використання доступних тепер засобів комп'ютерної аналітичної математики, наприклад, пакета MathCAD.

Розглянемо диференціальне рівняння першого порядку $Ty'+y=f(t)$ зі сталими коефіцієнтами і початковою умовою $y(0)=0$, але з довільною збурювальною (збуджувальною) функцією $f(t)$ і застосуємо до його лівої та правої частин перетворення Лапласа:

$$L\big(Ty'+y\big)=L\big(f(t)\big) \quad \text{або} \quad T\cdot L\big(y'\big)+L\big(y\big)=L\big(f(t)\big).$$

Теорема диференціювання для оригіналу є однією з найважливіших для практичних застосувань перетворення Лапласа:

$$f'(t) \;\Leftrightarrow\; sF(s)-f(0);$$

$$f''(t) \;\Leftrightarrow\; s^2F(s)-sf(0)-f'(0);$$

$$\ldots$$

$$f^{(n)}(t) \;\Leftrightarrow\; s^nF(s)-s^{n-1}f(0)-s^{n-2}f'(0)-\;\ldots\;-sf^{(n-2)}(0)-f^{(n-1)}(0).$$

Використавши теорему диференціювання для оригіналу наведеного диференціального рівняння першого порядку, отримаємо

$$T\cdot\big(s\cdot y(s)-y(0)\big)+y(s)=f(s).$$

Звідси

$$\big(Ts+1\big)\cdot y(s)=f(s)+Ty(0);$$

тоді

$$y(s)=\frac{f(s)}{Ts+1}+\frac{Ty(0)}{Ts+1} \;\Rightarrow\; L^{-1}\left(\frac{f(s)}{Ts+1}\right)+L^{-1}\left(\frac{Ty(0)}{Ts+1}\right).$$

Перша складова цього рівняння – добуток двох зображень $\left(f(s) \text{ і } \frac{1}{Ts+1}\right)$ – є згорткою оригіналів, а другу просто знаходять за таблицею перетворень Лапласа, тобто

$$y(t)=\frac{f(t)}{T}*e^{-\frac{t}{T}}+y(0)\cdot\left(1-e^{-\frac{t}{T}}\right)=e^{-\frac{t}{T}}\int_0^t f(\tau)\cdot e^{-\frac{t-\tau}{T}}d\tau+y(0)\left(1-e^{-\frac{t}{T}}\right).$$

Інтеграл згортки можна інтерпретувати у такий спосіб:

Нехай на проміжку часу від $\tau=0$ *до* $\tau=t$ *діє деякий змінний фактор* $f(\tau)$, *тоді сумарний ефект від його дії становитиме* $\int_0^t f(\tau)d\tau$. *Якщо ж кожному значенню фактора* $f(\tau)$ *надати ваговий коефіцієнт* $e^{-\frac{t-\tau}{T}}$, *який залежить від проміжку часу, що пройшов між моментом* τ *виникнення фактора і моментом* t *спостереження, тобто від* $t-\tau$, *то у такому разі сумарний ефект від дії усіх чинників визначатиметься інтегралом* $\int_0^t f(\tau)\cdot e^{-\frac{t-\tau}{T}}d\tau$.

Для диференціального рівняння другого порядку

$$y''+c_1y'+c_0y=f(t)$$

зі сталими дійсними коефіцієнтами, початковими умовами $y(0)$, $y'(0)$ і з довільною збурювальною (збуджувальною) функцією $f(t)$ відображенням буде

$$\left(s^2Y(s)-sy(0)-y'(0)\right)+c_1\left(sY(s)-y(0)\right)+c_0Y(s)=F(s).$$

Розв'язання цього рівняння в операторній формі матиме вигляд

$$Y(s)=F(s)\frac{1}{s^2+c_1s+c_0}+y(0)\frac{s+c_1}{s^2+c_1s+c_0}+y'(0)\frac{1}{s^2+c_1s+c_0},$$

а в часовій області може бути знайдений за таблицями перетворення Лапласа, розкладом на найпростіші дроби або за допомогою оберненого перетворення Лапласа у середовищі математичного пакета, як показано у прикладі нижче.

Приклад: Знайти значення струму і швидкості під час прямого запуску двигуна постійного струму з незалежним збудженням при номінальному потоці. Напруга на якорі двигуна становить $U_a = 110$ В, сумарний опір якірного кола $R_a = 0.1$ Ом, електромагнітна стала часу якірного кола $T_a = 0.05$ с, стала двигуна $C = 2.5$ В·с$^{-1}$, момент інерції привода $J = 2$ кг·м^2.

Пояснення: Передатну функцію двигуна постійного струму з незалежним для $\Phi = \Phi_н$ збудженням складають на основі його структурної моделі, що подана у розд. 2. Відображення стрибка напруги на вході моделі двигуна в операторній формі має вигляд $\frac{U_a}{s}$. Далі подано приклад розв'язування цієї задачі у середовищі пакета MathCAD.

MathCAD

Напруга на якорі двигуна $U_a := 110$

Сумарний опір якірного кола $R_a := 0.1$

Стала часу якірного кола $T_a := 0.05$

Стала двигуна $C := 2.5$

Момент інерції привода $J := 2$

Передатна функція за струмом:

$$W_i(s) := \frac{\frac{1}{R_a}}{\frac{T_a \cdot s + 1}{1 + \frac{1}{R_a} \cdot \frac{C}{J \cdot s} \cdot C}} \left| \begin{array}{l} \text{simplify} \\ \text{float}, 4 \end{array} \right. \rightarrow 200. \cdot \frac{s}{\left(s^2 + 20. \cdot s + 625.\right)}$$

Передатна функція за швидкістю:

$$W_\omega(s) := \frac{\dfrac{\dfrac{1}{R_a}\cdot\dfrac{C}{J\cdot s}}{T_a\cdot s+1}}{1+\dfrac{\dfrac{1}{R_a}\cdot\dfrac{C}{J\cdot s}}{T_a\cdot s+1}\cdot C} \left|\begin{array}{l}\text{simplify}\\ \text{float},4\end{array}\right. \rightarrow \frac{250.}{\left(s^2+20.\cdot s+625.\right)}$$

Відображення швидкості: $\Omega(s) := W_\omega(s)\cdot\dfrac{U_a}{s}$

Відображення струму якоря: $I(s) := W_i(s)\cdot\dfrac{U_a}{s}$

Знаходимо оригінали засобами аналітичної математики пакета:

$$\omega(t) := \Omega(s) \left|\begin{array}{l}\text{invlaplace},s\\ \text{float},4\end{array}\right. \rightarrow 44.-19.2\cdot\exp(-10.\cdot t)\cdot\sin(22.91\cdot t)-44.\cdot\exp(-10.\cdot t)\cdot\cos(22.91\cdot t)$$

$$i(t) := I(s) \left|\begin{array}{l}\text{invlaplace},s\\ \text{simplify}\\ \text{float},4\end{array}\right. \rightarrow 960.2\cdot\exp(-10.\cdot t)\cdot\sin(22.91\cdot t)$$

Будуємо часові залежності для $t := 0, 0.001..0.5$

Аналогічно розв'язують диференціальні рівняння вищого порядку.

Приклад: Знайти перехідну характеристику системи автоматичного регулювання з передатною функцією

$$W(s)=\frac{0.1s^2+0.5s+1}{0.2s^4+0.8s^3+2.4s^2+1.2s+1}.$$

Пояснення: Перехідна характеристика є реакцією системи на одиничний стрибкоподібний сигнал, який має відображення за Лапласом $1/s$. Відповідно перехідна характеристика матиме вигляд $H(s)=\frac{W(s)}{s}$.

Далі подано приклад розв'язування цієї задачі в середовищі пакета MathCAD.

MathCAD

Передатна функція системи:

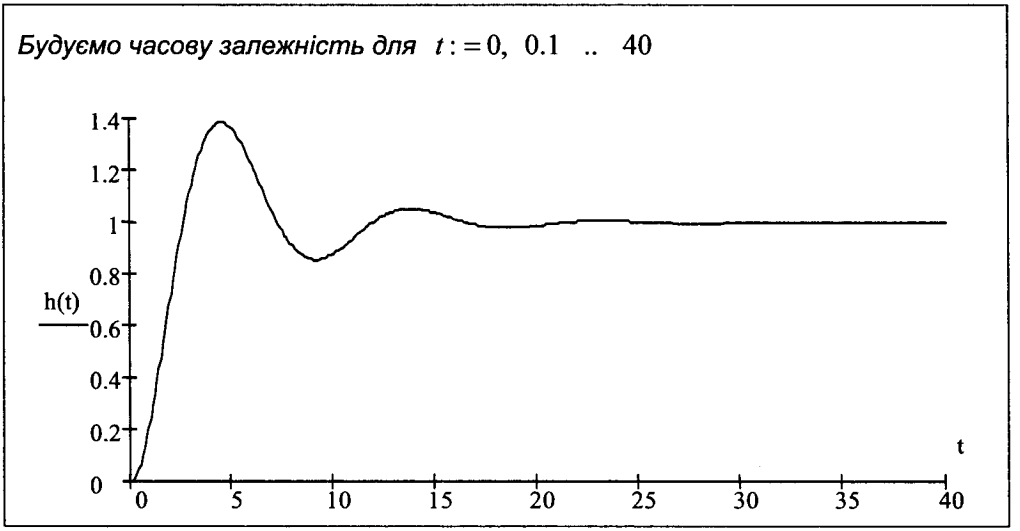
$$W(s):=\frac{0.1s^2+0.5s+1}{0.2s^4+0.8s^3+2.4s^2+1.2s+1}$$

Знаходимо оригінал засобами аналітичної математики пакета:

$$h(t):=\frac{W(s)}{s}\left|\begin{array}{l}\text{invlaplace, s}\\ \text{simplify}\\ \text{float, 3}\end{array}\right.\rightarrow 1.+2.72\cdot10^{-2}\cdot\exp(-1.79\cdot t)\cdot\cos(2.61\cdot t)-\ldots$$

$$-1.27\cdot10^{-2}\cdot\exp(-1.79\cdot t)\cdot\sin(2.61\cdot t)-$$

$$-1.03\cdot\exp(-.211\cdot t)\cdot\cos(.675\cdot t)\ -.199\cdot\exp(-.211\cdot t)\cdot\sin(.675\cdot t)$$



Як видно з прикладу, за допомогою засобів аналітичної (символьної) математики пакета MathCAD задачі такого типу розв'язують досить просто: математичний пакет дає змогу уникнути значного обсягу "ручних" аналітичних перетворень під час використання оберненого перетворення Лапласа для знаходження часових залежностей. Аналогічно встановлюють часові залежності для вхідних сигналів іншого типу (наприклад, синусоїдного), важливо лише, щоб такий сигнал можна було описати аналітично за допомогою перетворення Лапласа.

Засоби аналітичної математики також є і в пакеті MATLAB, але не в такому зручному вигляді, як у MathCAD. Для складних задач, перед якими засоби згаданих пакетів безсилі, можна спробувати або поділити задачу на окремі простіші складові, або розв'язати у потужніших математичних середовищах типу Maple або Mathematica.

Перетворення Лапласа є доволі зручним інструментом для розв'язування диференціальних рівнянь, що описують лінійні моделі електромеханічних систем. На жаль, такий підхід для моделювання нелінійних систем можна використовувати лише з певними обмеженнями, застосування яких вимагає глибокого розуміння перебігу фізичних процесів у модельованій системі.

3.2. Числові методи розв'язування диференціальних рівнянь, що записані у нормальній формі Коші

Аналітичне розв'язування диференціальних рівнянь або їх систем, які описують динаміку систем електроприводів і які є їх математичною моделлю, в інженерній практиці часто стикається з деякими труднощами. Є дві очевидні причини цьому. Перша – необхідність мати достатній обсяг знань з такого розділу вищої математики, як диференціальне та інтегральне числення. Особливо часто виникають труднощі під час розв'язування систем нелінійних диференціальних рівнянь, для більшості з яких взагалі не існує аналітичного розв'язання. Адже в реальних системах електроприводів здебільшого є ланки (елементи) з нелінійними характеристиками, наприклад, нелінійності намагнічування в зоні насичення, характеристики керування тиристорних перетворювачів, затримані зворотні зв'язки (відсікання), обмеження вихідного сигналу та нелінійні характеристики регуляторів, люфти тощо. Друга – наявність зворотних зв'язків у системах призводить до того, що аналітично розв'язати відповідні системи диференціальних рівнянь складно, а інколи і неможливо. Тому розроблялися інші способи аналізу динамічних процесів (перехідних процесів) без аналітичного розв'язування.

Ще три-чотири десятиліття тому широко використовували так звані частотні методи для оцінки стійкості систем та якості їх перехідних процесів, наприклад, методи, що побудовані на застосуванні логарифмічних амплітудних і фазних частотних характеристик. Для побудови графіків перехідних процесів користувалися методом трапецієподібних частотних характеристик з побудовою графіків дійсної частини частотної характеристики системи, розділенням її на трапеції та обчисленням коефіцієнтів нахилу в певних точках, а потім за таблицями так званих h_χ*-функцій будували графіки перехідних процесів. Процедура хоч і нескладна, проте громіздка, придатна тільки для лінійних систем загалом або лінійних на деякому проміжку. А головне, що точність такого методу побудови* міліметровий папір + лінійка + олівець *невисока.*

З появою обчислювальних машин і широким їх впровадженням в практику обчислень стало можливим використовувати числові методи розв'язування диференціальних рівнянь, які відомі вже давно.

Рівняння стану, що описують систему, яка моделюється, часто записують у нормальній формі Коші, тобто у вигляді системи диференціальних рівнянь, у лівій частині якої записують похідні координат, а у правій – вирази для їх обчислень:

$$\begin{cases} y_1' = f_1(x_1, \ \dots \ , x_m, y_1 \ , \ \dots \ , \ y_n, \ t) \ ; \\ \quad \dots \\ y_n' = f_n(x_1, \ \dots \ , \ x_m, \ y_1, \ \dots \ , \ y_n, \ t) \ ; \end{cases}$$

де $x_1, \dots, x_m$ – зовнішні збурення;

$y_1, \dots, y_n$ – координати системи;

t – незалежна змінна (час).

Переважна більшість відомих числових методів призначена для розв'язування диференціальних рівнянь саме в такій формі, і їх можна поділити на два типи методів:

- однокрокові, в яких враховується інформація тільки з попереднього кроку;
- багатокрокові, в яких використовується інформація з більш ніж одного попередніх кроків.

Кожен з цих методів має прихильників і критиків, які відстоюють переваги одних і підкреслюють недоліки інших, але ще раз відзначимо: *є раціональний метод розв'язування кожної задачі*, причому виграш у часі розрахунку інколи може становити 5–10 разів. Ще одним важливим моментом практичного застосування числових методів є точність отриманого розв'язку – немає потреби використовувати високоточні методи, якщо точність вимірювання чи розрахунку параметрів моделі становить 1–5 % (*а часто точність параметрів моделі ще нижча*), а саме з таким порядком точності найчастіше доводиться мати справу. Практика свідчить, що для моделювання автоматизованих електроприводів найефективнішими є формули числового інтегрування невисокого порядку, які будуть розглянуті далі.

Окремі елементи систем автоматизованих електроприводів описують звичайними диференціальними рівняннями вищих порядків, наприклад, багатомасові механічні системи з врахованими пружностями, електричні фільтри тощо. Водночас класичні числові методи розв'язування диференціальних рівнянь пристосовані для розв'язування диференціальних рівнянь першого поряд-

ку. У такому разі диференціальне рівняння вищого порядку треба звести до системи диференціальних рівнянь першого порядку, як описано нижче.

Нехай задане диференціальне рівняння *n*-го порядку:

$$a_n y^{(n)} + a_{n-1} y^{(n-1)} + \quad \ldots \quad + a_2 y'' + a_1 y' + a_0 y = \mathrm{f}(x).$$

Введемо проміжні змінні $y_1, y_2, \ldots, y_{n-1}$ та виконаємо заміну для похідних:

$$\frac{dy}{dt} = y_1 \ ;$$

$$\frac{d^2 y}{dt^2} = \frac{dy_1}{dt} = y_2 \ ;$$

$$\ldots$$

$$\frac{d^{n-1} y}{dt} = \ \ldots \ = \frac{dy_{n-2}}{dt} = y_{n-1} \ .$$

Тепер вихідне диференціальне рівняння *n*-го порядку можна записати так:

$$a_n \frac{dy_{n-1}}{dt} + a_{n-1} y_{n-1} + \quad \ldots \quad + a_2 y_2 + a_1 y_1 + a_0 y = \mathrm{f}(x).$$

Долучимо до нього рівняння, які визначають заміну похідних, і отримуємо систему з *n* диференціальних рівнянь першого порядку:

$$\frac{dy}{dt} = y_1;$$

$$\frac{dy_1}{dt} = y_2;$$

$$\ldots$$

$$\frac{dy_{n-2}}{dt} = y_{n-1};$$

$$\frac{dy_{n-1}}{dt} = \frac{\mathrm{f}(x) - a_{n-1} y_{n-1} - \quad \ldots \quad - a_2 y_2 - a_1 y_1 - a_0 y}{a_n}.$$

Приклад: Звести диференціальне рівняння третього порядку до системи диференціальних рівнянь першого порядку, поданих у нормальній формі Коші.

Задано диференціальне рівняння:

$$a_3 y''' + a_2 y'' + a_1 y' + a_0 y = \mathrm{f}(x).$$

Введемо проміжні змінні y_1, y_2 для заміни похідних:

$$\frac{dy}{dt} = y_1;$$
$$\frac{d^2y}{dt^2} = \frac{dy_1}{dt} = y_2.$$

Після підстановки проміжних змінних система диференціальних рівнянь матиме такий вигляд

$$\begin{aligned} &\frac{dy}{dt} = y_1; \\ &\frac{d^2y}{dt^2} = \frac{dy_1}{dt} = y_2; \\ &a_3\frac{dy_2}{dt} + a_2 y_2 + a_1 y_1 + a_0 y = \mathrm{f}(x); \end{aligned} \xrightarrow[\text{отримаємо}]{\text{звідси}} \begin{aligned} &\frac{dy}{dt} = y_1; \\ &\frac{dy_1}{dt} = y_2; \\ &\frac{dy_2}{dt} = \frac{\mathrm{f}(x) - a_2 y_2 - a_1 y_1 - a_0 y}{a_3}. \end{aligned}$$

Введення у числові методи

Найпростішим числовим методом розв'язування диференціальних рівнянь є **метод Ейлера** (*Euler*). Для лінійного диференціального рівняння першого порядку, яке записане у нормальній формі Коші,

$$y' = \mathrm{f}(x, y)$$

з початковими умовами $y(x_0) = y_0$ у рівновіддалених точках $x_0, x_1, ..., x_n$ з кроком $h = x_{i+1} - x_i$, наближені значення інтегральної функції його розв'язання послідовно обчислюють за формулою Ейлера:

$$y_{i+1} = y_i + h \cdot \mathrm{f}(x_i, y_i); \qquad i = 0,\ 1,\ 2,\ \cdots,\ n.$$

Приклад: Зміна ЕРС генератора постійного струму описується диференціальним рівнянням $e'_G = \dfrac{K_G \cdot U_d - e_G}{T_d}$. Обчислити методом Ейлера перші два значення ЕРС генератора e_G, якщо $K_G = 1$; $U_d = 100$ В; $T_d = 2$ с; $h = 0.1$ с; $e_{G0} = 0$.

Використовуючи формулу методу Ейлера, отримуємо

$$e_{G\,1} = e_{G\,0} + h\frac{(K_G U_d - e_{G\,0})}{T_d} = 0 + 0.1\frac{(1\cdot 100 - 0)}{2} = 5 \quad B;$$

$$e_{G\,2} = e_{G\,1} + h\frac{(K_G U_d - e_{G\,1})}{T_d} = 5 + 0.1\frac{(1\cdot 100 - 5)}{2} = 9.75 \quad B.$$

Метод Ейлера має ще іншу назву – метод ламаних дотичних, тому що під час обчислень інтегральну функцію інтерполюють дотичними лініями, як показано на рис. 3.2.

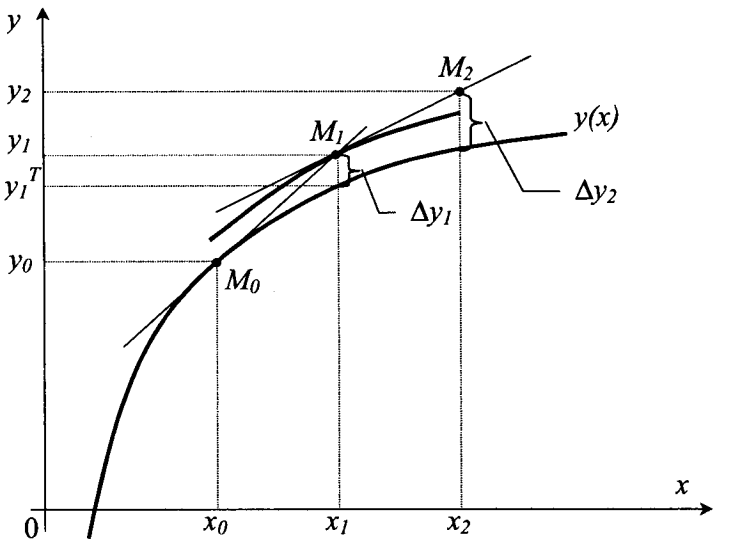


Рис. 3.2. Геометрична інтерпретація методу Ейлера

Нехай маємо аналітичну інтегральну функцію $y(x)$ розв'язання диференціального рівняння $y' = \mathrm{f}(x, y)$, графік якої показаний на рис. 3.2. Точка $M_0(x_0; y_0)$ належить цьому графікові. Рівняння дотичної до кривої в точці M_0

$$y - y_0 = k(x - x_0),$$

де $k = \mathrm{f}(x_0, y_0)$ – коефіцієнт нахилу дотичної. Це рівняння задовольняють координати деякої точки M_1, що належить дотичній, $y_1 - y_0 = \mathrm{f}(x_0; y_0) \cdot (x_1 - x_0)$. Позначивши $x_1 - x_0 = h$, загалом маємо вираз

$$y_{i+1} = y_i + h \cdot \mathrm{f}(x_i; y_i),$$

який є формулою Ейлера.

Рис. 3.2 ілюструє два важливі твердження:

- **значення інтегральної функції**, що обчислені методом Ейлера, **є наближеними** до y_i^T точних значень з похибкою, яка залежить від кроку інтегрування h: зменшення кроку підвищує точність обчислень;
- **похибка обчислень** на наступних кроках **накопичується**, чергова похибка Δy_i включає суму всіх попередніх.

Явне і неявне числове інтегрування

Нехай маємо лінійне диференціальне рівняння першого порядку $y' = \mathrm{f}(y, t)$. На рис. 3.3 зображено графіки диференціальної функції y'. Означений інтеграл у заданих межах інтегрування – це площа під диференціальною кривою в координатах $y' - x$ у цих самих межах. Наближено значення такої функції можна обчислити як суму площ прямокутників з основами $x_{i+1} - x_i = h$ і висотами, значення яких дорівнюють значенням похідної функції y' в певних точках. Для випадку, що відповідає рис. 3.3, *а*, маємо:

$$y_1 = y_0 + h \cdot y_0' = y_0 + h \cdot \mathrm{f}(x_0; y_0);\ y_2 = y_1 + h \cdot \mathrm{f}(x_1; y_1);\ \ldots;\ y_{i+1} = y_i + h \cdot \mathrm{f}(x_i; y_i).$$

Це формула числового явного інтегрування за Ейлером. Явного тому, що для обчислення на кожному кроці використовують значення похідної в попередній точці, яке є відомим, тобто **явним**.

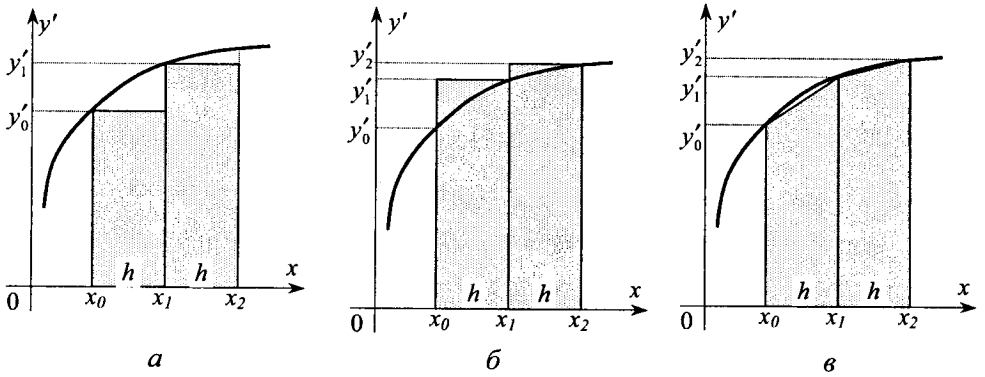


Рис. 3.3. Геометрична інтерпретація числового інтегрування
а – явне інтегрування; б – неявне інтегрування; в – метод трапецій

Неявному інтегруванню відповідає такий вигляд формули Ейлера, рис. 3.3, *б*:

$$y_{i+1} = y_i + h \cdot y'_{i+1} = y_i + h \cdot \mathrm{f}(x_{i+1};\, y_{i+1}).$$

Для обчислення значення функції y_{i+1} необхідно мати значення похідної в тій самій точці $\mathrm{f}(x_{i+1},\, y_{i+1})$, тобто y_{i+1} залежить саме від себе, а це означає, що цей метод **неявний**.

Як зрозуміло з рис. 3.3, *а–б*, інтегрування цими методами призводить до похибок протилежних знаків.

Точнішим методом числового інтегрування є неявний метод трапецій, рис. 3.3, *в*. У загальному випадку маємо таку формулу числового неявного інтегрування:

$$y_{i+1} = y_i + \frac{h}{2} \cdot \left(y'_i + y'_{i+1}\right) = y_i + \frac{h}{2} \cdot \left(\mathrm{f}(x_i;\, y_i) + \mathrm{f}(x_{i+1};\, y_{i+1})\right).$$

Розглянуті методи числового інтегрування належать до найпростіших. У наступних розділах будуть розглянуті складніші (і точніші) методи.

Однокрокові методи

Найвідомішими представниками однокрокових методів, що призначені для розв'язування диференціальних рівнянь виду $y' = \mathrm{f}(y,\ t)$, є методи Ейлера: зви-

чайний, вдосконалений і модифікований, а також методи Рунге–Кутта* (*Runge–Kutta*), формули яких для порядків від першого до четвертого наведено нижче.

Кожен числовий метод наближає розв'язок з певним порядком точності, який повинен узгоджуватись з розкладом в обмежений ряд Тейлора функції, що описує точний розв'язок диференціального рівняння. Порядок методу **p** *визначається залишковим членом цього розкладу.*

Формула Рунґе–Кутта 1-го порядку (метод Ейлера):

p = 1 $y_{i+1} = y_i + k_1,$

де

$$k_1 = h\,\mathrm{f}(y_i;\ t_i).$$

Формули Рунге–Кутта 2-го порядку:

метод Хойна (*Heun*), або вдосконалений метод Ейлера

p = 2 $y_{i+1} = y_i + (k_1 + k_2)/2,$

де

$$k_1 = h\,\mathrm{f}(y_i;\ t_i); \qquad k_2 = h\,\mathrm{f}(y_i + k_1;\ t_i + h).$$

модифікований метод Ейлера, або формула Ейлера–Коші

p = 2 $y_{i+1} = y_i + k_2,$

де

$$k_1 = h\,\mathrm{f}(y_i;\ t_i); \qquad k_2 = h\,\mathrm{f}(y_i + k_1/2;\ t_i + h/2).$$

Формула Рунге–Кутта 3-го порядку:

p = 3 $y_{i+1} = y_i + (k_1 + 4k_2 + k_3)/6,$

де

$$k_1 = h\,\mathrm{f}(y_i;\ t_i); \qquad k_2 = h\,\mathrm{f}(y_i + k_1/2;\ t_i + h/2).$$
$$k_3 = h\,\mathrm{f}(y_i - k_1 + 2k_2;\ t_i + h).$$

---

* *До речі, методи Ейлера належать до родини формул Рунге–Кутта.*

Формула Рунге–Кутта 4-го порядку:

p = 4 $y_{i+1} = y_i + (k_1 + 2k_2 + 2k_3 + k_4)/6,$

де

$$k_1 = hf(y_i;\ t_i); \qquad k_2 = hf(y_i + k_1/2;\ t_i + h/2);$$

$$k_3 = hf(y_i + k_2/2;\ t_i + h/2); \qquad k_4 = hf(y_i + k_3;\ t_i + h).$$

До речі, метод 4-го порядку є найуживанішим і саме він відомий багатьом користувачам комп'ютерів як "метод Рунге–Кутта"

Нижче подано тексти підпрограм для середовищ Microsoft Quick Basic і Turbo Pascal, що реалізують розв'язування системи диференціальних рівнянь за допомогою формули Рунге–Кутта 4-го порядку з фіксованим кроком і можуть використовуватись як стандартні.

Quick Basic

```
SUB RK4 (N%, t, h, y())
' ------------------------------------------------------
' Підпрограма призначена для розв'язування системи диференціальних
'  рівнянь, записаних у нормальній формі Коші, методом
'  Рунге-Кутта 4-го порядку з фіксованим кроком.
'
'  Вхідні дані:
'     N   - порядок системи
'     t   - незалежна змінна (час)
'     h   - крок
'     Y   - масив змінних
'
'  Вихідні дані:
'     t - незалежна змінна (час), після виконання кроку t = t + h
'     Y   - масив змінних
'
```

```
' Використовується підпрограма Dif, що обчислює значення
' правих частин системи диференціальних рівнянь.
' ------------------------------------------------------
DIM k1(N%), k2(N%), k3(N%), k4(N%), f(N%), y1(N%)
' ================================================
' Розрахунок 4-х наближень Y
' ================================================
   CALL dif(t, y(), f())
   FOR i = 1 TO N%
      k1(i) = h * f(i)
      y1(i) = y(i) + k1(i) / 2!
   NEXT i

   CALL dif(t + h / 2!, y1(), f())
   FOR i = 1 TO N%
      k2(i) = h * f(i)
      y1(i) = y(i) + k2(i) / 2!
   NEXT i

   CALL dif(t + h / 2!, y1(), f())
   FOR i = 1 TO N%
      k3(i) = h * f(i)
      y1(i) = y(i) + k3(i)
   NEXT i

   CALL dif(t + h, y(), f())
   FOR i = 1 TO N%
     k4(i) = h * f(i)
     y1(i) = y(i) + (k1(i) + 2! * (k2(i) + k3(i)) + k4(i)) / 6!
   NEXT i
' ================================================
' Закінчення кроку
' ================================================
   FOR i = 1 TO N%
      y(i) = y1(i)
   NEXT i
   t = t + h
END SUB
```

Turbo Pascal

```
{ Для роботи процедури RK4 використовується процедура  dif  }
{ ------------------------------------------------------- }
{ procedure  dif(x: double; var y, f : vector) ;          }
{ ------------------------------------------------------- }
{ Процедура призначена для обчислення значень правих частин }
{ системи диференціальних рівнянь, записаних у нормальній  }
{ формі Коші.                                              }
{ Форма  запису:                                           }
{      f[1] := функція_1(y[1],y[2],...,y[N],x)             }
{      f[2] := функція_2(y[1],y[2],...,y[N],x)             }
{                    .  .  .                               }
{      f[N] := функція_N(y[1],y[2],...,y[N],x)             }
{                                                          }
{ Вхідні дані:                                             }
{      x - незалежна змінна                                }
{      y - масив інтегрованих змінних                      }
{                                                          }
{ Вихідні дані:                                            }
{      y - масив інтегрованих змінних                      }
{      f - масив обчислених похідних                       }
{ ------------------------------------------------------- }

procedure  RK4( var x    : double ;
                var h    : double ;
                    N    : integer ;
                var y    : vector ) ;
{ ======================================================= }
{ Процедура розв'язування системи диференціальних рівнянь  }
{           методом Рунге-Кутта  4-го порядку              }
{ ------------------------------------------------------- }
{ Вхідні дані :                                            }
{     x   - незалежна змінна інтегрування                  }
{     h   - крок інтегрування                              }
{     N   - кількість інтегрованих змінних (порядок системи) }
{     y   - масив змінних, array [1..N] of DOUBLE = vector }
{ Вихідні дані :                                           }
{     x   := x + h                                         }
{     h   = var                                            }
```

```
{     Y   - масив вихідних значень                                  }
{ ------------------------------------------------------------- }

var
   y1, f, k1, k2, k3, k4   : vector;
   i        : integer;
   t        : REAL;
begin
  { ---------------------------- }
  { Обчислення 4-х наближень  Y  }
  { ---------------------------- }
  dif(x, y, f);
  for   i := 1  to   N   do
     begin
        k1[i] := h * f[i] ;
        y1[i] := y[i] + k1[i] ;
     end ;

  t := x + 0.5 * h ;
  dif(t, y1, f);
  for   i := 1  to   N   do
     begin
        k2[i] := h * f[i]/2.0 ;
        y1[i] := y[i] + k2[i] ;
     end;

  dif(t, y1, f) ;
  for   i := 1  to   N   do
     begin
        k3[i] := h * f[i]/2.0 ;
        y1[i] := y[i] + k3[i] ;
     end;
  t := x + h ;
  dif(t, y1, f);
  for   i := 1  to   N   do
     begin
      k4[i] := h * f[i] ;
      y[i] := y[i] + (k1[i] + 2.0*(k2[i] + k3[i]) + k4[i])/6.0;
     end;
{ ------------------------- }
{  Закінчення кроку         }
{ ------------------------- }
  x := t ;
end ;
```

Популярність формули Рунге–Кутта 4-го порядку, незважаючи на сторічну історію [В.11], дуже велика, бо вона придатна для розв'язування широкого кола задач, має доволі високу точність і просто програмується. Її недолік – відсутність оцінки похибки на кожному кроці, тому необхідно розв'язувати задачу з фіксованим кроком інтегрування, який треба вибирати для найгірших умов (див. далі "***Вибір кроку***"). Щоб усунути цей недолік, розроблено ряд інших формул інтегрування звичайних диференціальних рівнянь на підставі формул Кутта, серед яких найвідомішими є формули Мерсона (*Merson*) і Фельберґа (*Fehlberg*). Ці формули, крім значення змінної в наступній точці y_{i+1}, дають ще й оцінку локальної похибки ε_{i+1}, яку використовують для вибору поточного значення кроку. Далі подано деякі з цих формул [В.11]:

порядок формули методу — *порядок формули оцінки похибки*

Формула Мерсона порядку 4(5):

$$y_{i+1} = y_i + (k_1 + 4k_4 + k_5)/2; \qquad \varepsilon_{i+1} = (k_1 - (9k_3 + k_5)/2 + 4k_4)/5,$$

де

$$k_1 = h\,\mathrm{f}(y_i,\ t_i)/3; \qquad k_2 = h\,\mathrm{f}(y_i + k_1,\ t_i + h/3);$$

$$k_3 = h\,\mathrm{f}(y_i + (k_1 + k_2)/2,\ t_i + h/3)/3; \quad k_4 = h\,\mathrm{f}(y_i + (3k_1 + 9k_3)/8,\ t_i + h/2)/3;$$

$$k_5 = h\,\mathrm{f}(y_i + (3k_1 - 9k_3 + 12k_4)/2,\ t_i + h)/3.$$

*Нижче наведено текст підпрограми, що реалізує розв'язування системи диференціальних рівнянь за допомогою формули Мерсона 4-го порядку з автоматичним вибором кроку розв'язання і яка може використовуватись як стандартна. Алгоритм автоматичного вибору кроку описано далі в п. "**Вибір кроку**".*

Quick Basic

```
SUB Merson (N%, t, h, y(), Eps) STATIC
' ---------------------------------------------------------
'  Підпрограма призначена для розв'язування системи диференціальних
'  рівнянь, записаних у нормальній формі Коші, методом
'  Мерсона з автоматичним вибором кроку інтегрування
'
'  Вхідні дані:
'     N   - порядок системи
'     t   - незалежна змінна (час)
'     h   - крок (змінюється автоматично)
'     Y   - масив інтегрованих змінних
'     Eps - точність на кроці
'
'  Вихідні дані:
'     t   - незалежна змінна (час)
'     h   - крок (змінюється автоматично)
'     Y   - масив інтегрованих змінних
'
'  Використовується підпрограма  Dif , що обчислює праві
'  частини системи диференціальних рівнянь:
'        Dif( t , Y() , F())   , де
'             t - час
'             Y - ім'я масиву інтегрованих змінних
'             F - ім'я масиву похідних
' ---------------------------------------------------------
DIM k1(N), k2(N), k3(N), k4(N), k5(N), f(N), y1(N), ye(N)
DO
   '
   '  Розрахунок п'яти наближень  Y
   '
   CALL dif(t, y(), f())
   FOR i = 1 TO N%
      k1(i) = h * f(i) / 3!
      y1(i) = y(i) + k1(i)
   NEXT i

   CALL dif(t + h / 3!, y1(), f())
   FOR i = 1 TO N%
```

```
    k2(i) = (h * f(i) + k1(i)) / 2!
    y1(i) = y(i) + (k1(i) + k2(i)) / 2!
NEXT i

CALL dif(t + h / 3!, y1(), f())
FOR i = 1 TO N%
    k3(i) = h * f(i) / 3!
    k2(i) = (3! * k1(i) + 9! * k3(i)) / 8!
    y1(i) = y(i) + (3! * k1(i) + 9! * k3(i)) / 8!
NEXT i

CALL dif(t + h / 2!, y1(), f())
FOR i = 1 TO N%
    k4(i) = h * f(i) / 3!
    y1(i) = y(i) + 1.5 * (k1(i) - 3! * k3(i)) + 6! * k4(i)
NEXT i

CALL dif(t + h, y1(), f())
FOR i = 1 TO N%
    k5(i) = h * f(i) / 3!
    y1(i) = y(i) + (k1(i) + 4! * k4(i) + k5(i)) / 2!
    ye(i) = (k1(i) - (9! * k3(i) + k5(i)) / 2! + 4! * k4(i)) / 5!
NEXT i
'
'  Розрахунок похибки на кроці
'
delta =  1E-12                    ' Уникаємо ділення на нуль
FOR i = 1 TO N%
    Yerr = ye(i) / (ABS(y1(i)) + 1!)
    delta = delta + Yerr * Yerr
NEXT i
delta = SQR(delta) / N%
'
' Вибір кроку розв'язання
'
Hold = h
k = (Eps / delta) ^ (1! / 5!)
IF GoodStep% = 0 THEN        ' Змінна GoodStep% призначена
    Kmax = 1!                ' для обмеження збільшення кроку
    GoodStep% = 1            ' за невдалого попереднього
ELSE
    Kmax = 2.5
END IF
```

```
   IF k > Kmax THEN k = Kmax          ' Обмеження зміни кроку
   IF k < .2 THEN k = .2              ' розв'язання
   IF delta > Eps THEN GoodStep% = 0
   h = h * k * .9

LOOP UNTIL delta <= Eps
'
' Закінчення кроку інтегрування
'
   FOR i = 1 TO N%
      y(i) = y1(i)
   NEXT i
   t = t + Hold

END SUB
```

Turbo Pascal

```
{ Для роботи процедури Merson використовується процедура dif }
{ ------------------------------------------------------- }
{ procedure  dif(x: double; var y, f : vector) ;          }
{ ------------------------------------------------------- }
{  Процедура призначена для обчислення правих частин      }
{  системи диференціальних рівнянь, записаних у нормальній }
{  формі Коші.                                            }
{  Форма  запису:                                         }
{       f[1] := функція_1(y[1],y[2],...,y[N],x)           }
{       f[2] := функція_2(y[1],y[2],...,y[N],x)           }
{                    .  .  .                              }
{       f[N] := функція_N(y[1],y[2],...,y[N],x)           }
{                                                         }
{  Вхідні дані:                                           }
{       x - незалежна змінна                              }
{       y - масив інтегрованих змінних                    }
{                                                         }
{  Вихідні дані:                                          }
{       y - масив інтегрованих змінних                    }
{       f - масив обчислених похідних (вектор інтегрування) }
{ ------------------------------------------------------- }
```

```
procedure  Merson( var x    : DOUBLE ;
                   var h    : DOUBLE ;
                       N    : integer ;
                       Eps  : DOUBLE ;
                   var y    : matrix ) ;
{ ========================================================= }
{ Процедура для розв'язування системи диференціальних       }
{ рівнянь методом Мерсона з автоматичним вибором кроку      }
{ --------------------------------------------------------- }
{ Вхідні дані :                                             }
{     x   - незалежна змінна інтегрування                   }
{     h   - крок інтегрування                               }
{     N   - кількість змінних (порядок системи)             }
{     Eps - точність на кроці                               }
{     y   - масив змінних, array [1..N] of DOUBLE = vector  }
{ Вихідні дані :                                            }
{     x   := x + h                                          }
{     h   = VAR                                             }
{     y   - масив вихідних значень (вектор змінних)         }
{ --------------------------------------------------------- }

var
   y1, f, k1, k2, k3, k4, k5 : vector;
   i                         : integer;
   error, delta, t           : REAL;
   k_step                    : REAL;
begin
   repeat
   { ------------------------------ }
   {  Обчислення 5-ти наближень  Y   }
   { ------------------------------ }
      dif(x,y,f);
      for  i:= 1  to  N  do
         begin
            k1[i]:= h*f[i]/3.0;
            y1[i]:= y[i]+k1[i];
         end ;
      t := x+h/3.0;
      dif(t,y1,f);
      for  i:= 1  to  N  do
         begin
            k2[i]:= h*f[i]/3.0;
            y1[i]:= y[i]+(k1[i]+k2[i])/2.0;
```

```
        end;
      dif(t,y1,f);
      for  i:= 1  to  N  do
        begin
          k3[i]:= h*f[i]/3.0;
          y1[i]:= y[i]+(3.0*k1[i]+9.0*k3[i])/8.0;
        end;
      t := x+h/2.0;
      dif(t,y1,f);
      for  i:= 1  to  N  do
        begin
          k4[i]:= h*f[i]/3.0;
          y1[i]:= y[i]+1.5*(k1[i]-3.0*k3[i])+6.0*k4[i];
        end;
      t := x+h;
      dif(t,y1,f);
      for  i:= 1  to  N  do
        begin
          k5[i]:= h*f[i]/3.0;
          y1[i]:= y[i]+0.5*(k1[i]+4.0*k4[i]+k5[i]);
        end;
{ ---------------------- }
{   Обчислення похибки   }
{ ---------------------- }
      error := 1.0e-16 ;
      for  i:= 1 to  N  do
        begin
          delta:= abs(0.2*(k1[i]-
(9.0*k3[i]+k5[i])/2.0+4.0*k4[i]));
          error:= error+sqr(delta/(abs(y1[i])+1.0));
        end;
      error := sqrt(error)/N ;
{ ---------------------- }
{   Зміна кроку          }
{ ---------------------- }
      if  Eps < error then  k_step := sqrt(sqrt(Eps/error))
                    else  k_step := sqrt(sqrt(sqrt(Eps/error)));
      if  k_step > 2.0  then  k_step:= 2.0 ;
      if  k_step < 0.2  then  k_step:= 0.2 ;
      h := h * k_step * 0.9 ;
```

```
{ ---------------------- }
   until  error <= Eps ;
{ ---------------------------- }
{ Закінчення кроку інтегрування }
{ ---------------------------- }
   x := t;
   y := y1;
end ;
```

Формула Фельберґа порядку 2(3):

$$y_{i+1} = y_i + (k_1 + k_2)/2; \qquad \varepsilon_{i+1} = (2k_3 - (k_1 + k_2))/3,$$

де

$$k_1 = h\,\mathrm{f}(y_i;\ \ t_i); \qquad k_2 = h\,\mathrm{f}(y_i + k_1;\ \ t_i + h);$$

$$k_3 = h\,\mathrm{f}(y_i + (k_1 + k_2)/4;\ \ t_i + h/2).$$

Формула Фельберґа порядку 2(3)B:

$$y_{i+1} = y_i + (214k_1 + 27k_2 + 650k_3)/891;$$

$$\varepsilon_{i+1} = \frac{23}{1782}k_1 - \frac{k_2}{33} + \frac{350}{11583}k_3 - \frac{k_4}{78},$$

де

$$k_1 = h\,\mathrm{f}(y_i;\ \ t_i); \qquad k_2 = h\,\mathrm{f}(y_i + k_1/4;\ \ t_i + h/4);$$

$$k_3 = h\,\mathrm{f}(y_i + (729k_2 - 189k_1)/800;\ \ t_i + 27h/40);$$

$$k_4 = h\,\mathrm{f}(y_i + (214k_1 + 27k_2 + 650k_3)/891;\ \ t_i + h) = h\,\mathrm{f}(y_{i+1};\ \ t_i + h).$$

В останній формулі коефіцієнт k_4 *може бути використаний на наступному кроці інтегрування вже як* k_1*, що зменшує кількість звертань до функції обчислення правих частин системи диференціальних рівнянь і збільшує швидкодію числового інтегрування.*

*Нижче подано текст підпрограми, що реалізує розв'язування системи диференціальних рівнянь за допомогою формули Фельберґа порядку 2(3)B зі стратегією автоматичного вибору кроку розв'язання. Алгоритм автоматичного вибору кроку описано далі в п. "**Вибір кроку**".*

Автори рекомендують цю процедуру як одну з найефективніших для моделювання електроприводів. Щоб вона працювала, рекомендують задавати значення локальної точності **Eps** *в межах* $10^{-3}...10^{-5}$ *(здебільшого* 10^{-4} *є найкращим варіантом).*

Quick Basik

```
SUB Fehlberg2B (N, t, h, Eps, Y()) STATIC
' ------------------------------------------------------------
' Підпрограма призначена для розв'язування системи диференціальних
' рівнянь, записаних у нормальній формі Коші, методом
' Фельберґа 2(3)B  з автоматичним вибором кроку інтегрування
'
' Вхідні дані:
'      N      - порядок системи
'      t      - незалежна змінна (час)
'      h      - крок (змінюється автоматично)
'      Y      - масив інтегрованих змінних
'      Eps    - точність на кроці
'
' Вихідні дані:
'       t     - незалежна змінна (час)
'       h     - крок (змінюється автоматично)
'       Y     - масив інтегрованих змінних
'
' Використовується підпрограма  Dif , що обчислює значення
' правих частин системи диференціальних рівнянь:
'       Dif( t , Y() , F())    , де
'            t - час
'            Y - ім'я масиву інтегрованих змінних
'            F - ім'я масиву похідних (вектор інтегрування)
```

```
' ---------------------------------------------------------
'
'   Якщо перше звертання, то ініціалізація
'
IF first% = 0 THEN
  DIM k1(N), k2(N), k3(N), k4(N), f(N), y1(N)
  CALL dif(t, Y(), f())
  first% = 1
END IF
' ---------------------------------------------
'   Основна частина програми
' ---------------------------------------------
DO
    '
    '  Обчислення трьох наближень  Y
    '
   FOR i = 1 TO N
     k1(i) = h * f(i)
     y1(i) = Y(i) + .25 * k1(i)
   NEXT i

   CALL dif(t + .25 * h, y1(), f())
   FOR i = 1 TO N%
     k2(i) = h * f(i)
     y1(i) = Y(i) + (729! * k2(i) - 189! * k1(i)) / 800!
   NEXT i

   CALL dif(t + 27! * h / 40!, y1(), f())
   FOR i = 1 TO N%
    k3(i) = h * f(i)
    y1(i) = Y(i)+(214!*k1(i) + 27! * k2(i) + 650!*k3(i))/891!
   NEXT i
    '
    '  Обчислення похибки на кроці
    '
   delta = 1E-12                ' Уникаємо ділення на нуль
   CALL dif(t + h, y1(), f())
   FOR i = 1 TO N%
    k4(i) = h * f(i)
    Yerr = Y(i) + 533! * k1(i) / 2106! + 800! * k3(i) / 1053!
    Yerr = Yerr - k4(i) / 78!
    Yerr = (Yerr - y1(i)) / (ABS(y1(i)) + 1!)
    delta = delta + Yerr * Yerr
```

```
    NEXT i
    delta = SQR(delta / N%)
    '
    ' Вибір кроку розв'язання
    '
    Hold = h
    k = SQR(SQR(Eps / delta))
    IF GoodStep% = 0 THEN      ' Змінна  GoodStep%  призначена
      Kmax = 1!                ' для обмеження збільшення кроку,
      GoodStep% = 1            ' якщо попередній був невдалим
    ELSE                       '
      Kmax = 2.5
    END IF
    IF k > Kmax THEN k = Kmax    ' Обмеження зміни кроку
    IF k < .2 THEN k = .2        ' розв'язання
    h = h * k * .8
    IF delta > Eps THEN GoodStep% = 0
LOOP UNTIL delta <= Eps
'
' Закінчення кроку інтегрування
'
FOR i = 1 TO N%
  Y(i) = y1(i)
NEXT i
t = t + Hold

END SUB
```

Turbo Pascal

```
{$E+,N+,F+ }
unit  F23B;
{ ================================================ }
{     Реалізація формули Фельберґа порядку 2(3)B    }
{     з автоматичним вибором кроку розв'язування    }
{ ================================================ }
interface

type
   Vector = array [1..100]  of  DOUBLE ;
```

```
   Sys = procedure(t: DOUBLE; var y, f : Vector) ;
   { ------------------------------------------------------ }
   {  Процедура призначена для обчислення правих частин    }
   {  системи диференціальних рівнянь, записаних у         }
   {  нормальній формі Коші.                               }
   {  Форма  запису:                                       }
   {       f[1] := функція_1(y[1],y[2],...,y[N],t)         }
   {       f[2] := функція_2(y[1],y[2],...,y[N],t)         }
   {                  . . .                                }
   {       f[N] := функція_N(y[1],y[2],...,y[N],t)         }
   {  Вхідні дані:                                         }
   {       t - незалежна змінна                            }
   {       y - масив інтегрованих змінних                  }
   {  Вихідні дані:                                        }
   {       y - масив інтегрованих змінних                  }
   {       f - масив похідних (вектор інтегрування)        }
   { ------------------------------------------------------ }
procedure
   Fehlberg23B(Dif : Sys ;    { Ім'я процедури обчислення }
                              {правих частин системи }
            N   : integer ;    { Порядок системи          }
         var t   : DOUBLE ;    { Незалежна змінна (час) }
         var h   : DOUBLE ;    { Крок (змінюється автоматично) }
            Eps : DOUBLE ;     { Точність на кроці        }
         var y   : Vector ) ;  { Вектор змінних           }

implementation

var
   k1, k2, k3, k4   : Vector ;

procedure
   Fehlberg23B(Dif : Sys ;    { Ім'я процедури правих частин
системи }
            N   : integer ;    { Порядок системи          }
         var t   : DOUBLE ;    { Незалежна змінна (час) }
         var h   : DOUBLE ;    { Крок (змінюється автоматично) }
            Eps : DOUBLE ;     { Точність на кроці        }
         var y   : Vector ) ;  { Вектор змінних           }
{ ======================================================== }
{ Процедура розв'язує систему диференціальних              }
{ рівнянь, що записані у нормальній формі Коші, методом    }
{ Фельберґа 2(3)B  з автоматичним вибором кроку розв'язання }
```

```
{                                                                    }
{ Використовується процедура  Dif , що обчислює значення           }
{ правих частин системи диференціальних рівнянь:                    }
{       Dif(t, Y, F)    , де                                        }
{           t - час                                                 }
{           Y - ім'я вектора змінних                                }
{           F - ім'я вектора похідних                               }
{ ================================================================ }

var
   i    : INTEGER;
   y1, f  : Vector ;
   Hold, Yerr, k, Kmax, Delta : DOUBLE ;
const
   First     : boolean = true ;
   GoodStep  : boolean = true ;
begin
  {                                                  }
  {   Якщо перше звертання, то ініціалізація        }
  {                                                  }
  if First then
    begin
      dif(t, y, f) ;
      First := false ;
    end
  else
    for i := 1 to N do  f[i] := k1[i] / h ;
  { ---------------------------------- }
  {   Основна частина програми         }
  { ---------------------------------- }
  repeat
    {                                  }
    {  Розрахунок трьох наближень  Y   }
    {                                  }
    for i := 1 to N do
    begin
      k1[i] := h * f[i] ;
      y1[i] := y[i] + 0.25 * k1[i] ;
    end ;

    dif(t + 0.25 * h, y1, f) ;
    for i := 1 to N do
    begin
```

```
      k2[i] := h * f[i] ;
      y1[i] := y[i] + (729.0 * k2[i] - 189.0 * k1[i]) / 800.0 ;
    end ;

    dif(t + 27.0 * h / 40.0, y1, f) ;
    for i := 1 to N do
    begin
      k3[i] := h * f[i] ;
      y1[i] := y[i] + (214.0*k1[i]+27.0*k2[i]+650.0*k3[i])/891.0;
    end ;
    {                                    }
    {  Розрахунок похибки на кроці       }
    {                                    }
    Delta := 1e-12 ;      { Уникаємо ділення на нуль }
    dif(t + h, y1, f) ;
    for i := 1 to N do
    begin
      k4[i] := h * f[i] ;
      Yerr:=(299.0*k1[i]-702.0*k2[i]+700.0*k3[i]-
             297.0*k4[i])/23166.0;
      Yerr := Yerr / (abs(y1[i]) + 1.0) ;
      Delta := Delta + sqr(Yerr) ;
    end ;
    Delta := sqrt(Delta / N) ;
    {                                    }
    {     Вибір кроку розв'язання        }
    {                                    }
    Hold := h ;
    k := sqrt(sqrt(Eps / Delta)) ;
    if GoodStep = false then
    begin                    { Змінна  GoodStep  призначена   }
      Kmax := 1.0 ;          { для обмеження збільшення кроку }
      GoodStep := true ;     { за невдалого попереднього      }
    end
    else
      Kmax := 2.5 ;
    if k > Kmax then k := Kmax ;    { Обмеження зміни }
    if k < 0.2 then k := 0.2 ;      { кроку розв'язання }
    h := h * k * 0.8 ;
    if Delta > Eps then GoodStep := false ;
  until Delta <= Eps ;
{                                  }
{ Закінчення кроку інтегрування }
```

```
 {                                    }
 for i := 1 to N do
 begin
   y[i] := y1[i] ;
   k1[i] := k4[i] ;
 end ;
 t := t + Hold ;
end ;
{ ---------------------------------- }

begin
end .
```

Формула Фельберга порядку 4(5):

$$y_{i+1} = y_i + \frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4104}k_4 - \frac{k_5}{5};$$

$$\varepsilon_{i+1} = \frac{-2090k_1 + 22528k_3 + 21970k_4 - 15048k_5 - 27360k_6}{752400},$$

де

$$k_1 = h\,\mathrm{f}\left(y_i;\ t_i\right); \qquad k_2 = h\,\mathrm{f}\left(y_i + k_1/4;\ t_i + h/4\right);$$

$$k_3 = h\,\mathrm{f}\left(y_i + \left(3k_1 + 9k_2\right)/32;\ t_i + 3h/8\right);$$

$$k_4 = h\,\mathrm{f}\left(y_i + \frac{1932k_1 - 7200k_2 + 7296k_3}{2197};\ t_i + 12h/13\right);$$

$$k_5 = h\,\mathrm{f}\left(y_i + \frac{439}{216}k_1 - 8k_2 + \frac{3680}{513}k_3 - \frac{845}{4104}k_4;\ t_i + h\right);$$

$$k_6 = h\,\mathrm{f}\left(y_i - \frac{8}{27}k_1 + 2k_2 - \frac{3544}{2565}k_3 + \frac{1859}{4104}k_4 - \frac{11}{40}k_5;\ t_i + h/2\right).$$

Останнім часом стала популярною формула Дормана–Прінса (*Dormand–Prince*) порядку 5(4) [В.11], що відзначається високою точністю та ефективністю, але ще складніша за попередню.

Формула Дормана–Прінса порядку 5(4):

$$y_{i+1} = y_i + \frac{35}{384}k_1 + \frac{500}{1113}k_3 + \frac{125}{192}k_4 - \frac{2187}{6784}k_5 + \frac{11}{84}k_6;$$

$$\varepsilon_{i+1} = -\frac{71}{57600}k_1 + \frac{71}{16695}k_3 - \frac{71}{1920}k_4 + \frac{17253}{339200}k_5 - \frac{22}{525}k_6 + \frac{k_7}{40},$$

де

$$k_1 = hf(y_i;\ t_i); \qquad k_2 = hf(y_i + k_1/5;\ t_i + h/5);$$

$$k_3 = hf(y_i + (3k_1 + 9k_2)/40;\ t_i + 3h/10);$$

$$k_4 = hf\left(y_i + \frac{44}{45}k_1 - \frac{56}{15}k_2 + \frac{32}{9}k_3;\ t_i + \frac{4}{5}h\right);$$

$$k_5 = hf\left(y_i + \frac{19372}{6561}k_1 - \frac{25360}{2187}k_2 + \frac{64448}{6561}k_3 - \frac{212}{729}k_4;\ t_i + \frac{8}{9}h\right);$$

$$k_6 = hf\left(y_i + \frac{9017}{3168}k_1 - \frac{355}{33}k_2 + \frac{46732}{5247}k_3 + \frac{49}{176}k_4 - \frac{5103}{18656}k_5;\ t_i + h\right);$$

$$k_7 = hf\left(y_i + \frac{35}{384}k_1 + \frac{500}{1113}k_3 + \frac{125}{192}k_4 - \frac{2187}{6784}k_5 + \frac{11}{84}k_6;\ t_i + h\right).$$

У цій формулі коефіцієнт k_7 може бути використаний на наступному кроці інтегрування вже як k_1, що дає змогу зменшити кількість звертань до функції обчислення правих частин системи диференціальних рівнянь і пришвидшити процес інтегрування

Численні дослідження на багатьох моделях різної складності показали, що для задач розрахунку динаміки електроприводів найефективнішими є формули невисокого порядку, особливо формула Фельберґа 2(3)B. Використання формул високого порядку, наприклад, Фельберґа 4(5) і Дормана–Прінса 5(4), не дає переваги в швидкості обчислень, оскільки такі формули ефективні для високих значень локальної точності (10^{-6} і вище), а більшість параметрів моделей електроприводів відомі з точністю у межах 5–10 % (інколи до 20 %), тому нема потреби робити розрахунки динаміки з точністю, вищою за 10^{-3}–10^{-4}, а саме з такою точністю найкраще працюють методи невисоких порядків.

Багатокрокові методи

Недоліком однокрокових методів є те, що інформація з більш ніж одного попереднього кроку не застосовується. З одного боку, це спрощує використання таких методів, а програми на їх підставі є простішими і зрозумілішими. З іншого боку, брак інформації про поведінку інтегрованої функції (*яку можна було отримати з попередніх кроків інтегрування*) змушує на кожному кроці додатково звертатися до процедури, що обчислює праві частини системи диференціальних рівнянь, а це збільшує час розрахунку.

Багатокрокові методи, на відміну від однокрокових формул типу Рунге–Кутта, використовують інформацію з попередніх кроків, що робить їх економнішими в обчисленнях. Найвідомішими серед них є формули Адамса [В.2, В.13], які, незважаючи на півторавікову історію, досі належать до найефективніших числових методів:

- явні або Адамса–Бешфорта (*Adams-Bashforth*), для знаходження значення змінної потрібно мати її попереднє значення та значення похідних у попередніх точках;
- неявні або Адамса–Малтона (*Adams–Moulton*), потребують значення похідної в шуканій точці.

Нижче подано формули 1–5-го порядків (*формули вищого порядку є менш ефективними для моделювання електроприводів*) з виразами для оцінки похибок.

Явні формули Адамса

$$\mathrm{p}=1 \quad y_{i+1}=y_i+hy_i'; \qquad \varepsilon_{i+1}=\frac{1}{2}h^2y_{i+1}''$$

$$\mathrm{p}=2 \quad y_{i+1}=y_i+\frac{h}{2}\left(3y_i'-y_{i-1}'\right); \qquad \varepsilon_{i+1}=\frac{5}{12}h^3y_{i+1}'''$$

$$\mathrm{p}=3 \quad y_{i+1}=y_i+\frac{h}{12}\left(23y_i'-16y_{i-1}'+5y_{i-2}'\right); \qquad \varepsilon_{i+1}=\frac{3}{8}h^4y_{i+1}^{IV}$$

$$\mathrm{p}=4 \quad y_{i+1}=y_i+\frac{h}{24}\left(55y_i'-59y_{i-1}'+37y_{i-2}'-9y_{i-3}'\right); \qquad \varepsilon_{i+1}=\frac{251}{720}h^5y_{i+1}^{V}$$

$$\text{p}=5\quad y_{i+1}=y_i+\frac{h}{720}\begin{pmatrix}1901y'_i-2774y'_{i-1}+2616y'_{i-2}-\\-1274y'_{i-3}+251y'_{i-4}\end{pmatrix};\qquad \varepsilon_{i+1}=\frac{95}{288}h^6y^{VI}_{i+1}$$

Неявні формули Адамса

$$\text{p}=1\quad y_{i+1}=y_i+hy'_{i+1};\qquad \varepsilon_{i+1}=-\frac{1}{2}h^2y''_{i+1}$$

$$\text{p}=2\quad y_{i+1}=y_i+\frac{h}{2}\left(y'_{i+1}+y'_i\right);\qquad \varepsilon_{i+1}=-\frac{h^3}{12}y'''_{i+1}$$

$$\text{p}=3\quad y_{i+1}=y_i+\frac{h}{12}\left(5y'_{i+1}+8y'_i-y'_{i-1}\right);\qquad \varepsilon_{i+1}=-\frac{h^4}{24}y^{IV}_{i+1}$$

$$\text{p}=4\quad y_{i+1}=y_i+\frac{h}{24}\left(9y'_{i+1}+19y'_i-5y'_{i-1}+y'_{i-2}\right);\qquad \varepsilon_{i+1}=\frac{-19}{720}h^5y^{V}_{i+1}$$

$$\text{p}=5\quad y_{i+1}=y_i+\frac{h}{720}\begin{pmatrix}251y'_{i+1}+646y'_i-264y'_{i-1}+\\+106y'_{i-2}-19y'_{i-3}\end{pmatrix};\qquad \varepsilon_{i+1}=\frac{-3}{160}h^6y^{VI}_{i+1}$$

У формулах Адамса приймають, що всі точки рівновіддалені, тобто вважають, що інтегрування виконують зі сталим кроком.

Формули Адамса належать до методів, у яких похибка обчислень не накопичується [В.2], і тому можуть використовуватись на дуже довгих проміжках інтегрування, зокрема, вони ефективні для розв'язування осцилювальних (коливних) задач, наприклад, моделювання електроприводів з пружними зв'язками в механічній частині привода.

Переважна більшість створених на підставі методів Адамса програм поєднують явні та неявні формули для реалізації методу "прогноз–корекція":

1) прогноз (*prediction* – позначається літерою P) наступного i+1-го значення змінної $y_{p_{i+1}}$ за допомогою явної формули;
2) за отриманим прогнозованим значенням обчислюють (*evaluation* – позначається літерою E) похідну в i+1-й точці;

3) коригування (*correction* – позначається літерою С) значення змінної $y_{c_{i+1}}$ за допомогою неявної формули;
4) обчислення уточненого значення похідної в i+1-й точці (*evaluation* – E).

Такий спосіб побудови методу позначається як PECE і є оптимальним варіантом для обчислювальної процедури з використанням родини формул Адамса [В.2, В.10, В.13].

Приклад: Використаємо формули Адамса третього порядку для розв'язування диференціального рівняння $y' = \mathrm{f}(y,t)$.

1) Знаходимо для моменту часу t_{i+1} прогнозоване значення змінної:

$$y_{p_{i+1}} = y_i + \frac{h}{12}\left(23y'_i - 16y'_{i-1} + 5y'_{i-2}\right).$$

2) Встановимо прогнозоване значення похідної в $i+1$ точці:

$$y'_{p_{i+1}} = \mathrm{f}\left(y_{p_{i+1}},\ t_{i+1}\right).$$

3) Коригуємо значення змінної:

$$y_{c_{i+1}} = y_i + \frac{h}{12}\left(5y'_{p_{i+1}} + 8y'_i - y'_{i-1}\right).$$

4) Знаходимо похибку на кроці:

$$y_{c_{i+1}} - y_{p_{i+1}} = \underbrace{\left(y_{i+1} - \varepsilon_{ci+1}\right)}_{y_{c_{i+1}}} - \underbrace{\left(y_{i+1} - \varepsilon_{p_{i+1}}\right)}_{y_{p_{i+1}}} = \varepsilon_{p_{i+1}} - \varepsilon_{c_{i+1}},$$

де y_{i+1} – точний розв'язок на i+1-му кроці;

$\varepsilon_{p_{i+1}}$ – локальна похибка прогнозу;

$\varepsilon_{c_{i+1}}$ – локальна похибка коригування.

Співвідношення між похибками прогнозу і коригування відоме:

$$\varepsilon_{p_{i+1}} = \frac{3}{8}h^4 y^{IV};\ \varepsilon_{c_{i+1}} = -\frac{1}{24}h^4 y^{IV},\ \text{тоді}\ y_{c_{i+1}} - y_{p_{i+1}} = \varepsilon_{p_{i+1}} - \varepsilon_{c_{i+1}} = \frac{10}{24}h^4 y^{IV}$$

звідки визначаємо

$$\frac{\varepsilon_{c_{i+1}}}{\varepsilon_{p_{i+1}}-\varepsilon_{c_{i+1}}}=\frac{-\frac{1}{24}h^4y^{IV}}{\frac{10}{24}h^4y^{IV}}=-\frac{1}{10}\ .$$

Тоді

$$y_{i+1}=y_{c_{i+1}}+\varepsilon_{c_{i+1}}=y_{c_{i+1}}+\frac{\varepsilon_{c_{i+1}}}{\varepsilon_{p_{i+1}}-\varepsilon_{c_{i+1}}}\left(y_{c_{i+1}}-y_{p_{i+1}}\right)=y_{c_{i+1}}+\left(y_{p_{i+1}}-y_{c_{i+1}}\right)\Big/10\,.$$

5) Коригуємо значення похідної для наступних кроків: $y'_{i+1}=\mathrm{f}\left(y_{i+1},\ t_{i+1}\right)$, а за отриманим значенням похибки $\varepsilon_{p_{i+1}}-\varepsilon_{c_{i+1}}$ робимо висновки про необхідність зміни кроку інтегрування.

Неявні формули Адамса мають вищі точність і стійкість, а в поєднанні з явними дають змогу отримати порівняно нескладну обчислювальну процедуру з оцінкою похибки на кроці.

Як приклад використання методу "прогноз–корекція" на підставі формул Адамса третього порядку нижче подано тексти підпрограми, що реалізує розв'язування системи звичайних диференціальних рівнянь, що записана у нормальній формі Коші, з використанням стратегії автоматичного вибору кроку інтегрування. Оскільки інтегрування за формулами Адамса повинно виконуватися для рівновіддалених точок, після зміни кроку здійснюється інтерполяція попередніх точок.

Quick Basic

```
SUB adams3 (N, t, h, Eps, y()) STATIC
' ------------------------------------------------
'  Метод "прогноз-коригування" на основі явної та неявної формул
'  Адамса 3-го порядку з автоматичним вибором кроку
' ------------------------------------------------
```

```
'
'  Вхідні параметри:
'     N   - порядок системи
'     t   - незалежна змінна (час)
'     h   - крок (змінюється автоматично)
'     Y   - масив змінних інтегрування
'     Eps - точність на кроці
'
'  Вихідні параметри:
'     t   - незалежна змінна (час)
'     h   - крок (змінюється автоматично)
'     Y   - масив змінних інтегрування
'
'  Використовують підпрограму  Dif , що обчислює значення
'  правих частин системи диференціальних рівнянь:
'        Dif( t , Y() , F())   , де
'             t - час
'             Y - ім'я масиву змінних інтегрування
'             F - ім'я масиву похідних інтегрування
' ------------------------------------------------------------
'
'   Якщо перше звертання, то ініціалізація
'
IF first% = 0 THEN
   DIM f(N), f1(N), f2(N), f3(N), yp(N), yc(N), Delta(N)
   FOR i = 1 TO N
      f1(i) = 0
      f2(i) = 0
      f3(i) = 0
   NEXT i
   CALL Dif(t, y(), f1())
   first% = 1
END IF
' ------------------------------------------------
'   Основна частина програми
' ------------------------------------------------
DO
   '
   '  Прогноз з використанням явної формули Адамса
   '
   FOR i = 1 TO N
      yp(i) = y(i) + h * (23 * f1(i) - 16 * f2(i) + 5 * f3(i)) / 12
   NEXT i
   CALL Dif(t + h, yp(), f())
```

```
   Yerr = 1E-12        ' Уникаємо ділення на нуль
   '
   '   Коригування з використанням неявної формули  Адамса
   '
   FOR i = 1 TO N%
      yc(i) = y(i) + h * (5 * f(i) + 8 * f1(i) - f2(i)) / 12
   '
   '   Обчислення похибки на кроці
   '
      Delta(i) = (yp(i) - yc(i)) / 10
      Yerr = Yerr + Delta(i) * Delta(i)
   NEXT i

   Yerr = SQR(Yerr) / N
   '
   ' Вибір кроку інтегрування
   '
   Hold = h
   Kh = SQR(SQR((Eps / Yerr)))
   IF Kh > 3! THEN Kh = 3!        ' Обмеження величини зміни кроку
   IF Kh < .2 THEN Kh = .2        ' інтегрування
   h = h * Kh * .9
   '
   '   Інтерполяція попередніх точок після зміни кроку
   '
   FOR i = 1 TO N
        A = f1(i) * (2 * Hold * Hold - 3 * h * Hold + h * h)
        B = f2(i) * 2 * h * (2 * Hold - h)
        C = f3(i) * h * (h - Hold)
      fi2 = (A + B + C) / (2 * Hold * Hold)
        A = f1(i) * (Hold * Hold - 3 * Hold * h + 2 * h * h)
        B = f2(i) * 4 * h * (Hold - h)
        C = f3(i) * h * (2 * h - Hold)
      fi3 = (A + B + C) / (Hold * Hold)
      f2(i) = fi2
      f3(i) = fi3
   NEXT i

LOOP UNTIL Yerr <= Eps
'
' Закінчення кроку інтегрування
'
t = t + Hold
```

```
FOR i = 1 TO N%
   y(i) = yc(i) + Delta(i)
   f3(i) = f2(i)
   f2(i) = f1(i)
NEXT i
' Уточнюємо значення похідної
CALL Dif(t, y(), f1())

END SUB
```

Turbo Pascal

```
{$E+,N+,F+ }
unit  Adams3 ;
{ ================================================== }
{  Метод прогноз-коригування на підставі формул Адамса 3-го }
{  порядку з автоматичним вибором кроку розв'язування  }
{ ================================================== }
interface

type
   Vector = array [1..100]  of  DOUBLE ;

   Sys = procedure(t: DOUBLE; var y, f : Vector) ;
   { ------------------------------------------------- }
   {  Процедура призначена для обчислення значень      }
   {  правих частин системи диференціальних рівнянь,   }
   {  що записані в нормальній формі Коші.             }
   {  Форма  запису:                                   }
   {       f[1] := функція_1(y[1],y[2],...,y[N],t)     }
   {       f[2] := функція_2(y[1],y[2],...,y[N],t)     }
   {                  . . .                            }
   {       f[N] := функція_N(y[1],y[2],...,y[N],t)     }
   {  Вхідні величини:                                 }
   {       t - незалежна змінна                        }
   {       y - масив змінних інтегрування              }
   {  Вихідні величини:                                }
   {       y - масив змінних інтегрування              }
   {       f - масив обчислених похідних               }
   { ------------------------------------------------- }
```

```
var
   y, f1, f2, f3, f4   : Vector ;

procedure
   ABM3(Dif : Sys ;        { Ім'я процедури правих частин системи }
        N   : integer ;    { Порядок системи                      }
    var t   : DOUBLE ;     { Незалежна змінна (час)               }
    var h   : DOUBLE ;     { Крок (змінюється автоматично)        }
        Eps : DOUBLE ;     { Точність на кроці                    }
    var y   : Vector ) ;   { Вектор змінних                       }
{ ------------------------------------------------------------ }
{  Вхідні/вихідні параметри:                                     }
{     t   - незалежна змінна (час)                               }
{     y   - масив змінних інтегрування                           }
{                                                                }
{  Використовується підпрограма Dif, що обчислює праві         }
{  частини системи диференціальних рівнянь:                      }
{        Dif(t, Y, F)    , де                                    }
{            t - час                                             }
{            Y - ім'я вектора змінних інтегрування               }
{            F - ім'я вектора похідних                           }
{ ------------------------------------------------------------ }

implementation

procedure
   ABM3(Dif : Sys ;        { Ім'я процедури правих частин системи }
        N   : integer ;    { Порядок системи                      }
    var t   : DOUBLE ;     { Незалежна змінна (час)               }
    var h   : DOUBLE ;     { Крок (змінюється автоматично)        }
        Eps : DOUBLE ;     { Точність на кроці                    }
    var y   : Vector ) ;   { Вектор змінних                       }
{ ------------------------------------------------------------ }
const
   p21 : DOUBLE = 23.0 / 12.0 ;
   p22 : DOUBLE = -16.0 / 12.0 ;
   p23 : DOUBLE = 5.0 / 12.0 ;

   c21 : DOUBLE = 5.0 / 12.0 ;
   c22 : DOUBLE = 8.0 / 12.0 ;
   c23 : DOUBLE = -1.0 / 12.0 ;

  cErr : DOUBLE = 0.1 ;
```

```
   GoodStep : boolean = true ;
var
   i      : integer ;
   f, yp, yc, Delta : Vector ;
   error, Hold, KoefH, MaxKoefH  : DOUBLE ;
   A, B, C, fi2, fi3   : DOUBLE ;
begin
  repeat
   {                                                    }
   {  Прогноз з використанням явної формули Адамса     }
   {                                                    }
      for  i := 1  to  N  do
         yp[i] := y[i] + h*(p21*f1[i] + p22*f2[i] + p23*f3[i]);

      dif(t + h, yp, f) ;

      error := 1e-16;          { Уникаємо ділення на нуль }
   {                                                      }
   {   Коригування з використанням неявної формули Адамса  }
   {                                                      }
      for  i := 1  to  N  do
      begin
         yc[i] := y[i] + h*(c21*f[i] + c22*f1[i] + c23*f2[i]);
   {                                        }
   {   Обчислення похибки на кроці          }
   {                                        }
        Delta[i] := (yp[i] - yc[i]) * cErr / (abs(yc[i]) + 1.0);
         error := error + sqr(Delta[i]);
      end ;
      error := sqrt(error / N) ;
   {                           }
   { Вибір кроку інтегрування  }
   {                           }
      Hold := h ;
      KoefH := sqrt(sqrt((Eps / error))) ;
      if  GoodStep = false   then
        begin
          GoodStep := true ;
          MaxKoefH := 1.0 ;
        end
      else
        MaxKoefH := 2.0 ;
    { Обмеження зміни кроку інтегрування }
```

```
      if  KoefH > MaxKoefH then  KoefH := MaxKoefH ;
      if  KoefH < 0.2  then  KoefH := 0.2 ;
      h := h * KoefH * 0.9 ;
   {                                                    }
   { Інтерполяція попередніх точок після зміни кроку }
   {                                                    }
      if  error > Eps  then
        begin
          GoodStep := false ;
          for  i := 1  to  N  do
          begin
             A := f1[i] * (2.0*Hold*Hold - 3.0*h*Hold + h*h);
             B := f2[i] * 2.0 * h * (2.0 * Hold - h) ;
             C := f3[i] * h * (h - Hold) ;
             fi2 := (A + B + C) / (2.0 * Hold * Hold) ;
             A := f1[i]*(Hold*Hold - 3.0*Hold*h + 2.0*h*h);
             B := f2[i] * 4.0 * h * (Hold - h) ;
             C := f3[i] * h * (2.0 * h - Hold) ;
             fi3 := (A + B + C) / (Hold * Hold) ;
             f2[i] := fi2 ;
             f3[i] := fi3 ;
          end ;
        end ;
   until  error <= Eps;
   {                                  }
   { Закінчення кроку інтегрування }
   {                                  }
   t := t + Hold ;
   for  i := 1  to  N  do
   begin
      y[i] := yc[i] + Delta[i] ;
      f3[i] := f2[i] ;
      f2[i] := f1[i] ;
   end ;
   dif(t, y, f1) ;
end ;   { OF PROCEDURE }
{ ---------------------------------- }

begin
end .
```

Автори не претендують на абсолютну досконаліть наведених прикладів програм, зокрема, їх ефективність можна покращити використанням вектора Нордсика [В.13] і відповідного алгоритму зміни кроку інтегрування, але це вже виходить за межі цього посібника.

Від авторів

Багаторічний досвід комп'ютерного моделювання з використанням різних числових методів розв'язування звичайних диференціальних рівнянь одного з авторів показує, що програми на підставі формул Адамса дають "акуратніший" (іншого слова автори не знайшли) розв'язок, особливо у задачах з нелінійностями. Треба відзначити, що це твердження є лише особистою думкою автора і базується на його особистому досвіді моделювання певного класу систем. Необхідно врахувати, що поведінка числових методів розв'язування звичайних диференціальних рівнянь великою мірою визначається типом розв'язуваних задач, і в кожному випадку варто спробувати підібрати раціональний для поставленої задачі числовий метод – універсальних методів та рецептів не буває...

Під час розв'язування задач моделювання інколи виникає ситуація, коли співвідношення між найменшою та найбільшою сталими часу перевищує два–три порядки. Такі задачі належать до класу *жорстких*. Звичайні числові методи не зовсім придатні для розв'язування таких задач, оскільки з умови стійкості вимагають малого кроку навіть для гладкого розв'язання. Для жорстких задач розроблено низку числових методів, з яких найвідомішими є формули Ґіра (*Gear*) чи, як їх ще називають, формули диференціювання назад (ФДН) [В.13]. Варто зауважити, що стійкість формул диференціювання назад доведено лише до шостого порядку включно, а практично застосовують формули не вище ніж п'ятого порядку.

Формули прогнозу для ФДН

p = 1 $y_{i+1} = 2y_i - y_{i-1}$;

p = 2 $y_{i+1} = 3y_i - 3y_{i-1} + y_{i-2}$;

$p = 3 \quad y_{i+1} = 4y_i - 6y_{i-1} + 4y_{i-2} - y_{i-3};$

$p = 4 \quad y_{i+1} = 5y_i - 10y_{i-1} + 10y_{i-2} - 5y_{i-3} + y_{i-4};$

$p = 5 \quad y_{i+1} = 6y_i - 15y_{i-1} + 20y_{i-2} - 15y_{i-3} + 6y_{i-4} - y_{i-5}.$

Формули диференціювання назад

$p = 1 \quad y_{i+1} = y_i + hy'_{i+1};$ $\qquad \varepsilon_{i+1} = -\frac{1}{2}h^2 y''_{i+1}$

$p = 2 \quad y_{i+1} = (4y_i - y_{i-1} + 2hy'_{i+1})/3;$ $\qquad \varepsilon_{i+1} = -\frac{2}{9}h^3 y'''_{i+1}$

$p = 3 \quad y_{i+1} = (18y_i - 9y_{i-1} + 2y_{i-2} + 6hy'_{i+1})/11;$ $\qquad \varepsilon_{i+1} = -\frac{3}{22}h^4 y^{IV}_{i+1}$

$p = 4 \quad y_{i+1} = (48y_i - 36y_{i-1} + 16y_{i-2} - 3y_{i-3} + 12hy'_{i+1})/25;$ $\qquad \varepsilon_{i+1} = \frac{-12}{125}h^5 y^{V}_{i+1}$

$p = 5 \quad y_{i+1} = \begin{pmatrix} 300y_i - 300y_{i-1} + 200y_{i-2} - \\ -75y_{i-3} + 12y_{i-4} + 69hy'_{i+1} \end{pmatrix} \Big/ 137;$ $\qquad \varepsilon_{i+1} = \frac{-10}{137}h^6 y^{VI}_{i+1}$

Стосовно жорсткості систем звичайних диференціальних рівнянь, які описують реальну систему електропривода, треба відзначити, що така проблема у грамотно спроектованій моделі виникає вкрай рідко: нема потреби враховувати малі сталі часу (які відрізняються від основних більше ніж на два–три порядки), тому що здебільшого їх дія не буде помітною і найчастіше ними можна знехтувати. Дослідник завжди повинен свідомо ставитись до об'єкта моделювання і створеної ним моделі і розуміти, що істотно впливає на поведінку моделі, а чим можна і знехтувати.

Вибір кроку

Основним критерієм у виборі кроку інтегрування є здатність відтворення досліджуваних процесів з мінімальними втратами часу і точності. Передовсім це пов'язано з теоремою відліків Шеннона–Котєльнікова [В.8, Г.3, Г.6], яка

доводить, що для відтворення обмеженого частотою ω_{max} спектра сигналу частота відліків ω_0 повинна бути принаймні вдвічі вищою за максимальну частоту відтворюваного спектра ω_{max}, тобто, $\omega_0 \geq 2\omega_{\max}$, де $\omega_0 = \frac{2\pi}{h}$; h – крок інтегрування.

Вважають [В.8], що достатнім є крок, який забезпечує передачу 90–95 % енергії спектра вхідного сигналу. Відповідно, для моделювання повільних процесів достатнім є доволі великий крок, а для швидкоплинних крок інтегрування доводиться зменшувати. До речі, це ефективно здійснюється алгоритмами з автоматичним вибором кроку інтегрування, які описано далі. Завдяки цій властивості (можливості пристосування (*адаптації*) кроку до поведінки координат досліджуваної моделі) найпоширеніші саме алгоритми з автоматичним вибором кроку інтегрування, які становлять основну частину сучасних процедур розв'язування звичайних диференціальних рівнянь.

Іншим обмеженням на значення кроку інтегрування є стійкість використаного числового методу: для більшості формул числового інтегрування звичайних диференціальних рівнянь крок h не повинен перевищувати $(1...3)T_{\min}$, де $T_{\min}$ – найменша стала часу досліджуваної системи, інакше можливе виникнення числової нестійкості інтегрування (*у практичній роботі можна вважати, що для більшості числових методів значення кроку інтегрування не повинно перевищувати значення найменшої сталої часу*). Винятком є спеціально розроблені формули диференціювання назад, які стійкі для будь-якого кроку (*на жаль, це їх єдина перевага*).

Ці чинники враховуються автоматично у програмах, що використовують, як вже згадувалося вище, стратегію автоматичного вибору кроку інтегрування, інакше це доводиться робити дослідникові, якщо він застосовує процедури з фіксованим кроком інтегрування.

Моделювання з фіксованим кроком є простішим у реалізації, особливо з використанням різницевих рівнянь і Z-перетворення (див. далі). Попередній вибір фіксованого кроку розв'язування системи диференціальних рівнянь, що описують динаміку електропривода, є порівняно простою задачею. Загальні рекомендації є такими:

– для методів невисокого порядку (наприклад, формули трапецій) значення кроку становить приблизно 10–20 % від найменшої сталої часу системи [В.8];

– для методів вищих порядків (наприклад, Рунге–Кутта четвертого порядку) крок можна збільшити до 25–50 % від найменшої сталої часу.

Після попереднього вибору кроку інтегрування доцільно виконати уточнення кроку для зменшення можливих похибок і підвищення ефективності роботи програми. Уточнити крок можна, збільшуючи і зменшуючи його вдвічі: якщо процес стає нестійким чи вигляд його починає змінюватись (наприклад, зростає коливність розв'язку), крок зменшують.

Моделювання з автоматичним вибором кроку пришвидшує розв'язування з гарантованою точністю, тому що на гладких ділянках крок розв'язування автоматично збільшується. Процедура автоматичного вибору кроку інтегрування системи звичайних диференціальних рівнянь особливо просто реалізується для методів на підставі однокрокових формул (наприклад, формул Фельберга). Для багатокрокових методів (наприклад, на підставі формул Адамса) зміна кроку, як вже зазначалося вище, вимагає інтерполяції попередніх точок, оскільки в різницевій схемі цих багатокрокових методів вважають, що точки рівновіддалені. Тому програми на підставі багатокрокових методів значно складніші.

Від вибору алгоритму зміни кроку залежить ефективність роботи процедури розв'язування звичайних диференціальних рівнянь. До оптимального алгоритму автори відомих ефективних програм приходять після тривалих пошуків та уточнень алгоритму.

Найпростіший варіант реалізації алгоритму – зміна (збільшення чи зменшення) кроку вдвічі, що особливо зручно для багатокрокових методів, бо в такому разі простіше реалізується перехід між формулами різних порядків після зміни кроку інтегрування. Але такий спосіб не завжди є ефективним, тому в сучасних програмах реалізації багатокрокових методів використовується зміна кроку за допомогою вектора Нордсика [В.13]. Програми такого типу є складнішими, тому самому написати ефективну програму, що ґрунтується на формулах Адамса, з плавною зміною кроку є дуже і дуже складним завданням, тому доцільно скористатися готовою (наприклад, реалізованою у пакеті MATLAB функцією `ode113` чи знайти необхідну програму в Internet).

Далі описано один з найпоширеніших алгоритмів вибору кроку інтегрування, в якому нове значення кроку розраховують за формулою:

$$h_{i+1} = h_i K \left(\frac{\varepsilon_0}{\delta_i} \right)^{\frac{1}{p+1}},$$

де $K = 0.8–0.9$ – коефіцієнт гарантії;

ε_0 – задана похибка на кроці;

δ_i – оцінка відносної похибки на кроці;

p – порядок методу.

Оцінку відносної похибки обчислюють за формулою

$$\delta_i = \frac{\varepsilon_i}{1 + |y_i|},$$

де ε_i – оцінка похибки числового методу на кроці інтегрування;

y_i – значення змінної на кроці.

Оцінка відносної похибки, як запропоновано, призводить до завищення точності на кроці інтегрування для значень змінної, що нижчі за номінальне. Тому нормувати похибку під час моделювання електроприводів доцільно стосовно номінального значення кожної координати, що, як свідчать експерименти, відчутно покращує швидкодію алгоритму.

Щоб уникнути непотрібних різких змін кроку під час розв'язування задачі, зміна кроку здебільшого обмежується величиною

$$(0.1...0.2) \le \frac{h_{i+1}}{h_i} \le (1.5...5).$$

Після "забракованого" кроку (якщо довелось повторити розрахунок з меншим значенням кроку) на наступному кроці рекомендується його значення не збільшувати: $h_{i+1} \le h_i$.

Можна вважати доцільним примусове обмеження верхнього значення кроку інтегрування величиною, що становить 0.9–0.95 максимального кроку з умови збереження стійкості стосовно найменшої сталої часу T_{min} досліджуваної системи. Наприклад, для формули Фельберга 2(3)B найбільше значення кроку з умови стійкості не повинно перевищувати $2.5T_{min}$. Необхідність введення тако-

го обмеження пояснюється тим, що алгоритм автоматичного вибору кроку починає реагувати на зростання похибки лише після втрати стійкості числовим методом, після чого різко зменшує крок інтегрування. Такі різкі зміни кроку збільшують загалом тривалість розв'язування задачі.

У разі відсутності виразу для оцінки похибки числового методу однією з порівняно простих, але доволі ефективних процедур уточнення значення змінної та визначення похибки на кроці інтегрування є екстраполяція за Річардсоном (*Richardson*) [В.6, В.11], ідея якої є доволі простою та зрозумілою.

Похибка числового методу визначається його порядком p та кроком h і обчислюється за формулою $\varepsilon = K_\varepsilon \cdot h^p$, де K_ε – коефіцієнт похибки, що залежить від числового методу (див., наприклад, формули Адамса). Нехай знайдено наступну точку шуканого розв'язку y з двома різними кроками: h_1 і h_2, тоді точне значення розв'язку в обох випадках становитиме:

$$\left.\begin{aligned} y &= y_1 + K_\varepsilon \cdot h_1^p \\ y &= y_2 + K_\varepsilon \cdot h_2^p \end{aligned}\right\} \Rightarrow \text{ звідси } K_\varepsilon = \frac{y_1 - y_2}{h_2^p - h_1^p}.$$

Підставляючи це значення у перше рівняння, отримаємо

$$y = y_1 + \frac{y_1 - y_2}{(h_2 / h_1)^p - 1}.$$

Вираз $\dfrac{y_1 - y_2}{(h_2 / h_1)^p - 1}$ може використовуватись для оцінки похибки на кроці в алгоритмі автоматичного вибору кроку розв'язання. Такий підхід є практично єдиним способом оцінки похибки на кроці у разі реалізації різницевих схем чи застосування Z-перетворення (див. далі), а також за відсутності виразу для оцінки похибки у числовому методі інтегрування звичайних диференціальних рівнянь.

Найпоширенішим є варіант реалізації алгоритму екстраполяції за Річардсоном з одиничним і подвійним (або половинним і одиничним) кроками:

1) інтегрування системи диференціальних рівнянь, що описують модель, з подвійним кроком $2h$, внаслідок чого одержують значення змінної y_{2h};
2) знаходження розв'язку після двох одиничних кроків величиною h, внаслідок чого отримують значення змінної y_h;

3) знаходження уточненого значення змінної y за формулою

$$y = y_h + \frac{y_h - y_{2h}}{2^p - 1},$$

де p – порядок методу.

Додатково прочитати про алгоритми вибору кроку інтегрування звичайних диференціальних рівнянь можна в [В.11, В.13].

Стійкість числових методів

Для кожного методу можна визначити максимальне значення кроку, перевищення якого призводить до числової нестійкості. Область стійкості числового методу можна знайти різними способами [В.2, В.11, В.13], зокрема, можна проаналізувати, розв'язуючи тестове диференціальне рівняння $y' + y = 0$ з початковою умовою $y_0 = 1$. Для цього використовується підхід, що описаний в [В.5]: розглядати, наскільки точно числовий метод апроксимує аналітичний розв'язок цього диференціального рівняння – експоненту $y(t) = e^{-t}$, що зводиться до рекурентного рівняння $y_{i+1} = y_i \cdot e^{-h}$ з початковою умовою $y_0 = 1$. Для дійсного додатного кроку інтегрування h значення e^{-h} за модулем менше за одиницю. Це означає, що функція $f(t)$ буде згасною. Тому, щоб інтегрування було стійким, апроксимація значення e^{-h} числовим методом також повинна бути за модулем меншою від одиниці.

Для прикладу проаналізуємо неявну формулу Адамса другого порядку і формулу Рунге–Кутта третього порядку.

Неявна формула Адамса 2-го порядку

Підставляючи значення похідної $y' = -y$ з тестового рівняння $y' + y = 0$ у формулу методу $y_{i+1} = y_i + \frac{h}{2}(y'_{i+1} + y'_i)$, отримаємо $y_{i+1} = y_i - \frac{h}{2}(y_{i+1} + y_i)$. Зведемо подібні члени і одержимо $y_{i+1} = y_i\left(\frac{2-h}{2+h}\right)$. Це означає, що неявна

формула Адамса другого порядку апроксимує e^{-h} виразом $\frac{2-h}{2+h}$, який не перевищує за модулем одиницю для будь-якого додатного значення кроку h, що означає стійкість методу для будь-якого кроку інтегрування. Отже, областю стійкості неявної формули Адамса другого порядку є інтервал $h \in [0; \infty)$.

Формула Рунге–Кутта 3-го порядку

Підставивши значення похідної $y' = -y$ з тестового рівняння $y' + y = 0$ у формулу методу $y_{i+1} = y_i + (k_1 + 4k_2 + k_3)/6$, отримаємо:

$$k_1 = h\,\mathrm{f}(x_i, y_i) = -hy_i; \qquad k_2 = h\,\mathrm{f}\left(x_i + \frac{h}{2}, y_i + \frac{k_1}{2}\right) = -h\left(y_i - \frac{hy_i}{2}\right);$$

$$k_3 = h\,\mathrm{f}(x_i + h, y_i - k_1 + 2k_2) = h\,\mathrm{f}(x_{i+1}, y_i - k_1 + 2k_2) = -h\left(y_i - hy_i + 2h\left(y_i - \frac{hy_i}{2}\right)\right),$$

звідки

$$y_{i+1} = y_i\left(1 - h\left(1 - \frac{h}{2}\left(1 - \frac{h}{3}\right)\right)\right).$$

Для формули Рунге–Кутта вираз $1 - h\left(1 - \frac{h}{2}\left(1 - \frac{h}{3}\right)\right)$ є апроксимацією точного значення e^{-h} його обмеженим розкладом у ряд. Цей вираз за модулем стає більшим за одиницю, якщо значення кроку інтегрування h перевищує 2,5. Отже, областю стійкості формули Рунге–Кутта третього порядку є інтервал $h \in [0; 2{,}5]$.

Для частини числових методів графіки залежності апроксимувального виразу від кроку h порівняно із згасною експонентою e^{-h} (суцільна гладка лінія) показані на рис. 3.4. Для стійкого згасного розв'язку апроксимаційний вираз при y_i за модулем не повинен перевищувати 1, інакше формула починає давати нестійкий розв'язок (як вже згадувалося вище, за цим критерієм можна оцінити область допустимих значень кроку h). Якщо ж апроксимація числовим методом значення e^{-h} міститься в межах від –1 до 0, то отримуємо знакозмінний вираз біля y_i в рекурентній формулі та, відповідно, знакозмінний згасний розв'язок. У такому разі в розв'язку одержують коливання, спричинені числовим методом.

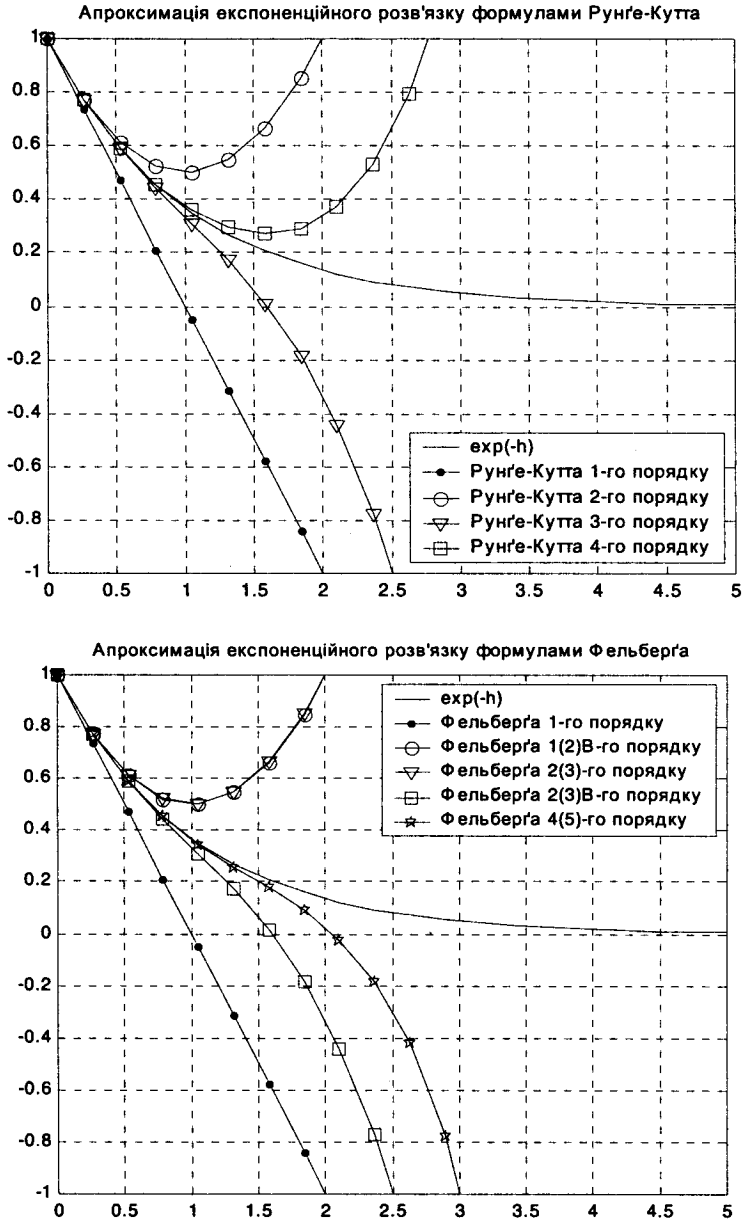
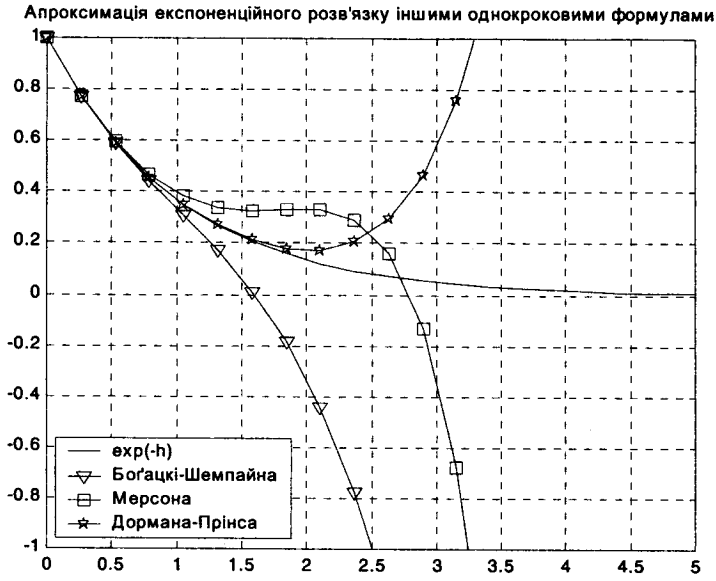


Рис. 3.4. Апроксимація згасної експоненти деякими числовими методами



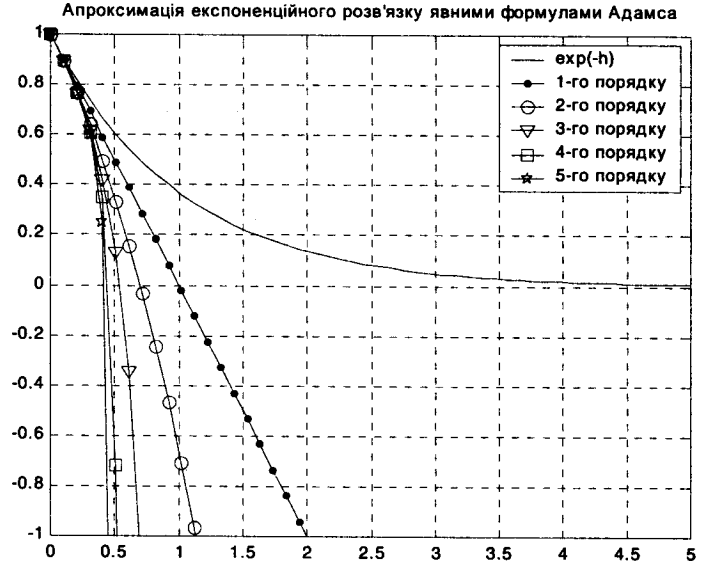


Рис. 3.4. Апроксимація згасної експоненти деякими числовими методами (продовження)

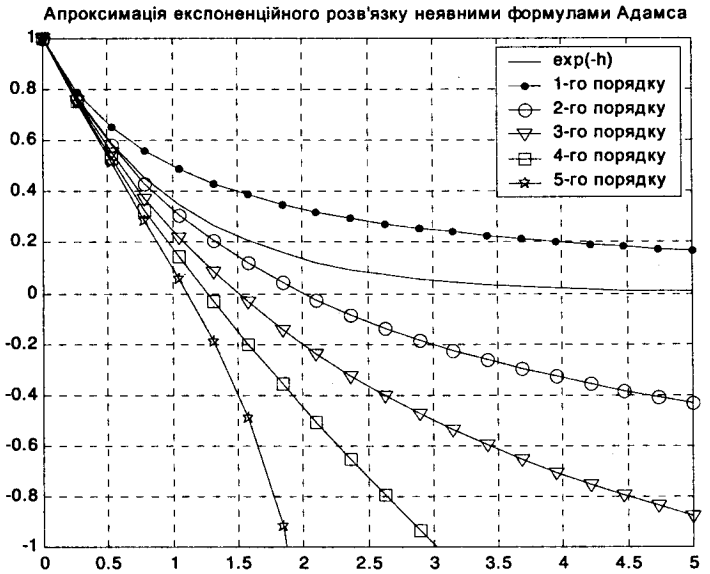


Рис. 3.4. Апроксимація згасної експоненти деякими числовими методами (закінчення)

Як видно з рис. 3.4, практично всі формули числового інтегрування забезпечують задовільну апроксимацію e^{-h} лише для $h \leq 1$, що і визначає область їх раціонального застосування. З поданих графіків видно перевагу неявних методів Адамса щодо розміру області стійкості. Також розширює область стійкості в однокрокових методах підвищення їх порядку (найбільшу область стійкості мають формули Фельберґа 4-го порядку і Дормана–Прінса 5-го порядку). Стосовно методів ФДН варто сказати, що врахування "передісторії" (попередніх кроків) розширяє область стійкості через точнішу апроксимацію експоненти, але ці формули вимагають додаткового дослідження у зв'язку з появою особливих точок на кривій апроксимації.

Реалізація числових методів у математичних пакетах

Перевагою використання математичних пакетів є наявність у них функцій, в яких вже реалізовані числові методи для розв'язування диференціальних

рівнянь, що спрощує моделювання із застосуванням таких пакетів. Нижче подано коротку інформацію про такі реалізації (слід пам'ятати, що фірми-виробники математичних пакетів постійно працюють над їх вдосконаленням, тому в нових версіях можливі зміни і доповнення).

MathCAD

`rkfixed` – функція призначена для розв'язування звичайних диференціальних рівнянь та системи з n диференціальних рівнянь за допомогою формули Рунге–Кутта четвертого порядку з фіксованим кроком і є базовою в пакеті. Виклик функції:

```
rkfixed(y, Xmin, Xmax, Npoints, D),
```

де `y` – вектор початкових умов у точці `Xmin` розміром n;

`Xmin`, `Xmax` – початкова і кінцева точки інтервалу інтегрування;

`Npoints` – бажана кількість точок розв'язку на інтервалі інтегрування, визначає `1+Npoints` рядків результівної матриці, яку повертає функція `rkfixed` ; для стійкого розв'язку необхідно подбати, щоб крок інтегрування не перевищував найменшої сталої часу;

`D` – функція-вектор, що містить перші похідні шуканої функції, складається з n рядків;

`rkfixed` повертає результівну матрицю (для прикладу назвемо її S), в якій:

(1) перший стовпець $S^{<0>}$ містить значення аргументу в точках розв'язку;

(2) наступні стовпці містять розв'язки за кожною змінною: наприклад, перший стовпець $S^{<1>}$ містить `(1+Npoints)` – елементний вектор розв'язку першої змінної, $S^{<2>}$ – вектор розв'язку другої змінної і т. д., відповідно, $S^{<n>}$ містить вектор розв'язку n-ї змінної.

`Rkadapt` – функція для розв'язування нежорстких систем з розв'язком, який змінюється повільно (є доволі універсальною, тому її можна

використовувати у багатьох випадках), застосовує алгоритм з автоматичним вибором кроку інтегрування на підставі формули Рунґе–Кутта четвертого порядку, але результат подається у рівновіддалених точках (як у `rkfixed`); функцію викликають аналогічно.

`Bulstoer` – реалізація методу Булірш–Штура (*Bulirsch-Stoer*) для розв'язування гладких функцій, для яких є дещо точнішою, ніж метод Рунґе–Кутта, що реалізований в `rkfixed` ; викликають її так само, як і попередні функції.

`Stiffb` – функція для розв'язування системи жорстких диференціальних рівнянь, застосовує метод Булірш–Штура.

`Stiffr` – функція для розв'язування системи жорстких диференціальних рівнянь, використовує метод Розенброка (*Rosenbrock*); обидві функції викликають аналогічно:

```
Stiffb(y, Xmin, Xmax, Npoints, D, J)

Stiffr(y, Xmin, Xmax, Npoints, D, J),
```

де `J` – функція, що повертає матрицю розміром $n\times(n+1)$, в якій перший стовпець містить похідні, а наступні стовпці утворюють матрицю Якобі системи диференціальних рівнянь.

Використання функцій розв'язування системи диференціальних рівнянь пакета MathCAD показано на прикладі: задано систему диференціальних рівнянь, що описує запуск двигуна постійного струму зі сталим потоком збудження.

$$\begin{cases} T_a\dfrac{di_a}{dt}+i_a=\dfrac{U_a-C\omega}{R_a}; \\ J\dfrac{d\omega}{dt}=C\left(i_a-I_c\right); \end{cases} \xrightarrow{\begin{array}{c}\textit{переходимо}\\ \textit{до нормальної}\\ \textit{форми Коші}\end{array}} \begin{cases} \dfrac{di_a}{dt}=\dfrac{\left(U_a-C\omega\right)/R_a-i_a}{T_a}; \\ \dfrac{d\omega}{dt}=\dfrac{C\left(i_a-I_c\right)}{J}. \end{cases}$$

Для розв'язування цієї системи застосовують поданий нижче документ MathCAD.

$N := 100$ *Кількість точок розв'язку*

$t_{min} := 0$ $t_{max} := 1$ *межі (діапазон) інтегрування*

$y_0 := 0$ $y_1 := 0$ *початкові умови*

Задаємо параметри двигуна:

$Ta := 0.05$ *стала часу якірного кола*

$Ua := 50$ *напруга на якорі*

$C := 2.5$ *стала двигуна*

$Ra := 0.1$ *опір якірного кола*

$J := 2$ *момент інерції привода*

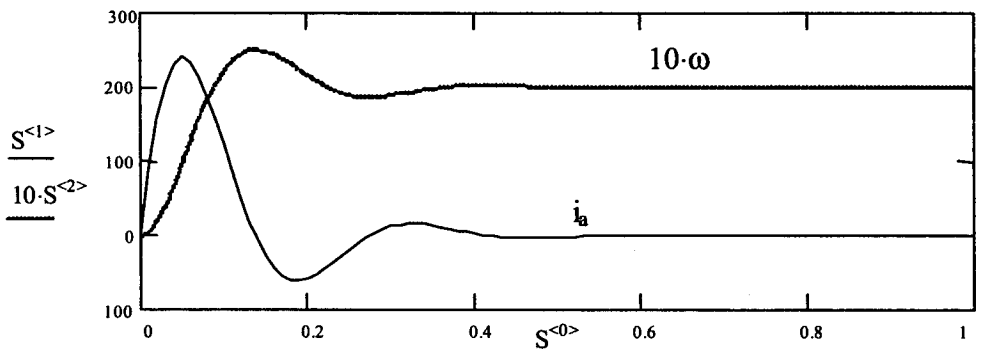
$Ic := 0$ *статичний струм*

$$DiffEq(x,y) := \begin{bmatrix} \dfrac{\dfrac{Ua - C \cdot y_1}{Ra} - y_0}{Ta} \\ \dfrac{C \cdot (y_0 - Ic)}{J} \end{bmatrix}$$

задаємо функцію-вектор для обчислення похідних

Отримуємо розв'язок і виводимо на графік:

$$S := rkfixed(y, t_{min}, t_{max}, N, DiffEq)$$



(Для наочності масштаб швидкості ω збільшено у 10 разів)

MATLAB

`ode23` – реалізація однокрокової формули Богацкі–Шемпайна (*Bogacki–Shampine*) порядку 2(3) з автоматичним вибором кроку інтегрування. За характеристиками ця формула дуже подібна до формул Фельберґа порядку 2(3)B і Рунґе–Кутта третього порядку. Цю функцію можна рекомендувати як стандартну для моделювання електроприводів, автори пакета MATLAB рекомендують її як ефективнішу за `ode45` для невисокої точності та для слабкожорстких систем.

`ode45` – реалізація формули Дормана–Прінса порядку 5(4) з автоматичним вибором кроку інтегрування. Рекомендується авторами пакета як універсальний метод, який може застосовуватися у першій спробі під час розв'язування системи диференціальних рівнянь, особливо ефективний для високої точності (10^{-6} і вище).

Спосіб виклику обох функцій є ідентичним. Текстова змінна `fileode` визначає назву функціонального файла, в якому записана система диференціальних рівнянь у нормальній формі Коші. Так, наприклад, для розв'язування системи диференціальних рівнянь в інтервалі часу $[t_0, t_k]$ з початковими умовами, визначеними вектором y_0, найпростіше звертання до функції має вигляд:

```
[t, x] = ode23('fileode', [tstart tfinish], y0),
[t, x] = ode45('fileode', [tstart tfinish], y0).
```

`ode113` – реалізує метод "прогноз – коригування" за формулами Адамса–Бешфорта–Малтона 1–13 порядків з автоматичним вибором порядку методу і кроку інтегрування і, на думку авторів посібника, є одним з найкращих засобів для розв'язування нежорстких систем диференціальних рівнянь, особливо для моделей з нелінійностями. Рекомендоване значення точності – не нижче за 10^{-4}. Ця функція особливо ефективна для складних моделей, що потребують значного часу обчислень.

`ode23s` – реалізує модифікований метод Розенброка (*Rosenbrock*) другого порядку для жорстких систем диференціальних рівнянь. Рекоменду-

ється для невисокої точності. Може застосовуватися для низки жорстких задач, для яких функція `ode15s` не є ефективною.

`ode15s` – універсальна функція, що реалізує два методи розв'язування жорстких систем диференціальних рівнянь з автоматичним вибором кроку розв'язування:

1) на підставі формул диференціювання назад (ФДН) порядку 1–5;
2) на основі родини формул числового диференціювання, запропонованих Клопфенштайном (*Klopfenstein*) і Райхером (*Reiher*) порядку 1–5, трохи ефективніших за ФДН.

`ode23t` – функція базується на методі трапецій, призначена для систем зі середньою жорсткістю і ефективна для невисокої точності розв'язання.

`ode23tb` – реалізує метод TR-BDF2, що подібний до методу Рунге–Кутта, в якому на першому кроці застосовується формула трапецій, а на другому – формула диференціювання назад другого порядку, ефективна для невисокої точності.

Використання функції розв'язування диференціальних рівнянь пакета MATLAB показано на попередньому прикладі.

Пояснення: Для знаходження правих частин системи диференціальних рівнянь створюється файл **`DIFF_RP.M`**, який містить однойменну функцію **`diff_rp`**, що обчислює ці праві частини заданої системи.

```
function f=diff_rp(t, y);
% Обчислення значень правих частин системи диференціальних рівнянь
Ua = 50;          % Напруга на якорі двигуна
Ra = 0.1;         % Опір якірного кола
Ta = 0.05;        % Стала часу якірного кола
C = 2.5;          % Стала двигуна
J = 2;            % Момент інерції привода
Ic = 0;           % Статичний струм
f=zeros(2,1);
f(1) = ((Ua-C*y(2))/Ra-y(1))/Ta;    % Похідна струму
f(2) = C*(y(1) - Ic)/J;             % Похідна швидкості
```

Далі міститься основний файл (або інакше, скрипт MATLAB), який розв'язує систему диференціальних рівнянь, що описана функцією **diff_rp** .

```
% Приклад використання вбудованої функції ode23 для
% розв'язування системи диференціальних рівнянь
tmax = 1;              % Час розрахунку
y0 = [0 0];            % Початкові умови

% Розв'язування диференціального рівняння
[t, y] = ode23('Diff_rp', [0 tmax], y0);
% Побудова графіка
plot(t, y(:,1), 'k', t, 10*y(:,2), 'k:')
title('Розв'язування диференціальних рівнянь');
legend('струм якоря', 'швидкість x 10');
```

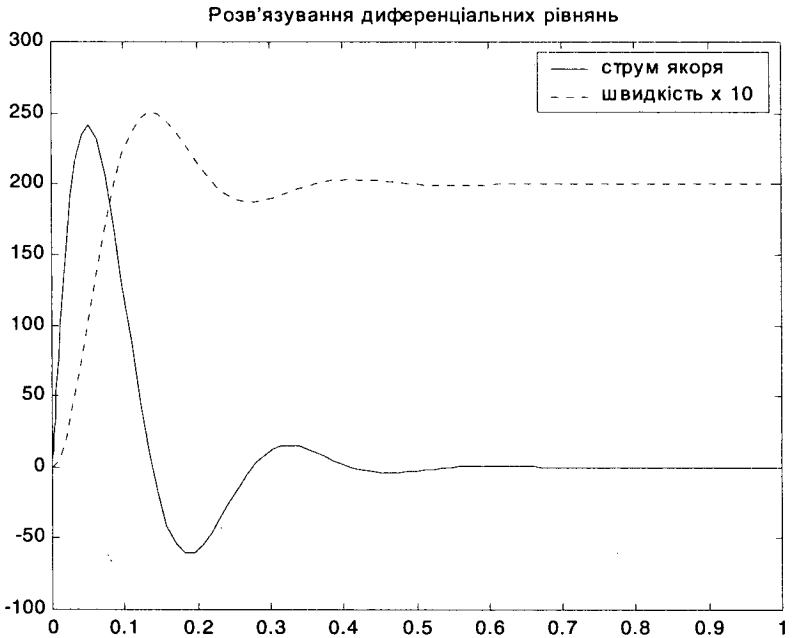


Рис. 3.5. Графік до прикладу розв'язування системи диференціальних рівнянь у MATLAB

У разі застосування числових методів розв'язування звичайних диференціальних рівнянь завжди треба пам'ятати, що вони апроксимують розв'язок обмеженим розкладом у ряд Тейлора. Це означає, що функція розв'язку є неперервною та диференційованою, і не повинно бути розривів як самої функції, так і її похідних. Саме тому числові методи розв'язування звичайних диференціальних рівнянь необхідно використовувати з певною обережністю та розуміти область їх допустимого застосування. Зокрема, числові методи працюють з певними труднощами на розривних функціях, наприклад, у моделях систем з релейними регуляторами.

3.3. Моделювання електроприводів різницевими рівняннями

Різницеве рівняння – це засіб описування динаміки систем. Воно подає розв'язок, який описується дискретними функціями.

Рекурентна формула дає змогу знайти i+1-те число на підставі попередніх i-х чисел.

Розв'язки різницевого рівняння і рекурентної формули важко розрізнити.

Використання цифрових інтеграторів

Нехай маємо рівняння

$$y(t) = y_0 e^{-\frac{t}{T}},$$

яке є розв'язком неперервного процесу, що описує однорідне диференціальне рівняння $y' + \frac{y}{T} = 0$. Дискретна функція розв'язку в часі $t = i \cdot h$ має вигляд

$$y(i) = y_0 e^{-\frac{i \cdot h}{T}},$$

а рекурентна формула –

$$y_{i+1} = y_i e^{-\frac{h}{T}}.$$

Тут ми маємо залежність i+1-го члена від i-го.

Нехай дано неоднорідне диференціальне рівняння

$$Ty' + y = x$$

і виразимо

$$y' = \frac{x - y}{T}.$$

З використанням для інтегрування явної формули Ейлера (явна формула Адамса першого порядку)

$$y_{i+1} = y_i + h \cdot y_i'$$

одержуємо

$$y_{i+1} = y_i + \frac{h}{T}(x_i - y_i),$$

звідки

$$y_{i+1} = y_i\left(1 - \frac{h}{T}\right) + \frac{h}{T}x_i.$$

Аналогічно, із застосуванням для інтегрування неявної формули Ейлера (неявна формула Адамса першого порядку)

$$y_{i+1} = y_i + h \cdot y_{i+1}'$$

отримаємо

$$y_{i+1} = y_i + \frac{h}{T}(x_{i+1} - y_{i+1}),$$

звідки

$$y_{i+1} = y_i \frac{T}{T+h} + \frac{h}{T+h}x_{i+1}.$$

Під час складання різницевих рівнянь за цим способом використовують формули відомих багатокрокових цифрових інтеграторів, наприклад, формули Адамса чи ФДН. Треба відзначити, що використання неявних методів не тільки не ускладнює задачу, а й призводить до одержання рекурентних рівнянь з

вищими стійкістю (див. підрозділ "**Стійкість різницевих рівнянь**") та точністю порівняно із застосуванням явних методів. Однією з найкращих формул для використання за цим способом є неявний метод трапецій (неявна формула Адамса другого порядку), яка дає змогу отримати нескладні рекурентні рівняння з доброю стійкістю та достатньою для практичних застосувань точністю.

Наприклад:

Аперіодичну ланку описують диференціальним рівнянням першого порядку

$$Ty' + y = kx \quad \text{або} \quad y' = \frac{kx - y}{T},$$

де T – стала часу ланки;
k – коефіцієнт підсилення ланки;
x – вхідний сигнал.

Для формування різницевого рівняння використовують формулу ФДН третього порядку $11y_{i+1} = 18y_i - 9y_{i-1} + 2y_{i-2} + 6hy'_{i+1}$. Підставимо з диференціального рівняння вираз для похідної y' у формулу методу

$$11y_{i+1} = 18y_i - 9y_{i-1} + 2y_{i-2} + + \frac{6h}{T}\left(kx_{i+1} - y_{i+1}\right),$$

після чого переносимо y_{i+1} в ліву частину

$$\left(11 + \frac{6h}{T}\right)y_{i+1} = 18y_i - 9y_{i-1} + 2y_{i-2} + \frac{6h}{T}kx_{i+1},$$

звідки

$$y_{i+1} = \frac{18y_i - 9y_{i-1} + 2y_{i-2} + \frac{6h}{T}kx_{i+1}}{11 + \frac{6h}{T}}.$$

За отриманим рекурентним рівнянням складають програму. Далі, як приклад, показано її фрагменти.

Quick Basic

```
Y = 0              ' Початкова умова
Yold1 = 0          ' Початкова умова
Yold2 = 0          ' Початкова умова
' Початок циклу
FOR t = Tmin TO Tmax STEP h
   X = ( ... вираз для розрахунку вхідного сигналу ... )

   Ynew = (18*Y - 9*Yold1 + 2*Yold2 + 6*h/T*k*X) / (11+6*h/T)
   (  ...   Оператори виведення результатів на кроці  ...   )

   ' Готуємось до нового кроку
   Yold2 = Yold1
   Yold1 = Y
   Y = Ynew
NEXT t
```

Turbo Pascal

```
Y := 0.0 ;             { Початкова умова }
Yold1 := 0.0 ;         { Початкова умова }
Yold2 := 0.0 ;         { Початкова умова }
{ Початок циклу }
t := Tmin ;
repeat
   X := ( ... вираз для розрахунку вхідного сигналу ... ) ;

   Ynew := (18*Y - 9*Yold1 + 2*Yold2 + 6*h/T*k*X) / (11+6*h/T) ;

   (  ...   Оператори виведення результатів на кроці  ...   )

   { Готуємось до нового кроку  }
   Yold2 := Yold1 ;
   Yold1 := Y ;
   Y := Ynew ;
   t := t + h ;
until t > Tmax ;
```

MathCAD

$j := 0 .. N$ $t_j := j \cdot h$ *Час*

$i := 2 .. N$

$X := (\ldots$ *розрахунок напруги завдання* $\ldots)$

$$Y_{i+1} := \frac{18.Y_i - 9.Y_{i-1} + 2.Y_{i-2} + \frac{6.h}{T}.K.X}{11 + \frac{6.h}{T}}$$

MATLAB

```
N = Tmax/h + 1 ;      % Кількість точок розрахунку
t = 0:h:Tmax ;        % Час
Y(1) = 0 ;            % Початкова умова
Y(2) = 0 ;            % Початкова умова
Y(3) = 0 ;            % Початкова умова
% Початок циклу
for i = 4:N
 X(i) = ( ... вираз для розрахунку вхідного сигналу ... );

 Y(i) = (18*Y(i-1) - 9*Y(i-2) + 2*Y(i-3) + 6*h/T*k*X(i))/(11+6*h/T);

  (  ...   Оператори виведення результатів на кроці   ...   )
end
```

Стійкість різницевих рівнянь

Оцінимо числову стійкість різницевих рівнянь, що отримані за явним і неявним методами Ейлера (формула прямокутників). Якщо прийняти $x = 0$, то для явного методу матимемо

$$y_{i+1} = y_i - \frac{h}{T} y_i = y_i \left(1 - \frac{h}{T}\right).$$

Для неявного методу Ейлера матимемо

$$y_{i+1} = y_i - \frac{h}{T} y_{i+1} \quad \Rightarrow \quad y_{i+1}\left(1 + \frac{h}{T}\right) = y_i \quad \Rightarrow \quad y_{i+1} = y_i \left(\frac{T}{T+h}\right).$$

Порівняємо вирази для явного і неявного методів з позиції числової стійкості. Числова стійкість цих різницевих рівнянь зумовлена значеннями коефіцієнтів правої частини, тобто

$1 - \frac{h}{T}$ – для явної формули;

$\frac{T}{T+h}$ – для неявної формули.

Якщо значення цих виразів за модулем перевищує одиницю, то таке різницеве рівняння стає нестійким. З цієї умови можна визначити значення кроку інтегрування h, яке для заданого T забезпечує стійкість розв'язку різницевого рівняння.

З умови стійкості явного методу

$$\left|1 - \frac{h}{T}\right| \le 1, \qquad \text{звідки} \qquad 0 \le \frac{h}{T} \le 2,$$

а неявного –

$$\left|\frac{T}{T+h}\right| \le 1, \qquad \text{звідки} \qquad 0 \le \frac{h}{T}$$

тобто неявний метод стійкий для будь-якого кроку.

Неявне інтегрування забезпечує кращу стійкість і вищу точність.

Тепер оцінимо стійкість неявних різницевих рівнянь інтегруванням неоднорідних диференціальних рівнянь, тому що в однорідних диферен-

ціальних рівняннях кінцеві значення інтегральних функцій дорівнюють нулеві. Оцінку стійкості виконано на прикладі аперіодичної ланки, передатна функція якої за Лапласом

$$\frac{y(s)}{x(s)} = \frac{1}{Ts+1},$$

де $y(s)$, $x(s)$ – відображення вхідного і вихідного сигналів. Відповідне диференціальне рівняння

$$Ty' + y = x$$

є неоднорідним і його інтегральна функція є точним розв'язком

$$y(t) = y_0 e^{-\frac{t}{T}} + x\left(1 - e^{-\frac{t}{T}}\right).$$

Усталеного кінцевого значення функція досягає, коли $t \to \infty$, тоді $y(t) = x$. Різницеві рівняння, що отримані за явною та неявною формулами, такі:

$$y_{i+1} = y_i\left(1 - \frac{h}{T}\right) + \frac{h}{T}x$$

– для явної формули інтегрування;

$$y_{i+1} = y_i \frac{T}{T+h} + \frac{h}{T+h}x$$

– для неявної формули інтегрування.

Усталеного стійкого значення ці різницеві рівняння досягають на i-му кроці для $t \to \infty$: тоді $y_{i+1} = y_i$. Усталені значення дорівнюватимуть:

$$y_{i+1} = y_i\left(1 - \frac{h}{T}\right) + \frac{h}{T}x \qquad \Rightarrow \qquad y_{i+1} = x$$

– для явної формули;

$$y_{i+1} = y_i \frac{T}{T+h} + \frac{h}{T+h}x \qquad \Rightarrow \qquad y_{i+1} = x$$

– для неявної формули.

Звідси робимо такий висновок:

і явні, і неявні методи досягають одного і того самого кінцевого значення, яке дорівнює кінцевому значенню ступеневої функції вхідного сигналу, але рекурентна формула неявного інтегрування, як вже згадувалося, забезпечує вищу точність і є стійкішою за явну.

Під час використання неявної формули інтегрування трапляються труднощі, які зумовлені обчисленням значень функції збурення (вхідного сигналу), якщо в системі є зворотні зв'язки. Тоді загалом

$$x = x(y,\ t),$$

а для дискретних величин

$$x_i = x(y_i,\ i \cdot h).$$

Ми маємо значення y_i, а для обчислення y_i необхідно мати x_i, тобто y_i залежить від себе. Для простої структури системи це вирішується доволі просто – алгебричними перетвореннями знаходимо вираз для обчислень x_i. Для складних же систем це може бути доволі складною проблемою, яка вимагає інших методів, наприклад, використанням розв'язання системи алгебричних рівнянь чи екстраполяції сигналу зворотного зв'язку.

Вибір кроку

Вибір кроку для різницевих рівнянь повністю відповідає засадам, викладеним у п. ***Вибір кроку*** розд. 3.2. У зв'язку з наявною різницевою схемою, що передбачає рівномірну сітку, найчастіше застосовують розв'язання з фіксованим кроком, у такому разі слід користуватися методикою з вказаного розділу. Варто зауважити, що найкращий ефект отримують з використанням багатокрокових неявних формул порядку, не вищого за 3–4. Підвищення порядку формули, **по-перше**, відчутно ускладнює отримане різницеве рівняння, **по-друге**, зменшує область стійкості одержаного різницевого рівняння. Оптимальним вибором для задач моделювання електроприводів можна вважати неявні формули Адамса другого (*неявна формула трапецій*) і третього порядків.

Реалізація алгоритму зі змінним кроком дає змогу збільшити швидкість моделювання і підвищити точність, але процедура зміни кроку у разі різницевих рівнянь вимагає відчутного ускладнення процедури розрахунку:

1) різницеву схему будують для рівномірної сітки, тому після зміни кроку потрібно виконати інтерполяцію точок для нової сітки різницевої схеми;
2) відсутні чіткі критерії (такі, як, наприклад, для однокрокових формул Фельберґа) для зміни кроку різницевої схеми; практично єдиним способом визначення похибки на кроці для реалізації критерію зміни кроку є екстраполяція за Річардсоном.

3.4. Моделювання електроприводів за допомогою Z-перетворення

Формування різницевих рівнянь за допомогою Z-перетворення є ефективним числово-аналітичним методом, що забезпечує високу точність і швидкість моделювання і поширений за рубежем. Недоліком цього способу є необхідність певної аналітичної роботи у ході формування різницевих рівнянь, відсутність простої оцінки похибки на кроці та проблеми з точністю для складніших систем, які для знаходження дискретних передатних функцій необхідно ділити на ланки, передатні функції яких відповідають табличним. Якщо ж потрібно отримати швидкодійну модель, наприклад, для систем реального часу, то такий підхід є найефективнішим [Г.6].

У цьому розділі не описано теорію та принципи Z-перетворення – для цього є відповідна література [Г.3, Г.6, Г.7], основною його метою є показати, як апарат Z-перетворення та його переваги можуть бути використані для отримання швидких та якісних дискретних моделей систем автоматизованого електропривода. Практично лише за допомогою табличного Z-перетворення можуть бути одержані швидкодійні числові моделі, які є стійкими для будь-якого кроку. Не можуть зменшити цінності використання цього методу і окремі його недоліки:

1) необхідність певних аналітичних перетворень під час формування різницевих рівнянь, що описують модель, але з появою математичних пакетів, що мають засоби символьної математики (наприклад, MathCAD, MATLAB, Mathematica, Maple V, Derive тощо) ця проблема практично вирішена;

2) складність оцінки похибки на кроці розв'язання і, відповідно, реалізації алгоритму автоматичного вибору кроку; практично більшість відомих застосувань використовує фіксований крок розв'язування.

Однією з основних проблем із застосуванням Z-перетворення є отримання дискретної передатної функції *W(z)*, яка описує систему, що моделюється. Вирішення цієї проблеми розглядають далі.

Після одержання дискретної передатної функції постає задача знаходження вихідного сигналу в часовій області *y(t)*. Найзручнішою процедурою для розв'язування такої задачі за допомогою обчислювальної техніки є отримання рекурентної формули, за якою можна обчислювати послідовні значення шуканої вихідної функції. Як це зробити, показано нижче.

Нехай для певної неперервної передатної функції *W(s)* за допомогою аналітичних перетворень отримано дискретну передатну функцію *W(z)*:

$$W(z)=\frac{a_m z^m + a_{m-1}z^{m-1} + \ldots + a_1 z + a_0}{b_n z^n + b_{n-1}z^{n-1} + \ldots + b_1 z + b_0} = \frac{y(z)}{x(z)} \quad \text{для } n \ge m .$$

З дискретної передатної функції для моменту часу t_i одержуємо

$$y_i\left(b_n z^n + b_{n-1}z^{n-1} + \ldots + b_1 z + b_0\right) = x_i\left(a_m z^m + a_{m-1}z^{m-1} + \ldots + a_1 z + a_0\right);$$

після чого ділимо почленно на z^n

$$y_i\left(b_n + b_{n-1}z^{-1} + \ldots + b_1 z^{1-n} + b_0 z^{-n}\right) = x_i\left(a_m z^{m-n} + a_{m-1}z^{m-1-n} + \ldots + a_1 z^{1-n} + a_0 z^{-n}\right).$$

Враховуючи теорему зміщення для Z-перетворення $\left(y_{i-1} = y_i \cdot z^{-1}\right)$, отримуємо

$$y_i b_n + y_{i-1}b_{n-1} + \ldots + y_{i+1-n}b_1 + y_{i-n}b_0 =$$

$$= x_{i+m-n}a_m + x_{i+m-1-n}a_{m-1} + \ldots + x_{i+1-n}a_1 + x_{i-n}a_0 ,$$

звідки знаходимо остаточний вираз

$$y_i = \frac{x_{i+m-n}a_m + x_{i+m-1-n}a_{m-1} + \ldots + x_{i+1-n}a_1 + x_{i-n}a_0 - y_{i-1}b_{n-1} - \ldots - y_{i+1-n}b_1 - y_{i-n}b_0}{b_n}.$$

***Наприклад*:**

$$W(z)=\frac{a_1 z + a_0}{b_2 z^2 + b_1 z + b_0}.$$

Для моменту часу t_i одержуємо

$$y_i\left(b_2z^2+b_1z+b_0\right)=x_i\left(a_1z+a_0\right),$$

далі почленно ділимо на z^2:

$$y_i\left(b_2+b_1z^{-1}+b_0z^{-2}\right)=x_i\left(a_1z^{-1}+a_0z^{-2}\right)\ \Rightarrow$$

$$y_ib_2+y_{i-1}b_1+y_{i-2}b_0=x_{i-1}a_1+x_{i-2}a_0;$$

звідки

$$y_i=\frac{x_{i-1}a_1+x_{i-2}a_0-y_{i-1}b_1-y_{i-2}b_0}{b_2}.$$

Великою перевагою Z-перетворення є те, що з його допомогою, як і за допомогою перетворення Лапласа, для відомого вхідного сигналу і для лінійної системи можна отримати аналітично точний розв'язок для вихідного сигналу. На жаль, така ситуація в моделюванні трапляється рідко (*одним з винятків є, як вже згадувалося вище, знаходження перехідної характеристики лінійної системи – відгуку (реакції) на одиничний стрибкоподібний імпульс, який має аналітичне відображення для Z-перетворення*), тому що в складних системах зі зворотними зв'язками вхідні сигнали внутрішніх блоків часто є сумою сигналів з виходів попередніх блоків та зворотних зв'язків. У такому разі практично неможливо отримати аналітичний вираз перетворення Лапласа для вхідного сигналу окремого блока чи системи, тому реально доводиться застосовувати процедури перетворення вхідного сигналу – його апроксимації простішими функціями, для яких існує перетворення Лапласа [Г.3, Г.6, Г.7] (рис. 3.6).

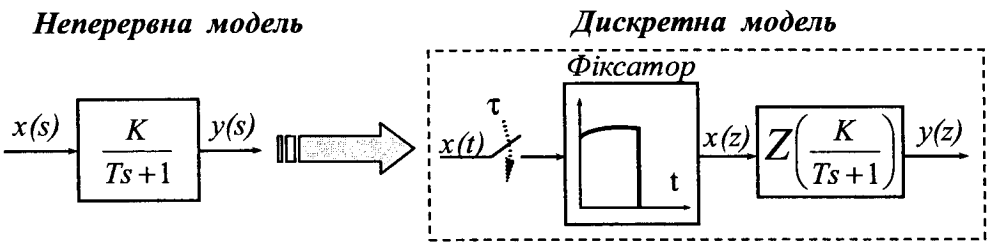


Рис. 3.6. Побудова дискретної моделі

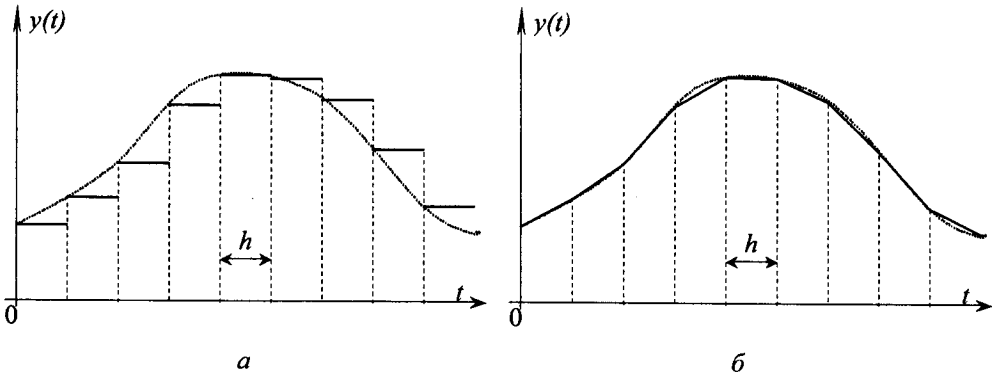
Варто зауважити, що розуміння необхідності перетворення (чи апроксимації) вхідного сигналу та фізичного змісту такого процесу в Z-перетворенні є нетривіальною задачею. Допомогти може усвідомлення того, що знаходження вихідного сигналу системи за допомогою Z-перетворення є ***аналітичним*** *процесом, як і перетворення Лапласа.*

Якщо неможливо отримати аналітичний вираз для сигналу на вході досліджуваного блока, на допомогу приходять пристрої, які в теорії автоматичного керування називають фіксаторами, – на проміжку довжиною τ вони виконують апроксимацію (фіксацію) довільного вхідного сигналу шматками поліномів невисокого порядку (зазвичай, не вищим від першого). У такому разі аналітичним описом вхідного сигналу стає передатна функція відповідного фіксатора. Щоправда, в такому разі цей опис є наближеним, бо поліноміальна апроксимація для довільної кривої виконується лише з певною точністю (наближенням).

Найчастіше для моделювання використовують два види апроксимації вхідного сигналу (апроксимації вищого порядку є значно складнішими і практично не застосовуються):

1) за допомогою полінома нульового порядку (*апроксимація прямокутниками*) – цій апроксимації відповідає фіксатор нульового порядку (рис. 3.7, *а*), який має передатну функцію $\frac{1-e^{-hs}}{s}$;
2) за допомогою полінома першого порядку (*апроксимація трапеціями*) – цій апроксимації відповідає фіксатор першого порядку (рис. 3.7, *б*), який має передатну функцію $\frac{\left(1-e^{-hs}\right)^2 e^{hs}}{hs^2}$.

Треба відзначити, що апроксимація вхідного сигналу трапеціями є точнішою, тому в моделюванні застосовується частіше. У реальних системах керування фізично реалізувати можна тільки апроксимацію прямокутниками [Г.3, Г.7], яка не вимагає інформації про стан системи на наступному, ще не виконаному кроці.



а *б*

Рис. 3.7. Види апроксимації вхідного сигналу: а – прямокутниками; б – трапеціями

Використання таблиць Z-перетворення

Для великої кількості типових лінійних ланок існує пряма відповідність між неперервною передатною функцією (за Лапласом) і дискретною передатною функцією (Z-перетворення). Такі таблиці відповідності можна знайти у будь-якій книзі з теорії дискретних систем керування, але найповніші містяться у [Г.3, Г.7], там подана також ґрунтовна теорія Z-перетворення. У додатку подана таблиця найуживаніших неперервних передатних функцій та відповідних їм дискретних.

Різницеві рівняння, що отримані за допомогою табличного Z-перетворення, стійкі для будь-якого кроку.

Приклад: Знайти за допомогою Z-перетворення перехідну характеристику (реакцію на одиничний стрибок) для аперіодичної ланки з передатною функцією $\frac{1}{Ts+1}$ і сталою часу $T = 0.8$ с.

Така задача може бути розв'язана двома способами: за допомогою використання аналітичного описання вхідного сигналу або ввімкненням на вході аперіодичної ланки фіксатора нульового порядку.

Перший спосіб

Як вже згадувалося раніше, перехідна характеристика системи є її реакцією $y(t)$ на вхідний одиничний стрибкоподібний сигнал $x(t)$, що має відображення за Лапласом $\frac{1}{s}$. Передатна функція аперіодичної ланки має вигляд $\frac{1}{Ts+1}$, тому отримуємо зображення перехідної характеристики аперіодичної ланки $H(s)=\frac{1}{s}\cdot\frac{1}{Ts+1}$. Для побудови рекурентного рівняння знаходимо Z-перетворення для визначеної перехідної характеристики: $Z\left(\frac{1}{s}\cdot\frac{1}{Ts+1}\right)$. За таблицею Z-перетворення (див. додаток) знаходимо відповідну до неперервної дискретну передатну функцію:

$$\frac{a}{s(s+a)} \quad \Rightarrow \quad \frac{y(z)}{x(z)}=\frac{\left(1-e^{-ah}\right)z}{(z-1)\left(z-e^{-ah}\right)}.$$

Підставивши $a=\frac{1}{T}$, отримаємо дискретну передатну функцію моделі

$$\frac{z\left(1-e^{-h/T}\right)}{(z-1)\left(z-e^{-h/T}\right)},$$

звідки

$$y(z)\cdot(z-1)\left(z-e^{-h/T}\right)=x(z)\cdot z\left(1-e^{-h/T}\right).$$

Після розкривання дужок матимемо

$$y(z)\cdot\left(z^2-\left(1+e^{-h/T}\right)z+e^{-h/T}\right)=x(z)\cdot z\left(1-e^{-h/T}\right),$$

а з урахуванням теореми зміщення для Z-перетворення ($y_{i+1} = y_i \cdot z$) одержуємо:

$$y_{i+2} - y_{i+1}\left(1+e^{-h/T}\right)+y_i e^{-h/T} = x_{i+1}\left(1-e^{-h/T}\right),$$

звідси виводимо рекурентне рівняння для розрахунку перехідної характеристики ($x_0 = 1$, у всіх інших випадках $x_i = 0$ – це пояснюється тим, що в момент часу t_0 аналітично заданий вхідний сигнал "активізується") з початковою умовою $y_0 = 0$:

$$y_{i+2} = y_{i+1}\left(1+e^{-h/T}\right)-y_i e^{-h/T}+x_i\left(1-e^{-h/T}\right).$$

Як видно з рекурентної формули, щоб знайти наступне значення змінної, потрібно мати два її попередні значення. Виняток становить обчислення значення першої точки y_1, для якого відоме лише початкове значення y_0. Тому y_1 розраховують за виразом

$$y_1 = -y_0 e^{-h/T}+x_0\left(1-e^{-h/T}\right).$$

Нижче подано фрагменти програм для розрахунку перехідної характеристики за отриманим рекурентним рівнянням. Для їх розуміння треба пам'ятати, що, як вже згадувалося вище, вхідний сигнал подається в момент часу t_0, тому $x_0 = 1$, в усіх інших випадках $x_i = 0$; початкова умова є нульовою: $y_0 = 0$. Враховуючи вищесказане, необхідно зауважити, що лише з навчальною метою в тексті програм використовують такі вирази, як $1\cdot\left(1-e^{-h/T}\right)$ чи $0\cdot\left(1-e^{-h/T}\right)$.

Quick Basic

```
CONST Ta = 0.8     ' Стала часу ланки
CONST h = 0.05     ' Крок виведення результату
CONST Tmax = 2     ' Час розрахунку
Y = 0              ' Нульова початкова умова
Et = EXP(-h/Ta)    ' Допоміжна величина
```

```
Y1 = -Y * Et + 1*(1 - Et)    ' Перший крок
PRINT "t="; h, "Y="; Y1
' Всі наступні кроки
FOR t = h TO Tmax STEP h
   Y2 = Y1*(1 + Et)- Y * Et + 0*(1 - Et)
   PRINT "t="; t, "Y="; Y2
   ' Підготовка до наступного кроку
   Y = Y1
   Y1 = Y2
NEXT t
```

Turbo Pascal

```
const
   Ta = 0.8;       { Стала часу ланки           }
   h = 0.05;       { Крок виведення результату  }
   Tmax = 2.0;     { Час розрахунку             }
var
   t, Y, Y1, Y2, Et : double;
begin
   Y := 0.0;            { Нульова початкова умова }
   Et := EXP(-h/Ta);    { Допоміжна величина }
   t := 0.0;
   Y1 := -Y*Et + 1*(1 - Et);     { Перший крок }
   WriteLn('t = ', h:5:2, '       Y = ', Y1:8:3);
   repeat
      Y2 := Y1*(1 + Et) - Y*Et + 0*(1 - Et);
      t := t + h;
      WriteLn('t = ', t:5:2, '       Y = ', Y2:8:3);
      { Підготовка до наступного кроку }
      Y := Y1;
      Y2 := Y1;
   until t > Tmax;
end.
```

MathCAD

$T_a := 0.8$ *Стала часу ланки*

$h := 0.05$ *Крок розрахунку*

$N := 40$ *Кількість точок графіка* $i := 0 .. N$

$y_0 := 0$ *Нульові початкові умови* $t_i := i \cdot h$

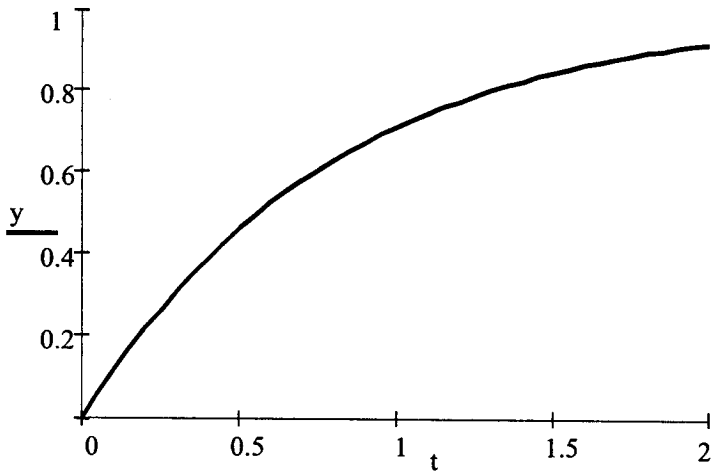
Перший крок

$$y_1 = -y_0 \cdot \exp\left(\frac{-h}{T}\right) + 1 \cdot \left(1 - \exp\left(\frac{-h}{T}\right)\right)$$

Наступні кроки

$$y_{i+2} = y_{i+1} \cdot \left(1 + \exp\left(\frac{-h}{T}\right)\right) - y_i \cdot \exp\left(\frac{-h}{T}\right) + 0 \cdot \left(1 - \exp\left(\frac{-h}{T}\right)\right)$$

Отримані результати



MATLAB

```
Ta=0.8;              % Стала часу ланки
h=0.05;              % Крок виведення результату
Tmax=2;              % Час розрахунку
y(1)=0;              % Нульові початкові умови
t=0:h:Tmax;          % Час
Np=Tmax/h;           % Кількість кроків
Et=exp(-h/Ta);       % Допоміжна величина
y(2)=-y(1)*Et + 1*(1-Et);    % Перший крок
for i=1:Np
   y(i+2)= y(i+1)*(1+Et) - y(i)*Et + 0*(1-Et);
end
% Виведення графіка
plot(t(1:Np), y(1:Np)),grid
```

Другий спосіб

Передатна функція моделі аперіодичної ланки і фіксатора нульового порядку на її вході матиме вигляд: $\frac{1-e^{-sh}}{s} \cdot \frac{1}{Ts+1}$. Для побудови рекурентного рівняння знаходимо Z-перетворення для отриманого виразу:

$$Z\left(\frac{1-e^{-sh}}{s} \cdot \frac{1}{Ts+1}\right)=\left(1-z^{-1}\right)\cdot Z\left(\frac{1}{s(Ts+1)}\right).$$

За таблицею Z-перетворення (табл. Д.3) знаходимо відповідну до неперервної дискретну передатну функцію:

$$\frac{a}{s(s+a)} \quad \Rightarrow \quad \frac{\left(1-e^{-ah}\right)z}{(z-1)\left(z-e^{-ah}\right)}.$$

Підставивши $a = \frac{1}{T}$, одержуємо дискретну передатну функцію моделі

$$\left(1 - z^{-1}\right) \frac{z\left(1 - e^{-h/T}\right)}{(z-1)\left(z - e^{-h/T}\right)} \quad \Rightarrow \quad \frac{1 - e^{-h/T}}{z - e^{-h/T}},$$

звідки $y_i\left(z - e^{-h/T}\right) = x_i\left(1 - e^{-h/T}\right)$, звідси знаходимо рекурентне рівняння для розрахунку перехідної характеристики ($x_i = 1$):

$$y_{i+1} = y_i e^{-h/T} + x_i\left(1 - e^{-h/T}\right).$$

У такому разі через наявність на вході фіксатора нульового порядку всі значення вхідного сигналу є одиничними ($x_i = 1$) з нульовими початковими умовами ($y_0 = 0$).

Нижче подано фрагменти програм для розрахунку перехідної характеристики за отриманим рекурентним рівнянням.

Quick Basic

```
CONST Ta = 0.8     ' Стала часу ланки
CONST h = 0.05     ' Крок виведення результату
CONST Tmax = 2     ' Час розрахунку
Y = 0              ' Нульові початкові умови
Et = EXP(-h/Ta)    ' Допоміжна величина

FOR t = 0 TO Tmax STEP h
   Y = Y * Et + (1 - Et)
   PRINT "t="; t, "Y="; Y
NEXT t
```

Turbo Pascal

```
const
   Ta = 0.8 ;     { Стала часу ланки          }
   h = 0.05 ;     { Крок виведення результату }
   Tmax = 2.0 ;   { Час розрахунку            }
var
   t, Y, Et : double ;
begin
   Y := 0.0 ;              { Нульові початкові умови }
   Et := EXP(-h/Ta) ;      { Допоміжна величина }
   t := 0.0 ;
   repeat
      Y := Y*Et + (1 - Et) ;
      WriteLn('t = ', t:5:2, '       Y = ', Y:8:3) ;
      t := t + h ;
   until t > Tmax ;
end .
```

MathCAD

$T_a := 0.8$ *Стала часу ланки*

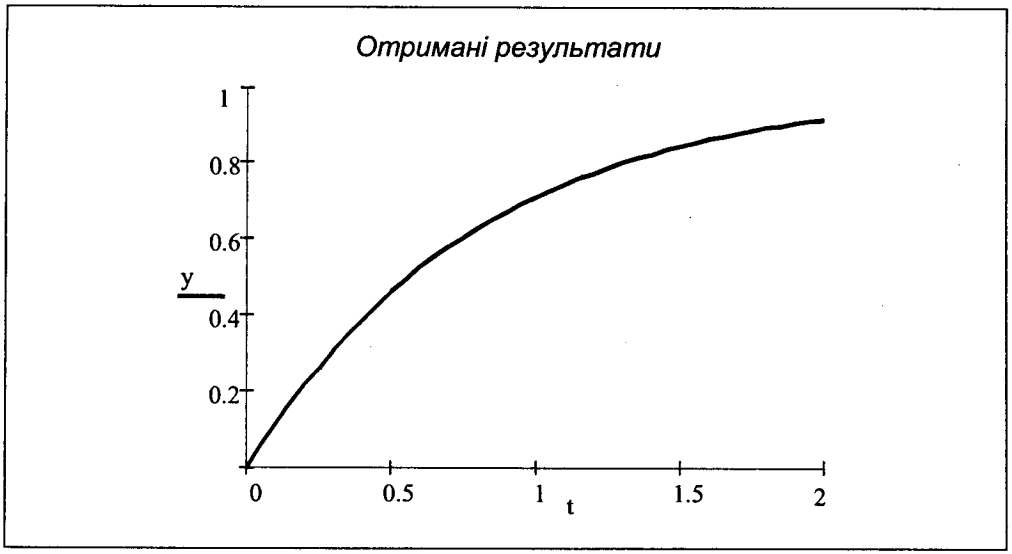
$h := 0.05$ *Крок розрахунку*

$N := 40$ *Кількість точок графіка* $i := 1 .. N$

$y_0 := 0$ *Нульові початкові умови*

$$y_i := y_{i-1} \cdot \exp\left(\frac{-h}{T}\right) + \left(1 - \exp\left(\frac{-h}{T}\right)\right)$$

$i := 0 .. N$ $t_i := i \cdot h$



MATLAB

```
Ta=0.8;              % Стала часу ланки
h=0.05;              % Крок виведення результату
Tmax=2;              % Час розрахунку
y(1)=0;              % Нульові початкові умови
t=0:h:Tmax;          % Час
Et=exp(-h/Ta);       % Допоміжна величина
for i=2:(Tmax/h+1)
   y(i)=y(i-1)*Et+(1-Et);
end
% Виведення графіка
plot(t,y),grid
```

Може скластися враження, що в другому випадку і рекурентне рівняння, і складені за ним програми є простішими, хоча результат отримуємо аналогічний. Треба відзначати, що це справедливо лише для вхідного стрибкоподібного сигналу, а для сигналів іншого типу другий спосіб даватиме нижчу точність внаслідок перетворення сигналу фіксатором нульового порядку.

Від авторів

На жаль, обмежений обсяг навчального посібника не дає змоги висвітлити всі сторони як теорії, так і практики Z-перетворення. Цей надзвичайно потужний апарат має багато тонкощів, розкриття яких потребує значного часу, а використання – глибокого розуміння як фізики модельованих процесів, так і теорії цифрового перетворення сигналів.

Для додаткового ознайомлення із Z-перетворенням автори рекомендують класичні роботи його творців Е. Джурі [Г.3] та Ю. Ту [Г.7]. Практичні аспекти застосування Z-перетворення можна знайти у роботі Дж. Сміта [Г.6].

Як підсумок нижче показано приклади синтезу рекурентних рівнянь для моделювання аперіодичної ланки першого порядку зі сталою часу T і коефіцієнтом підсилення K (*аналог звичайного диференціального рівняння першого порядку*) за допомогою двох видів апроксимації вхідного сигналу і використання таблиць Z-перетворення (див. рис. 3.6).

Фіксатор нульового порядку

Виконаємо Z-перетворення для фіксатора нульового порядку і аперіодичної ланки:

$$Z\left(\left(\frac{1-e^{-sh}}{s}\right)\cdot\left(\frac{K}{Ts+1}\right)\right) = (1-z^{-1})Z\left(\frac{K}{s(Ts+1)}\right) =$$

$$=(1-z^{-1})\cdot K\cdot\frac{\left(1-e^{-ah}\right)z}{(z-1)\left(z-e^{-ah}\right)} = \frac{\left(1-e^{-\frac{h}{\mathrm{T}}}\right)K}{z-e^{-\frac{h}{\mathrm{T}}}},$$

за дискретною передатною функцією отримуємо рекурентне рівняння для моделювання:

$$y_{i+1} = y_i\cdot e^{-\frac{h}{\mathrm{T}}} + \left(1-e^{-\frac{h}{\mathrm{T}}}\right)\cdot K\cdot x_i. \qquad (3.9)$$

Фіксатор першого порядку

Виконаємо Z-перетворення для фіксатора першого порядку і аперіодичної ланки:

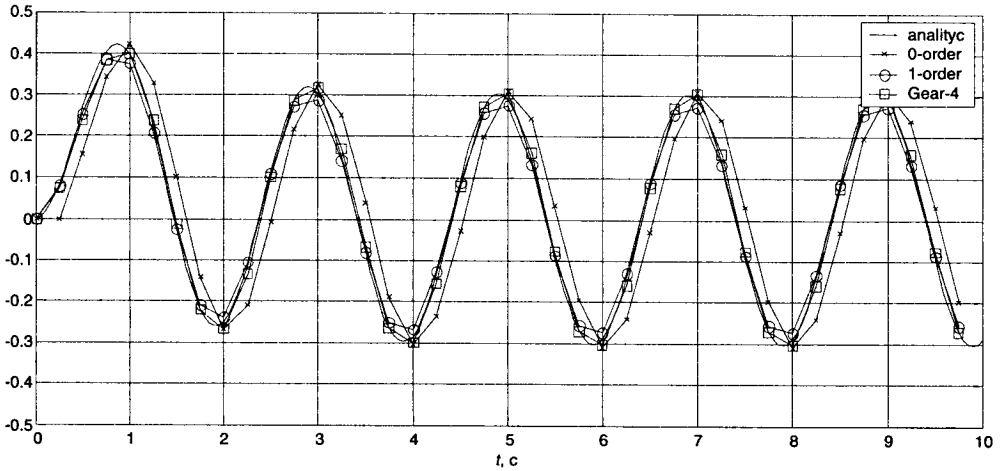
$$Z\left(\left(\frac{\left(1-e^{-sh}\right)^2 e^{sh}}{hs^2}\right)\cdot\left(\frac{K}{Ts+1}\right)\right)=\frac{z\left(1-z^{-1}\right)^2}{h}\cdot Z\left(\frac{K}{s^2(Ts+1)}\right)\frac{z\left(1-z^{-1}\right)^2}{h}\cdot K\times$$

$$\times\left(\frac{hz}{(z-1)^2}-\frac{z\cdot\left(1-e^{-\frac{h}{\mathrm{T}}}\right)}{\frac{1}{T}(z-1)\left(z-e^{-\frac{h}{\mathrm{T}}}\right)}\right)=\frac{z-e^{-\frac{h}{\mathrm{T}}}-\frac{T}{h}\left(1-e^{-\frac{h}{\mathrm{T}}}\right)(z-1)}{z-e^{-\frac{h}{\mathrm{T}}}}K,$$

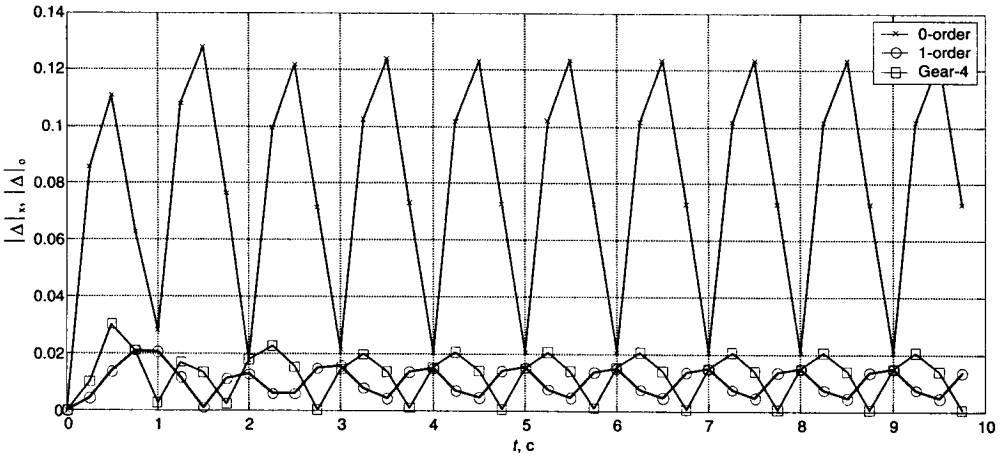
за дискретною передатною функцією одержуємо рекурентне рівняння для моделювання:

$$y_{i+1}=y_i\cdot e^{-\frac{h}{\mathrm{T}}}+K\cdot\left(x_{i+1}-x_i\cdot e^{-\frac{h}{\mathrm{T}}}\right)-K\cdot\frac{T}{h}\cdot(x_{i+1}-x_i)\cdot\left(1-e^{-\frac{h}{\mathrm{T}}}\right). \qquad (3.10)$$

Отримані рекурентні формули забезпечують стійкий розв'язок для **будь-якого** кроку, причому перша за своєю точністю не поступається класичним формулам інтегрування звичайних диференціальних рівнянь не нижче від другого порядку, а друга – формулам інтегрування третього–четвертого порядків. Результати застосування цих формул показано на рис. 3.8. У цьому числовому експерименті на вхід аперіодичної ланки першого порядку з коефіцієнтом підсилення $K=1$ і сталою часу $T=1$ с подано сигнал – одиничну синусоїду з кутовою частотою π рад./с. Вихідну реакцію для кроку моделювання $h=0.25$ с за формулами (3.9) і (3.10) (позначено на графіку відповідно хрестиком –×– і кружечком –о–) порівнюють з аналітичним розв'язком (суцільна лінія —) і розв'язком, отриманим за допомогою формули диференціювання назад (*Gear*) четвертого порядку (позначено квадратиком –□–).



а



б

Рис. 3.8. Результати числового експерименту з цифровими моделями аперіодичної ланки: а – графік реакції моделей аперіодичної ланки на синусоїдний вхідний сигнал; б – графік абсолютних похибок цифрових моделей аперіодичної ланки

Як видно з графіка вихідних координат цифрових моделей (рис. 3.8, *а*) і з графіка абсолютних похибок стосовно точного розв'язку (рис. 3.8, *б*), застосування Z-перетворення може бути доволі ефективним.

Метод підстановки нулів та полюсів

Вдосконалення математичних пакетів для персональних комп'ютерів, що підтримують символьну (аналітичну) математику, наприклад, MathCAD, MATLAB, Derive, Maple V тощо, дало змогу спростити процедуру отримання дискретних передатних функцій для систем, що описуються неперервними передатними функціями за допомогою перетворення Лапласа. Для цього використовують метод підстановки нулів та полюсів передатної функції, короткий алгоритм якого подано нижче.

1. Знайти всі нулі Z_i ($i = 1 \ldots m$) і полюси P_j ($j = 1 \ldots n$) неперервної передатної функції моделі досліджуваної системи, для якої порядок чисельника становить m, а порядок знаменника – n, причому $n \geq m$.
2. З використанням відомого співвідношення $z = e^{sh}$, де h – крок часової дискретизації, перейти до дискретної передатної функції, записаної в термінах нулів і полюсів:
   - дискретні нулі подають виразом $Zd_i = e^{z_i h}$, де $i = 1 \ldots m$;
   - дискретні записують виразом $Pd_j = e^{p_j h}$, де $j = 1 \ldots n$.

   Узагальнити цей підхід можна так:

$$\frac{\sum_{i=1}^{m}(s - Z_i)}{\sum_{j=1}^{n}(s - P_j)} \Rightarrow \frac{\sum_{i=1}^{m}\left(z - e^{Z_i \cdot h}\right)}{\sum_{j=1}^{n}\left(z - e^{P_j \cdot h}\right)}.$$

Варто зауважити, що цей метод дає аналітично точний результат для лінійних і лінеаризованих моделей систем і відомих вхідних сигналів. Для довільних вхідних сигналів на вході моделі знов-таки повинен визначатись фіксатор нульового чи першого порядку, його наявність треба врахувати у передатній функції моделі системи.

Використання Z-форм (підстановок)

Якщо у таблиці Z-перетворення немає відповідної дискретної передатної функції, у техніці прийнято застосовувати дискретні апроксимації оператора Лапласа [Г.3, Г.6, Г.7]. Для цього у неперервну передатну функцію замість оператора Лапласа підставляється отримана тим чи іншим методом його дискретна апроксимація (див. табл. 3.1), у результаті чого одержується дискретна передатна функція, за якою формують відповідне рекурентне рівняння. Вважають [Г.7, Г.9], що найточнішою підстановкою є Z-форма. У складних випадках для алгебричних перетворень можна використати математичні пакети, які спрощують дослідникові отримання кінцевих рівнянь.

Таблиця 3.1

Дискретні апроксимації оператора Лапласа

Аналоговий інтегратор	Табличне Z-перетворення	Z-форма	Метод Тастіна
s^{-1}	$\frac{hz}{z-1}$	$\frac{h}{2}\frac{z+1}{z-1}$	$\frac{h}{2}\frac{z+1}{z-1}$
s^{-2}	$\frac{h^2 z}{(z-1)^2}$	$\frac{h^2}{12}\frac{z^2+10z+1}{(z-1)^2}$	$\left(\frac{h}{2}\frac{z+1}{z-1}\right)^2$
s^{-3}	$\frac{h^3}{2}\frac{z(z+1)}{(z-1)^3}$	$\frac{h^3}{2}\frac{z(z+1)}{(z-1)^3}$	$\left(\frac{h}{2}\frac{z+1}{z-1}\right)^3$
s^{-4}	$\frac{h^4}{6}\frac{z\left(z^2+4z+1\right)}{(z-1)^4}$	$\frac{h^4}{6}\frac{z\left(z^2+4z+1\right)}{(z-1)^4}$	$\left(\frac{h}{2}\frac{z+1}{z-1}\right)^4$
s^{-5}	$\frac{h^5}{24}\frac{z\left(z^3+11z^2+11z+1\right)}{(z-1)^5}$	$\frac{h^5}{24}\frac{z\left(z^3+11z^2+11z+1\right)}{(z-1)^5}$	$\left(\frac{h}{2}\frac{z+1}{z-1}\right)^5$

Практичне використання таких підстановок розглянуто у [Г.6, Г.9]. Рекомендоване значення кроку становить $h \le T_{\min}/(10 \dots 15)$ [Г.6], де T_{min} – найменша стала часу системи, але практика показує, що в багатьох випадках без відчутної втрати точності крок можна збільшити до $h \le T_{\min}/(2 \dots 5)$.

Приклад: Створити математичну і цифрову моделі двигуна постійного струму незалежного збудження зі сталим потоком збудження для дослідження його динамічних режимів для стрибкоподібної зміни напруги на якорі (рис. 3.9) з використанням Z-перетворення за допомогою методу підстановок.

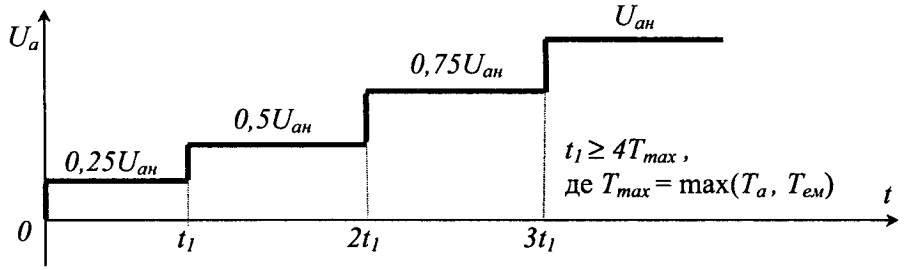


Рис. 3.9. Приклад графіка зміни напруги на якорі двигуна

Пояснення: Моделі двигуна постійного струму відповідає відома структура і відповідна система диференціальних рівнянь:

$$\begin{cases} T_a \dfrac{di_a}{dt} + i_a = \dfrac{U_a - C \cdot \omega}{R_a} ; \\ J \dfrac{d\omega}{dt} = C \cdot (i_a - I_c) ; \end{cases}$$

або в операторній формі

$$\begin{cases} i_a(s) = \dfrac{(U_a - C \cdot \omega(s))/R_a}{T_a \cdot s + 1} ; \\ \omega(s) = \dfrac{C}{J \cdot s}(i_a(s) - I_c) . \end{cases}$$

Виконуючи підстановку $\frac{1}{s}=\frac{h}{2}\frac{z+1}{z-1}$, отримаємо:

$$\begin{cases} i_a(z)=\dfrac{\left(U_a-C\cdot\omega(z)\right)/R_a}{\dfrac{2T_a}{h}\left(\dfrac{z-1}{z+1}\right)+1}; \\ \omega(z)=\dfrac{h\cdot C}{2J}\left(\dfrac{z+1}{z-1}\right)\left(i_a(z)-I_c\right). \end{cases}$$

Після відповідних алгебричних перетворень з урахуванням теореми зміщення для Z-перетворення ($y_{i-1}=y_i\cdot z^{-1}$) перейдемо до системи рекурентних рівнянь:

$$\begin{cases} i_{a_{i+1}}=i_{a_i}\dfrac{2T_a-h}{2T_a+h}+\dfrac{h}{R_a\cdot(2T_a+h)}\left(2U_{a_i}-C\cdot(\omega_{i+1}+\omega_i)\right); \\ \omega_{i+1}=\omega_i+\dfrac{C\cdot h\cdot\left(i_{a_{i+1}}+i_{a_i}-2I_c\right)}{2J}. \end{cases}$$

Під час розрахунку $i_{a_{i+1}}$ значення ω_{i+1} ще невідоме і можливі два варіанти:

1) розглянути отриману систему як систему лінійних алгебричних рівнянь, звідки знайти шукані значення $i_{a_{i+1}}$ та ω_{i+1};
2) з урахуванням того, що швидкість змінюється повільніше за струм, можна використати замість значення ω_{i+1} його попереднє значення – ω_i (*це не зовсім точно, але в багатьох випадках допустимо*).

Якщо використати другий, простіший варіант, то остаточно система рекурентних рівнянь запишеться так:

$$\begin{cases} i_{a_{i+1}}=i_{a_i}\dfrac{2T_a-h}{2T_a+h}+\dfrac{2h}{R_a\cdot(2T_a+h)}\left(U_{ai}-C\cdot\omega_i\right); \\ \omega_{i+1}=\omega_i+\dfrac{C\cdot h\cdot\left(i_{a_{i+1}}+i_{a_i}-2I_c\right)}{2J}. \end{cases}$$

Нижче пропонуються варіанти програм розв'язування цієї системи для конкретного двигуна і отримані результати (рис. 3.9).

Quick Basic

```
'-------------------------------------------------------------
'   Програма розрахунку запуску двигуна постійного струму
'   для стрибкоподібної зміни напруги на якорі.
'-------------------------------------------------------------
CONST Tmax = 5              ' Максимальний час розрахунку
CONST Xmin = 0              ' Ліва межа графіка
CONST Xmax = Tmax           ' Права межа графіка
CONST Ymin = 0              ' Нижня межа графіка
CONST Ymax = 200            ' Верхня межа графіка
CONST h = .01               ' Крок моделювання

CONST Uanom = 220           ' Номінальна напруга на якорі
CONST C = 2.5               ' Стала двигуна
CONST Ra = .21              ' Сумарний опір якірного кола
CONST Ta = .05              ' Стала часу якірного кола
CONST J = 2.9               ' Сумарний момент інерції привода
CONST Ic = 20!              ' Статичний струм якоря

Ia = 0                      '
Ia.old = 0                  ' Початкові умови
w = 0                       '

FOR t = 0 TO Tmax STEP h
   ' Використовується підпрограма виведення графіка
   CALL Graphic(9, t, Ia)        ' Виведення графіка струму якоря
   CALL Graphic(10, t, 2 * w)    ' Виведення графіка швидкості

   ' Формування напруги на якорі
   Ua = .25 * Uanom
   IF t > 1.25 THEN Ua = .5 * Uanom
   IF t > 2.5 THEN Ua = .75 * Uanom
   IF t > 3.75 THEN Ua = Uanom

   ' Модель якірного кола. Розрахунок струму якоря
   Ia.old = Ia
   Ia = (Ia*(2*Ta - h) + 2*h*(Ua - C*w)/Ra)/(2*Ta + h)
```

```
    ' Опис механічної частини привода: розрахунок швидкості
    w = w + C * h * (Ia + Ia.old - 2 * Ic) / J / 2

NEXT t
END
```

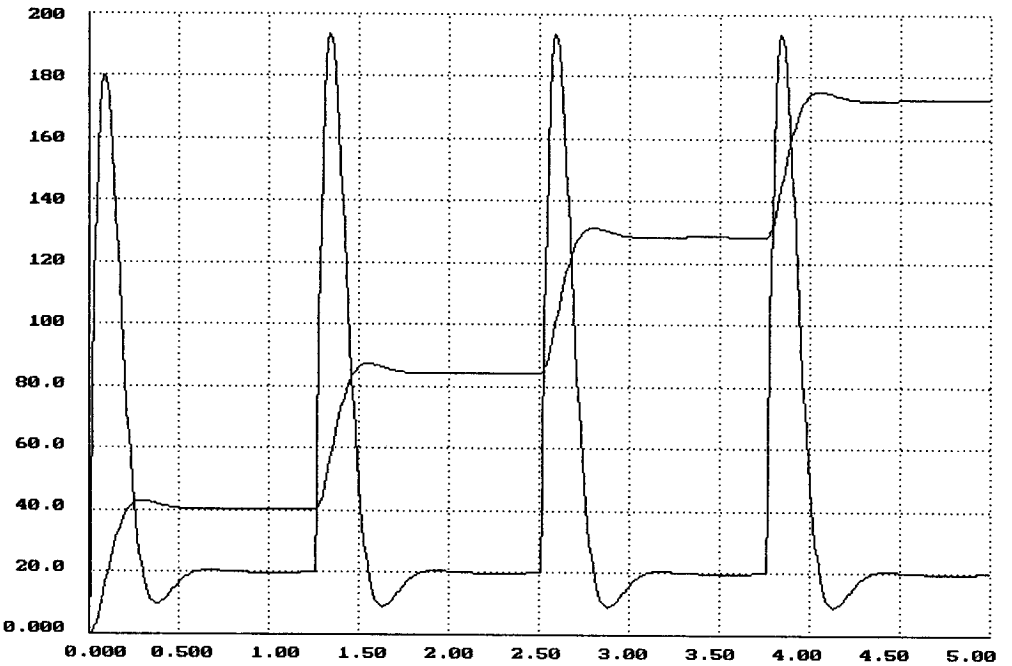


Рис. 3.10. Графіки режиму запуску двигуна

Turbo Pascal

```
{ ------------------------------------------------------------ }
{   Програма розрахунку запуску двигуна постійного струму     }
{   для стрибкоподібної зміни напруги на якорі.               }
{ ------------------------------------------------------------ }
```

```
uses
   Plot ;
const
   Tmax = 5.0 ;          { Максимальний час розрахунку }
   Ymin = 0.0 ;          { Нижня межа графіка          }
   Ymax = 200.0 ;        { Верхня межа графіка         }
   h = 0.01 ;            { Крок моделювання            }

   Uanom = 220.0 ;       { Номінальна напруга на якорі }
   C = 2.5 ;             { Стала двигуна               }
   Ra = 0.21 ;           { Сумарний опір якірного кола }
   Ta = 0.05 ;           { Стала часу якірного кола    }
   J = 2.9 ;             { Сумарний момент інерції привода }
   Ic = 20.0 ;           { Статичний струм якоря       }
var
   t, Ua, Ia, Ia_old, w : double ;
begin
   Ia := 0.0 ;              {                        }
   Ia_old := 0.0 ;          { Початкові умови        }
   w := 0.0 ;               {                        }
   t := 0.0 ;
   { ======================================== }
   {         Підготовка до виведення графіка    }
   { ======================================== }
    InitGraphic(0, Tmax, Ymin, Ymax, 1.0, 20.0) ;
   { -------   Основний цикл програми   ------- }
   repeat
      { Використовується підпрограма виведення графіка }
      Graphic(9, t, Ia) ;     { Виведення графіка струму якоря }
      Graphic(10, t, 2 * w); { Виведення графіка швидкості  }

      { Формування напруги на якорі }
      Ua := 0.25 * Uanom ;
      if t > 1.25 then Ua := 0.5 * Uanom ;
      if t > 2.5 then Ua := 0.75 * Uanom ;
      if t > 3.75 then Ua := Uanom ;

      { Модель якірного кола }
      Ia_old := Ia ;
      Ia := (Ia*(2*Ta - h) + 2*h*(Ua - C*w)/Ra)/(2*Ta + h) ;

      { Опис механічної частини привода: розрахунок швидкості }
      w := w + C * h * (Ia + Ia_old - 2 * Ic) / J / 2 ;
```

```
      t := t + h ;
   until t > Tmax ;
end .
```

MathCAD

$T_{max} := 5$ *Час розв'язування*

$h := 0.01$ *Крок моделювання*

$U_{nom} := 220$ *Номінальна напруга якоря*

$C := 2.5$ *Стала двигуна*

$R_a := 0.21$ *Сумарний опір якоря*

$T_a := 0.05$ *Стала часу якоря*

$J := 2.9$ *Сумарний момент інерції*

$I_c := 20$ *Статичний струм*

$$N := \frac{T_{max}}{h}$$

Кількість точок на графіку

$i := 0 .. N$ *Час* $t_i := i \cdot h$

$Ia_0 := 0$ $\omega_0 := 0$ *Початкові умови*

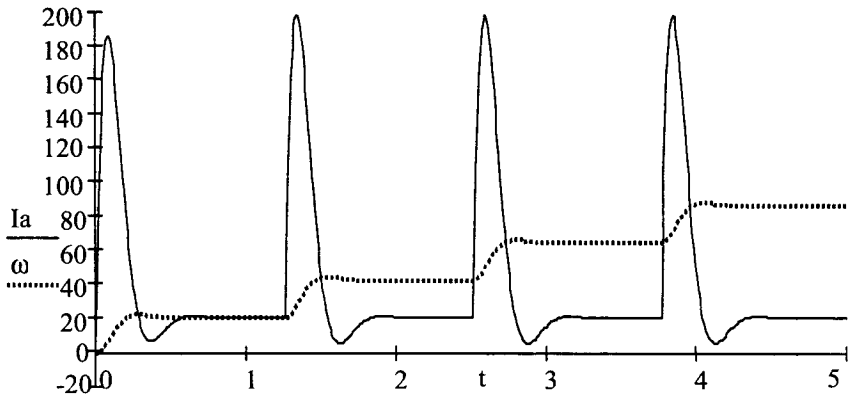
Формування напруги завдання:

$$U_a(t) := \begin{cases} 0.5 \cdot Ua_{nom} & \text{if } t > 1.25 \\ 0.75 \cdot Ua_{nom} & \text{if } t > 2.5 \\ Ua_{nom} & \text{if } t > 3.75 \\ 0.25 \cdot Ua_{nom} & \text{otherwise} \end{cases}$$

Розрахунок:

$$\begin{bmatrix} Ia_{i+1} \\ \omega_{i+1} \end{bmatrix} := \begin{bmatrix} Ia_i \cdot \frac{2 \cdot T_a - h}{2 \cdot T_a + h} + \frac{2 \cdot h}{R_a \cdot (2 \cdot T_a + h)} \cdot \left(U_a(t_i) - C \cdot \omega_i \right) \\ \omega_i + \frac{h \cdot C \cdot (Ia_i - I_c)}{J} \end{bmatrix}$$

Виведення результату:



MATLAB

```
%-------------------------------------------------------------
%   Програма розрахунку запуску двигуна постійного струму
%   для стрибкоподібної зміни напруги на якорі.
%-------------------------------------------------------------
Tmax = 5;              % Максимальний час розрахунку
h = 0.01;              % Крок моделювання
t = 0:h:Tmax;          % Час
N = Tmax/h+1;          % Кількість точок на графіку
```

```
Uanom = 220;          % Номінальна напруга на якорі
C = 2.5;              % Стала двигуна
Ra = 0.21;            % Сумарний опір якірного кола
Ta = 0.05;            % Стала часу якірного кола
J = 2.9;              % Сумарний момент інерції привода
Ic = 20;              % Статичний струм якоря

Ia(1) = 0;            % Початкові умови
w(1) = 0;

for i = 2:N
   % Формування напруги на якорі
   Ua = 0.25 * Uanom;
   if t(i) > 1.25
      Ua = .5 * Uanom;
   end
   if t(i) > 2.5
      Ua = .75 * Uanom;
   end
   if t(i) > 3.75
      Ua = Uanom;
   end

   % Модель якірного кола. Розрахунок струму якоря
   Ia(i) = (Ia(i-1)*(2*Ta-h) + 2*h*(Ua-C*w(i-1))/Ra)/(2*Ta+h);

   % Опис механічної частини привода: розрахунок швидкості
   w(i) = w(i-1) + C*h*(Ia(i)+Ia(i-1)-2*Ic)/J/2;
end
plot(t, w, t, Ia), grid
```

Застосування різницевих рівнянь, виведених на підставі Z-перетворення, дає змогу отримати порівняно прості й водночас швидкодійні моделі електроприводів. Такі моделі часто простіші та структурно зрозуміліші, ніж подані системою диференціальних рівнянь у нормальній формі Коші.

Вибір кроку

Вибір кроку для різницевих рівнянь, що отримані за допомогою Z-перетворення, повністю відповідає засадам, викладеним у п. "***Вибір кроку***"

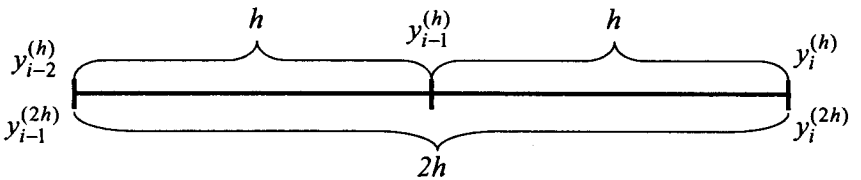
розд. 3.2. У зв'язку з наявною різницевою схемою, що передбачає рівновіддалені точки, найчастіше застосовують розв'язок з фіксованим кроком, у такому разі слід користуватися методикою з вказаного розділу.

Реалізація алгоритму зі змінним кроком, як вже раніше згадувалося, дає змогу підвищити швидкість моделювання і точність, але процедура зміни кроку для різницевих рівнянь, що отримані за допомогою Z-перетворення, вимагає відчутного ускладнення процедури розрахунку:

1) різницеву схему будують для рівномірної сітки, тому, якщо використовується не одне, а декілька попередніх значень змінної, після зміни кроку необхідна інтерполяція точок для нової сітки різницевої схеми;
2) відсутні чіткі критерії для зміни кроку розрахунку, практично єдиним способом оцінки похибки на кроці є екстраполяція за Річардсоном [В.6].

Підвищення точності отриманих рекурентних формул за допомогою екстраполяції за Річардсоном як корисний побічний ефект дає змогу одержати вираз для оцінки похибки (див. розд. 3.2). Найпростіший варіант реалізації такого способу для Z-перетворення є з використанням схем з одиничним і подвійним кроками розв'язування, який описано нижче з позначеннями за допомогою верхніх індексів: $^{(h)}$ – для одиничного кроку, $^{(2h)}$ – для подвійного кроку:

1) знаходять i-те значення змінної для двох послідовних кроків розміром h – $y_i^{(h)}$;
2) знаходять i-те значення змінної для одного подвійного кроку розміром $2h$ – $y_i^{(2h)}$;



3) розраховують уточнене значення змінної за формулою $y_i = y_i^{(h)} + \dfrac{y_i^{(h)} - y_i^{(2h)}}{2^p - 1}$,

де p – порядок числового методу інтегрування, що відповідає використаному фіксатору:

- фіксатор нульового порядку відповідає інтегратору першого порядку ($p = 1$ – явний метод прямокутників або Ейлера);
- фіксатор першого порядку – інтегратору другого порядку ($p = 2$ – неявний метод трапецій).

Крім того, величина $\frac{y_i^{(h)} - y_i^{(2h)}}{2^p - 1}$ є оцінкою похибки на кожному виконаному кроці. На підставі такої оцінки нескладно розробити алгоритм з автоматичним вибором кроку розв'язування.

Уточнення за допомогою екстраполяції за Річардсоном забезпечує підвищення порядку точності розрахованої змінної: для фіксатора нульового порядку – другий порядок точності (*тобто якщо зменшити крок вдвічі, то похибка зменшується в чотири рази*), для фіксатора першого порядку – четвертий порядок точності (*тобто зменшення кроку вдвічі зменшує похибку в шістнадцять разів*) [В.6]. Ефективність цього підходу підтверджена числовими експериментами. Ще одним позитивним наслідком використання екстраполяції за Річардсоном є придатність отриманих рекурентних формул для розв'язування нелінійних моделей [В.6].

Нижче наведено приклад застосування цього підходу з використанням Z-перетворення та екстраполяції за Річардсоном для отримання рекурентних рівнянь для моделювання аперіодичної ланки першого порядку (аналог звичайного диференціального рівняння першого порядку) за допомогою апроксимації вхідного сигналу фіксатором нульового порядку.

За відповідними таблицями (наприклад, [Г.3, Г.6, Г.7]) виконаємо Z-перетворення для фіксатора нульового порядку і аперіодичної ланки:

$$Z\left(\left(\frac{1-e^{-sh}}{s}\right)\cdot\left(\frac{K}{Ts+1}\right)\right) \Rightarrow \frac{(1-e^{-\frac{h}{T}})K}{z-e^{-\frac{h}{T}}},$$

звідки за дискретною передатною функцією одержуємо рекурентне рівняння для моделювання:

$$y_{i+1} = y_i \cdot e^{-\frac{h}{T}} + (1 - e^{-\frac{h}{T}}) \cdot K \cdot x_i .$$

Отримане рівняння є базовим для уточнення за допомогою екстраполяції за Річардсоном. Реалізація процедури уточнення для i+1-ї точки матиме такий вигляд:

1) Знаходять i+1-те значення змінної для одиничного кроку h:

$$y_{i+1}^{(h)} = y_i \cdot e^{-\frac{h}{T}} + (1 - e^{-\frac{h}{T}}) \cdot K \cdot x_i .$$

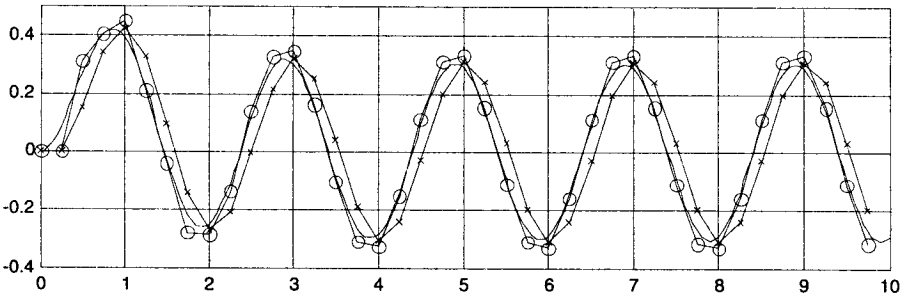
2) Встановлюють i+1-те значення змінної для подвійного кроку $2h$:

$$y_{i+1}^{(2h)} = y_{i-1} \cdot e^{-\frac{2h}{T}} + (1 - e^{-\frac{2h}{T}}) \cdot K \cdot x_{i-1} .$$

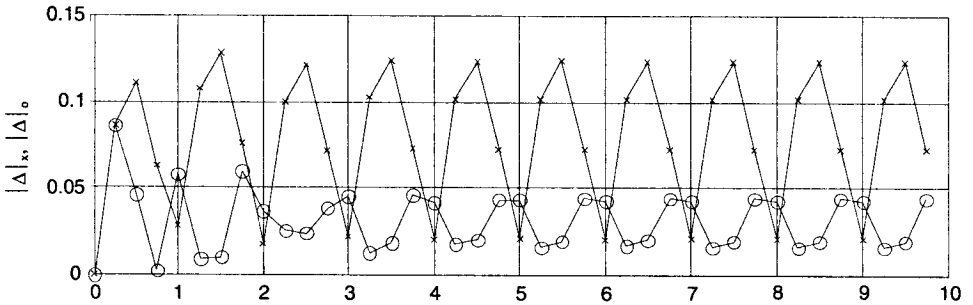
3) Обчислюють уточнене значення змінної: $y_{i+1} = y_{i+1}^{(h)} + \frac{y_{i+1}^{(h)} - y_{i+1}^{(2h)}}{2^1 - 1}$. Після спрощень отримуємо:

$$y_{i+1} = y_{i-1} \cdot e^{-\frac{2h}{T}} + (1 - e^{-\frac{h}{T}}) \cdot K \cdot (2x_i - x_{i-1} \cdot (1 - e^{-\frac{h}{T}})) .$$

Ефект від використання екстраполяції за Річардсоном можна показати на прикладі дослідження реакції аперіодичної ланки зі сталою часу $T = 1$ с і одиничним коефіцієнтом підсилення $K = 1$ (передатна функція $W(s) = \frac{1}{s+1}$) на одиничний синусоїдний вхідний сигнал з частотою π с$^{-1}$ (на такому сигналі найкраще видно похибки від дії фіксатора нульового порядку) – відповідає диференціальному рівнянню $\frac{dy}{dt} + y = \sin(\pi t)$. Результати моделювання з кроком $h = 0{,}2$ с для двох типів моделювальних рекурентних формул показано на рис. 3.10, *а*, а на рис. 3. 10, *б* наведено графіки абсолютних похибок Δ_x і Δ_o стосовно аналітичного розв'язку для обох моделювальних виразів.



а



б

Рис. 3.11. Результати моделювання (а) і графіки абсолютних похибок (б) для різних цифрових моделей аперіодичної ланки:
—— – аналітичний розв'язок; –x– – фіксатор нульового порядку;
–o– – екстраполяція за Річардсоном

Отже, екстраполяція за Річардсоном дає змогу відчутно підвищити точність цифрових моделей, що отримані на основі Z-перетворення.

4 ПРАКТИЧНІ ЗАДАЧІ МОДЕЛЮВАННЯ ЕЛЕКТРОПРИВОДІВ

У цьому розділі подано низку прикладів розрахунку динаміки електроприводів із застосуванням як середовищ програмування (Microsoft Quick Basic, Turbo Pascal), так і математичних пакетів (MathCAD, MATLAB). Пропоновані приклади можуть бути використані як для демонстрації чи пояснення тих чи інших особливостей мови програмування, так і для використання як основи (шаблона) для створення діючих програм моделювання реальних систем електропривода. Автори наголошують, що наведені приклади є нескладними, бо подані з навчальною метою: прості приклади легше сприймаються та аналізуються, а на їх підставі можна створити, якщо необхідно, складнішу модель.

Під час роботи в середовищі MATLAB треба враховувати проблеми з підтримкою кирилиці у цьому пакеті, зокрема, у текстах програм не повинно бути малої букви "я" – замість неї необхідно використовувати велику літеру "Я", що і зроблено в пропонованих прикладах.

4.1. Електроприводи постійного струму

Форсоване збудження генератора постійного струму

Розрахувати перехідні процеси режиму форсованого збудження генератора постійного струму з урахуванням індуктивності якірного кола (див. рис. 4.1). Для розв'язання використати різницеві рівняння на підставі неявного методу Ейлера (див. розд. 3): $y_{i+1} = y_i + hy'_{i+1}$ з кроком h = 0.002 с для часу інтегрування T_{max} = 2.5 с.

Задано: $U_{dн}$ = 220 В – номінальне збудження генератора;
$U_н$ = 460 В – номінальна напруга якоря генератора;
K_G = 2.09 – коефіцієнт підсилення генератора;

$K_ф = U_d/U_{dн} = 3$ – коефіцієнт форсування;
$R_a = 0.15$ Ом – опір якоря генератора;
$R_н = 4.1$ Ом – опір навантаження;
$T_a = 0.1$ с – стала часу якірного кола;
$R_d = 4.5$ Ом – опір обмотки збудження генератора;
$L_d = 5.5$ Г – індуктивність обмотки збудження генератора;
$R_1 = 9$ Ом – додатковий опір в колі збудження генератора.

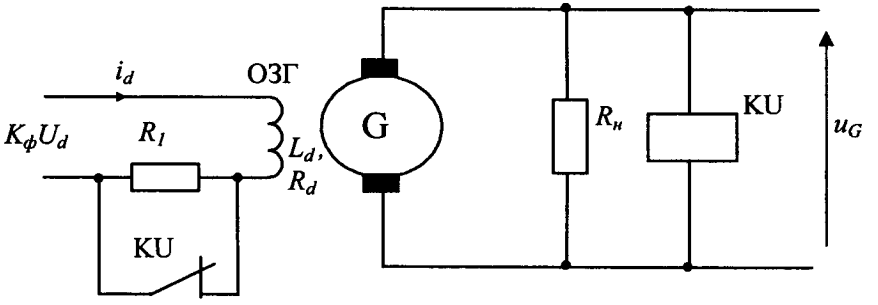


Рис. 4.1. Схема форсованого збудження генератора постійного струму

Виведемо різницеві рівняння для струму збудження i_d та струму якоря i_a генератора

$$Ty' + y = x \quad \Rightarrow \quad y' = \frac{x - y}{T};$$

підставляючи у формулу числового методу, отримаємо

$$y_{i+1} = y_i + \frac{h}{T}\left(x_{i+1} - y_{i+1}\right),$$

звідки

$$y_{i+1} = \frac{y_i + \frac{h}{T}x_{i+1}}{1 + \frac{h}{T}}.$$

Тоді

$$i_{d_{i+1}} = \frac{i_{d_i} + \frac{h}{T_G}\frac{K_\Phi U_d}{R_{d\Sigma}}}{1+\frac{h}{T_G}}; \qquad i_{a_{i+1}} = \frac{i_{a_i} + \frac{h}{T_a}\frac{K_G i_{d_{i+1}} R_d}{R_a + R_н}}{1+\frac{h}{T_a}},$$

де $R_{d\Sigma} = R_d$ для режиму форсування і $R_{d\Sigma} = R_d + R_1$ після закінчення форсування;

R_1 – додатковий опір в колі ОЗГ, вибирають з умови забезпечення номінального струму $I_{dн}$ ОЗГ після закінчення форсування для напруги $K_\phi U_d$ ($I_{dн} = \frac{K_\Phi U_{dн}}{R_1 + R_d}$).

Quick Basic

```
'-------------------------------------------------------------
' Розрахунок перехідних процесів ГПС для режиму
' форсування збудження генератора з використанням
' різницевого рівняння за неявним методом Ейлера.
'-------------------------------------------------------------
CONST Kg = 2.09          ' Коефіцієнт підсилення генератора
CONST Lg = 5.5           ' Індуктивність ОЗГ
CONST Rd = 4.5           ' Опір ОЗГ
CONST R1 = 9!            ' Додатковий опір кола збудження
CONST Udnom = 220        ' Номінальна напруга збудження
CONST Ugnom = 460        ' Номінальна напруга якоря
CONST Ra = .15           ' Опір якоря генератора
CONST Rn = 4.1           ' Опір навантаження
CONST Ta = .1            ' Стала часу якірного кола
CONST Kf = 3             ' Коефіцієнт форсування
CONST h = .002           ' Крок моделювання
CONST Tmin = 0           ' Початковий час розрахунку
CONST Tmax = 2.5         ' Максимальний час розрахунку
CONST Ymin = 0           ' Нижня межа графіка
CONST Ymax = 500         ' Верхня межа графіка
' ========================================
```

```
'          Підготовка до виведення графіка
' ========================================
COMMON SHARED Xmin, Xmax
Xmin = Tmin
Xmax = Tmax
' ========================================
'            Основний цикл програми
' ========================================
Id = 0                ' Початкова умова (струм збудження)
FOR t = Tmin TO Tmax STEP h

  CALL Graphic(12, t, Ia)   ' Виведення графіка струму якоря
  CALL Graphic(11, t, Ug)   ' Виведення графіка напруги генератора

  ' Знаходимо значення опору в колі збудження
  IF Ug < Ugnom THEN Rds = Rd ELSE Rds = Rd + R1
  Tg = Lg / Rds                      ' Стала часу ОЗГ

  ' Розрахунок за різницевим рівнянням струму збудження
  Id = (Id + h / Tg * Kf * Udnom / Rds) / (1 + h / Tg)
  Ud = Id * Rd                       ' Значення напруги збудження

  ' Розрахунок за різницевим рівнянням струму якоря
  Ia = (Ia + h / Ta * Kg * Ud / (Ra + Rn)) / (1 + h / Ta)
  Ug = Ud * Kg - Ia * Ra             ' Значення напруги на якорі
NEXT t
END
```

Turbo Pascal

```
{ ------------------------------------------- }
{ Розрахунок перехідних процесів ГПС для режиму   }
{ форсування збудження генератора з використанням }
{ різницевого рівняння за неявним методом Ейлера. }
{ ------------------------------------------- }
uses
   Plot ;
const
   Kg = 2.09 ;          { Коефіцієнт підсилення генератора }
   Lg = 5.5 ;           { Індуктивність ОЗГ                }
```

```
   Rd = 4.5 ;          { Опір ОЗГ                              }
   R1 = 9.0 ;          { Додатковий опір кола збудження        }
   Udnom = 220 ;       { Номінальна напруга збудження          }
   Ugnom = 460 ;       { Номінальна напруга якоря              }
   Ra = 0.15 ;         { Опір якоря генератора                 }
   Rn = 4.1 ;          { Опір навантаження                     }
   Ta = 0.1 ;          { Стала часу якірного кола              }
   Kf = 3 ;            { Коефіцієнт форсування                 }
   h = 0.002 ;         { Крок моделювання                      }
   Tmin = 0 ;          { Початковий час розрахунку             }
   Tmax = 2.5 ;        { Максимальний час розрахунку           }
   Ymin = 0 ;          { Нижня межа графіка                    }
   Ymax = 500 ;        { Верхня межа графіка                   }
var
   t, Id, Ia, Ug, Ud, Rds, Tg  : double ;

begin
{ ======================================== }
{         Підготовка до виведення графіка      }
{ ======================================== }
    InitGraphic(Tmin, Tmax, Ymin, Ymax, 0.2, 100) ;
{ ======================================== }
{           Основний цикл програми           }
{ ======================================== }
Id := 0 ;    { Початкова умова (струм збудження) }
Ug := 0 ;    { Початкова умова (напруга генератора) }
Ia := 0 ;    { Початкова умова (струм якоря) }
t := Tmin ; { Початкове значення часу       }
repeat
  Graphic(12, t, Ia) ;  { Виведення графіка струму якоря }
  Graphic(11, t, Ug) ;  { Виведення графіка напруги генератора }

  { Знаходимо значення опору в колі збудження }
  if Ug < Ugnom then
    Rds := Rd                  { З форсуванням  }
  else
    Rds := Rd + R1 ;           { Без форсування }
  Tg := Lg / Rds ;             { Стала часу ОЗГ }

  { Розрахунок за різницевим рівнянням струму збудження }
  Id := (Id + h / Tg * Kf * Udnom / Rds) / (1 + h / Tg) ;
  Ud := Id * Rd ;           { Значення напруги збудження }
```

```
  { Розрахунок за різницевим рівнянням струму якоря      }
  Ia := (Ia + h / Ta * Kg * Ud / (Ra + Rn)) / (1 + h / Ta) ;
  Ug := Ud * Kg - Ia * Ra ;     { Значення напруги на якорі }

  t := t + h ;
until  t > Tmax ;
end.
```

MathCAD

$U_{dн} := 220$	*Номінальна напруга збудження генератора*
$U_н := 460$	*Номінальна напруга якоря генератора*
$K_G := 2.09$	*Коефіцієнт підсилення генератора*
$K_ф := 3$	*Коефіцієнт форсування*
$R_a := 0.15$	*Опір якоря генератора*
$R_н := 4.1$	*Опір навантаження*
$T_a := 0.1$	*Стала часу якірного кола*
$R_d := 4.5$	*Опір обмотки збудження генератора*
$L_d := 5.5$	*Індуктивність обмотки збудження генератора*
$R_l := 9$	*Додатковий опір в колі збудження генератора*
$h := 0.002$	*Крок моделювання*
$T_{max} := 2.5$	*Час моделювання*

$$N := \frac{T_{max}}{h} \qquad i := 0 .. N \qquad t_i := i \cdot h$$

Номери розрахункових точок

$$\begin{pmatrix} i_{d_{i+1}} \\ i_{a_{i+1}} \end{pmatrix} :=$$

"Напруга генератора"

$U_G \leftarrow K_G \cdot i_{d_i} \cdot R_d - i_{a_i} \cdot R_a$

"Введення додаткового опору в коло ОЗГ"

$R_{\Sigma d} \leftarrow R_d + R_1 \quad \text{if } U_G \geq Ua_{nom}$

$R_{\Sigma d} \leftarrow R_d \quad \text{otherwise}$

"Стала часу обмотки збудження"

$$T_d \leftarrow \frac{L_d}{R_{\Sigma d}}$$

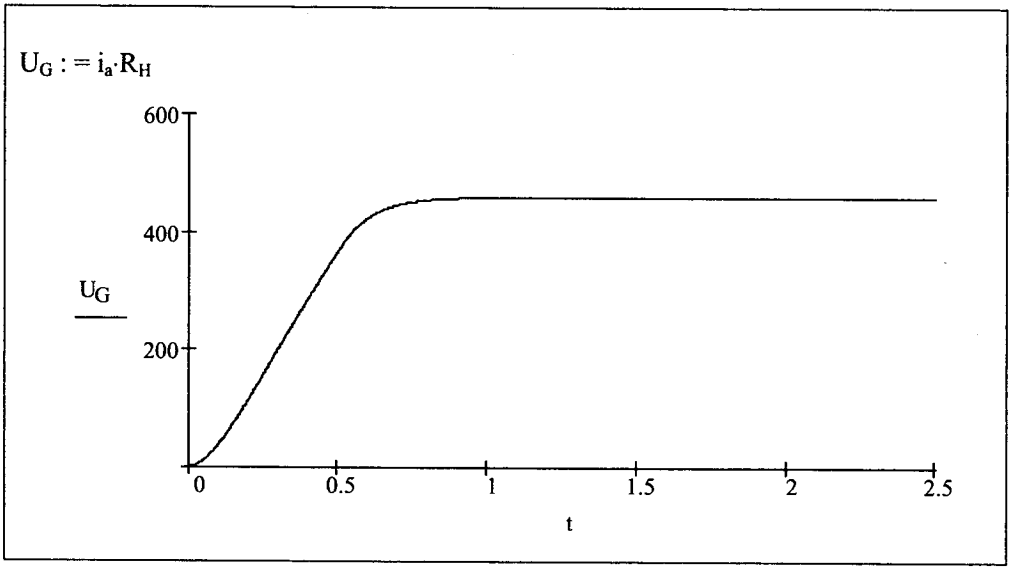
"Наступне значення струму збудження"

$$Id1 \leftarrow \frac{i_{d_i} + \frac{h}{T_d} \cdot \frac{Kf \cdot U_d}{R_{\Sigma d}}}{1 + \frac{h}{T_d}}$$

"Наступне значення струму якоря"

$$Ia1 \leftarrow \frac{i_{a_i} + \frac{h}{T_a} \cdot \frac{K_G \cdot Id1 \cdot R_d}{R_a + R_H}}{1 + \frac{h}{T_a}}$$

$$\begin{pmatrix} Id1 \\ Ia1 \end{pmatrix}$$



MATLAB

```
% ------------------------------------------------------------
% Розрахунок перехідних процесів ГПС для режиму
% форсування збудження генератора з використанням
% різницевого рівняння за неявним методом Ейлера
% ------------------------------------------------------------
Kg = 2.09;          % Коефіцієнт підсилення генератора
Lg = 5.5;           % Індуктивність ОЗГ
Rd = 4.5;           % Опір ОЗГ
R1 = 9;             % Додатковий опір кола збудження
Udnom = 220;        % Номінальна напруга збудження
Ugnom = 460;        % Номінальна напруга Якоря
Ra = .15;           % Опір Якоря генератора
Rn = 4.1;           % Опір навантаження
Ta = .1;            % Стала часу Якірного кола
Kf = 3;             % Коефіцієнт форсування
h = .002;           % Крок моделювання
Tmin = 0;           % Початковий час розрахунку
Tmax = 2.5;         % Максимальний час розрахунку
```

```
% ========================================
%          Основний цикл програми
% ========================================
Id = 0;                  % Початкова умова (струм збудження)
Ia = 0;                  % Початкова умова (струм Якоря)
Ug(1) = 0;
t(1) = 0;
N = Tmax/h;                 % Крок розв'язування
for i = 1:N
  t(i+1) = i*h;
  % Знаходимо значення опору в колі збудження
  if Ug(i) < Ugnom
      Rds = Rd;
  else
      Rds = Rd + R1;
  end
  Tg = Lg / Rds;                        % Стала часу ОЗГ

  % Розрахунок за різницевим рівнянням струму збудження
  Id = (Id + h / Tg * Kf * Udnom / Rds) / (1 + h / Tg);

  % Розрахунок за різницевим рівнянням струму Якоря
  Ia = (Ia + h/Ta * Kg*Id*Rd / (Ra+Rn)) / (1 + h/Ta);
  Ug(i+1) = Ia * Rn;              % Значення напруги на Якорі
end
plot(t, Ug), grid
```

Реостатний запуск двигуна постійного струму незалежного збудження

Розрахувати перехідні процеси для режиму триступеневого реостатного запуску двигуна постійного струму у функції струму якоря.

Задано:

$P_н = 4.8$ кВт	– потужність двигуна;
$U_н = 220$ В	– номінальна напруга двигуна;
$I_н = 24.2$ А	– номінальний струм якоря;
$R_н = 0.38$ Ом	– опір якоря;
$n_н = 1500$ об/хв	– номінальна швидкість обертання.

Для реостатного запуску прийнято:

- максимальний струм $I_1 = 2.5I_н = 60.5$ А;
- коефіцієнт $\lambda = I_1 / I_2 = 2.12$, де I_1, I_2 – струми перемикання під час реостатного запуску, які отримують з пускової діаграми; для забезпечення задовільної динаміки приймають $I_2 \geq (1.1 \ \ldots \ 1.2)I_c$;
- значення пускових опорів (знайдені з використанням пускової діаграми, див. відповідний документ у MathCAD):

  $R_1 = 3.26$ Ом – сумарний додатковий опір першого ступеня;
  $R_2 = 1.33$ Ом – сумарний додатковий опір другого ступеня;
  $R_3 = 0.43$ Ом – сумарний додатковий опір третього ступеня.

Для розв'язування поставленої задачі використано запропоновану в розд. 3 процедуру з використанням формули Фельберга 2(3)В з автоматичним вибором кроку інтегрування. Результати розрахунку показано на рис. 4.2.

Quick Basic

```
'-----------------------------------------------------
' Розрахунок перехідних процесів режиму реостатного
' запуску двигуна постійного струму незалежного збудження
' для трьох пускових ступенів.
'-----------------------------------------------------
CONST N = 2                ' Порядок системи
CONST Eps = .0001          ' Точність розв'язку на кроці
CONST Tmin = 0             ' Початковий час розрахунку
CONST Tmax = 2.5           ' Максимальний час розрахунку
CONST Xmin = Tmin          ' Ліва межа графіка
CONST Xmax = Tmax          ' Права межа графіка
CONST Ymin = 0             ' Нижня межа графіка
CONST Ymax = 100           ' Верхня межа графіка

DIM y(N)                   ' Опис масиву змінних
FOR i = 1 TO N             ' Обнулення масиву змінних
   y(i) = 0                ' Нульові початкові умови
NEXT i                     '
```

```
' ========================================
'          Основний цикл програми
' ========================================
t = Tmin                                  ' Початковий час
h = .001                                  ' Початковий крок
DO
  CALL Graphic(4, t, y(1))                ' Виведення графіка струму
якоря
  CALL Graphic(3, t, y(2)/2)              ' Виведення графіка
швидкості
  CALL Fehlberg2B(N, t, h, Eps, y())      ' Виклик числового методу
LOOP UNTIL t > Tmax
END

SUB dif (t, y(), f()) STATIC
' ------------------------------------------------------
' Підпрограма призначена для обчислення значень правих частин
' системи диференціальних рівнянь, записаних у нормальній формі
  Коші
'  Форма  запису:
'       f(1) = функція_1(y(1),y(2),...,y(N),t)
'       f(2) = функція_2(y(1),y(2),...,y(N),t)
'                    ...
'       f(N) = функція_N(y(1),y(2),...,y(N),t)
'
'  Вхідні величини:
'       t - незалежна змінна
'       y - масив змінних величин
'
'  Вихідні величини:
'       y - масив змінних
'       f - масив похідних
' ------------------------------------------------------
CONST Unom = 220!    ' Uд  - напруга на якорі двигуна
CONST Inom = 24.2    ' Inom - номінальний струм якоря
CONST Rad = .38      ' Rяд - опір якоря двигуна
CONST Lad = .017     ' Lяд - індуктивність якоря двигуна
CONST C = 1.34       ' C   - стала двигуна
CONST J = .5         ' J   - сумарний момент інерції
CONST Mc = 16.2      ' Mc  - статичний момент
CONST Imax = 2.5 * Inom  ' Imax - макс. пусковий струм
```

```
CONST Lambda = 2.12      ' Коеф. кратності пуск. струму
CONST R1 = 3.26          ' R1 - сумарний дод. опір першого ступеня
CONST R2 = 1.33          ' R2 - сумарний дод. опір другого ступеня
CONST R3 = .43           ' R3 - сумарний дод. опір третього ступеня

' Першим вмикається 1-й пусковий ступінь
                         ' Під час запуску програми
IF first% = 0 THEN       ' всі змінні отримують
    M% = 1               ' нульові значення, зокрема ціла змінна
    first% = 1           ' First% . У першому звертанні до процедури
END IF                   ' встановлюємо 1-й номер пускового ступеня.

' Визначення сумарного опору якірного кола Ra
' залежно від номера пускового ступеня ( змінна  M% )
SELECT CASE M%
   CASE 1
       Ra = Rad + R1     ' перший пусковий ступінь
   CASE 2
       Ra = Rad + R2     ' другий пусковий ступінь
   CASE 3
       Ra = Rad + R3     ' третій пусковий ступінь
   CASE IS > 3
       Ra = Rad          ' природна характеристика
END SELECT
Ta = Lad / Ra            ' Стала часу якірного кола

'   Модель якірного кола (розрахунок похідної струму якоря)
f(1) = ((Unom - y(2) * C) / Ra - y(1)) / Ta

' Перемикання ступеня здійснюється, якщо
' I2 <= Imax / Lambda та значення похідної струму
' від'ємне (струм спадає).
'
'       Номер пускового ступеня збільшується на одиницю
IF (y(1) <= Imax / Lambda) AND (f(1) <= 0) THEN M% = M% + 1

'   Модель механічної частини (розрахунок похідної швидкості)
f(2) = (y(1) * C - Mc) / J

END SUB
```

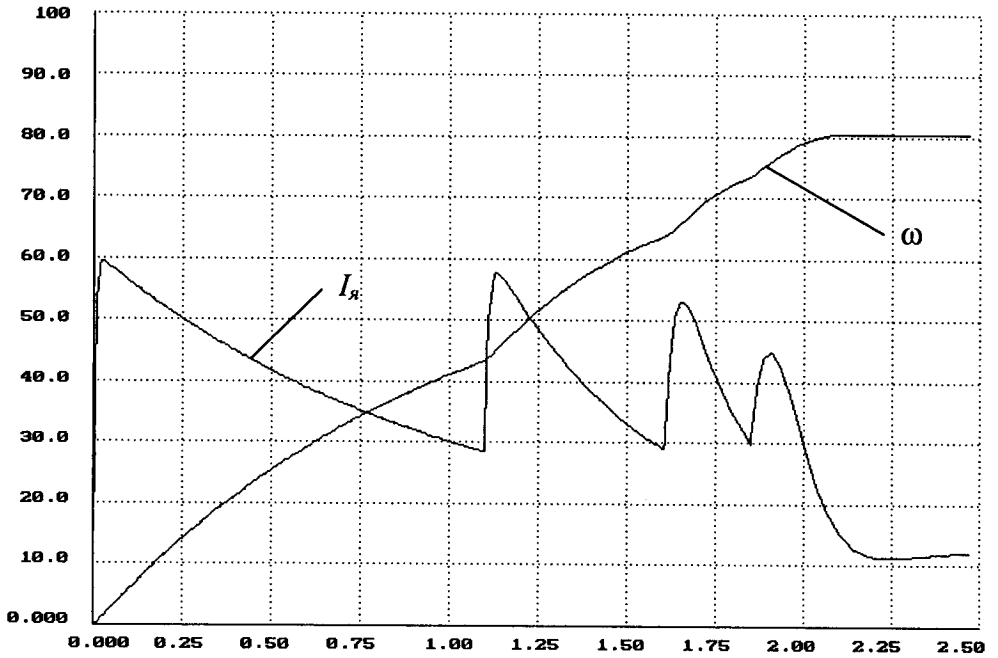


Рис. 4.2. Результати розрахунку перехідних процесів струму якоря та швидкості для режиму реостатного запуску двигуна постійного струму

Turbo Pascal

```
{$E+,N+,F+ }
program  DC_Reostat;
uses
   f23b, Crt, Plot;
const
   N = 2;                    { Порядок системи                }
   Tmax = 2.5;               { Максимальний час розрахунку    }
   Eps  : DOUBLE = 1e-4;     { Точність обчислення на кроці }
   Ymin : DOUBLE = 0.0;
   Ymax : DOUBLE = 100.0;
var
```

```
   t, h  : DOUBLE;
   i  : INTEGER;
   y  : Vector;

procedure  dif(x: DOUBLE; var y, f : Vector);
{ ------------------------------------------------------- }
{ Процедура призначена для обчислення правих частин      }
{ системи диференціальних рівнянь, записаної у нормальній }
{ формі Коші.                                            }
{ Форма  запису:                                         }
{       f[1] := ...                                      }
{       . . .                                            }
{       f[N] := ...                                      }
{                                                        }
{ Вхідні величини:                                       }
{       t - незалежна змінна                             }
{       y - масив змінних величин                        }
{ Вихідні величини:                                      }
{       y - масив змінних                                }
{       f - масив обчислених похідних                    }
{ ------------------------------------------------------- }
const
   Unom = 220.0;       { Номінальна напруга якоря    }
   Inom = 24.2;        { Номінальний струм якоря     }
   Rad = 0.38;         { Опір якірного кола          }
   Lad = 0.017;        { Індуктивність якірного кола }
   C = 1.34;           { Стала двигуна               }
   J = 0.5;            { Сумарний момент інерції привода  }
   Mc = 16.2;          { Статичний момент            }
   Imax = 2.5*Inom;    { Максимальний пусковий струм}
   Lambda = 2.12;      { Кратність пускового струму }
   R1 = 3.26;          { Сум.дод.опір 1-го ступеня  }
   R2 = 1.33;          { Сум.дод.опір 2-го ступеня  }
   R3 = 0.43;          { Сум.дод.опір 3-го ступеня  }
   M : integer = 1;    { Номер пускового ступеня     }
var
   Ra, Ta : double;
begin
   { Визначення сумарного опору якірного кола }
   { залежно від номера ступеня }
   case M of
```

```
        1 : Ra := Rad + R1;
        2 : Ra := Rad + R2;
        3 : Ra := Rad + R3;
    else
        Ra := Rad     { природна характеристика }
    end;
    { Визначення сталої часу якоря }
    Ta := Lad / Ra;

    { Обчислення значення похідної струму якоря }
    f[1] := ((Unom - y[2]*C)/Ra - y[1])/Ta;
    { Перемикання ступеня здійснюється, якщо          }
    { I2 <= Imax / Lambda та значення похідної струму }
    { від'ємне (струм спадає). У такому разі номер    }
    { пускового ступеня збільшується на одиницю       }
    if (y[1]<=Imax/Lambda) and (f[1]<0) then M := M + 1;

    { Обчислення значення похідної швидкості двигуна }
    f[2] := (y[1]*C - Mc)/J;
end;

begin
    h := 0.01;        { Початковий крок    }
    {   Ініціалізація графіка      }
    InitGraphic(0, Tmax, Ymin, Ymax, 0.2, 20.0);
    for i := 1 to N do y[i]:= 0.0;          { Початкові умови }
{ ------------------------------------  }
{  Розрахунок перехідного процесу запуску }
{ ------------------------------------  }
    repeat
        Fehlberg23B(Dif, N, t, h, Eps, y);
        Graphic(1, t, y[1]);     { виведення на екран Ia }
        Graphic(2, t, y[2]/2);   { виведення на екран w  }
    until t > Tmax;
    ReadLn;     { Натиснути  ENTER  для продовження }
end.
```

MathCAD

Технічні дані

$U_{nom} := 220$ *Номінальна напруга двигуна*

$I_{nom} := 24.2$ *Номінальний струм якоря*

$n_{nom} := 1500$ *Номінальна кутова швидкість* ($^{об.}/_{хв.}$)

$R_a := 0.38$ *Опір якоря двигуна*

$L_a := 0.035$ *Індуктивність якоря двигуна*

$J := 0.5$ *Сумарний момент інерції привода*

$p_p := 1$ *Кількість пар полюсів*

$M_c := 16.2$ *Статичний момент*

Розрахункові величини

$\omega_{nom} := n_{nom} \frac{\pi}{30}$ *Номінальна кутова швидкість* ($^{рад.}/_{с}$)

$C := \frac{U_{nom} - I_{nom} R_a}{\omega_{nom}}$ *Стала двигуна*

Розрахунок реостатних характеристик

$m := 3$ $j := 1 .. m$ *Кількість пускових ступенів*

$I_1 := 2.5 \cdot I_{nom}$ *Максимальний пусковий струм*

$R_m := \frac{U_{nom}}{I_1}$ *Сумарний пусковий опір*

$\lambda := \sqrt[m]{\frac{R_m}{R_a}}$ *Кратність пускових струмів*

$I_2 := \frac{I_1}{\lambda}$ *Струм, за якого перемикаються ступені*

$r_j := R_a \cdot \lambda^{j-1} \cdot (\lambda - 1)$ $r_0 := 0$ *Значення опору кожного ступеня*

$$\omega 2_j := \frac{U_{nom} - I_2 \cdot \left(\sum_{k=0}^{j} r_k + R_a \right)}{C}$$

Швидкості, за яких перемикаються ступені

Розрахунок перехідного процесу режиму реостатного запуску

$T_{max} := 2.5$ *Кінцевий час розрахунку*

$N := 500$ *Кількість інтервалів інтегрування*

Вектор-функція правих частин системи диференціальних рівнянь:

y_0 – *струм якоря*

y_1 – *швидкість*

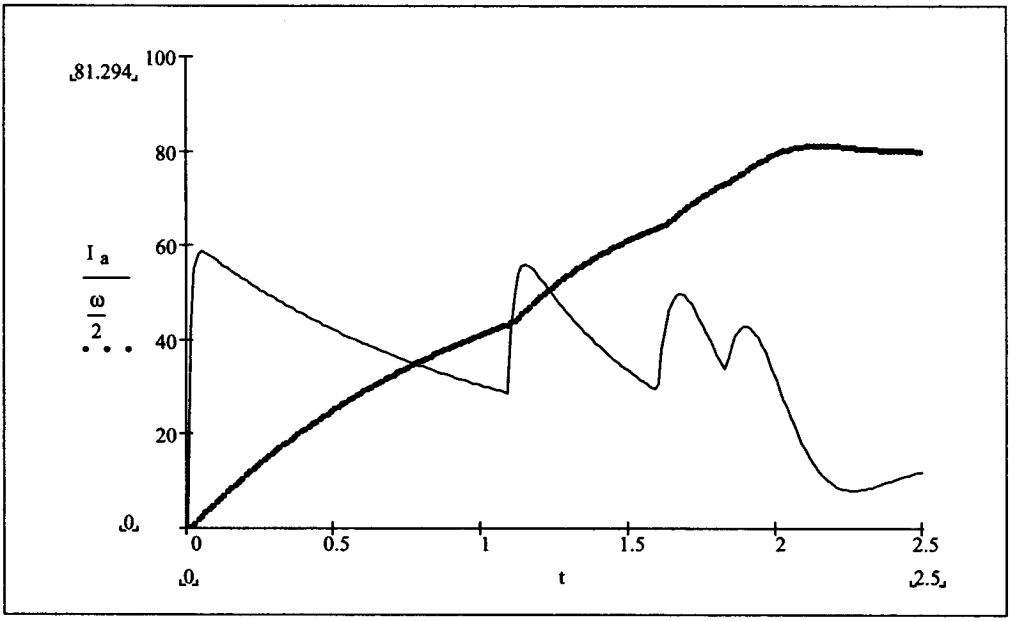
Початкові умови: $y0 := \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

$$Diff(t, y) := \left(\begin{array}{l} R_{a\Sigma} \leftarrow R_a \\ \text{for } j \in 0..m \\ \quad R_{a\Sigma} \leftarrow R_{a\Sigma} + r_j \text{ if } y_1 < \omega 2_j \\ T_a \leftarrow \dfrac{L_a}{R_{a\Sigma}} \\ \dfrac{\dfrac{U_{nom} - C \cdot y_1}{R_{a\Sigma}} - y_0}{T_a} \\ \dfrac{C \cdot y_0 - M_c}{J} \end{array} \right)$$

Розв'язуємо систему диференціальних рівнянь: $Res := rkfixed(y0, 0, T_{max}, N, Diff)$

З отриманої матриці розв'язку виділяємо необхідні стовпці:

$t := Res^{<0>}$ $I_a := Res^{<1>}$ $\omega := Res^{<2>}$



MATLAB

```
% Розрахунок перехідного процесу реостатного запуску
% двигуна постійного струму
global Mr;
Mr = 1;                 % Перший пусковий ступінь
Tmax = 2.5;             % Час розрахунку
y0 = [0 0];             % Початкові умови

% Розв'ЯзуванняЯ системи диференціальних рівнЯнь
 [t, y] = ode23('dcm_reost', [0 Tmax], y0);

% Побудова графіка
plot(t, y(:,1), 'k', t, y(:,2)/2, 'k:');
axis([0 Tmax 0 inf]);
xlabel('t, c'); ylabel('I_a, \omega/2'); legend('I_a', '\omega/2');
```

```
function f=dcm_reost(t, y);
% ------------------------------------------------
%  Розрахунок динаміки режиму реостатного
%  запуску двигуна постійного струму
% ------------------------------------------------
global Mr;
Unom = 220.0;      % Напруга на Якорі двигуна
Inom = 24.2;       % Номінальний струм Якоря
Rad = 0.38;        % Опір Якоря двигуна
Lad = 0.017;       % Індуктивність Якоря двигуна
C = 1.340;         % Стала двигуна
J = 0.5;           % Сумарний момент інерції
Mc = 16.2;         % Статичний момент
Imax = 2.5*Inom;   % Макс. пусковий струм
Lambda = 2.12;     % Коеф. кратності пуск. струму
R1 = 3.26;         % Сумарний дод. опір 1-го ступеня
R2 = 1.33;         % Сумарний дод. опір 2-го ступеня
R3 = 0.43;         % Сумарний дод. опір 3-го ступеня

% Розрахунок правих частин системи диференціальних рівнянь
f=zeros(2,1);      % формування вектор-стовпця
% Визначення сумарного опору Якірного кола Ra залежно
% від номера пускового ступеня (глобальна змінна Mr)
switch Mr
   case 1
      Ra = Rad + R1;     % перший пусковий ступінь
   case 2
      Ra = Rad + R2;     % другий пусковий ступінь
   case 3
      Ra = Rad + R3;     % третій пусковий ступінь
   otherwise
      Ra = Rad;          % природна характеристика
end
Ta = Lad / Ra;           % Стала часу Якірного кола

%   Модель Якірного кола (розрахунок похідної струму ЯкоряЯ)
f(1) = ((Unom - y(2)*C)/Ra - y(1))/Ta;
% Перемикання ступеня здійснюється, Якщо
% I2 <= Imax / Lambda та значення похідної струму
% від'ємне (струм спадає).
%
%      Номер пускового ступеня збільшується на одиницю
if (y(1) <= Imax/Lambda) & (f(1) <= 0)
```

```
   Mr = Mr + 1;
end
%   Модель механічної частини (розрахунок похідної швидкості)
f(2) = (y(1) * C - Mc) / J;
```

Математична модель системи Г–Д з тиристорним збудником і паралельним коригуванням

Далі наведено приклади моделювання динамічних режимів систем електроприводів постійного струму. Ці системи з різноманітними зворотними зв'язками побудовані за принципами паралельного та послідовного коригування.

Математична модель системи Г-Д з тиристорним збудником (тиристорним перетворювачем ТП) і паралельним коригуванням (рис. 4.3) подається системою диференціальних рівнянь:

$$\frac{du_{mn}}{dt} = \frac{u_{вх}K_{mn} - u_{mn}}{T_{mn}};$$

$$\frac{de_G}{dt} = \frac{u_{mn}K_G - e_G}{T_G};$$

$$\frac{di_a}{dt} = \frac{(e_G - C\omega)/R_a - i_a}{T_a};$$

$$\frac{d\omega}{dt} = \frac{(i_a - I_c)C}{J},$$

де $u_{вх} = U_з - \left(K_U \cdot U_G + (I_{відс} - i_a)\cdot K_i + \omega \cdot K_\omega\right)$ – вхідна напруга ТП;

$U_з$ – напруга завдання;

K_U, K_i, K_ω – коефіцієнти зворотного зв'язку відповідно за напругою генератора, струмом якоря та швидкістю двигуна відповідно;

u_{mn} – вихідна напруга ТП;

$I_{відс}$ – струм відсікання.

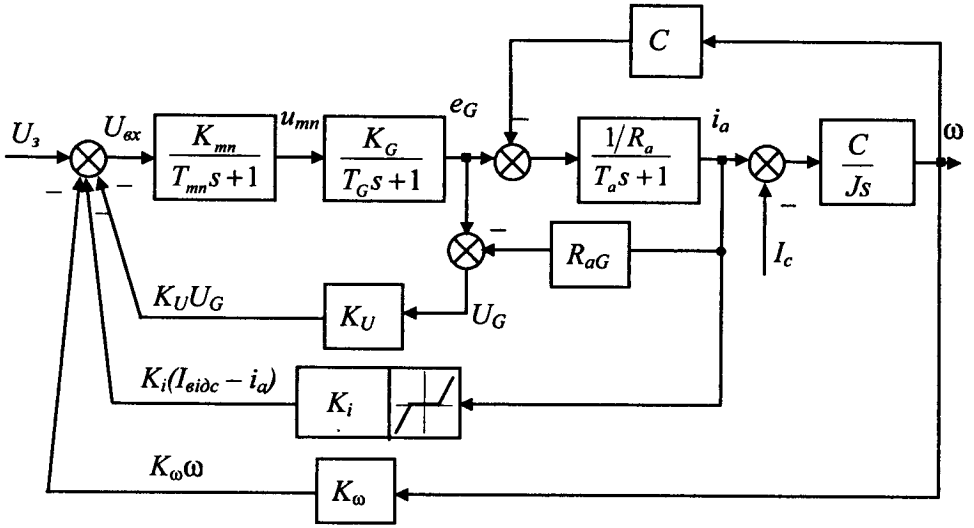


Рис. 4.3. Структурна схема системи Г–Д з ТЗ і паралельним коригуванням координат

Приклад: Нижче подано приклад розрахунку перехідних процесів режиму "запуск – реверс – зупинка" у системі Г–Д з паралельними зворотними зв'язками: із затриманим за струмом (струмовим відсіканням) та лінійним за швидкістю. Результати моделювання цього режиму показані на рис. 4.4.

Задано:

$I_{відс} = 320$ А	– струм якоря, за якого спрацьовує відсікання;
$U_{znom} = 20$ В	– номінальна напруга завдання;
$R_a = 0.34$ Ом	– сумарний опір якірного кола системи Г–Д;
$C = 2.34$ В·с$^{-1}$	– стала двигуна;
$K_d = 20$	– коефіцієнт підсилення ТП;
$K_G = 21$	– коефіцієнт підсилення генератора;
$K_ω = 0.05$	– коефіцієнт зворотного зв'язку за швидкістю;
$K_I = 0.25$	– коефіцієнт зворотного зв'язку за струмом якоря;
$T_μ = 0.01$с	– стала часу ТП;
$T_G = 0.7$ с	– стала часу ОЗГ;
$T_a = 0.05$ с	– електромагнітна стала часу якірного кола;

$M_C = 64.5$ Н·м – статичний момент навантаження привода;

$J = 9.2$ кг·м^2 – сумарний момент інерції привода, зведений до вала двигуна;

$T_{max} = 20$ с – час розрахунку.

Quick Basic

```
' ========================================================
'  Розрахунок динаміки режиму ЗАПУСК/РЕВЕРС/СТОП
'  для системи Г-Д з паралельним коригуванням
' ========================================================
CONST N = 4                ' Порядок системи
CONST Tmax = 20!           ' Максимальний час розрахунку
CONST Eps = .001           ' Точність розв'язання на кроці
'    Необхідно для виведення графіка
COMMON SHARED Xmin, Xmax
Xmin = 0!                  ' Ліва межа графіка
Xmax = Tmax                ' Права межа графіка
CONST Ymin = -500          ' Нижня межа графіка
CONST Ymax = 500           ' Верхня межа графіка
' ========================================================
DIM y(N)                   ' Описання масиву змінних
FOR i = 1 TO N             ' Обнулення масиву змінних
   y(i) = 0                ' (початкові умови перед запуском
NEXT i                     ' двигуна - нульові)
t = 0!
h = .001                   ' Початковий крок моделювання
DO
   CALL Graphic(11, t, y(3))     ' Виведення на графік струму якоря
   CALL Graphic(12, t, y(4))     ' Виведення на графік швидкості
   CALL Fehlberg2B(N, t, h, Eps, y()) ' Виклик числового методу
LOOP UNTIL t > Tmax
END

SUB dif (t, y(), f())
' ------------------------------------------------------------
' Підпрограма призначена для обчислення значень правих частин
' системи диференційних рівнянь у нормальній формі Коші.
' ------------------------------------------------------------
```

```
'     Позначення змінних:
'     y(1) - напруга ТП
'     y(2) - напруга генератора
'     y(3) - струм якоря
'     y(4) - швидкість двигуна
' ------------------------------------------------------------
CONST Uznom = 20        ' Номінальна напруга завдання
CONST Kw = .05          ' Коеф. зворотного зв'язку за швидкістю
CONST Kc = .25          ' Коеф. зворотного зв'язку за струмом
CONST Ia.lim = 320      ' Струм спрацювання відсікання
CONST Ttp = .01         ' Стала часу ТП
CONST Ktp = 20!         ' Коеф. підсилення ТП
CONST Tg = .7           ' Стала часу генератора
CONST Kg = 2.1          ' Коеф. підсилення генератора
CONST C = 2.34          ' Стала двигуна
CONST Ra = .34          ' Сумарний опір якірного кола
CONST Ta = .05          ' Стала часу якірного кола
CONST J = 9.2           ' Сумарний момент інерції привода
CONST Mc = 64.5         ' Момент опору на валу

'   Формування напруги завдання
Uz = Uznom
IF t > 6 THEN Uz = -Uznom
IF t > 14 THEN Uz = 0
'   Розрахунок вхідної напруги ТП
IF ABS(y(3)) < Ia.lim THEN
   ' Струмове відсікання відсутнє
   Ubx = Uz - Kw * y(4)
ELSE
   ' Струмове відсікання спрацювало
   Ubx = Uz - Kw * y(4) - Kc * (y(3) - Ia.lim * SGN(y(3)))
END IF
'   Модель тиристорного перетворювача
f(1) = (Ubx * Ktp - y(1)) / Ttp
'   Модель генератора
f(2) = (y(1) * Kg - y(2)) / Tg
'   Модель якірного кола Г-Д
f(3) = ((y(2) - C * y(4)) / Ra - y(3)) / Ta
'   Розрахунок похідної швидкості
f(4) = (C * y(3) - Mc) / J
END SUB
```

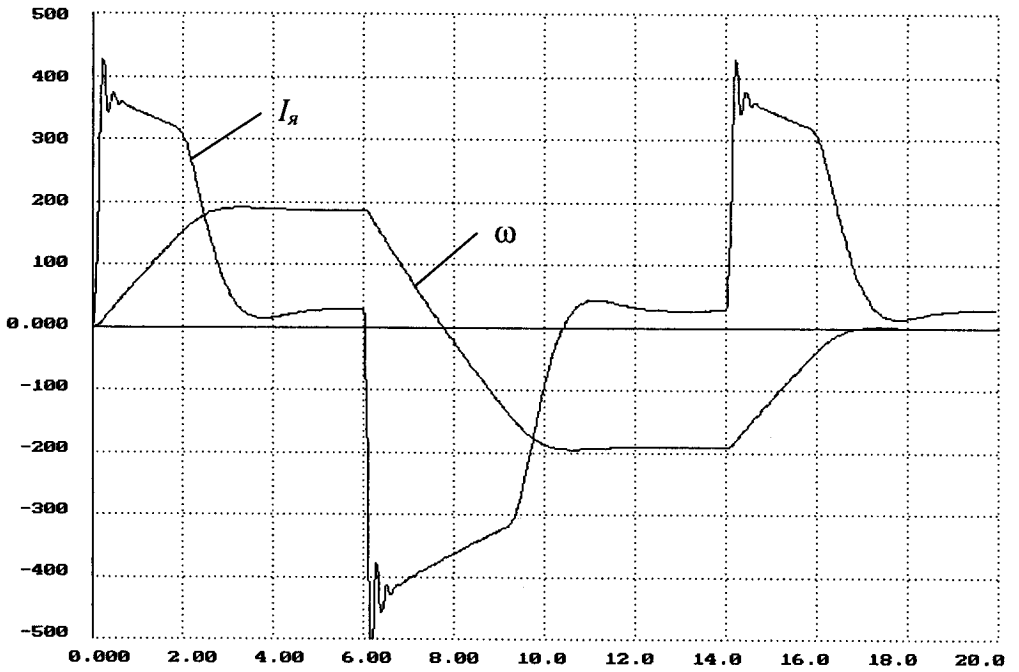


Рис. 4.4. Перехідні процеси у системі Г–Д з паралельним коригуванням

Turbo Pascal

```
{$E+,N+,F+ }
program  Parallel_Correction;
uses
   F23B, Crt, Plot;
const
   N = 4;                    { Порядок системи                }
   Tmax = 20.0;              { Максимальний час розрахунку    }
   Eps  : DOUBLE = 1e-4;     { Точність на кроці }
   Ymin : DOUBLE = -500.0;
   Ymax : DOUBLE = 500.0;
   dT   : DOUBLE = 2.0;      { Крок сітки графіка за часом    }
   dY   : DOUBLE = 100.0;    { Крок сітки графіка за віссю Y }
```

```
var
   t, h  : DOUBLE;
   i   : INTEGER;
   y   : Vector;

procedure  dif(x: DOUBLE; var y, f : Vector);
{ ------------------------------------------------------- }
{ Процедура призначена для обчислення значень правих частин }
{ системи диференціальних рівнянь у нормальній формі Коші:  }
{       f[1] = функція_1(y[1],y[2],...,y[N],x)              }
{       f[2] = функція_2(y[1],y[2],...,y[N],x)              }
{                      . . .                                }
{       f[N] = функція_N(y[1],y[2],...,y[N],x)              }
{ Вхідні величини:                                          }
{       x - незалежна змінна                                }
{       y - масив змінних величин                           }
{Вихідні величини:                                          }
{        y - масив змінних                                  }
{        f - масив обчислених похідних                      }
{      y[1] - напруга ТП                                    }
{      y[2] - ЕРС генератора                                }
{      y[3] - струм якоря                                   }
{      y[4] - швидкість двигуна                             }
{ ------------------------------------------------------- }
const
   Uznom = 20.0; { Напруга завдання для номінального режиму }
   Kw = 0.05;    { Коефіцієнт зворотного зв'язку за швидкістю }
   Kc = 0.25;    { Коефіцієнт зворотного зв'язку за струмом  }
   Ia_lim = 320.0; { Величина струмового відсікання          }
   Ktp = 20.0;     { Коефіцієнт підсилення ТП                }
   Ttp = 0.01;     { Стала часу ТП                           }
   Kg = 2.1;       { Коефіцієнт підсилення генератора        }
   Tg = 0.7;       { Стала часу ОЗГ                          }
   Ce = 2.34;      { Стала двигуна                           }
   Ra = 0.34;      { Сумарний опір якірного кола             }
   Ta = 0.05;      { Стала часу якірного кола                }
   J = 9.2;        { Сумарний момент інерції привода         }
   Mc = 64.5;      { Номінальний момент опору навантаження   }
var
   Uz, Ubx: DOUBLE;
begin
{  Формування напруги завдання  }
Uz := Uznom;
```

```
if t > 6 then Uz := -Uznom;
if t > 14 then Uz := 0.0;

{ Розрахунок вхідної напруги ТП }
if abs(y[3]) < Ia_lim then
  { Струмове відсікання відсутнє }
   Ubx := Uz - Kw*y[4]
else
  { Струмове відсікання спрацювало }
   Ubx := Uz - Kw*y[4]- Kc*(y[3] - Ia_lim*y[3]/abs(y[3]));

{  Модель тиристорного перетворювача  }
f[1] := (Ubx*Ktp - y[1])/Ttp;  { Врахування інерційності ТП }

{  Модель генератора  }
f[2] := (y[1]*Kg - y[2])/Tg;

{  Опис якірного кола ТП-Д  }
f[3] := ((y[2] - Ce*y[4])/Ra - y[3])/Ta;

{   Опис механічної частини привода      }
{   Розрахунок похідної швидкості        }
f[4] := (Ce*y[3] - Mc)/J;

end;

begin
   h := 0.001;  { Початковий крок   }
   {   Ініціалізація графіка     }
   InitGraphic(0, Tmax, Ymin, Ymax, dT, dY);
   { Початкові умови }
   for  i := 1  to  N  do y[i] := 0.0;
{ ------------------------------- }
{  Розрахунок перехідного процесу  }
{ ------------------------------- }
   repeat
      Fehlberg23B(Dif, N, t, h, Eps, y);
      Graphic(1, t, y[3]);  { виведення на екран Ia }
      Graphic(2, t, y[4]);  { виведення на екран w  }
   until  t > Tmax;
   ReadLn;  { Натиснути ENTER для продовження }
end.
```

MathCAD

$I_{lim} := 320$ *Струм якоря, за якого спрацьовує відсікання*

$U_{znom} := 20$ *Номінальна напруга завдання*

$R_a := 0.34$ *Сумарний опір якірного кола*

$C := 2.34$ *Стала двигуна*

$K_d := 20$ *Коефіцієнт підсилення ТП*

$K_G := 21$ *Коефіцієнт підсилення генератора*

$K_\omega := 0.05$ *Коефіцієнт зворотного зв'язку за швидкістю*

$K_I := 0.25$ *Коефіцієнт зворотного зв'язку за струмом якоря*

$T_\mu := 0.01$ *Стала часу ТП*

$T_G := 0.7$ *Стала часу ОЗГ*

$T_a := 0.05$ *Електромагнітна стала часу якірного кола*

$M_C := 64.5$ *Статичний момент навантаження*

$J := 9.2$ *Сумарний момент інерції привода*

$T_{max} := 20$ *Час розрахунку*

Напруга завдання:

$$U_z(t) := \begin{cases} U_{znom} & \text{if } t \le 6 \\ -U_{znom} & \text{if } 6 < t \le 14 \\ 0 & \text{otherwise} \end{cases}$$

Початкові умови:

$$y_0 := \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Кількість точок інтегрування:

$N := 2000$

Вектор-функція правих частин системи диференціальних рівнянь, що описують модель електропривода зі зворотним зв'язком за швидкістю та струмовим відсіканням:

y_0 – *напруга ТП;*

y_1 – *напруга генератора;*

y_2 – *струм якоря;*

y_3 – *швидкість.*

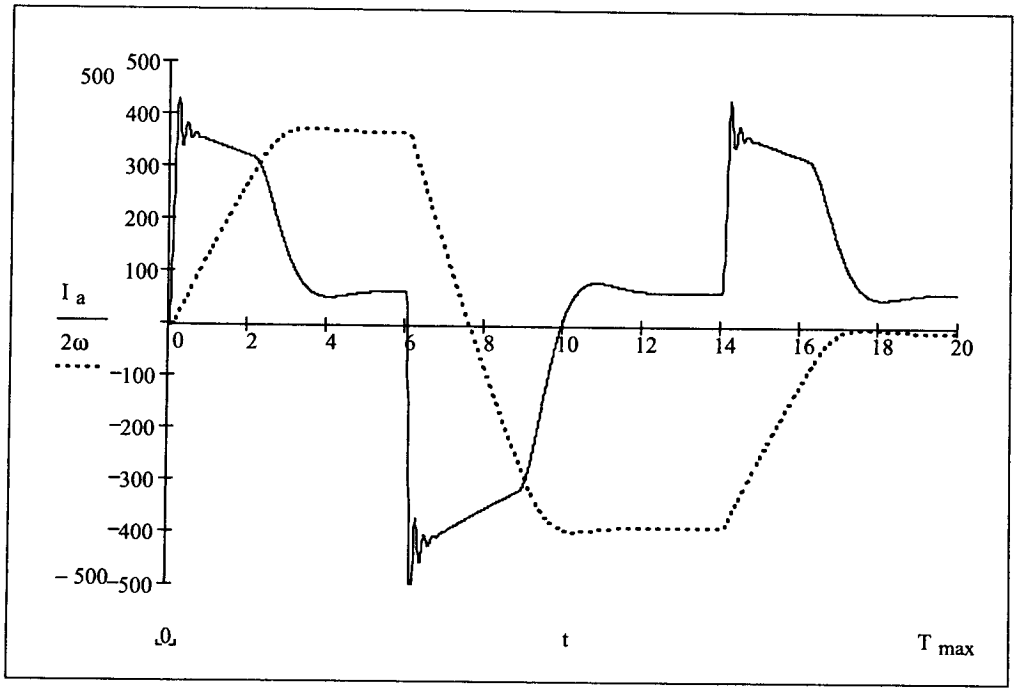
$$\text{Diff}(t,y) := \begin{bmatrix} \dfrac{K_d \cdot \left(U_z(t) - K_\omega \cdot y_3 - K_I \cdot \begin{cases} y_2 - I_{lim} \cdot \dfrac{y_2}{\lvert y_2 \rvert} & \text{if } \lvert y_2 \rvert > I_{lim} \\ 0 & \text{otherwise} \end{cases} \right) - y_0}{T_\mu} \\ \dfrac{K_G \cdot y_0 - y_1}{T_G} \\ \dfrac{\dfrac{y_1 - C \cdot y_3}{R_a} - y_2}{T_a} \\ \dfrac{C \cdot (y_2 - M_c)}{J} \end{bmatrix}$$

Розв'язуємо систему диф. рівнянь:

$$\text{Res} := \text{rkfixed}(y_0, 0, T_{max}, N, \text{Diff})$$

З матриці розв'язку виділяємо необхідні стовпці:

$$t := \text{Res}^{\langle 0 \rangle} \qquad I_a := \text{Res}^{\langle 3 \rangle} \qquad \omega := \text{Res}^{\langle 4 \rangle}$$



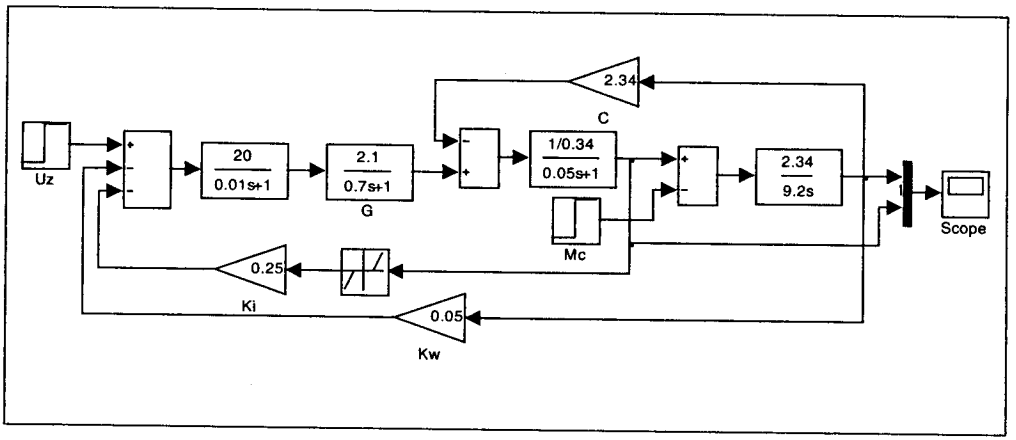
MATLAB

```
% Розрахунок перехідного процесу
% системи з паралельним коригуванням
Tmax = 20;              % Час розрахунку
y0(1:4) = 0;            % Початкові умови

% Розв'ЯзуваннЯ системи диференціальних рівнянь
[t, y] = ode23('parallel_correction', [0 Tmax], y0);

% Побудова графіка
plot(t, y(:,3), 'k', t, y(:,4), 'k:'); grid
xlabel('t, c'); ylabel('I_a, \omega'); legend('I_a', '\omega', 3);
```

```
function f = parallel_correction(t, y);
% ------------------------------------------------------------
% ФункціЯ, що описує модель системи з паралельним коригуваннЯм
%         y(1) - напруга ТП
%         y(2) - напруга генератора
%         y(3) - струм якорЯ
%         y(4) - швидкість
% ------------------------------------------------------------
Uznom = 20;          % Номінальна напруга завдання
Kw = 0.05;           % Коефіцієнт зворотного зв'Язку за швидкістю
Kc = 0.25;           % Коефіцієнт зворотного зв'Язку за струмом
Ialim = 320.0;       % Струм відсікання
Ktp = 20;            % Коефіцієнт підсиленнЯ ТП
Ttp = 0.01;          % Стала часу ТП
Kg = 2.1;            % Коефіцієнт підсиленнЯ ТП
Tg = 0.7;            % Стала часу ТП
C = 2.34;            % Стала двигуна
Ra = 0.34;           % Сумарний опір Якірного кола
Ta = 0.05;           % Стала часу Якірного кола
J = 9.2;             % Сумарний момент інерції привода
Mc = 64.5;           % Момент опору навантаженнЯ

f=zeros(4, 1);
% ФормуванняЯ напруги завданнЯ
Uz = Uznom;
if t > 6
   Uz = -Uznom;
end;
if t > 14
   Uz = 0;
end;
% ----------------------------------------
%    Модель тиристорного перетворювача
% ----------------------------------------
% Розрахунок вхідної напруги ТП
if abs(y(3)) < Ialim
    % Струмове відсіканнЯ відсутнє
    Ubx = Uz - Kw*y(4);
else
    % Струмове відсіканнЯ спрацювало
    Ubx = Uz - Kw*y(4) - Kc*(y(3) - Ialim*sign(y(3)));
end
f(1) = (Ubx*Ktp - y(1))/Ttp;
```

```
% Модель генератора
f(2) = (y(1)*Kg - y(2))/Tg;

% Модель Якірного кола ТП-Д
f(3) = ((y(2) - C*y(4))/Ra - y(3))/Ta;

% Обчислення похідної швидкості
f(4) = (C*y(3) - Mc)/J;
```

Simulink



Моделювання динамічних режимів одноразово інтегрувальної системи підпорядкованого регулювання (СПР) координат за схемою ТП–Д

Структурна схема одноразово інтегрувальної СПР швидкості двигуна постійного струму показана на рис. 4.5 [Д.3, Д.13, Д.21, Д.27].

Налагоджувальні параметри системи регулювання обчислюють за відомими виразами: $T_{ic} = 2T_{mn}K_{mn}K_i/R_a$; $K_{pc} = T_a/T_{ic}$; $K_{pш} = T_{ем}CK_i/4T_{mn}K_\omega R_a$; $K_i = U_i^{\max}/I_a^{\max}$; $K_\omega = U_\omega^{\max}/\omega_{\max}$. Значення $U_i^{\max}$, $U_\omega^{\max}$ – максимальні зна-

чення вихідних напруг відповідно давача струму і швидкості, приймають залежно від типу елементної бази системи автоматичного регулювання і найчастіше становлять ±10 В. Значення напруги завдання $U_з$ співмірне зі значенням $U_\omega^{\max}$ і залежить від усталеного значення кутової швидкості ω.

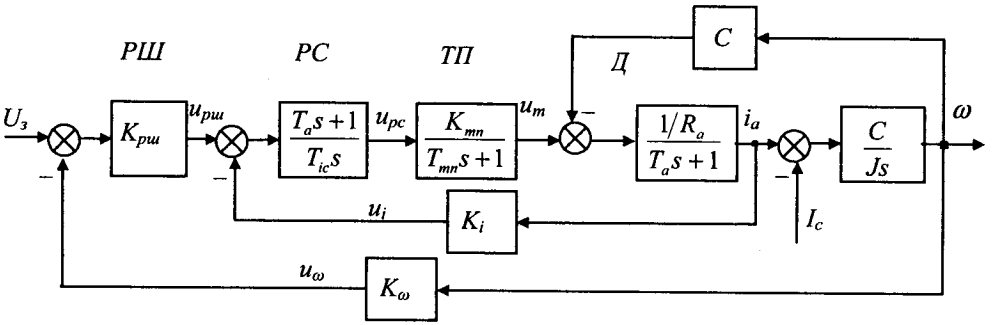


Рис. 4.5. Структурна схема СПР швидкості ДПС з одноразовим інтегруванням

Математична модель такої системи, записаної системою алгебричних і диференціальних рівнянь першого порядку, поданих у нормальній формі Коші, має вигляд (див. розд. 2):

$$u_{рш} = (U_з - K_\omega \omega) K_{рш}\ ;\ \text{необхідно передбачити обмеження}\ |u_{рш}| \le K_i I_a^{\max}\ ;$$

$$\frac{du_{ic}}{dt} = \frac{u_{рш} - K_i i_a}{T_{ic}};$$

$$\frac{du_{mn}}{dt} = \frac{\left(u_{ic} + (u_{рш} - K_i i_a) K_{pc}\right) K_{mn} - u_{mn}}{T_{mn}};$$

$$\frac{di_a}{dt} = \frac{(u_{mn} - C\omega)/R_a - i_a}{T_a};$$

$$\frac{d\omega}{dt} = \frac{(i_a - I_c) C}{J}.$$

Приклад: Нижче подано приклад розрахунку перехідних процесів режимів "запуск – реверс – зупинка" і зміни моменту навантаження в одноразово інтегрувальній системі підпорядкованого керування. Результати моделювання цих режимів показано на рис. 4.6.

Задано:

$U_{znom} = 10$ В	– номінальна напруга завдання;
$U_{Pmax} = 10$ В	– максимальна вихідна напруга регуляторів;
$K_\omega = 0.045$	– коефіцієнт зворотного зв'язку за швидкістю;
$K_I = 0.045$	– коефіцієнт зворотного зв'язку за струмом якоря;
$K_{P\omega} = 23.5$	– коефіцієнт підсилення регулятора швидкості;
$K_{Pi} = 0.34$	– коефіцієнт підсилення регулятора струму;
$T_{Pi} = 0.147$ с	– стала часу інтегратора регулятора струму;
$K_d = 50$	– коефіцієнт підсилення ТП;
$T_{тп} = 0.01$ с	– стала часу ТП;
$R_a = 0.34$ Ом	– сумарний опір якірного кола;
$C = 2.5$ В·с$^{-1}$	– стала двигуна;
$T_a = 0.05$ с	– електромагнітна стала часу якірного кола;
$M_{Cnom} = 100$ Н·м	– сумарний статичний момент інерції привода;
$J = 2.2$ кг·м^2	– момент інерції двигуна;
$T_{max} = 10$ с	– час розрахунку.

Quick Basic

```
DECLARE SUB Graphic (N%, Xin!, Yin!)
DECLARE SUB Felberg2B (N!, t!, h!, Eps!, y!())
DECLARE SUB dif (t!, y!(), f!())
' ========================================================
'  Розрахунок динаміки у режимах ЗАПУСК/РЕВЕРС/ЗУПИНКА
```

```
'  для одноразово інтегрувальної СПР електропривода
' ========================================================
CONST N = 4                 ' Порядок системи
CONST Tmax = 10             ' Максимальний час розрахунку
CONST Eps = .0001           ' Точність розв'язання на кроці
'    Необхідно для виведення графіка
COMMON SHARED Xmin, Xmax
Xmin = 0!                   ' Ліва межа графіка
Xmax = Tmax                 ' Права межа графіка
CONST Ymin = -250           ' Нижня межа графіка
CONST Ymax = 250            ' Верхня межа графіка
' ========================================================
DIM y(N)                    ' Опис масиву змінних
FOR i = 1 TO N              ' Обнулення масиву змінних
   y(i) = 0                 ' (нульові початкові умови)
NEXT i                      '
t = 0!
h = .001                    ' Початковий крок розв'язання
DO
   CALL Graphic(11, t, y(3))      ' Побудова графіка струму якоря
   CALL Graphic(12, t, y(4))      ' Побудова графіка швидкості
   CALL Felberg2B(N, t, h, Eps, y()) ' Виклик числового методу
LOOP UNTIL t > Tmax

END

SUB dif (t, y(), f())
' ------------------------------------------------------------
' Підпрограма призначена для обчислення значень правих частин
' системи диференціальних рівнянь, записаних у нормальній
' формі Коші
' Форма  запису:
'       f(1) = функція_1(y(1),y(2),...,y(N),t)
'       f(2) = функція_2(y(1),y(2),...,y(N),t)
'                       . . .
'       f(N) = функція_N(y(1),y(2),...,y(N),t)
'
' Вхідні величини:
'       t - незалежна змінна
'       y - масив змінних (координат електропривода)
' Вихідні величини:
'       y - масив змінних
'       f - масив похідних
```

```
' ------------------------------------------------------------
'     y(1) - інтегральна складова ПІ-регулятора струму
'     y(2) - напруга ТП
'     y(3) - струм якоря
'     y(4) - швидкість двигуна
' ------------------------------------------------------------
      Uz = 10
CONST KPW = 23.5    ' Коефіцієнт підсилення рег-ра швидкості
CONST KDW = .045    ' Коефіцієнт зворотного зв'язку за швидкістю
CONST UPWmax = 10!  ' Напруга обмеження рег-ра швидкості
CONST KPC = .34     ' Коефіцієнт підсилення рег-ра струму
CONST Ti = .147     ' Стала часу регулятора струму
CONST KDC = .045    ' Коефіцієнт зворотного зв'язку за струмом
CONST Ktp = 50      ' Коефіцієнт підсилення ТП
CONST Ttp = .01     ' Стала часу ТП
CONST C = 2.34      ' Стала двигуна
CONST Ra = .34      ' Сумарний опір якірного кола
CONST Ta = .05      ' Стала часу якірного кола ТП-Д
CONST J = 2.2       ' Сумарний момент інерції привода
CONST Mcnom = 100   ' Момент опору на валу
' Формування напруги завдання
IF t > 4.5 THEN Uz = -10
IF t > 7.5 THEN Uz = 0
'   Вихід регулятора швидкості
UPW = (Uz - KDW * y(4)) * KPW
'   Тепер обмежимо, за потреби, його вихідну напругу
IF ABS(UPW) > UPWmax THEN UPW = UPWmax * SGN(UPW)
'   Вихід регулятора струму
UPCbx = (UPW - KDC * y(3))      ' Вхідна напруга регулятора
UPCp = UPCbx * KPC              ' Пропорційна складова регулятора
f(1) = UPCbx / Ti               ' Інтегральна складова регулятора
Upc = UPCp + y(1)               ' Вихід ПІ-регулятора струму
'  Модель тиристорного перетворювача
f(2) = (Upc * Ktp - y(2)) / Ttp
'  Модель якірного кола ТП-Д
f(3) = ((y(2) - C * y(4)) / Ra - y(3)) / Ta
'  Реалізація зміни моменту навантаження
Mc = Mcnom
IF t > 2.5 THEN Mc = 2 * Mcnom
IF t > 3.5 THEN Mc = Mcnom
'   Розрахунок похідної швидкості
f(4) = (C * y(3) - Mc) / J
END SUB
```

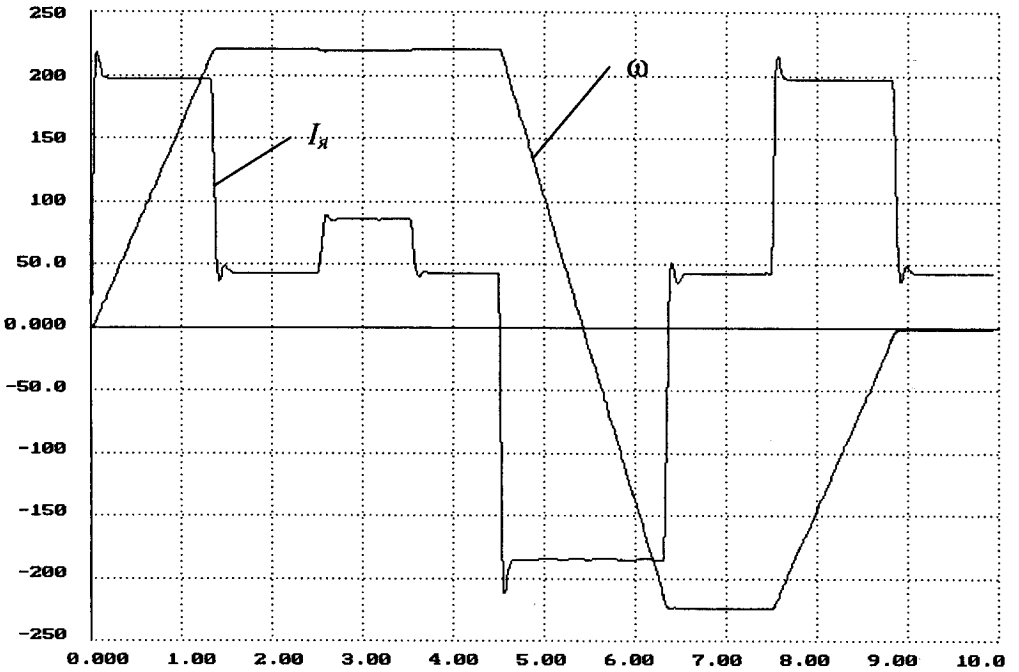


Рис. 4.6. Результати моделювання динаміки у системі ТП–Д з одноразовим інтегруванням

Turbo Pascal

```
{$E+,N+,F+ }
program SPR_1int;
{ ************************************************ }
uses
   F23B, Crt, Plot;
const
   N = 4;                    { Порядок системи              }
   Tmax = 10.0;              { Максимальний час розрахунку  }
   Eps  : DOUBLE = 1e-4;     { Точність на кроці }
   Ymin : DOUBLE = -250.0;
```

```
   Ymax : DOUBLE = 250.0;
var
   t, h  : DOUBLE;
   i  : INTEGER;
   y  : Vector;

procedure  dif(x: DOUBLE; var y, f : Vector);
{ ------------------------------------------------------- }
{ Процедура призначена для обчислення значень правих частин }
{ системи диференціальних рівнянь, записаних у нормальній   }
{ формі Коші.                                               }
{ Форма  запису:                                            }
{      f[1] = функція_1(y[1],y[2],...,y[N],x)               }
{      f[2] = функція_2(y[1],y[2],...,y[N],x)               }
{                    . . .                                  }
{      f[N] = функція_N(y[1],y[2],...,y[N],x)               }
{                                                           }
{ Вхідні величини:                                          }
{      x - незалежна змінна                                 }
{      y - масив змінних (координат електропривода)         }
{                                                           }
{ Вихідні величини:                                         }
{      y - масив змінних                                    }
{      f - масив похідних                                   }
{                                                           }
{   y[1] - інтегральна складова ПІ-регулятора струму        }
{   y[2] - напруга ТП                                       }
{   y[3] - струм якоря                                      }
{   y[4] - швидкість двигуна                                }
{ ------------------------------------------------------- }
const
   Uznom = 10.0;  { Напруга завдання для номінальної швидкості }
   Kpw = 23.5;    { Коефіцієнт підсилення рег-ра швидкості    }
   KDW = 0.045;    { Коефіцієнт зворотного зв'язку за швидкістю }
   UPWmax = 10.0; { Напруга обмеження рег-ра швидкості        }
   Kpc = 0.34;    { Коефіцієнт підсилення рег-ра струму       }
   Tpc = 0.147;   { Стала часу інтегратора РС                 }
   KDC = 0.045;    { Коефіцієнт зворотного зв'язку за струмом }
   Ed0 = 514.8;   { Напруга неробочого ходу ТП                }
   Ktp = 50.0;    { Коефіцієнт підсилення ТП                  }
   Ttp = 0.01;    { Стала часу ТП                             }
   C = 2.34;      { Стала двигуна                             }
   Ra = 0.34;     { Сумарний опір якірного кола               }
```

```
   Ta = 0.05;        { Стала часу якірного кола                }
   J = 2.2;          { Сумарний момент інерції привода         }
   Mcnom = 100.0;    { Номінальний момент опору на валу        }
var
   Upw : DOUBLE;
   Upc , UPCbx , UPCp : DOUBLE;
   Uz , Mc            : DOUBLE;
begin
{  Формування напруги завдання  }
Uz := Uznom;
if t > 4.5 then Uz := -Uznom;
if t > 7.5 then Uz := 0.0;

{  Вихід регулятора швидкості   }
Upw := (Uz - KDW*y[4])*Kpw;
{  Тепер обмежимо, якщо необхідно, його вихідну напругу  }
if abs(Upw) > UPWmax then Upw := UPWmax * abs(Upw)/Upw;

{   Вихід регулятора струму                                  }
UPCbx := (Upw - KDC * y[3]); { Вхідна напруга регулятора  }
UPCp := UPCbx * Kpc;          { Пропорційна складова регулятора }
f[1] := UPCbx / Tpc;          { Інтегральна складова регулятора }
Upc := UPCp + y[1];           { Вихід ПІ-регулятора струму    }

{  Модель тиристорного перетворювача  }
f[2] := (Upc*Ktp - y[2])/Ttp;  { Врахування інерційності ТП }

{  Модель якірного кола ТП-Д  }
f[3] := ((y[2] - C*y[4])/Ra - y[3])/Ta;

{   Опис механічної частини привода      }
{ Реалізація зміни моменту навантаження }
Mc := Mcnom;
if t > 2.5 then Mc := 2.0 * Mcnom;
if t > 3.5 then Mc := Mcnom;

{   Розрахунок похідної швидкості        }
f[4] := (C*y[3] - Mc)/J;
end;
{ ================================== }
{      Основна частина програми      }
{ ================================== }
begin
```

```
   h := 0.001;  { Початковий крок   }
   {   Ініціалізація графіка     }
   InitGraphic(0, Tmax, Ymin, Ymax, 1.0, 50.0);
   { Початкові умови }
   for  i := 1  to  N  do y[i] := 0.0;
{ ---------------------------------- }
{  Розрахунок динаміки електропривода   }
{ ---------------------------------- }
   repeat
      Fehlberg23B(Dif, N, t, h, Eps, y);
      Graphic(1, t, y[3]);  { виведення на екран графіка Ia }
      Graphic(2, t, y[4]);  { виведення на екран графіка w  }
   until  t > Tmax;
   ReadLn;  { Натиснути ENTER для продовження }
end.
```

MathCAD

$U_{znom} := 10$	*Номінальна напруга завдання*
$U_{Pmax} := 10$	*Максимальна вихідна напруга регуляторів*
$K_{\omega} := 0.045$	*Коефіцієнт зворотного зв'язку за швидкістю*
$K_i := 0.045$	*Коефіцієнт зворотного зв'язку за струмом якоря*
$K_{P\omega} := 23.5$	*Коефіцієнт підсилення регулятора швидкості*
$K_{Pi} := 0.34$	*Коефіцієнт підсилення регулятора струму*
$T_{Pi} := 0.147$	*Стала часу інтегратора регулятора струму*
$K_d := 50$	*Коефіцієнт підсилення ТП*
$T_{\mu} := 0.01$	*Стала часу ТП*
$R_a := 0.34$	*Сумарний опір якірного кола*
$C := 2.5$	*Стала двигуна*
$T_a := 0.05$	*Електромагнітна стала часу якірного кола*
$M_{Cnom} := 100$	*Номінальний статичний момент навантаження*
$J := 2.2$	*Сумарний момент інерції привода*
$T_{max} := 10$	*Час розрахунку*
$N := 1000$	*Кількість точок інтегрування*

Напруга завдання:

$$U_z(t) := \begin{cases} Uz_{nom} & \text{if } t < 4.5 \\ -Uz_{nom} & \text{if } 4.5 \le t < 7.5 \\ 0 & \text{otherwise} \end{cases}$$

Статичний момент:

$$M_c(t) := \begin{cases} 2 \cdot M_{cnom} & \text{if } 2.5 < t < 3.5 \\ M_{cnom} & \text{otherwise} \end{cases}$$

Початкові умови:

$$y_0 := \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Вектор-функція правих частин системи диференціальних рівнянь, що описують систему:

$$Diff(t, y) := \begin{cases} U_{P\omega} \leftarrow \left(U_z(t) - K_{\omega} \cdot y_3\right) \cdot K_{P\omega} \\ U_{P\omega} \leftarrow Up_{max} \cdot \dfrac{U_{P\omega}}{\left|U_{P\omega}\right|} & \text{if } \left|U_{P\omega}\right| > Up_{max} \\ U_{bxPC} \leftarrow U_{P\omega} - K_i \cdot y_2 \\ f_0 \leftarrow \dfrac{U_{bxPC}}{T_{Pi}} \\ U_{Pi} \leftarrow y_0 + K_{Pi} \cdot U_{bxPC} \end{cases}$$

$$f_1 \leftarrow \frac{K_d \cdot U_{Pi} - y_1}{T_\mu}$$

$$f_2 \leftarrow \frac{\frac{y_1 - C \cdot y_3}{R_a} - y_2}{T_a}$$

$$f_3 \leftarrow \frac{C \cdot y_2 - M_c(t)}{J}$$

f

Позначення змінних:

$U_{P\omega}$ - *напруга РШ;*

U_{bxPC} - *напруга на вході РС;*

U_{Pi} - *напруга РС;*

y_0 - *напруга інтегратора РС;*

y_1 - *напруга ТП;*

y_2 - *струм якоря;*

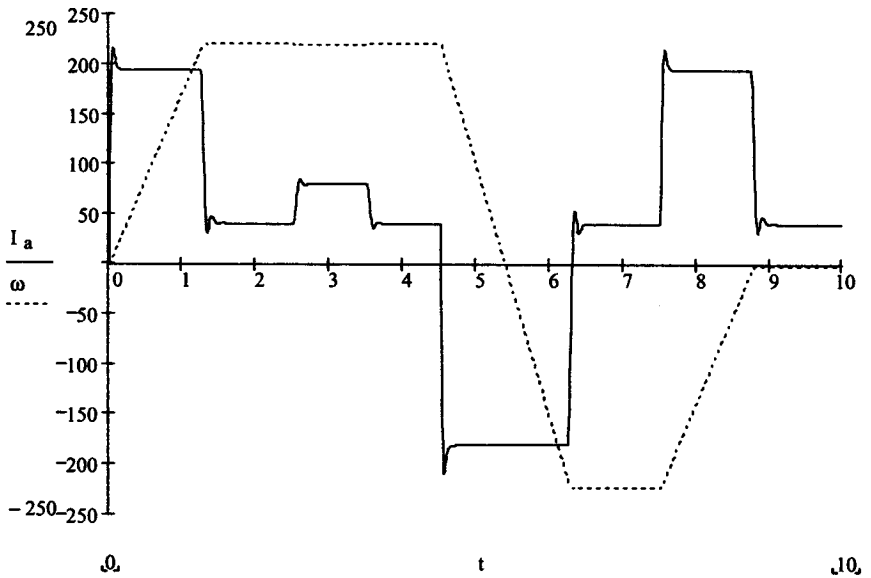
y_3 - *швидкість.*

Розв'язуємо систему диф. рівнянь:

$$Res := rkfixed(y_0, 0, T_{max}, N, Diff)$$

З матриці розв'язку виділяємо необхідні стовпці:

$t := Res^{<0>}$ $\quad I_a := Res^{<3>}$ $\quad \omega := Res^{<4>}$



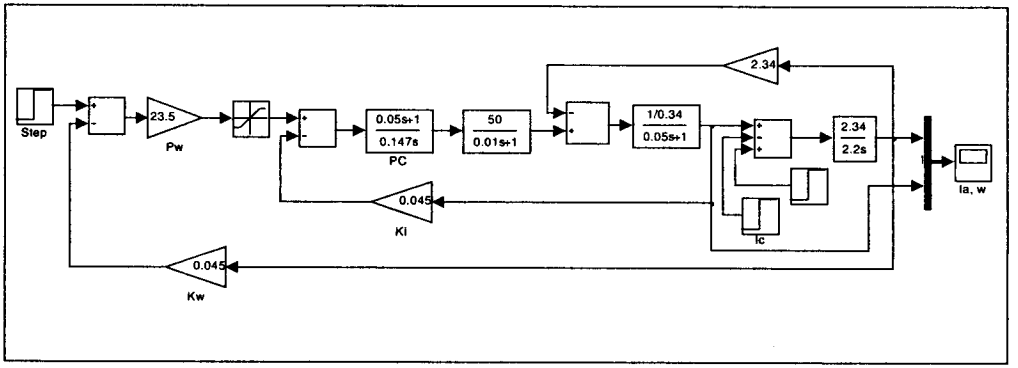
MATLAB

```
% Розрахунок динаміки одноразово інтегрувальної СПР
Tmax = 10;              % Час розрахунку
y0(1:4) = 0;            % Початкові умови

% Розв'ЯзуваннЯ системи диференціальних рівнЯнь
 [t, y] = ode23('sprlint', [0 Tmax], y0);
% Побудова графіка
plot(t, y(:,3), 'k', t, y(:,4), 'k:'); grid
xlabel('t, c'); ylabel('I_a, \omega'); legend('I_a', '\omega', 3);
```

```
function f = sprlint(t, y);
% ------------------------------------------------------------
%  Функція, що описує модель одноразово інтегрувальної САР ЕП
%        y(1) - інтегральна складова РС
%        y(2) - напруга ТП
%        y(3) - струм ЯкорЯ
%        y(4) - швидкість
% ------------------------------------------------------------
Uznom = 10;         % Номінальна напруга завданнЯ
Kpw = 23.5;         % Коефіцієнт підсилення рег-ра швидкості
KDW = 0.045;        % Коефіцієнт зворотного зв'Язку за швидкістю
UPWmax = 10;        % Напруга обмеженнЯ рег-ра швидкості
Kpc = 0.34;         % КоефіцієнтпідсиленнЯ рег-ра струму
Tpc = 0.147;        % Стала часу інтегратора РС
KDC = 0.045;        % Коефіцієнт зворотного зв'Язку за струмом
Ktp = 50;           % КоефіцієнтпідсиленнЯ ТП
Ttp = 0.01;         % Стала часу ТП
C = 2.34;           % Стала двигуна
Ra = 0.34;          % Сумарний опір Якірного кола
Ta = 0.05;          % Стала часу Якірного кола
J = 2.2;            % Сумарний момент інерції привода
Mcnom = 100;        % Момент опору на валу

% Розрахунок правих частин системи диференціальних рівнЯнь
f=zeros(4, 1);
% ФормуваннЯ напруги завданнЯ
```

```
Uz = Uznom;
if t > 4.5
   Uz = -Uznom;
end;
if t > 7.5
   Uz = 0;
end;
% ==========================================
%       РегулЯтор швидкості
% ==========================================
Upw = (Uz - KDW*y(4))*Kpw;
%   Тепер обмежимо, Якщо необхідно, його вихідну напругу
if abs(Upw) > UPWmax
  Upw = UPWmax * sign(Upw);
end
% ==========================================
%       РегулЯтор струму
% ==========================================
UPCbx = (Upw - KDC * y(3));   % Вхідна напруга регулЯтора
UPCp = UPCbx * Kpc;           % Пропорційна складова регулЯтора
f(1) = UPCbx / Tpc;           % Інтегральна складова регулЯтора
Upc = UPCp + y(1);            % Вихід ПІ-регулЯтора
% ----------------------------------------
%    Модель  тиристорного  перетворювача
% ----------------------------------------
f(2) = (Upc*Ktp - y(2))/Ttp;

%   Опис Якірного кола ТП-Д
f(3) = ((y(2) - C*y(4))/Ra - y(3))/Ta;

%  Реалізація зміни моменту навантаженнЯ
Mc = Mcnom;
if t > 2.5
   Mc = 2 * Mcnom;
end
if t > 3.5
   Mc = Mcnom;
end
%  Розрахунок похідної швидкості
f(4) = (C*y(3) - Mc)/J;
```

Simulink



Моделювання динамічних режимів одноразово інтегрувальної СПР координат за схемою Г–Д за допомогою Z-перетворення

Приклад: З використанням Z-перетворення розрахувати перехідні процеси в одноразово інтегрувальній системі підпорядкованого керування системою Г–Д з урахуванням нелінійностей тиристорного перетворювача та генератора постійного струму. Результати моделювання динамічних режимів системи електропривода показано на рис. 4.7.

Задано:

$U_{znom} = 10$ В	– номінальна напруга завдання швидкості;
$U_{Pmax} = 10$ В	– максимальна вихідна напруга регуляторів;
$K_\omega = 0.055$	– коефіцієнт зворотного зв'язку за швидкістю;
$K_I = 0.06$	– коефіцієнт зворотного зв'язку за струмом якоря;
$K_{P\omega} = 10.5$	– коефіцієнт підсилення регулятора швидкості;
$K_{Pi} = 3.4$	– коефіцієнт підсилення регулятора струму;
$T_{Pi} = 0.278$ с	– стала часу інтегратора регулятора струму;
$E_{d0} = 514.8$ В	– ЕРС неробочого ходу ТП;
$U_{Gном} = 460$ В	– номінальна напруга якоря генератора;
$U_{dном} = 220$ В	– номінальна напруга збудження генератора;
$T_G = 1.5$ с	– стала часу обмотки збудження генератора;

$R_a = 0.1$ Ом – сумарний опір якірного кола;

$C = 2.34$ В·с$^{-1}$ – стала двигуна;

$T_a = 0.05$ с – електромагнітна стала часу якірного кола;

$M_C = 64.5$ Н·м – статичний момент навантаження;

$J = 3.8$ кг·м^2 – сумарний момент інерції привода;

$T_{max} = 5$ с – час розрахунку.

Пояснення до програми. Застосування Z-перетворення [Г.3, Г.6, Г.7] дає змогу отримати доволі прості, але водночас швидкодійні програми, яким не притаманне поняття числової нестійкості методу. Для моделювання використано готові різницеві рівняння, що отримані за допомогою Z-перетворення (див. розд. 3 і [Г.6]):

для інтегратора:

$$y_{i+1} = y_i + \frac{h}{T} x_i ;$$

для аперіодичної ланки:

$$y_{i+1} = y_i e^{-\frac{h}{T}} + \left(1 - e^{-\frac{h}{T}}\right) x_i ,$$

де h – крок моделювання;

T – стала часу (інтегратора чи аперіодичної ланки);

y_i, y_{i+1} – вихідний сигнал в i-й та i+1-й моменти часу відповідно;

x_i – вхідний сигнал в i-й момент часу.

Імпульсний характер роботи ТП імітується використанням фіксованого кроку розв'язку $h = 0.01$ с. Регулювальна характеристика ТП подається залежністю

$$E_d = E_{d0} \sin(\alpha) ,$$

де $\alpha = \alpha_{min} + (\alpha_{max} - \alpha_{min}) \dfrac{|U_{вх}|}{U_{вх}^{max}}$ – кут регулювання ТП (для лінійної залежності кута відкривання тиристорів від керуючої напруги);

α_{min} – мінімальний кут відкривання тиристорів перетворювача;

α_{max} – максимальний кут відкривання тиристорів перетворювача;

$U_{вх}$ – вхідна напруга керування ТП;

$U_{вх}^{max}$ – максимальна вхідна напруга керування ТП.

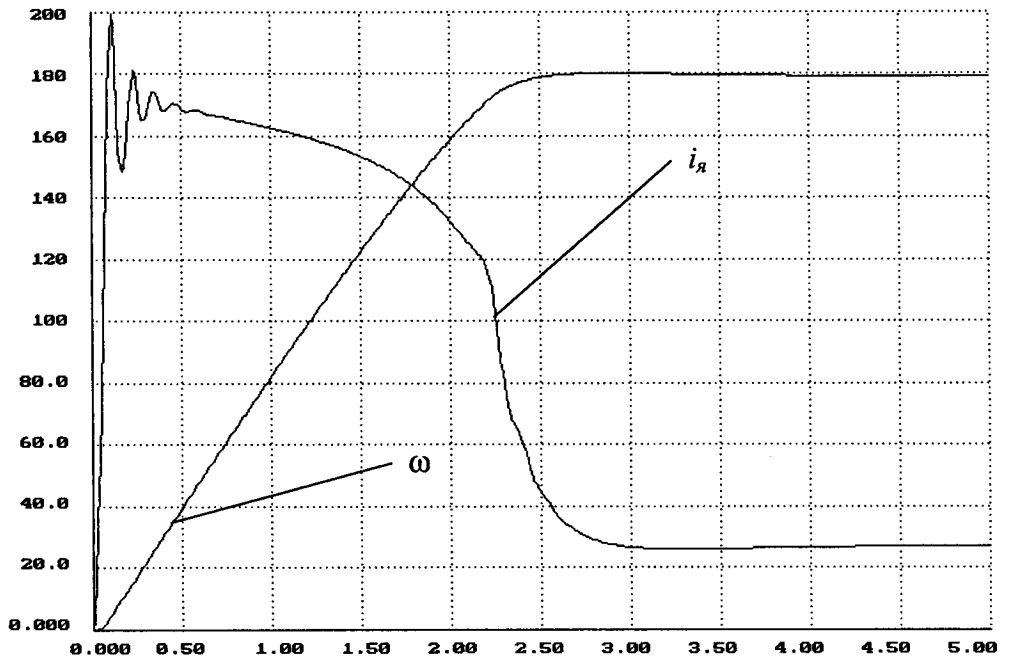


Рис. 4.7. Перехідні процеси в системі з одноразовим інтегруванням і врахуванням нелінійностей ТП і генератора

Для врахування нелінійності характеристики намагнічування генератора використовується арктангенційна залежність

$$E_G = A \cdot \text{arctg}(B \cdot U_d) \text{ [Д.2]},$$

де A, B – коефіцієнти апроксимації кривої намагнічування;

U_d – напруга збудження генератора.

Коефіцієнти залежності $E_G(U_d)$ можна отримати двома способами:

1) за простішим способом коефіцієнти одержують інтерполяцією за двома вузлами кривої намагнічування: (0; 0) і ($U_d^{ном}$; $E_G^{ном}$), тоді $A = E_G^{ном}$; $B = \tan(1)/U_d^{ном}$ – у такому разі точність відтворення кривої намагнічування не нижча за 5–10 %, що часто є достатнім для багатьох практичних застосувань;

2) для отримання точнішої моделі значення коефіцієнтів треба отримати за мінімумом суми середньоквадратичних відхилень від експериментальних точок (наприклад, за методом найменших квадратів). У такому разі точність апроксимації кривої намагнічування буде не гіршою за 2–3 % (див. розд. 2).

Quick Basic

```
CONST Tmax = 5          ' Максимальний час розрахунку
CONST h = .01           ' Крок моделювання
CONST Xmin = 0!         ' Ліва межа графіка
CONST Xmax = Tmax       ' Права межа графіка
CONST Ymin = 0          ' Нижня межа графіка
CONST Ymax = 200        ' Верхня межа графіка
Ug = 0                  ' Початкова умова для напруги генератора
Ia = 0                  ' Початкова умова для струму якоря
W = 0                   ' Початкова умова для швидкості
UPCi = 0                ' Початкова умова для ПІ-регулятора струму
      Uz = 10!          ' Напруга завдання  Uз
CONST KPW = 10.5        ' Коефіцієнт підсилення регулятора швидкості
CONST KDW = .055        ' Коефіцієнт зворотного зв'язку за швидкістю
CONST UPWmax = 10!      ' Напруга обмеження регулятора швидкості
CONST Kpc = 3.4         ' Коефіцієнт підсилення РС
CONST Tpc = .278        ' Стала часу інтегрування РС
CONST KDC = .06         ' Коефіцієнт зворотного зв'язку за струмом
CONST Ed0 = 514.8       ' Напруга неробочого ходу ТП
CONST Ugnom = 460       ' Номінальна напруга якоря генератора
CONST Uozgnom = 220     ' Номінальна напруга збудження генератора
CONST Tg = 1.5          ' Стала часу обмотки збудження генератора
CONST C = 2.34          ' Стала двигуна
CONST Ra = .1           ' Сумарний опір якірного кола
CONST Ta = .05          ' Стала часу якірного кола
CONST J = 3.8           ' Сумарний момент інерції привода
CONST Mc = 64.5         ' Момент опору на валу
'   Допоміжні змінні
ETg = EXP(-h / Tg)      ' Застосовується у моделі генератора
ETa = EXP(-h / Ta)      ' Застосовується у моделі якірного кола
' ------------------------------------
'          Основна частина програми
' ------------------------------------
```

```
FOR T = 0 TO Tmax STEP h
   CALL Graphic(11, T, Ia)       ' Побудова графіка струму якоря
   CALL Graphic(12, T, W)        ' Побудова графіка швидкості

'   Модель регулятора швидкості
UPW = (Uz - KDW * W) * KPW
'   Тепер обмежимо, якщо необхідно, вихідну напругу РШ
IF ABS(UPW) > UPWmax THEN UPW = UPWmax * SGN(UPW)

'   Модель регулятора струму
UPCbx = (UPW - KDC * Ia)        ' Вхідна напруга регулятора
UPCp = UPCbx * Kpc              ' Пропорційна складова регулятора
UPCi = UPCi + h / Tpc * UPCbx   ' Інтегральна складова регулятора
UPC = UPCp + UPCi               ' Вихід ПІ-регулятора струму
' ----------------------------------------
'    Модель тиристорного перетворювача
' ----------------------------------------
CONST Amin = .2     ' Мінімальний кут відкривання ТП в радіанах
CONST Amax = 1.57   ' Максимальний кут відкривання ТП в радіанах
CONST Ubxmax = 10!  ' Максимальна вхідна напруга ТП
Ubx = UPC           ' Вхідна напруга ТП
'   Обмеження вхідної напруги СІФК
IF Ubx > Ubxmax THEN Ubx = Ubxmax
IF Ubx < -Ubxmax THEN Ubx = -Ubxmax
' Розрахунок кута керування СІФК (лінійна хар-ка керування)
A = Amin + (Amax - Amin) * (Ubxmax - abs(Ubx)) / Ubxmax
'   Вихідна напруга ТП
Ed = Ed0 * SIN(A) * SGN(Ubx)
' ------------------------------------------------------------
'          Модель генератора постійного струму
' ------------------------------------------------------------
CONST Ag = Ugnom                    ' Коефіцієнт апроксимації
CONST Bg = 1.557408 / Uozgnom       ' Коефіцієнт апроксимації
Ug = Ug * ETg + (1 - ETg) * Ag * ATN(Bg * Ed)

'   Модель якірного кола. Розрахунок струму якоря
Ia = Ia * ETa + (1 - ETa) * (Ug - C * W) / Ra

'   Модель механічної частини привода. Розрахунок швидкості
W = W + h / J * (C * Ia - Mc)
NEXT T
END
```

Turbo Pascal

```
{$E+,N+}
program Demo_Z;
{  Використання методу Z-перетворення на прикладі     }
{  розрахунку динаміки режиму запуску електропривода }
uses
   Crt, Plot;
const
   Tmax : DOUBLE = 5.0;       { Максимальний час розрахунку  }
   h    : DOUBLE = 0.01;      { Крок моделювання             }
   Tmin : DOUBLE = 0.0;       { Ліва межа графіка            }
   Ymin : DOUBLE = 0.0;       { Нижня межа графіка           }
   Ymax : DOUBLE = 200.0;     { Верхня межа графіка          }
   { Параметри моделі привода }
   Uz   : DOUBLE = 10.0;  { Напруга завдання  Uз             }
   KPW  : DOUBLE = 10.5;  { Коефіцієнт підсилення РШ         }
   KDW  : DOUBLE = 0.055; { Коефіцієнт зв. зв. за швидкістю  }
   UPWmax : DOUBLE = 10.0;{ Напруга обмеження РШ             }
   KPC  : DOUBLE = 3.4;   { Коефіцієнт підсилення РС         }
   Tpc  : DOUBLE = 0.278; { Стала часу інтегрування РС       }
   KDC  : DOUBLE = 0.06;  { Коефіцієнт зв. зв. за струмом    }
   Ed0  : DOUBLE = 514.8; { Напруга неробочого ходу ТП       }
   C    : DOUBLE = 2.34;  { Стала двигуна                    }
   Ra   : DOUBLE = 0.1;   { Сумарний опір якірного кола      }
   Ta   : DOUBLE = 0.05;  { Стала часу якірного кола         }
   J    : DOUBLE = 3.8;   { Сумарний момент інерції привода  }
   Mc   : DOUBLE = 64.5;  { Момент опору на валу             }
   { Параметри моделі тиристорного перетворювача }
   Amin : DOUBLE = 0.2;     { Мінім. кут відкривання ТП в радіанах }
   Amax : DOUBLE = 1.57;    { Макс. кут відкривання ТП в радіанах  }
   Ubxmax : DOUBLE = 10.0; { Максимальна напруга керування ТП }
   { Параметри моделі генератора  }
   Ugnom   = 460.0;    { Номінальна напруга якоря генератора  }
   Uozgnom = 220.0;    { Номінальна напруга збудження генератора }
   Tg : DOUBLE = 1.5;  { Стала часу обмотки збудження генератора }
   Ag : REAL = Ugnom;                  { Коефіцієнт апроксимації }
   Bg : DOUBLE = 1.557408/Uozgnom; { Коефіцієнт апроксимації }
var
   UPW, UPC, UPCbx, UPCi, UPCp : DOUBLE;
```

```
   Ubx, A, Ed, Ug, Ia, W      : DOUBLE;
   ETg, ETa    : DOUBLE;
   t           : DOUBLE;

begin
   Ug := 0;          { Початкова умова для напруги генератора }
   Ia := 0;          { Початкова умова для струму якоря       }
   W := 0;           { Початкова умова для швидкості          }
   UPCi := 0;        { Початкова умова для ПІ-регулятора струму }
   t := Tmin;
   {   Допоміжні змінні   }
   ETg := exp(-h/Tg); { Використовується у моделі генератора }
   ETa := exp(-h/Ta);  { Застосовується у моделі якірного кола }

   {  Ініціалізація графіка   }
   InitGraphic(0, Tmax, Ymin, Ymax, 1, 20.0);
   { ------------------------------- }
   {      Основна частина програми     }
   { ------------------------------- }
   repeat
      Graphic(1, t, Ia);     { Побудова графіка струму якоря }
      Graphic(2, t, W);      { Побудова графіка швидкості    }

     { Модель регулятора швидкості  }
      UPW := (Uz - KDW*W)*KPW;
     { Тепер обмежимо, якщо необхідно, вихідну напругу РШ }
      if abs(UPW) > UPWmax then UPW := UPWmax * UPW/abs(UPW);

      { Модель регулятора струму   }
      UPCbx := (UPW - KDC * Ia); { Вхідна напруга регулятора }
      UPCp := UPCbx * KPC;       { Пропорційна складова регулятора }
      UPCi := UPCi+h/Tpc*UPCbx; { Інтегральна складова
регулятора }
      UPC := UPCp + UPCi;        { Вихід ПІ-регулятора струму }

      { ------------------------------------ }
      { Модель тиристорного перетворювача  ТП }
      { ------------------------------------ }
      Ubx := UPC;          { Вхідна напруга ТП  }
      {  Обмеження вхідної напруги СІФК        }
      if  Ubx > Ubxmax   then Ubx := Ubxmax;
      if  Ubx < -Ubxmax  then Ubx := -Ubxmax;
      { Розрахунок кута керування СІФК      }
```

```
        { (лінійна характеристика керування)  }
        A := Amin + (Amax - Amin)*(Ubxmax - ABS(Ubx))/Ubxmax;
        { Вихідна напруга ТП   }
        if Ubx >= 0.0 then
           Ed := Ed0*cos(A)
        else
           Ed := -Ed0*cos(A);
        { ------------------------------------ }
        { Модель генератора постійного струму  }
        { ------------------------------------ }
        Ug := Ug*ETg + (1.0 - ETg)*Ag*ArcTan(Bg*Ed);

        {  Модель якірного кола. Розрахунок струму якоря }
        Ia := Ia*ETa + (1.0 - ETa)*(Ug - C*W)/Ra;

         { Модель механічної частини привода. Розрахунок швидкості }
        W := W + h/J*(C*Ia - Mc);

        t := t + h;
    until  t > Tmax;
    ReadLn;    { Натиснути ENTER для продовження }
end.
```

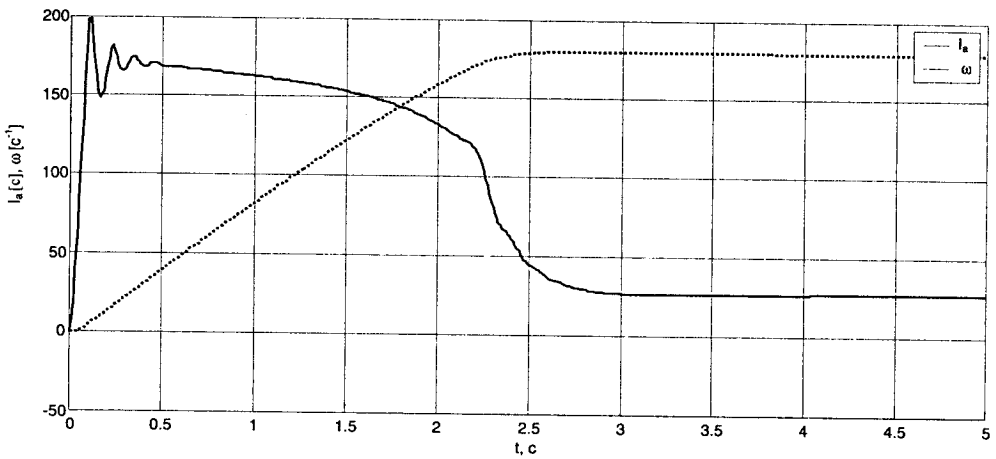


Рис. 4.8. Отримані в середовищі MATLAB перехідні процеси в системі Г–Д з одноразовим інтегруванням і врахуванням нелінійностей ТП і генератора

MATLAB

```
Tmax = 5;          % Максимальний час розрахунку
h = 0.01;          % Початковий крок розв'язування
N = Tmax/h;        % Кількість точок

Uz = 10;           % Напруга завдання  Uз
KPW = 10.5;        % Коефіцієнт підсилення регулятора швидкості
KDW = 0.055;       % Коефіцієнт зворотного зв'язку за швидкістю
UPWmax = 10;       % Напруга обмеження регулятора швидкості
Kpc = 3.4;         % Коефіцієнт підсилення РС
Tpc = 0.278;       % Стала часу інтегрування РС
KDC = 0.06;        % Коефіцієнт зворотного зв'язку за струмом
Ed0 = 514.8;       % Напруга неробочого ходу ТП
Ugnom = 460;       % Номінальна напруга якоря генератора
Uozgnom = 220;     % Номінальна напруга збудження генератора
Tg = 1.5;          % Стала часу обмотки збудження генератора
Ag = Ugnom;                % Коефіцієнти залежності кривої
Bg = 1.557408/Uozgnom;     % Намагнічування генератора
C = 2.34;          % Стала двигуна
Ra = 0.1;          % Сумарний опір якірного кола
Ta = 0.05;         % Стала часу якірного кола
J = 3.8;           % Сумарний момент інерції привода
Mc = 64.5;         % Момент опору на валу

Amin = 0.2;        % Мінімальний кут відкривання ТП в радіанах
Amax = 1.57;       % Максимальний кут відкривання ТП в радіанах
Ubxmax = 10;       % Максимальна вхідна напруга ТП

t = zeros(N+1,1);        % Час
Ug = t;          % Початкова умова для напруги генератора
Ia = t;          % Початкова умова для струму якоря
w = t;           % Початкова умова для швидкості
UPCi = 0;        % Початкова умова для ПІ-регулятора струму

%   Допоміжні змінні
ETg = exp(-h/Tg);      % Використовується у моделі генератора
ETa = exp(-h/Ta);      % Застосовується у моделі якірного кола
% ---------------------------------------
%  Основна частина програми
% ---------------------------------------
```

```
for i = 1:N
   t(i) = (i-1)*h;
   %   Модель регулЯтора швидкості
   UPW = (Uz - KDW*w(i))*KPW;
   %   Тепер обмежимо, якщо необхідно, вихідну напругу РШ
   if abs(UPW) > UPWmax
      UPW = UPWmax * sign(UPW);
   end

   %  Модель регулЯтора струму
   UPCbx = (UPW - KDC*Ia(i));  % Вхідна напруга регулЯтора
   UPCp = UPCbx * Kpc;         % Пропорційна складова регулЯтора
   UPCi = UPCi + h/Tpc*UPCbx;  % Інтегральна складова регулЯтора
   UPC = UPCp + UPCi;          % Вихід ПІ-регулЯтора струму
   % ----------------------------------------
   %    Модель тиристорного перетворювача
   % ----------------------------------------
   Ubx = UPC;                  % Напруга керування ТП
   %   ОбмеженнЯ вхідної напруги СІФК
   if Ubx > Ubxmax
      Ubx = Ubxmax*sign(Ubx);
   end
   % Розрахунок кута керуванЯ СІФК
   % (лінійна характеристика керування)
   A = Amin + (Amax - Amin)*Ubx/Ubxmax;
   %   Вихідна напруга ТП
   Ed = Ed0 * sin(A);
   % ------------------------------------------------------
   %          Модель генератора постійного струму
   % ------------------------------------------------------
   Ug(i+1) = Ug(i)*ETg + (1 - ETg)*Ag*atan(Bg*Ed);

   % Модель Якірного кола. Розрахунок струму ЯкорЯ
   Ia(i+1) = Ia(i)*ETa + (1 - ETa)*(Ug(i+1) - C*w(i))/Ra;

   % Модель механічної частини привода. Розрахунок швидкості
   w(i+1) = w(i) + h/J*(C*Ia(i+1) - Mc);
end
t(N+1) = N*h;
plot(t, Ia, 'k-', t, w, 'k:'),grid
xlabel('t, c'); ylabel('I_a [c], \omega [c^{-1}]');
legend('I_a', '\omega');
```

Моделювання динамічних режимів дворазово інтегрувальної системи підпорядкованого регулювання (СПР) координат за схемою ТП–Д

Структурна схема моделі дворазово інтегрувальної СПР електропривода постійного струму за системою ТП–Д показана на рис. 4.9. Математична модель такої системи електропривода подається системою рівнянь (див. розд. 2):

$$\frac{du_{зі}}{dt} = \frac{U_з - u_{зі}}{T_{зі}};$$

$$\frac{du_{іш}}{dt} = \frac{u_{зі} - K_\omega \omega}{T_{іш}};$$

$u_{рш} = u_{іш} + (u_{зі} - K_\omega \omega) K_{рш}$; необхідно передбачити обмеження $|u_{рш}| \le u_{рш}^{\max}$;

$$\frac{du_{іс}}{dt} = \frac{u_{рш} - K_i i_a}{T_{іс}};$$

$$u_{рс} = u_{іс} + (u_{рш} - K_i i_a) K_{рс};$$

$$\frac{du_{тп}}{dt} = \frac{u_{рс} K_{тп} - u_{тп}}{T_{тп}};$$

$$\frac{di_a}{dt} = \frac{(u_{тп} - C\omega)/R_a - i_a}{T_a};$$

$$\frac{d\omega}{dt} = \frac{(i_a - I_c)C}{J}.$$

Важливим моментом під час моделювання ПІ-регулятора з обмеженням вихідної координати (наприклад, регулятора швидкості) є правильна реалізація в моделі власне обмеження: після досягнення максимального значення варто не просто обмежити умовним оператором рівень виходу регулятора, а й зупинити інтегрування (див. розд. 2).

Параметри налагодження регулятора контуру швидкості за "технічним оптимумом" у дворазово інтегрувальній СПР розраховують за формулами:

$$T_{зі} = 8T_{\mu}\,;\quad T_{іш} = \frac{32T_{\mu}^2 K_{\omega} R_a}{T_{ем} K_i C}\,;\quad K_{рш} = 8T_{\mu}/T_{іш}\,;\quad T_{\mu} = T_{mn} + T_{ф}\,.$$

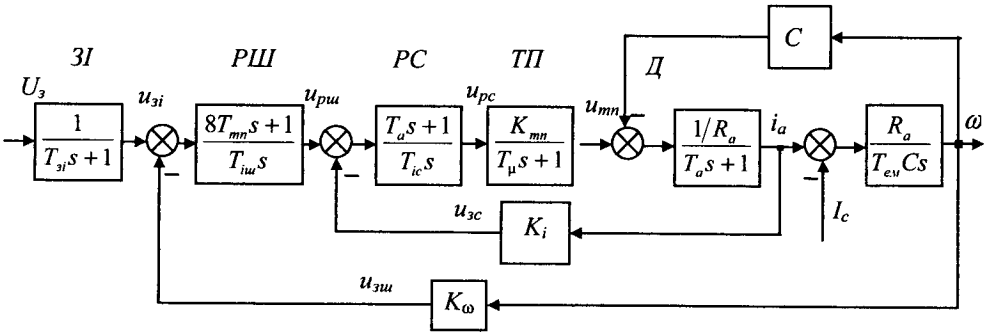


Рис. 4.9. Структурна схема дворазово інтегрувальної СПР

Приклад: Нижче подано приклад розрахунку перехідних процесів режимів "запуск – реверс – зупинка" і зміни моменту навантаження в дворазово інтегрувальній системі підпорядкованого регулювання. Результати моделювання цих режимів показано на рис. 4.10.

Задано:

$U_{znom} = 10$ В	– номінальна напруга завдання;
$U_{Pmax} = 10$ В	– максимальна вихідна напруга регуляторів;
$K_{\omega} = 0.045$	– коефіцієнт зворотного зв'язку за швидкістю;
$K_i = 0.045$	– коефіцієнт зворотного зв'язку за струмом якоря;
$K_{рш} = 23.5$	– коефіцієнт підсилення регулятора швидкості;
$T_{іш} = 0.0034$ с	– стала часу інтегратора регулятора швидкості;
$K_{рс} = 0.34$	– коефіцієнт підсилення регулятора струму;
$T_{Pi} = 0.147$ с	– стала часу інтегратора регулятора струму;
$K_d = 50$	– коефіцієнт підсилення ТП;
$T_{\mu} = 0.01$ с	– еквівалентна стала часу ТП;
$R_a = 0.34$ Ом	– сумарний опір якірного кола;
$C = 2.5$ В·с$^{-1}$	– стала двигуна;
$T_a = 0.05$ с	– електромагнітна стала часу якірного кола;
$M_{Cnom} = 100$ Н·м	– статичний момент навантаження електропривода;

$J = 2.2$ кг·м^2 – сумарний момент інерції привода;
$T_{max} = 10$ с – час розрахунку.

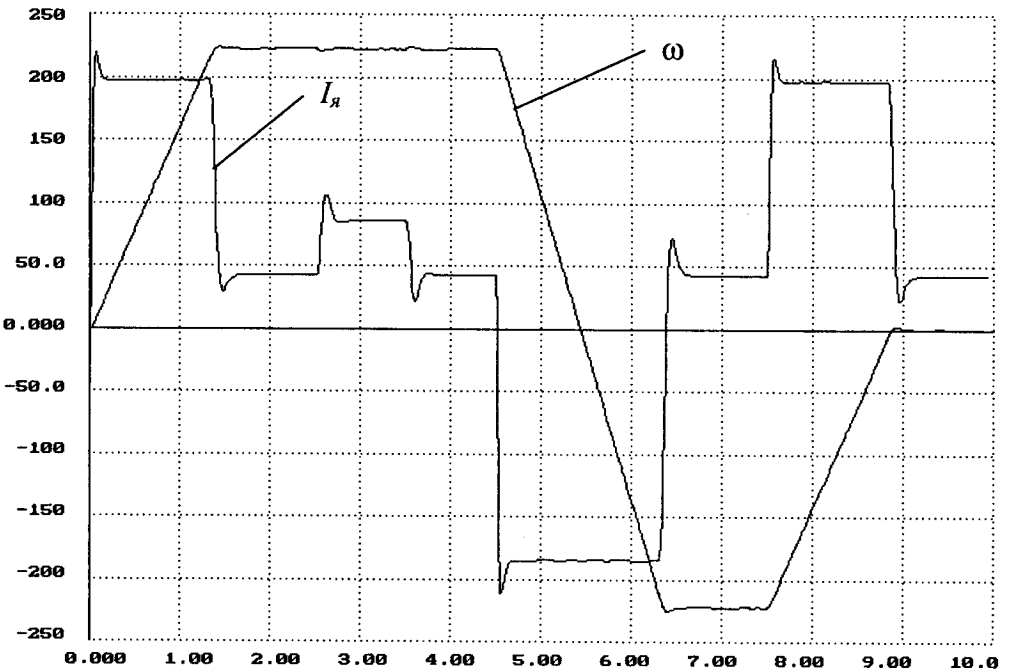


Рис. 4.10. Результати розрахунку перехідних процесів у системі ТП–Д з дворазовим інтегруванням

Quick Basic

```
DECLARE SUB Graphic (N%, Xin!, Yin!)
DECLARE SUB dif (t!, y!(), f!())
DECLARE SUB Fehlberg2B (N!, t!, h!, Eps!, y!())
' ============================================================
'  Розрахунок динаміки у режимах ЗАПУСК/РЕВЕРС/ЗУПИНКА
'  для дворазово інтегрувальної СПР електропривода
' ============================================================
```

```
CONST N = 6                  ' Порядок системи
CONST Tmax = 10              ' Максимальний час розрахунку
CONST Eps = .0001            ' Точність розв'язання на кроці
' ========================================================
'    Необхідно для виведення графіка
' ========================================================
COMMON SHARED Xmin, Xmax
Xmin = 0!                    ' Ліва межа графіка
Xmax = Tmax                  ' Права межа графіка
CONST Ymin = -250            ' Нижня межа графіка
CONST Ymax = 250             ' Верхня межа графіка
' ========================================================
DIM y(N)    .                ' Опис масиву змінних

FOR i = 1 TO N               ' Обнулення масиву змінних
   y(i) = 0                  ' (нульові початкові умови)
NEXT i                       '
t = 0!
h = .001                     ' Початковий крок моделювання
DO
   CALL Graphic(11, t, y(5))     ' Побудова графіка струму якоря
   CALL Graphic(12, t, y(6))     ' Побудова графіка швидкості
   CALL Fehlberg2B(N, t, h, Eps, y()) ' Виклик числового методу
LOOP UNTIL t > Tmax

END
SUB dif (t, y(), f())
' ------------------------------------------------------------
'  Підпрограма призначена для обчислення значень правих
'  частин системи диференціальних рівнянь, що записані у
'  нормальній формі Коші:
'       f(1) = функція_1(y(1),y(2),...,y(N),t)
'       f(2) = функція_2(y(1),y(2),...,y(N),t)
'                    . . .
'       f(N) = функція_N(y(1),y(2),...,y(N),t)
'  Вхідні величини:
'       t - незалежна змінна
'       y - масив змінних (координати системи електропривода)
'  Вихідні величини:
'       y - масив змінних
'       f - масив похідних координат системи електропривода
' ------------------------------------------------------------
'     y(1) - задавач інтенсивності
```

```
'     y(2) - інтегральна складова ПІ-регулятора швидкості
'     y(3) - інтегральна складова ПІ-регулятора струму
'     y(4) - напруга ТП
'     y(5) - струм якоря
'     y(6) - швидкість двигуна
' ------------------------------------------------------------
CONST Uznom = 10    ' Номінальна напруга завдання швидкості
CONST Tzi = .08     ' Стала часу задавача інтенсивності
CONST Kpw = 23.5    ' Коефіцієнт підсилення рег-ра швидкості
CONST Tpw = .0034   ' Стала часу інтегрування РШ
CONST KDW = .045    ' Коефіцієнт зворотного зв'язку за швидкістю
CONST UPWmax = 10!  ' Напруга обмеження рег-ра швидкості
CONST Kpc = .34     ' Коефіцієнт підсилення рег-ра струму
CONST Tpc = .147    ' Стала часу інтегрування РС
CONST KDC = .045    ' Коефіцієнт зворотного зв'язку за струмом
CONST Ktp = 50      ' Коефіцієнт підсилення ТП
CONST Ttp = .01     ' Стала часу ТП
CONST C = 2.34      ' Стала двигуна
CONST Ra = .34      ' Сумарний опір якірного кола
CONST Ta = .05      ' Стала часу якірного кола
CONST J = 2.2       ' Сумарний момент інерції привода
CONST Mcnom = 100   ' Момент опору на валу

' Формування напруги завдання
Uz = Uznom
IF t > 4.5 THEN Uz = -Uznom
IF t > 7.5 THEN Uz = 0
'  Задавач інтенсивності
f(1) = (Uz - y(1)) / Tzi
' ============================================================
'       Регулятор швидкості
' ============================================================
UPWbx = y(1) - KDW * y(6)    ' Вхідна напруга регулятора
UPWp = UPWbx * Kpw           ' Пропорційна складова регулятора
f(2) = UPWbx / Tpw           ' Інтегральна складова регулятора
Upw = UPWp + y(2)            ' Вихід ПІ-регулятора
'   Тепер обмежимо, за потреби, вихідну напругу
IF ABS(Upw) > UPWmax THEN
  Upw = UPWmax * SGN(Upw)
  f(2) = 0                   ' Зупинка інтегрування
END IF
' ============================================================
'       Регулятор струму
' ============================================================
```

```
UPCbx = (Upw - KDC * y(5))      ' Вхідна напруга регулятора
UPCp = UPCbx * Kpc              ' Пропорційна складова регулятора
f(3) = UPCbx / Tpc              ' Інтегральна складова регулятора
Upc = UPCp + y(3)               ' Вихід ПІ-регулятора
' ----------------------------------------
'    Модель тиристорного перетворювача
' ----------------------------------------
f(4) = (Upc * Ktp - y(4)) / Ttp

'   Модель якірного кола ТП-Д
f(5) = ((y(4) - C * y(6)) / Ra - y(5)) / Ta

'  Реалізація зміни навантаження привода
Mc = Mcnom
IF t > 2.5 THEN Mc = 2 * Mcnom
IF t > 3.5 THEN Mc = Mcnom
'   Розрахунок похідної швидкості
f(6) = (C * y(5) - Mc) / J          ' Для активного навантаження
END SUB
```

Turbo Pascal

```
{$E+,N+,F+ }
program  SPR_2INT;
uses
   F23B, Crt, Plot;
const
   N = 6 ;                      { Порядок системи              }
   Tmax = 10.0 ;                { Максимальний час розрахунку  }
   Eps  : DOUBLE = 1e-4 ;       { Точність на кроці }
   Ymin : DOUBLE = -250.0 ;
   Ymax : DOUBLE = 250.0 ;
var
   t, h  : DOUBLE ;
   i  : INTEGER ;
   y  : Vector ;

procedure  dif(x: DOUBLE; var y, f : Vector) ;
{ ------------------------------------------------------- }
{ Процедура призначена для обчислення значень правих частин }
```

```
{ системи диференціальних рівнянь, записаних у нормальній   }
{ формі Коші:                                                }
{      f[1] = функція_1(y[1],y[2],...,y[N],x)                }
{       f[2] = функція_2(y[1],y[2],...,y[N],x)               }
{                        ...                                 }
{       f[N] = функція_N(y[1],y[2],...,y[N],x)               }
{ Вхідні величини:                                           }
{       x - незалежна змінна                                 }
{       y - масив змінних (координати системи електропривода) }
{ Вихідні величини:                                          }
{       y - масив змінних                                    }
{       f - масив похідних координат системи електропривода  }
{ ---------------------------------------------------------- }
{     y[1] - задавач інтенсивності                           }
{     y[2] - інтегральна складова ПІ-регулятора швидкості    }
{     y[3] - інтегральна складова ПІ-регулятора струму       }
{     y[4] - напруга ТП                                      }
{     y[5] - струм якоря                                     }
{     y[6] - швидкість двигуна                               }
{ ---------------------------------------------------------- }
const
   Uznom = 10.0 ;    { Напруга завдання для номінальної швидкості }
   Tzi = 0.08 ;      { Стала часу задавача інтенсивності          }
   Kpw = 23.5 ;      { Коефіцієнт підсилення регулятора швидкості }
   Tpw = 0.0034 ;    { Стала часу інтегрування РШ                 }
   KDW = 0.045 ;     { Коефіцієнт зворотного зв'язку за швидкістю }
   UPWmax = 10.0 ;   { Напруга обмеження регулятора швидкості     }
   Kpc = 0.34 ;      { Коефіцієнт підсилення рег-ра струму        }
   Tpc = 0.147 ;     { Стала часу інтегрування РС                 }
   KDC = 0.045 ;     { Коефіцієнт зворотного зв'язку за струмом   }
   Ed0 = 514.8 ;     { Напруга неробочого ходу ТП                 }
   Ktp = 50.0 ;      { Коефіцієнт підсилення ТП                   }
   Ttp = 0.01 ;      { Стала часу ТП                              }
   C = 2.34 ;        { Стала двигуна                              }
   Ra = 0.34 ;       { Сумарний опір якірного кола                }
   Ta = 0.05 ;       { Стала часу якірного кола                   }
   J = 2.2 ;         { Сумарний момент інерції привода            }
   Mcnom = 100.0 ;   { Номінальний момент статичного навантаження }
var
   Upw , UPWbx , UPWp : DOUBLE ;
   Upc , UPCbx , UPCp : DOUBLE ;
   Uz , Mc            : DOUBLE ;
```

```
begin
{  Формування напруги завдання  }
Uz := Uznom ;
if t > 4.5 then Uz := -Uznom ;
if t > 7.5 then Uz := 0.0 ;
{  Задавач інтенсивності  }
f[1] := (Uz - y[1]) / Tzi ;
{ ========================== }
{       Регулятор швидкості    }
{ ========================== }
UPWbx := y[1] - KDW * y[6] ;    { Вхідна напруга регулятора  }
UPWp := UPWbx * Kpw ;           { Пропорційна складова       }
f[2] := UPWbx / Tpw ;           { Інтегральна складова       }
Upw := UPWp + y[2] ;            { Вихід ПІ-регулятора        }
{  Тепер обмежимо, якщо необхідно, вихідну напругу  }
if abs(Upw) > UPWmax then
   begin
      Upw := UPWmax * abs(Upw)/Upw ;
      f[2] := 0 ;                { Зупинка інтегрування  }
   end ;
{ ========================== }
{       Регулятор струму       }
{ ========================== }
UPCbx := (Upw - KDC * y[5]) ;   { Вхідна напруга регулятора }
UPCp := UPCbx * Kpc ;           { Пропорційна складова       }
f[3] := UPCbx / Tpc ;           { Інтегральна складова       }
Upc := UPCp + y[3] ;            { Вихід ПІ-регулятора        }

{  Модель тиристорного перетворювача  }
f[4] := (Upc * Ktp - y[4]) / Ttp ; { Врахування інерційності ТП }

{  Модель якірного кола ТП-Д  }
f[5] := ((y[4] - C * y[6]) / Ra - y[5]) / Ta ;

{ Реалізація зміни моменту навантаження  }
Mc := Mcnom ;
if t > 2.5 then Mc := 2.0 * Mcnom ;
if t > 3.5 then Mc := Mcnom ;
{   Розрахунок похідної швидкості       }
f[6] := (C * y[5] - Mc) / J ;
end ;
```

```
begin
   h := 0.001 ;  { Початковий крок   }
   {   Формування "вікна" для виведення графіка  }
   InitGraphic(0, Tmax, Ymin, Ymax, 1.0, 50.0) ;
   for i := 1 to N do y[i] := 0.0;
{ ----------------------------------  }
{  Розрахунок динаміки електропривода   }
{ ----------------------------------  }
   repeat
      Fehlberg23B(Dif, N, t, h, Eps, y) ;
      Graphic(1, t, y[5]) ;  { виведення на екран Ia }
      Graphic(2, t, y[6]) ;  { виведення на екран w  }
   until  t > Tmax ;
   ReadLn;
end.
```

MathCAD

$U_{znom} := 10$	*Номінальна напруга завдання*
$U_{Pmax} := 10$	*Максимальна вихідна напруга регуляторів*
$K_{\omega} := 0.045$	*Коефіцієнт зворотного зв'язку за швидкістю*
$K_i := 0.045$	*Коефіцієнт зворотного зв'язку за струмом якоря*
$K_{P\omega} := 23.5$	*Коефіцієнт підсилення регулятора швидкості*
$T_{P\omega} := 0.0034$	*Стала часу інтегрування регулятора швидкості*
$K_{Pi} := 0.34$	*Коефіцієнт підсилення регулятора струму*
$T_{Pi} := 0.147$	*Стала часу інтегрування регулятора струму*
$K_d := 50$	*Коефіцієнт підсилення ТП*
$T_{\mu} := 0.01$	*Стала часу ТП*
$R_a := 0.34$	*Сумарний опір якірного кола*
$C := 2.5$	*Стала двигуна*
$T_a := 0.05$	*Електромагнітна стала часу якірного кола*
$M_{Cnom} := 100$	*Номінальний статичний момент*
$J := 2.2$	*Сумарний момент інерції привода*
$T_{max} := 10$	*Час розрахунку*

$N := 1000$ *Кількість точок інтегрування*

Напруга завдання:

$$U_z(t) := \begin{cases} Uz_{nom} & \text{if } t < 4.5 \\ -Uz_{nom} & \text{if } 4.5 \le t < 7.5 \\ 0 & \text{otherwise} \end{cases}$$

Статичний момент:

$$M_c(t) := \begin{cases} 2 \cdot M_{cnom} & \text{if } 2.5 < t < 3.5 \\ M_{cnom} & \text{otherwise} \end{cases}$$

Початкові умови:

$$y_0 := \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Вектор-функція правих частин системи диференціальних рівнянь, що описують модель:

$$Diff(t, y) := \begin{cases} f_0 \leftarrow \dfrac{U_z(t) - y_0}{8 \cdot T_\mu} \\ U_{bxP\omega} \leftarrow y_0 - K_\omega \cdot y_5 \\ f_1 \leftarrow \dfrac{U_{bxP\omega}}{T_{P\omega}} \end{cases}$$

$$U_{P\omega} \leftarrow y_1 + U_{bxP\omega} \cdot K_{P\omega}$$

$$U_{P\omega} \leftarrow \begin{cases} \text{if } |U_{P\omega}| > Up_{max} \\ \quad f_1 \leftarrow 0 \\ \quad Up_{max} \cdot \dfrac{U_{P\omega}}{|U_{P\omega}|} \\ U_{P\omega} \text{ otherwise} \end{cases}$$

$$U_{bxPC} \leftarrow U_{P\omega} - K_i \cdot y_4$$

$$f_2 \leftarrow \frac{U_{bxPC}}{T_{Pi}}$$

$$U_{Pi} \leftarrow y_2 + K_{Pi} \cdot U_{bxPC}$$

$$f_3 \leftarrow \frac{K_d \cdot U_{Pi} - y_3}{T_\mu}$$

$$f_4 \leftarrow \frac{\dfrac{y_3 - C \cdot y_5}{R_a} - y_4}{T_a}$$

$$f_5 \leftarrow \frac{C \cdot y_4 - M_c(t)}{J}$$

$$f$$

Позначення змінних:

$U_{bxP\omega}$ - *напруга на вході РШ;*

$U_{P\omega}$ - *напруга РШ;*

U_{bxPC} - *напруга на вході РС;*

U_{Pi} - *напруга РС;*

y_0 - *напруга фільтра на вході РШ;*

y_1 - *напруга інтегратора РШ;*

y_2 - *напруга інтегратора РС;*

y_3 - *напруга ТП;*

y_4 - *струм якоря;*

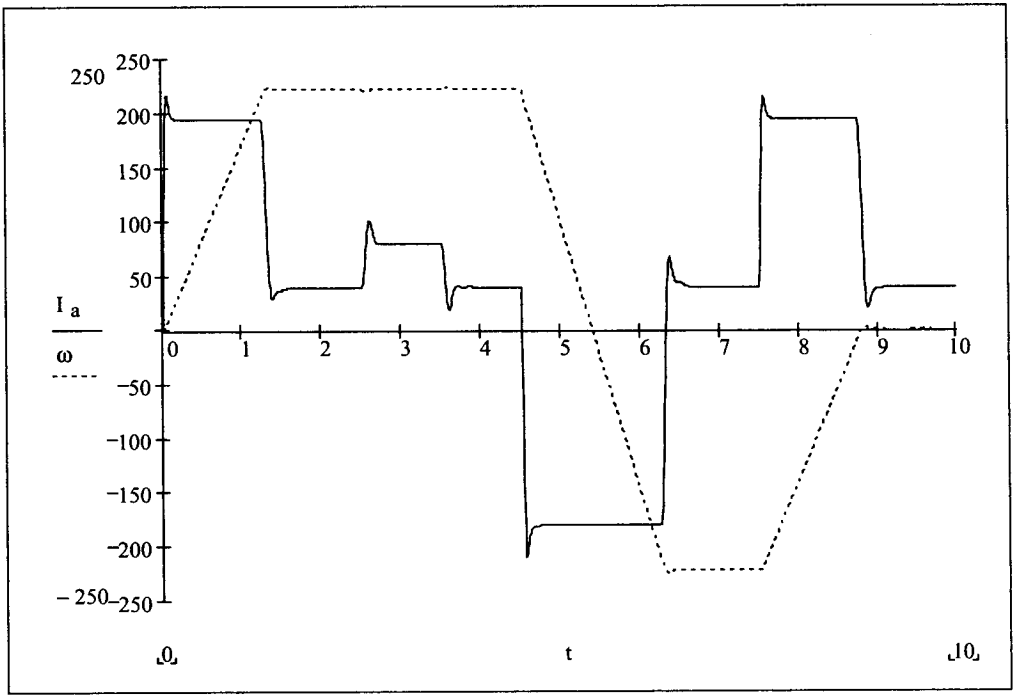
y_5 - *швидкість.*

Розв'язуємо систему диф. рівнянь:

$$Res := rkfixed(y_0, 0, T_{max}, N, Diff)$$

З матриці розв'язку виділяємо необхідні стовпці:

$t := Res^{<0>}$ $\quad I_a := Res^{<5>}$ $\quad \omega := Res^{<6>}$



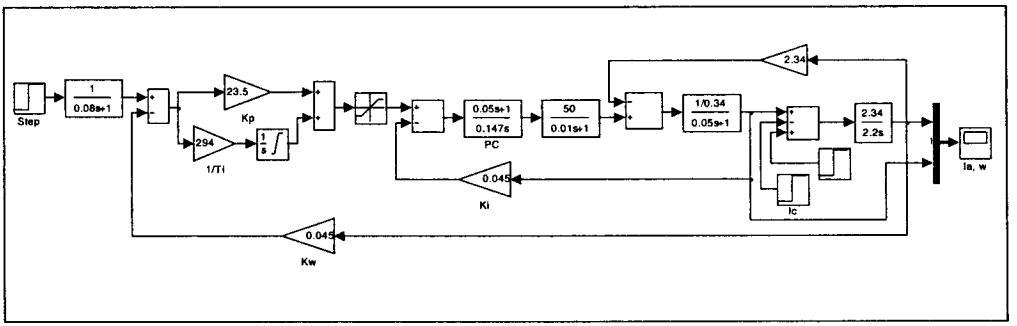
MATLAB

```
% Розрахунок перехідних процесів
% дворазово інтегрувальної СПР ЕП
Tmax = 10;              % Час розрахунку
y0(1:6) = 0;            % Початкові умови

% Розв'ЯзуванняЯ системи диференціальних рівнЯнь
[t, y] = ode23('spr2int', [0 Tmax], y0);

% Побудова графіка
plot(t, y(:,5), 'k', t, y(:,6), 'k:'), grid
xlabel('t, c'); ylabel('I_a, \omega'); legend('I_a', '\omega', 3);
```

```
function f = spr2int(t, y);
% ------------------------------------------------------------
%     ФункціЯ, що описує модель дворазово інтегрувальної САР ЕП
%
%     y(1) - вихід задавача інтенсивності
%     y(2) - інтегральна складова РШ
%     y(3) - інтегральна складова РС
%     y(4) - напруга ТП
%     y(5) - струм ЯкорЯ
%     y(6) - швидкість
% ------------------------------------------------------------
Uznom = 10;         % Номінальна напруга завданнЯ
Tzi = 0.08;         % Стала часу задавача інтенсивності
Kpw = 23.5;         % КоефіцієнтпідсиленнЯ рег-ра швидкості
Tpw = 0.0034;       % Стала часу інтегруванняЯ РШ
KDW = 0.045;        % Коефіцієнт зворотного зв'Язку за швидкістю
UPWmax = 10;        % Напруга обмеженняЯ регулЯтора швидкості
Kpc = 0.34;         % КоефіцієнтпідсиленнЯ рег-ра струму
Tpc = 0.147;        % Стала часу інтегруванняЯ РС
KDC = 0.045;        % Коефіцієнт зворотного зв'Язку за струмом
Ktp = 50;           % КоефіцієнтпідсиленнЯ ТП
Ttp = 0.01;         % Стала часу ТП
C = 2.34;           % Стала двигуна
Ra = 0.34;          % Сумарний опір Якірного кола
Ta = 0.05;          % Стала часу Якірного кола
J = 2.2;            % Сумарний момент інерції привода
Mcnom = 100;        % Момент опору на валу

% Розрахунок значень правих частин системи диф. рівнЯнь
f=zeros(6, 1);
% ФормуванняЯ напруги завданнЯ
Uz = Uznom;
if t > 4.5
   Uz = -Uznom;
end
if t > 7.5
   Uz = 0;
end

%  Задавач інтенсивності
f(1) = (Uz - y(1)) / Tzi;
```

```
% ============================================================
%       РегулЯтор швидкості
% ============================================================
UPWbx = y(1) - KDW * y(6);     % Вхідна напруга регулЯтора
UPWp = UPWbx * Kpw;            % Пропорційна складова регулЯтора
f(2) = UPWbx / Tpw;            % Інтегральна складова регулЯтора
Upw = UPWp + y(2);             % Вихід ПІ-регулЯтора
%   Тепер обмежимо, Якщо необхідно, його вихідну напругу
if abs(Upw) > UPWmax
  Upw = UPWmax * sign(Upw);
  f(2) = 0;                    % "НасиченнЯ" інтегратора
end

% ============================================================
%       РегулЯтор струму
% ============================================================
UPCbx = (Upw - KDC * y(5));    % Вхідна напруга регулЯтора
UPCp = UPCbx * Kpc;            % Пропорційна складова регулЯтора
f(3) = UPCbx / Tpc;            % Інтегральна складова регулЯтора
Upc = UPCp + y(3);             % Вихід ПІ-регулЯтора

% ------------------------------------------
%    Модель  тиристорного  перетворювача
% ------------------------------------------
f(4) = (Upc*Ktp - y(4))/Ttp;

%  Модель Якірного кола ТП-Д
f(5) = ((y(4) - C*y(6))/Ra - y(5))/Ta;

%   Модель механічної частини привода
%   РеалізацІЯ зміни моменту навантаженнЯ
Mc = Mcnom;
if t > 2.5
   Mc = 2 * Mcnom;
end
if t > 3.5
   Mc = Mcnom;
end

%   Розрахунок похідної швидкості
f(6) = (C*y(5) - Mc)/J;
```

Simulink



4.2. Електроприводи змінного струму

Реостатний запуск двигуна змінного струму

Розрахувати процес зміни моменту $M(t)$ та швидкості $\omega(t)$ для режиму триступеневого реостатного запуску асинхронного двигуна з фазним ротором типу 4АК160М4 у функції швидкості. Для знаходження моменту двигуна використати спрощену формулу Клоса. Результати моделювання режиму показано на рис. 4.11 і 4.12.

Задано:

$P_н = 14$ кВт	– потужність двигуна;
$U_н = 380$ В	– номінальна напруга двигуна;
$\lambda = 3.5$	– кратність максимального моменту;
$J = 0.5$ кг·м^2	– сумарний момент інерції привода;
$M_c = 53.03$ Н·м	– статичний момент навантаження;
$n_н = 1440$ $^{об}/_{хв}$	– номінальна швидкість обертання.

Пояснення до програми: Для спрощення програми використано попередньо розраховані значення критичного ковзання і швидкості для кожного пускового ступеня.

Quick Basic

```
COMMON SHARED Xmin, Xmax, Ymin, Ymax  ' Для виведення графіків
'-------------------------------------------------------
' Розрахунок динамічних режимів реостатного запуску
' асинхронного двигуна з фазним ротором у функції швидкості
'-------------------------------------------------------
CONST N = 2                ' Порядок системи
CONST Tmax = 1!            ' Максимальний час розрахунку
CONST Eps = .001           ' Точність розв'язання на кроці
h = .01                    ' Початковий крок інтегрування

DIM Y(2)
Y(1) = 0: Y(2) = 0     ' Початкові умови
Xmin = 0!              ' Ліва межа графіка
Xmax = Tmax            ' Права межа графіка
Ymin = 0               ' Нижня межа графіка
Ymax = 200             ' Верхня межа графіка
t = 0!                 ' Початковий час
' -------   Основний цикл програми   -------
DO
  CALL Graphic(1, t, Y(2))        ' Побудова графіка швидкості
  CALL Graphic(2, t, Y(1))        ' Побудова графіка моменту АД

  CALL adams3(N, t, h, Eps, Y()) ' Виклик числового методу
LOOP UNTIL t > Tmax

END

SUB Dif (t, Y(), f()) STATIC
' ---------------------------------------------------------
'  Підпрограма призначена для обчислення значень правих частин
'  системи диференціальних рівнянь, записаних у нормальній
'  формі Коші.
'  Форма  запису:
'         f(1) = функція_1(y(1),y(2),...,y(N),t)
'         f(2) = функція_2(y(1),y(2),...,y(N),t)
'         . . .
'         f(N) = функція_N(y(1),y(2),...,y(N),t)
'
```

```
'  Вхідні величини:
'        t - незалежна змінна
'        y - масив змінних (координат електропривода)
'  Вихідні величини:
'        y - масив змінних
'        f - масив похідних координат електропривода
' ---------------------------------------------------------
' Розрахувати запуск асинхронного двигуна (АД) з фазним ротором
' з параметрами:
'        Тип     - 4АК160М4У3
'        Рн      = 14 кВт
'        Uн      = 380 В
'        Nн      = 1440 об./хв.
'        Мк/Мн   = 3.5 (кратність максимального моменту),
'        позначимо LambdaM
'
'        У розрахунку використана проста формула Клоса.
' ---------------------------------------------------------
CONST Pn = 14000          ' Номінальна потужність АД
CONST W0 = 157.08         ' Синхронна частота АД
CONST Wn = 150.8          ' Номінальна частота обертання АД
CONST LambdaM = 3.5       ' Кратність максимального моменту
CONST Mn = Pn / Wn        ' Номінальний момент АД
CONST Mk = Mn * LambdaM     ' Макс. (критичний) момент АД
CONST Sn = .04              ' Номінальне ковзання АД
CONST J = .5          ' Сумарний момент інерції привода
CONST Mc = 53.03      ' Статичний момент навантаження
CONST W1 = 83.21      ' Швидкість перемикання з першого ступеня
CONST W2 = 122.46     ' Швидкість перемикання з другого ступеня
CONST W3 = 141!       ' Швидкість перемикання з третього ступеня

Sk = 2.98
'  Y(2) - швидкість обертання АД
SELECT CASE Y(2)
  CASE 0 TO W1
    Sk = 2.98           ' Перший ступінь
  CASE W1 TO W2
    Sk = 1.405          ' Другий ступінь
  CASE W2 TO W3
    Sk = .668           ' Третій ступінь
  CASE IS > W3
    Sk = .274           ' Природна характеристика
END SELECT
```

```
S = (W0 - Y(2)) / W0                ' Розрахунок ковзання
' Розрахунок моменту за формулою Клоса
M = 2 * Mk / (S / Sk + Sk / S)

' Визначення електромагнітної сталої часу
Te = 1! / (W0 * Sk)

f(1) = (M - Y(1)) / Te              ' Похідна моменту на валі
f(2) = (Y(1) - Mc) / J              ' Похідна швидкості

END SUB
```

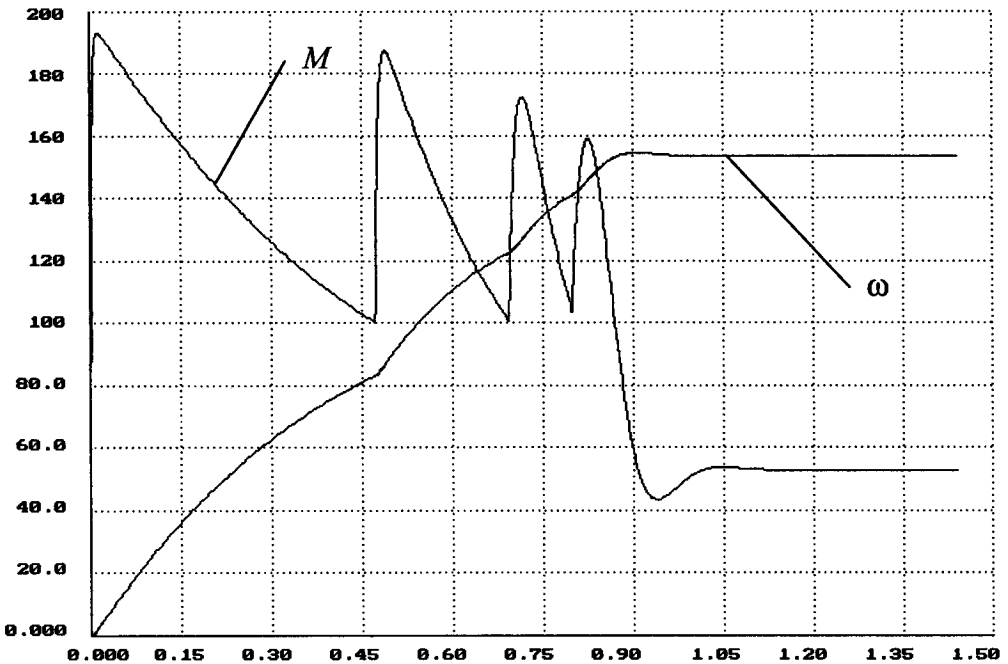


Рис. 4.11. Результати розрахунку перехідних процесів реостатного запуску асинхронного двигуна з фазним ротором у середовищі Quick Basic

Turbo Pascal

```
{$E+,N+,F+ }
program  AM_FR;
{ ********************************************** }
uses
   Adams3, Plot;
const
   N = 2;                    { Порядок системи              }
   Tmax = 1.0;               { Максимальний час розрахунку  }
   Eps  : DOUBLE = 1e-4;     { Точність на кроці }
   Ymin : DOUBLE = 0.0;
   Ymax : DOUBLE = 200.0;
var
   t, h  : DOUBLE;
   i  : INTEGER;
   y  : Vector;

procedure  dif(x: DOUBLE; var y, f : Vector);
{ ------------------------------------------------------- }
{ Процедура призначена для обчислення значень правих частин }
{ системи диференціальних рівнянь, записаних у нормальній   }
{ формі Коші.                                               }
{ Форма  запису:                                            }
{        f[1] := ...                                        }
{        . . .                                              }
{        f[N] := ...                                        }
{                                                           }
{  Вхідні величини:                                         }
{        t - незалежна змінна                               }
{        y - масив змінних (координат електропривода)       }
{ Вихідні величини:                                         }
{        y - масив змінних                                  }
{        f - масив похідних координат електропривода        }
{ ------------------------------------------------------- }
{ Задача:                                                   }
{                                                           }
{ Розрахувати запуск асинхронного двигуна (АД) з фазним ротором }
{ з такими параметрами:                                     }
{                                                           }
```

```
{       Тип     - 4АК160М4У3                                    }
{       Рн      = 14 кВт                                        }
{       Uн      = 380 В                                         }
{       Nн      = 1440 об./хв.                                  }
{       Мк/Мн   = 3.5 (кратність максимального моменту),        }
{       позначимо LambdaM                                       }
{                                                               }
{       У розрахунку використана проста формула Клоса.          }
{ ------------------------------------------------------------- }
const
   Pn = 14000.0;       { Номінальна потужність АД              }
   W0 = 157.08;        { Синхронна кутова швидкість АД         }
   Wn = 150.8;         { Номінальна частота обертання АД       }
   LambdaM = 3.5;      { Кратність максимального моменту АД    }
   Mn = Pn / Wn;       { Номінальний момент АД                 }
   Mk = Mn * LambdaM;      { Макс. (критичний) момент АД       }
   Sn = 0.04;              { Номінальне ковзання АД            }
   J = 0.5;        { Сумарний момент інерції привода           }
   Mc = 53.03;     { Статичний момент навантаження             }
   W1 = 83.21;     { Швидкість перемикання з першого ступеня   }
   W2 = 122.46;    { Швидкість перемикання з другого ступеня   }
   W3 = 141.0;     { Швидкість перемикання з третього ступеня  }
var
   S, Sk, M, Te : double;
begin
   {  Y[2] - швидкість обертання АД  }
   { Розрахунок критичного ковзання залежно від ступеня }
   if (y[2]<W1) then Sk := 2.98;                  { перший ступінь }
   if (y[2]>=W1) and (y[2]<W2) then Sk := 1.405; { другий ступінь }
   if (y[2]>=W2) and (y[2]<W3) then Sk := 0.668; { третій ступінь }
   if (y[2]>=W3) then Sk := 0.274;  { природна характеристика }

   S := (W0 - Y[2]) / W0;              { Розрахунок ковзання }
   { Розрахунок моменту АД за формулою Клоса }
   M := 2 * Mk / (S / Sk + Sk / S);

   { Обчислення електромагнітної сталої часу }
   Te := 1.0/(W0*Sk);

   f[1] := (M - Y[1]) / Te;       { Похідна моменту АД   }
   f[2] := (Y[1] - Mc) / J;       { Похідна швидкості АД }

end;
```

```
begin
   h := 0.01;        { Початковий крок    }
{   Ініціалізація графіка      }
   InitGraphic(0, Tmax, Ymin, Ymax, 0.2, 20.0);
   for i := 1  to N do y[i] := 0.0;  { Початкові умови }
{ ------------------------------------ }
{  Розрахунок динаміки режиму запуску АД     }
{ ------------------------------------ }
   repeat
      ABM3(Dif, N, t, h, Eps, y);
      Graphic(1, t, y[1]);  { Побудова графіка моменту АД   }
      Graphic(2, t, y[2]);  { Побудова графіка швидкості АД }
   until  t > Tmax;
end.
```

MathCAD

$P_{nom} := 14000$ *Потужність двигуна*

$n_0 = 1500$ *Синхронна кутова швидкість обертання* $\omega_0 := n_0 \frac{\pi}{30}$

$n_{nom} = 1440$ *Номінальна кутова швидкість обертання* $\omega_{nom} := n_{nom} \frac{\pi}{30}$

$M_{nom} := \frac{P_{nom}}{\omega_{nom}}$ *Номінальний момент двигуна*

$\lambda := 3.5$ *Кратність критичного (максимального) моменту*

$M_k := M_{nom} \cdot \lambda$ *Критичний момент двигуна*

$J := 0.5$ *Сумарний момент інерції привода*

$M_c = 53.03$ *Статичний момент навантаження*

$\omega_1 := 83.21$ *Швидкість АД у разі перемикання на другий ступінь*

$\omega_2 := 122.45$ *Швидкість АД у разі перемикання на третій ступінь*

$\omega_3 := 141$ *Швидкість АД у разі перемикання на природну характеристику*

$T_{max} := 1$ *Кінцевий час розрахунку*

$N := 1000$ *Кількість кроківі* $:= 1 .. N$

Функція-вектор правих частин системи диференціальних рівнянь:

$$Diff(t,y) := \left(\begin{array}{l} s_k \leftarrow 2.98 \text{ if } y_1 \le \omega_1 \\ s_k \leftarrow 1.405 \text{ if } \omega_1 < y_1 \le \omega_2 \\ s_k \leftarrow 0.668 \text{ if } \omega_2 < y_1 \le \omega_3 \\ s_k \leftarrow 0.274 \text{ otherwise} \\ s \leftarrow \dfrac{\omega_0 - y_1}{\omega_0} \\ M \leftarrow \dfrac{2 \cdot M_k}{\dfrac{s}{s_k} + \dfrac{s_k}{s}} \\ T_e \leftarrow \dfrac{1}{\omega_0 \cdot s_k} \\ \dfrac{M - y_0}{T_e} \\ \dfrac{y_0 - M_c}{J} \end{array} \right)$$

y_0 - електромагнітний момент

y_1 - кутова швидкість

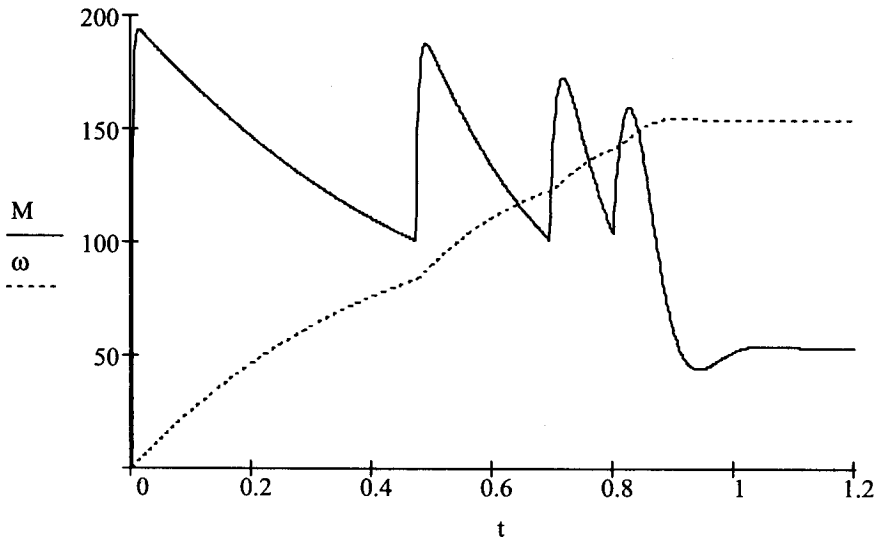
Початкові умови: $y_0 := \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

Отримання розв'язку системи диференціальних рівнянь:

$$Res := Rkadapt(y_0, 0, T_{max}, N, Diff)$$

Формування стовпців координат АД у зручному вигляді:

$$t := Res^{<0>} \qquad M := Res^{<1>} \qquad \omega := Res^{<2>}$$



MATLAB

```
% Розрахунок динаміки режиму реостатного запуску
% асинхронного двигуна з фазним ротором у функції швидкості
Tmax = 1;              % Час розрахунку
y0 = [0 0];            % Початкові умови
```

```
% Розв'язуванням системи диференціальних рівнянь
 [t, y] = ode23('am_fr', [0 Tmax], y0);

% Побудова графіка
plot(t, y(:,1), 'k', t, y(:,2), 'k:');
axis([0 Tmax 0 inf]);
xlabel('t, c'); ylabel('M, \omega'); legend('M', '\omega');
```

```
function f=am_fr(t, y);
% ------------------------------------------------------
% Обчисленням значень правих частин системи диференціальних
% рівнянь, що описують динаміку запуску АД з фазним ротором
% ------------------------------------------------------
Pnom = 14000;           % Номінальна потужність АД
W0 = 157.08;            % Синхронна кутова частота обертання АД
Wnom = 150.8;           % Номінальна частота обертання АД
LambdaM = 3.5;          % Кратність максимального моменту АД
Mnom = Pnom / Wnom;     % Номінальний момент АД
Mk = Mnom * LambdaM;    % Максимальний (критичний) момент АД
Snom = 0.04;            % Номінальне ковзання АД
J = 0.5;                % Сумарний момент інерції привода
Mc = 53.03;             % Статичний момент навантаження
W1 = 83.21;             % Швидкість перемикання з 1-го ступеня
W2 = 122.46;            % Швидкість перемикання з 2-го ступеня
W3 = 141;               % Швидкість перемикання з 3-го ступеня
Sk = 2.98;              % Критичне ковзання

% Розрахунок правих частин системи диференціальних рівнянь
f=zeros(2,1);
% Задання критичного ковзання залежно від пускового ступеня
if y(2)>W3
    Sk = 0.274;
elseif y(2)>W2
    Sk = 0.668;
elseif y(2)>W1
    Sk = 1.405;
else
    Sk = 2.98;
end
```

```
S = (W0 - y(2)) / W0;        % Розрахунок ковзанняЯ
M = 2*Mk/(S/Sk + Sk/S);      % Розрахунок моменту за формулою Клоса
Te = 1.0/(W0*Sk);            % Розрахунок електромагнітної сталої часу

f(1) = (M - y(1)) / Te; % Похідна електромагнітного моменту АД
f(2) = (y(1) - Mc) / J; % Похідна швидкості АД
```

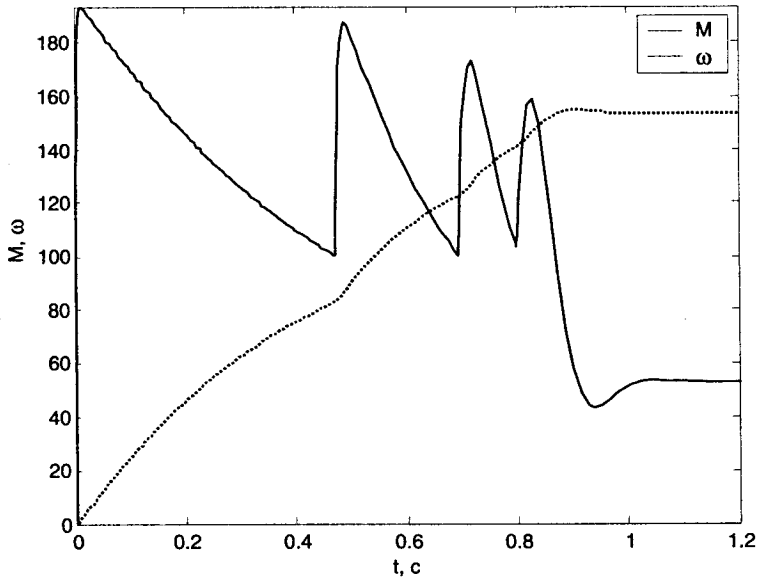


Рис. 4.12. Результати розрахунку перехідних процесів реостатного запуску асинхронного двигуна з фазним ротором у середовищі MATLAB

Моделювання режиму прямого запуску АД з короткозамкненим ротором у фазних координатах

Дослідити режим прямого запуску асинхронного двигуна з короткозамкненим ротором, використовуючи уточнену модель у фазних координатах. На рис. 4.13 показано отримані на основі цієї моделі часові залежності моменту (масштаб моменту – 1:30) та швидкості, а на рис. 4.14 – фазних струмів статора для режиму прямого запуску АД.

Задано:

$I_н = 104$ А	– номінальний струм фази двигуна;
$U_н = 220$ В	– номінальна фазна напруга двигуна;
$p_n = 4$	– кількість пар полюсів;
$R_1 = 0.103$ Ом	– активний опір обмотки фази статора;
$X_1 = 0.172$ Ом	– індуктивний опір розсіювання обмотки фази статора;
$R_2' = 0.237$ Ом	– зведений до статора активний опір обмотки фази ротора;
$X_2' = 0.366$ Ом	– зведений до статора індуктивний опір розсіювання обмотки фази ротора;
$J = 4.5$ кг·м^2	– сумарний момент інерції привода;
$M_c = 520$ Н·м	– статичний момент навантаження;
$n_н = 1440$ $^{об}/_{хв}$	– номінальна швидкість обертання.

Quick Basic

```
DECLARE SUB adams3 (N!, t!, h!, Eps!, y!())
DECLARE SUB Graphic (N%, x!, y!)
DECLARE SUB dif (t!, y!(), f!())
'-------------------------------------------------------------
' Розрахунок динаміки режиму запуску АД з короткозамкненим
ротором
' за уточненою моделлю у фазних координатах
'-------------------------------------------------------------
CONST N = 5                 ' Порядок системи
CONST Tmax = 1              ' Максимальний час розрахунку
CONST Eps = .001            ' Точність розв'язку на кроці
CONST Pi = 3.141593         ' Число "Пі"
CONST f0 = 50!              ' Частота напруги мережі живлення
CONST w = 2 * f0 * Pi       ' Кутова частота напруги мережі

h = .001                    ' Початковий крок інтегрування
DIM y(N)
FOR i = 1 TO N
   y(i) = 0                 ' Нульові початкові умови
NEXT i
```

```
CONST Xmin = 0!           ' Ліва межа графіка
CONST Xmax = Tmax         ' Права межа графіка
' Для виведення швидкості та моменту АД
CONST Ymin = -10          ' Нижня межа графіка
CONST Ymax = 90           ' Верхня межа графіка
' Для виведення фазних струмів статора АД
' CONST Ymin = -750       ' Нижня межа графіка
' CONST Ymax = 750        ' Верхня межа графіка

t = 0!                    ' Початковий час
COMMON SHARED M           ' Змінна моменту - глобальна
' -------   Основний цикл програми   -------
DO
   ' Розрахунок фазних струмів статора АД
  Ia = y(3) * COS(w * t) - y(4) * SIN(w * t)
  Ib = y(3) * COS(w*t - 2*Pi/3) - y(4)*SIN(w*t - 2*Pi/3)
  Ic = y(3) * COS(w*t + 2*Pi/3) - y(4)*SIN(w*t + 2*Pi/3)

  ' CALL Graphic(3, t, Ia)            ' Побудова графіків
  ' CALL Graphic(4, t, Ib)            ' фазних струмів
  ' CALL Graphic(5, t, Ic)            ' статора АД

  CALL Graphic(9, t, y(5))     ' Побудова графіка швидкості АД
  CALL Graphic(10, t, M / 30)  ' Побудова графіка моменту АД

  CALL adams3(N, t, h, Eps, y()) ' Виклик числового методу
LOOP UNTIL t > Tmax

END

SUB dif (t, y(), f()) STATIC
' ------------------------------------------------------
'  Вектор координат АД:
'     y(1) - потік за альфа-координатою
'     y(2) - потік за бета-координатою
'     y(3) - струм за альфа-координатою
'     y(4) - струм за бета-координатою
'     y(5) - швидкість
' ------------------------------------------------------
CONST Ufn = 220!        ' Номінальна фазна напруга
CONST Ifn = 104!        ' Номінальний струм фази статора
CONST Zb = Ufn / Ifn    ' Базовий опір
```

```
CONST J = 4.5           ' Сумарний момент інерції привода
CONST Mc = 520!         ' Статичний момент навантаження

CONST Em = 311!         ' Амплітуда фазної напруги
CONST p = 4             ' Кількість пар полюсів
CONST R1 = .103         ' Активний опір обмотки статора
CONST X1 = .172         ' Індуктивний опір розсіювання статора
CONST R2 = .237         ' Зведений до статора активний опір ротора
' Зведений до статора індуктивний опір розсіювання обмотки ротора
CONST X2 = .366
CONST L1 = X1 / w           ' Індуктивність розсіювання статора
CONST L2 = X2 / w           ' Індуктивність розсіювання ротора
CONST Lm = 3! * Zb / w      ' Взаємна індуктивність
CONST Ls = L1 + Lm          ' Індуктивність обмотки статора
CONST Lr = L2 + Lm          ' Індуктивність обмотки ротора
CONST Lh = Lr / (Ls * Lr - Lm * Lm)

' Обчислення значень фазних напруг
Ufa = Em * SIN(w * t)                       ' Фаза A
Ufb = Em * SIN(w * t - 2 * Pi / 3)          ' Фаза B
Ufc = Em * SIN(w * t + 2 * Pi / 3)          ' Фаза C

Ua = (2 * Ufa - Ufb - Ufc) / 3              ' U-Alfa
Ub = (Ufb - Ufc) / SQR(3)                   ' U-Beta

' Обчислення похідних потоків АД
f(1) = R2 * (Lm * y(3) - y(1)) / Lr - p * y(5) * y(2)
f(2) = R2 * (Lm * y(4) - y(2)) / Lr + p * y(5) * y(1)

' Розрахунок похідних струмів АД
f(3) = Lh * (Ua - R2 * y(3) - Lm * f(1) / Lr)
f(4) = Lh * (Ub - R1 * y(4) - Lm * f(2) / Lr)

' Розрахунок моменту АД
M = 1.5 * p * Lm * (y(1) * y(4) - y(2) * y(3)) / Lr

'  y(5) - швидкість обертання АД
f(5) = (M - Mc) / J                         ' Похідна швидкості

END SUB
```

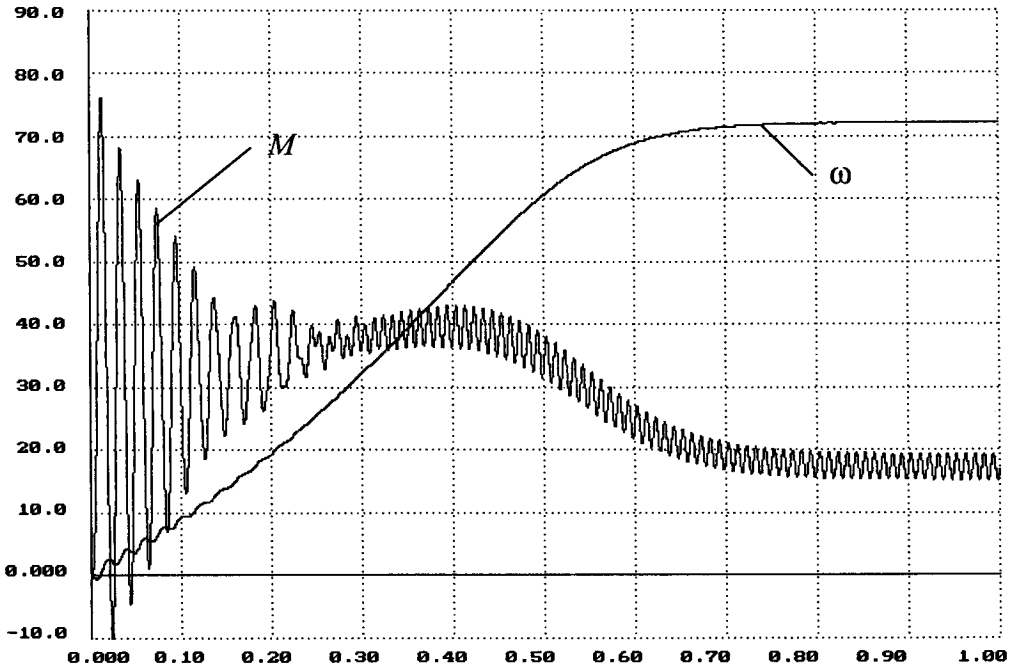


Рис. 4.13. Результати розрахунку швидкості та моменту під час запуску АД з короткозамкненим ротором для моделі у координатах α–β у середовищі Quick Basic

Примітка:

Спробуйте у цій програмі змінити числовий метод: замінити підпрограму розв'язування системи диференціальних рівнянь `Adams3` *на підпрограму* `Merson` *чи* `Fehlberg2B` *(розд. 3), а тоді подивитися, як зміниться час розрахунку (до речі, такий експеримент можна виконати не лише в середовищі Quick Basic, але і в будь-якому іншому – результати будуть доволі повчальними). Для визначення часу можна скористатись системною функцією* `Timer` *середовища Quick Basic, для чого ввести у текст програми такі рядки:*

Quick Basic

На початку програми (перед основним циклом):

```
' Початок відліку часу
StartTime = TIMER
```

У кінці програми (перед оператором END):

```
' Виведення часу розрахунку
PRINT USING "Час розрахунку ###.# с" ; TIMER - StartTime
```

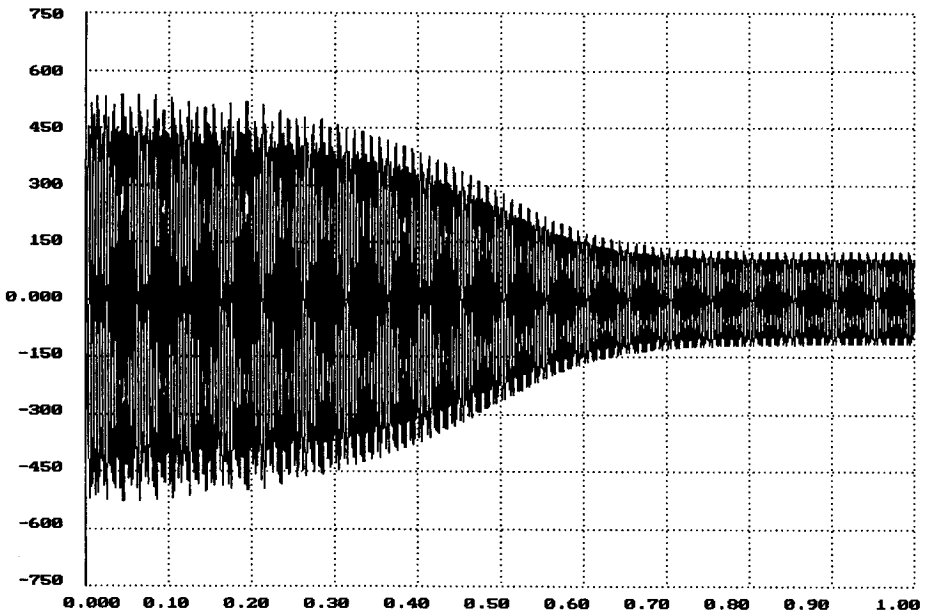


Рис. 4.14. Результати розрахунку фазних струмів під час запуску АД з короткозамкненим ротором для моделі у координатах α–β у середовищі Quick Basic

Turbo Pascal

```
{$E+,N+,F+ }
program  AD_phase_coord;
{ ------------------------------- }
{  Модель АД у фазних координатах  }
{ ------------------------------- }
uses
   Adams3, Crt, Plot;
const
   N = 5;                    { Порядок системи              }
   Tmax = 1.0;               { Максимальний час розрахунку  }
   Eps  : double = 1e-4;     { Точність на кроці }
   Ymin : double = -10.0;
   Ymax : double = 90.0;
   w = 100.0*3.1415926; { Кутова частота напруги живлення }
var
   t, h, M { момент АД }  : double;
   i  : integer;
   y  : Vector;

procedure  dif(x: double; var y, f : Vector);
{ ------------------------------------------------------ }
{ Процедура призначена для обчислення значень правих     }
{ частин системи диференціальних рівнянь, записаних у    }
{ нормальній формі Коші                                  }
{  Форма  запису:                                        }
{        f[1] := ...                                     }
{        . . .                                           }
{        f[N] := ...                                     }
{                                                        }
{  Вхідні величини:                                      }
{        t - незалежна змінна                            }
{        y - масив змінних (координат АД)                }
{  Вихідні величини:                                     }
{        y - масив змінних                               }
{        f - масив похідних координат АД                 }
{ ------------------------------------------------------ }
{ Моделювання режиму запуску АД з к.з. ротором           }
{ на основі моделі у фазних координатах                  }
```

```
{     y[1] - потік за альфа-координатою                          }
{     y[2] - потік за бета-координатою                           }
{     y[3] - струм за альфа-координатою                          }
{     y[4] - струм за бета-координатою                           }
{     y[5] - швидкість АД                                        }
{ ---------------------------------------------------------- }
const
   Ufn = 220.0;       { Номінальна фазна напруга статора АД  }
   Em = 311.1;        { Амплітуда фазної напруги АД          }
   Ifn = 104.0;       { Номінальний струм фази статора АД    }
   p = 4;             { Кількість пар полюсів                }
   Zb = Ufn/Ifn;      { Базовий опір                         }
   R1 = 0.103;        { Активний опір обмотки фази статора   }
   X1 = 0.172;        { Індуктивний опір розсіювання статора }
   { Зведені до статора:    }
      R2 = 0.237;          { Активний опір обмотки ротора     }
      X2 = 0.366;          { Індуктивний опір обмотки ротора  }
   L1 = X1/w;      { Індуктивність розсіювання обмотки статора }
   L2 = X2/w;      { Індуктивність розсіювання обмотки ротора  }
   Lm = 3.0*Zb/w;      { Взаємна індуктивність                 }
   Ls = L1 + Lm;       { Індуктивність обмотки статора         }
   Lr = L2 + Lm;       { Індуктивність обмотки ротора          }
   J = 4.5;            { Момент інерції двигуна                }
   Mc = 520.0;         { Статичний момент навантаження         }
var
   Ufa, Ufb, Ufc, Ua, Ub, Lh : double;
begin
   { Обчислення фазних напруг }
   Ufa := Em*sin(w*t);                    { Фаза A }
   Ufb := Em*sin(w*t - 2*Pi/3);           { Фаза B }
   Ufc := Em*sin(w*t + 2*Pi/3);           { Фаза C }
   Ua := (2*Ufa - Ufb - Ufc)/3;           { U-Alfa }
   Ub := (Ufb - Ufc)/sqrt(3);             { U-Beta }

   Lh := Lr/(Ls*Lr - sqr(Lm));
   { Обчислення похідних потоків за координатами }
   f[1] := R2*(Lm*y[3] - y[1])/Lr - p*y[5]*y[2];
   f[2] := R2*(Lm*y[4] - y[2])/Lr + p*y[5]*y[1];

   { Обчислення похідних струмів за координатами }
   f[3] := Lh*(Ua - R2*y[3] - Lm*f[1]/Lr);
   f[4] := Lh*(Ub - R1*y[4] - Lm*f[2]/Lr);
```

```
   M := 1.5*p*Lm*(y[1]*y[4] - y[2]*y[3])/Lr; { Обчислення моменту }
   f[5] := (M - Mc)/J;    { Обчислення похідної швидкості АД }
end;

begin
   h := 0.001;        { Початковий крок    }
   {   Ініціалізація графіка      }
   InitGraphic(0, Tmax, Ymin, Ymax, 0.1, 10.0);
   for i := 1 to N do y[i] := 0.0;  { Початкові умови }
   { ------------------------------------------  }
   {  Розрахунок динаміки режиму прямого запуску АД }
   { ------------------------------------------  }
   repeat
      ABM3(Dif, N, t, h, Eps, y);
      Graphic(1, t, M/30);  { Побудова графіка моменту АД    }
      Graphic(2, t, y[5]);  { Побудова графіка швидкості АД }
   until  t > Tmax;
   ReadLn;    { Натиснути ENTER для продовження }
end.
```

MathCAD

$U_{fn} := 220$	*Номінальна напруга фази двигуна*
$I_{fn} := 104$	*Номінальний струм фази двигуна*
$E_m := 220 \cdot \sqrt{2}$	*Амплітуда фазної напруги*
$p := 4$	*Кількість пар полюсів*
$f := 50$	*Частота напруги мережі живлення*
$\omega := 2 \cdot \pi \cdot f$	*Кутова частота напруги мережі*
$Z_b := \frac{U_{fn}}{I_{fn}}$	*Базовий опір*
$R_1 := 0.103$	*Активний опір обмотки статора*
$X_1 := 0.172$	*Індуктивний опір обмотки статора*
$L_1 := \frac{X_1}{\omega}$	*Індуктивність розсіювання обмотки статора*
$R_2 := 0.237$	*Зведений до статора активний опір обмотки ротора*

$X_2 := 0.366$ *Зведений до статора індуктивний опір обмотки ротора*

$L_2 := \frac{X_2}{\omega}$ *Зведена до статора індуктивність розсіювання обмотки ротора*

$L_m := \frac{3 \cdot Z_b}{\omega}$ *Взаємна індуктивність*

$L_s := L_1 + L_m$ *Індуктивність статора*

$L_r := L_2 + L_m$ *Зведена до статора індуктивність ротора*

$L_h := \frac{L_r}{L_s \cdot L_r - L_m^2}$

$J := 4.5$ *Момент інерції АД*

$M_c = 520$ *Статичний момент навантаження*

$T_{max} := 1$ *Кінцевий час розрахунку*

$N := 1000$ *Кількість кроків*

Функція-вектор правих частин системи диференціальних рівнянь:

$$
\begin{aligned}
Diff(t, y) := \;& Uf_A \leftarrow E_m \cdot \sin(\omega \cdot t) \\
& Uf_B \leftarrow E_m \cdot \sin\left(\omega \cdot t - \frac{2 \cdot \pi}{3}\right) \\
& Uf_C \leftarrow E_m \cdot \sin\left(\omega \cdot t + \frac{2 \cdot \pi}{3}\right) \\
& U_\alpha \leftarrow \frac{2 \cdot Uf_A - Uf_B - Uf_C}{3} \\
& U_\beta \leftarrow \frac{Uf_B - Uf_C}{\sqrt{3}} \\
& M \leftarrow 1.5 \cdot \frac{p \cdot L_m \cdot (y_0 \cdot y_3 - y_1 \cdot y_2)}{L_r}
\end{aligned}
$$

$$f_0 \leftarrow \frac{R_2 \cdot (L_m \cdot y_2 - y_0)}{L_r} - p \cdot y_4 \cdot y_1$$

$$f_1 \leftarrow \frac{R_2 \cdot (L_m \cdot y_3 - y_1)}{L_r} + p \cdot y_4 \cdot y_0$$

$$\begin{bmatrix} f_0 \\ f_1 \\ L_h \cdot \left(U_\alpha - R_2 \cdot y_2 - \frac{L_m}{L_r} \cdot f_0 \right) \\ L_h \cdot \left(U_\beta - R_1 \cdot y_3 - \frac{L_m}{L_r} \cdot f_1 \right) \\ \frac{M - M_c}{J} \end{bmatrix}$$

y_0 - потік за α-координатою

y_1 - потік за β-координатою

y_2 - струм за α-координатою

y_3 - струм за β-координатою

y_4 - швидкість АД

Початкові умови:

$$y_0 := \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Отримуємо розв'язок системи диференціальних рівнянь:

Res : = Rkadapt(y_0 , 0 , T_{max} , N , Diff)

Формування стовпців координат АД у зручному вигляді:

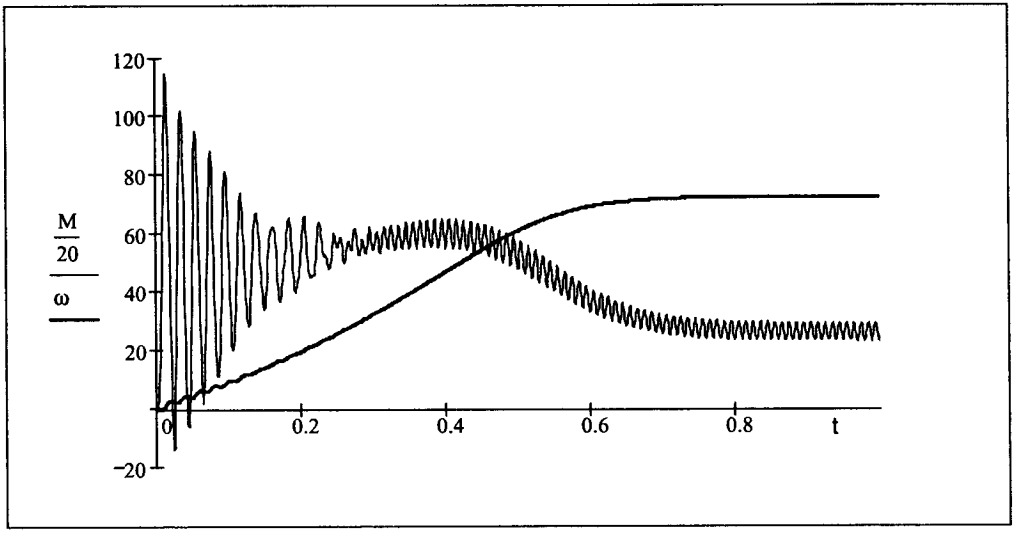
$t := Res^{<0>}$ $\Psi_\alpha := Res^{<1>}$ $\Psi_\beta := Res^{<2>}$ $i_\alpha := Res^{<3>}$

$i_\beta := Res^{<4>}$ $\omega := Res^{<5>}$

N : = last(t) i : = 0 .. N

$$M_i := \frac{1.5 \cdot p \cdot L_m \cdot (\Psi_{\alpha i} \cdot i_{\beta i} - \Psi_{\beta i} \cdot i_{\alpha i})}{L_r}$$

Обчислення моменту АД



MATLAB

```
% Розрахунок динаміки режиму прЯмого запуску АД з коротко-
% замкненим ротором на підставі моделі у фазних координатах
global p  Lr  Lm
Tmax = 1.0;          % Час розрахунку
y0 = [0 0 0 0 0];    % Початкові умови

% Розв'Язування диференціального рівнЯннЯ
[t, y] = ode113('am_phase', [0 Tmax], y0);

% ОбчисленнЯ фазних струмів статора
Ia = y(:,3)*cos(w*t) - y(:,4)*sin(w*t);
Ib = y(:,3)*cos(w*t - 2*pi/3) - y(:,4)*sin(w*t - 2*pi/3);
Ic = y(:,3)*cos(w*t + 2*pi/3) - y(:,4)*sin(w*t + 2*pi/3);

% ОбчисленнЯ моменту АД
M = 1.5*p*Lm*(y(:,1).*y(:,4) - y(:,2).*y(:,3))/Lr;

% Побудова графіка
plot(t, M(:)/20, '-k', t, y(:,5), 'k:');
```

```
axis([0 Tmax 0 inf]);
xlabel('t, c'); ylabel('M, \omega');
legend('M (1:20)', '\omega');
```

```
function f=am_phase(t, y);
% ------------------------------------------------------
%  Розрахунок режиму прЯмого запуску АД з короткозамкненим ротором
%     y(1) - потік за альфа-координатою
%     y(2) - потік за бета-координатою
%     y(3) - струм за альфа-координатою
%     y(4) - струм за бета-координатою
%     y(5) - швидкість
% ------------------------------------------------------
global p  Lr  Lm
Ufn = 220.0;            % Номінальна фазна напруга АД
Ifn = 220.0;            % Номінальний струм фази статора
Em = 311.1;             % Амплітуда фазної напруги
p = 4;                  % Кількість пар полюсів
w = 2*pi*50;           % Кутова частота напруги мережі живленнЯ АД
Zb = Ufn/Ifn;           % Базовий опір
Lm = 3.0*Zb/w;          % Взаємна індуктивність
R1 = 0.103;             % Активний опір статора
X1 = 0.172;             % Індуктивний опір розсіюваннЯ статора
L1 = X1/w;              % Індуктивність розсіюваннЯ статора
Ls = L1 + Lm;           % Індуктивність статора
% Зведені до статора:
R2 = 0.237;             % Активний опір ротора
X2 = 0.366;             % Індуктивний опір розсіюваннЯ ротора
L2 = X2/w;              % Індуктивність розсіюваннЯ ротора
Lr = L2 + Lm;           % Індуктивність ротора
Lh = Lr/(Ls*Lr - Lm*Lm);

J = 4.5;                % Момент інерції АД
Mc = 53.03;             % Статичний момент навантаженнЯ

% ОбчисленнЯ правих частин системи диференціальних рівнЯнь
f=zeros(5,1);
% ОбчисленнЯ фазних напруг
Ufa = Em*sin(w*t);                          % Фаза A
Ufb = Em*sin(w*t - 2*pi/3.0);               % Фаза B
```

```
Ufc = Em*sin(w*t + 2*pi/3.0);                    % Фаза C
Ua = (2*Ufa - Ufb - Ufc)/3.0;                    % U-Alfa
Ub = (Ufb - Ufc)/sqrt(3);                        % U-Beta

% Обчислення потоків АД
f(1) = R2*(Lm*y(3) - y(1))/Lr - p*y(5)*y(2);
f(2) = R2*(Lm*y(4) - y(2))/Lr + p*y(5)*y(1);

% Обчислення струмів АД
f(3) = Lh*(Ua - R2*y(3) - Lm*f(1)/Lr);
f(4) = Lh*(Ub - R1*y(4) - Lm*f(2)/Lr);

% Обчислення моменту АД
M = 1.5*p*Lm*(y(1)*y(4) - y(2)*y(3))/Lr;
f(5) = (M - Mc)/J;                               % Похідна швидкості АД
```

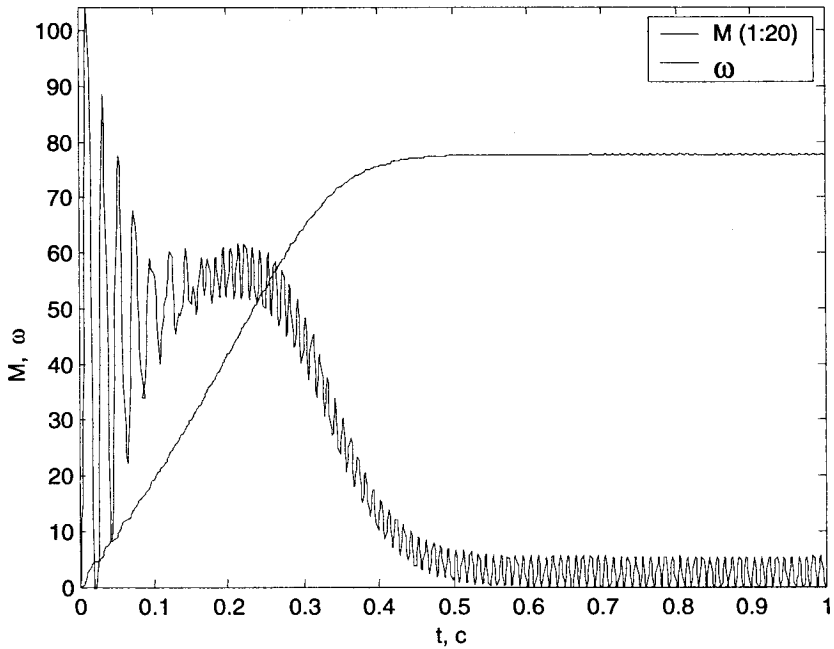
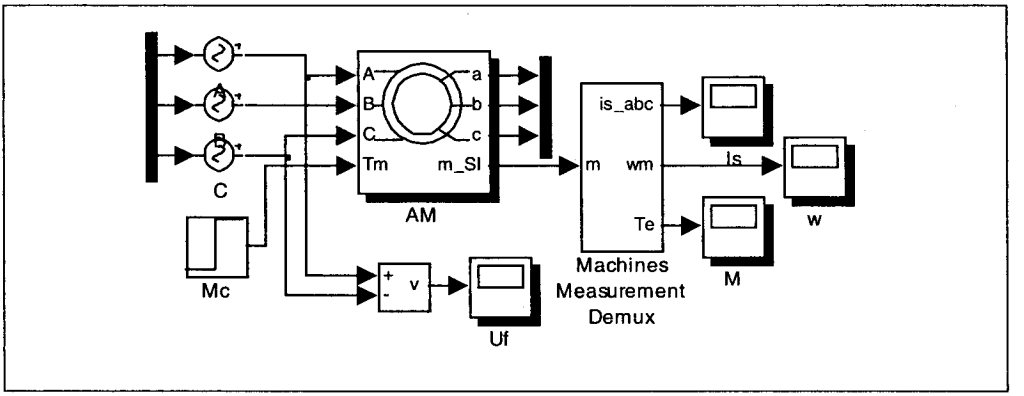


Рис. 4.15. Результати розрахунку динаміки режиму прямого запуску АД з короткозамкненим ротором для моделі у фазних координатах у середовищі MATLAB

Структурна схема моделі для дослідження динаміки режиму прямого запуску АД в середовищі Simulink спрощується з використанням блоків з бібліотеки елементів Power System Toolbox.

Simulink + Power System Toolbox



Прямий запуск АД з двомасовою моделлю механічного навантаження

Розрахувати динаміку режиму прямого запуску АД з короткозамкненим ротором типу МТКН 512-8 потужністю $P_н$ = 45 кВт (див. попередній приклад) з двомасовою моделлю механічної частини. Результати розрахунку режиму запуску показано на рис. 4.16 і 4.17 (масштаб моменту – 1:20).

Quick Basic

```
DECLARE SUB Graphic (N%, x!, y!)
DECLARE SUB Dif (t!, y!(), f!())
DECLARE SUB adams3 (N!, t!, h!, Eps!, y!())
```

```
'-------------------------------------------------------
' Розрахунок динаміки режиму прямого запуску
' АД з короткозамкненим ротором з урахуванням
' двомасової моделі механічної частини привода
'-------------------------------------------------------
CONST N = 4                ' Порядок системи
DIM SHARED M12             ' Пружний момент - глобальна змінна
DIM y(N)                   ' Масив змінних
CONST Tmax = 2.5           ' Максимальний час розрахунку
CONST Eps = .001           ' Точність розв'язку на кроці
CONST Xmin = 0!            ' Ліва межа графіка
CONST Xmax = Tmax          ' Права межа графіка
CONST Ymin = -10           ' Нижня межа графіка
CONST Ymax = 90            ' Верхня межа графіка

FOR i = 1 TO N             '
   y(i) = 0                ' Нульові початкові умови
NEXT                       '
h = .01                    ' Початковий крок інтегрування
t = 0!                     ' Початковий час
' -------   Основний цикл програми   -------
DO
   CALL Graphic(1,t,y(2))    ' Побудова графіка швидкості АД
   CALL Graphic(2,t,y(4))    ' Побудова графіка швидкості 2-ї маси
   CALL Graphic(3,t,M12/20)  ' Побудова графіка пружного моменту
                             ' у масштабі 1/20
   CALL adams3(N, t, h, Eps, y())   ' Виклик числового методу
LOOP UNTIL t > Tmax

END

SUB Dif (t, y(), f()) STATIC
' ------------------------------------------------------------
' Підпрограма призначена для обчислення значень правих частин
' системи диференціальних рівнянь, записаних у нормальній
' формі Коші.
' Форма  запису:
'       f(1) = функція_1(y(1),y(2),...,y(N),t)
'       f(2) = функція_2(y(1),y(2),...,y(N),t)
'                       . . .
'       f(N) = функція_N(y(1),y(2),...,y(N),t)
'
```

```
'  Вхідні величини:
'       t - незалежна змінна
'       y - масив змінних (координат електропривода)
'  Вихідні величини:
'       y - масив змінних
'       f - масив похідних координат електропривода
' ------------------------------------------------------
'      y(1)  - похідна моменту двигуна
'      y(2)  - похідна кутової швидкості двигуна
'      y(3)  - похідна кута скручування вала
'      y(4)  - похідна кутової швидкості другої маси
' ------------------------------------------------------
CONST Pn = 45000            ' Номінальна потужність АД
CONST W0 = 78.5             ' Синхронна кутова швидкість АД
CONST Wn = 71.2             ' Номінальна кутова швидкість АД
CONST LambdaM = 2.5         ' Кратність максимального моменту
CONST Mn = Pn / Wn          ' Номінальний момент АД
CONST Mk = Mn * LambdaM         ' Макс. (критичний) момент АД
CONST Sn = (W0 - Wn) / W0       ' Номінальне ковзання АД
CONST J1 = 4.5            ' Момент інерції двигуна
CONST R1 = .103           ' Активний опір обмотки статора
CONST X1 = .172           ' Індуктивний опір обмотки статора
CONST R2 = .237           ' Зведений до статора акт-ний опір ротора
CONST X2 = .366           ' Зведений до статора інд-ний опір ротора
CONST Xk = X1 + X2        ' Індуктивний опір к.з. АД
CONST a = R1 / R2         ' Використовується для розрахунку моменту
' Зведені до валу двигуна
CONST C12 = 2950         ' Пружність вала
CONST J2 = 12.3          ' Момент інерції 2-ї маси
CONST Beta = 15!         ' Коефіцієнт природного демпфування
CONST Mc = 520!          ' Статичний момент навантаження

Sk = R2 / SQR(R1 ^ 2 + Xk ^ 2)       ' Критичне ковзання
'  Y(2) - швидкість обертання АД
S = (W0 - y(2)) / W0                 ' Обчислення ковзання
' Обчислення моменту за уточненою формулою Клоса
M = 2 * Mk * (1 + a * Sk) / (S / Sk + Sk / S + 2 * a * Sk)

' Обчислення електромагнітної сталої часу
Te = 1! / (W0 * Sk)
```

```
f(1) = (M - y(1)) / Te          ' Обчислення похідної моменту АД
' ======================================
'   Модель механічної частини
' ======================================
M12 = C12 * y(3) + Beta * (y(2) - y(4))     ' Пружний момент
' Обчислення похідної швидкості двигуна
f(2) = (y(1) - M12) / J1
' Обчислення похідної кута скручування вала
f(3) = y(2) - y(4)
' Обчислення похідної швидкості другої маси
f(4) = (M12 - Mc) / J2

END SUB
```

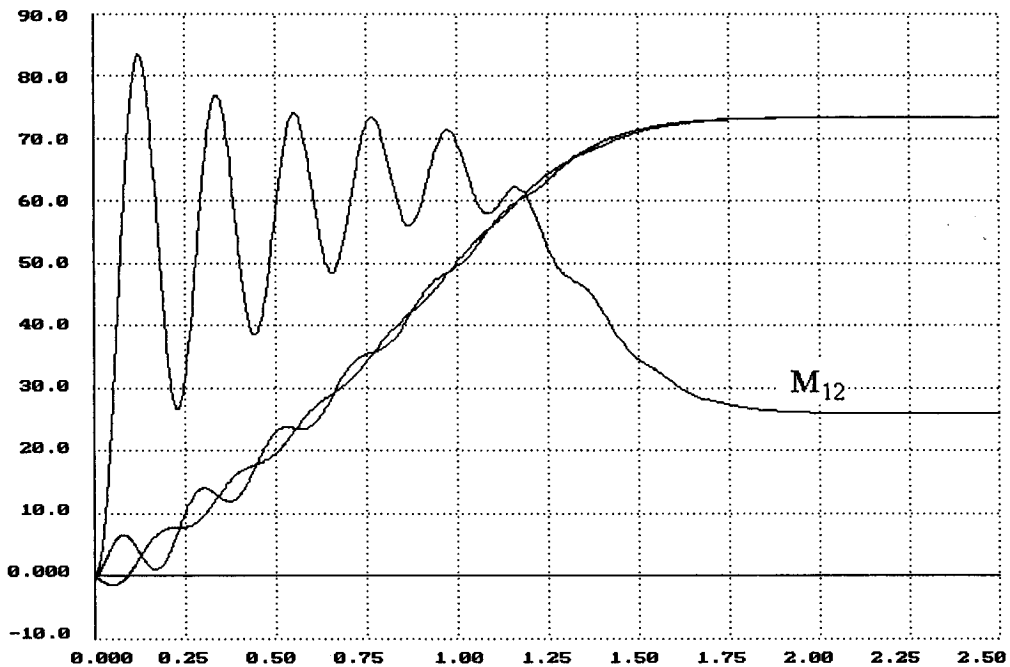


Рис. 4.16. Результати розрахунку запуску АД з короткозамкненим ротором і двомасовою механічною частиною привода в середовищі Quick Basic

Turbo Pascal

```
{$E+,N+,F+ }
program AD_Mech;
uses
   Adams3, Crt, Plot;
const
   N = 4;                    { Порядок системи                }
   Tmax = 2.5;               { Максимальний час розрахунку    }
   Eps  : DOUBLE = 1e-4;     { Точність на кроці }
   Ymin : DOUBLE = -10.0;
   Ymax : DOUBLE = 90.0;
var
   t, h, M12 {пружний момент}  : DOUBLE;
   i  : INTEGER;
   y  : Vector;

procedure  dif(x: DOUBLE; var y, f : Vector);
{ ------------------------------------------------ }
{ Процедура обчислення значень правих частин системи }
{ диференціальних рівнянь у нормальній формі Коші    }
{  Форма  запису:                                    }
{         f[1] := ...                                }
{         . . .                                      }
{         f[N] := ...                                }
{                                                    }
{  Вхідні величини:                                  }
{       t - незалежна змінна                         }
{       y - масив змінних (координат електропривода) }
{  Вихідні величини:                                 }
{       y - масив змінних                            }
{       f - масив похідних координат електропривода  }
{ ------------------------------------------------ }
{ Запуск АД з короткозамкненим ротором і врахуванням }
{ двомасової моделі механічної частини привода       }
{ ------------------------------------------------ }
const
   Pn = 45000.0;      { Номінальна потужність АД              }
   W0 = 78.5;         { Синхронна кутова частота обертання АД }
   Wn = 71.2;         { Номінальна частота обертання АД       }
   LambdaM = 2.5;     { Кратність максимального моменту АД    }
```

```
   Mn = Pn / Wn;      { Номінальний момент АД                    }
   Mk = Mn * LambdaM;    { Макс. (критичний) момент АД           }
   R1 = 0.103;        { Активний опір обмотки статора            }
   X1 = 0.172;        { Індуктивний опір обмотки статора         }
   R2 = 0.237;        { Зведений до статора акт.опір ротора      }
   X2 = 0.366;        { Зведений до статора інд.опір ротора      }
   Xk = X1 + X2;      { Індуктивний опір короткого замикання     }
   a = R1/R2;         { Використовується для розрах. моменту     }
   J1 = 4.5;          { Момент інерції АД                        }
   { Зведені до вала двигуна:    }
   C12 = 2950.0;  { Пружність вала    }
   J2 = 12.3;         { Момент інерції другої маси               }
   Beta = 15.0;    { Коефіцієнт природного демпфування           }
   Mc = 520.0;     { Статичний момент навантаження               }
var
   S, Sk, M, Te : double;
begin
   Sk := R2/sqrt(sqr(R1)+sqr(Xk));     { Критичне ковзання }
   {  Y[2] - швидкість обертання АД  }
   S := (W0 - Y[2]) / W0;             { Обчислення ковзання }
   { Обчислення моменту за уточненою формулою Клоса }
   M := 2*Mk*(1.0+a*Sk)/(S/Sk + Sk/S + 2*a*Sk);

   { Обчислення електромагнітної сталої часу }
   Te := 1.0 / (W0 * Sk);

   f[1] := (M - Y[1]) / Te;       { Похідна моменту АД }
   { --------------------------- }
   {  Модель механічної частини  }
   { --------------------------- }
   M12 := C12*y[3] + Beta*(y[2] - y[4]);    { Пружний момент }
   f[2] := (y[1] - M12)/J1;           { Похідна швидкості АД }
   f[3] := y[2] - y[4];             { Похідна скручування вала }
   f[4] := (M12 - Mc)/J2;    { Похідна швидкості другої маси }
end;

begin
   h := 0.001;       { Початковий крок    }
{   Ініціалізація графіка      }
   InitGraphic(0, Tmax, Ymin, Ymax, 0.5, 10.0);
   for i := 1  to N do y[i] := 0.0;  { Початкові умови }
{ ------------------------------------  }
{  Розрахунок динаміки режиму запуску АД      }
{ ------------------------------------  }
```

```
   repeat
     ABM3(Dif, N, t, h, Eps, y);
     Graphic(1, t, M12/20); { Побудова графіка пружного моменту }
     Graphic(2, t, y[2]);   { Побудова графіка швидкості АД }
     Graphic(3, t, y[4]); { Побудова графіка швидкості мех-му }
   until  t > Tmax;
   ReadLn;    { Натиснути ENTER для продовження }
end.
```

MathCAD

$P_{nom} := 45000$	*Потужність двигуна*
$\omega_0 = 78.5$	*Синхронна кутова швидкість обертання*
$\omega_{nom} = 71.2$	*Номінальна кутова швидкість обертання*
$M_{nom} := \frac{P_{nom}}{\omega_{nom}}$	*Номінальний момент двигуна*
$\lambda := 2.5$	*Кратність критичного (максимального) моменту АД*
$M_k := M_{nom} \cdot \lambda$	*Критичний момент двигуна*
$J_1 := 4.5$	*Момент інерції АД*
$R_1 := 0.103$	*Активний опір статора*
$X_1 := 0.172$	*Індуктивний опір статора*
$R_2 := 0.237$	*Зведений до статора активний опір фази ротора*
$X_2 := 0.366$	*Зведений до статора індуктивний опір фази ротора*
$X_k := X_1 + X_2$	*Індуктивний опір короткого замикання*
$a := \frac{R_1}{R_2}$	
Зведені до валу АД:	
$C_{12} := 2950$	*Пружність вала*

$J_2 := 12.3$ *Момент інерції навантаження*

$\beta := 15$ *Коефіцієнт природного демпфування*

$M_c = 520$ *Статичний момент навантаження*

$T_{max} := 2.5$ *Кінцевий час розрахунку*

$N := 1000$ *Кількість кроків* $M_{12} := 0$

Функція-вектор правих частин системи диференціальних рівнянь:

$$\text{Diff}(t,y) := \begin{bmatrix} s_k \leftarrow \dfrac{R_2}{\sqrt{R_1^{\,2} + X_k^{\,2}}} \\ s \leftarrow \dfrac{\omega_0 - y_1}{\omega_0} \\ M \leftarrow \dfrac{2 \cdot M_k \cdot (1 + a \cdot s_k)}{\dfrac{s}{s_k} + \dfrac{s_k}{s} + 2 \cdot a \cdot s_k} \\ T_e \leftarrow \dfrac{1}{\omega_0 \cdot s_k} \\ M_{12} \leftarrow C_{12} \cdot y_2 + \beta \cdot (y_1 - y_3) \\ \dfrac{M - y_0}{T_e} \\ \dfrac{y_0 - M_{12}}{J_1} \\ y_1 - y_3 \\ \dfrac{M_{12} - M_c}{J_2} \end{bmatrix}$$

y_0 – електромагнітний момент

y_1 – кутова швидкість АД

y_2 – кут скручування вала

y_3 – кутова швидкість механізму

Початкові умови:

$$y_0 := \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

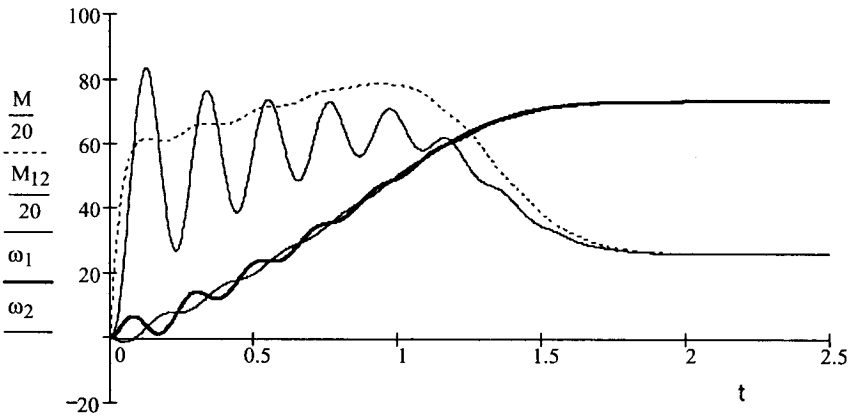
Розв'язок системи диференціальних рівнянь:

$$Res := Rkadapt(y_0, 0, T_{max}, N, Diff)$$

Формування стовпців координат електропривода в зручному вигляді:

$t := Res^{<0>}$ $M := Res^{<1>}$ $\omega_1 := Res^{<2>}$ $\omega_2 := Res^{<4>}$

$M_{12} := C_{12} \cdot Res^{<3>} + \beta \cdot (\omega_1 - \omega_2)$



MATLAB

```
%  Розрахунок динаміки режиму запуску АД з к.з. ротором
%  і урахуванням двомасової моделі привода
Tmax = 2.5;          % Час розрахунку
y0 = [0 0 0 0];      % Початкові умови
C12 = 2050;          % Пружність вала
Beta = 15.0;         % Коефіцієнт природного демпфування

% Розв'язування диференціального рівняння
[t, y] = ode23('am_mech', [0 Tmax], y0);
```

```
M12 = C12*y(:,3) + Beta*(y(:,2) - y(:,4));  % Пружний момент
% Побудова графіка
plot(t,y(:,1)/20,'-k.',t,y(:,2),'k:',t,M12/20,'k',t,y(:,4),'k');
axis([0 Tmax 0 inf]);
xlabel('t, c'); ylabel('M, M_{12}, \omega_1, \omega_2');
legend('M (1:20)', '\omega_1', 'M_{12} (1:20)', '\omega_2');
```

```
function f=am_fr(t, y);
% ------------------------------------------------------
%  Розрахунок динаміки режиму запуску АД з к.з. ротором
%  і урахуванням двомасової моделі привода
% ------------------------------------------------------
Pnom = 45000;           % Номінальна потужність АД
W0 = 78.5;              % Синхронна частота обертання АД
Wnom = 71.2;            % Номінальна частота обертання АД
LambdaM = 2.5;          % Кратність максимального моменту АД
Mnom = Pnom/Wnom;       % Номінальний момент АД
Mk = Mnom * LambdaM;    % Максимальний (критичний) момент АД
Sn = (W0-Wnom)/W0;      % Номінальне ковзання АД
J1 = 4.5;               % Момент інерції АД
R1 = 0.103;             % Активний опір обмотки статора
X1 = 0.172;             % Індуктивний опір обмотки статора
R2 = 0.237;             % Зведений до статора акт. опір ротора
X2 = 0.366;             % Зведений до статора інд. опір ротора
Xk = X1 + X2;           % Індуктивний опір короткого замикання
a = R1/R2;             % Використовується для розрахунку моменту
% Зведені до вала АД:
C12 = 2050;             % Пружність вала
J2 = 12.3;              % Момент інерції механізму
Beta = 15.0;            % Коефіцієнт природного демпфування
Mc = 53.03;             % Статичний момент навантаження

% Обчислення правих частин системи диференціальних рівнянь
f=zeros(4,1);
Sk = R2/sqrt(R1^2 + Xk^2); % Критичне ковзання АД
S = (W0 - y(2))/W0;        % Розрахунок ковзання АД
% Розрахунок моменту АД за уточненою формулою Клоса
M = 2*Mk*(1 + a*Sk)/(S/Sk + Sk/S + 2*a*Sk);
Te = 1.0/(W0*Sk);     % Обчислення електромагнітної сталої часу АД

f(1) = (M - y(1))/Te;  % Похідна електромагнітного моменту АД
% ---------------------------------
%  Модель механічної частини
% ---------------------------------
```

```
M12 = C12*y(3) + Beta*(y(2) - y(4));   % Пружний момент
f(2) = (y(1) - M12)/J1;                % Похідна швидкості АД
f(3) = y(2) - y(4);              % Похідна кута скручуванняя вала
f(4) = (M12 - Mc)/J2;            % Похідна швидкості механізму
```

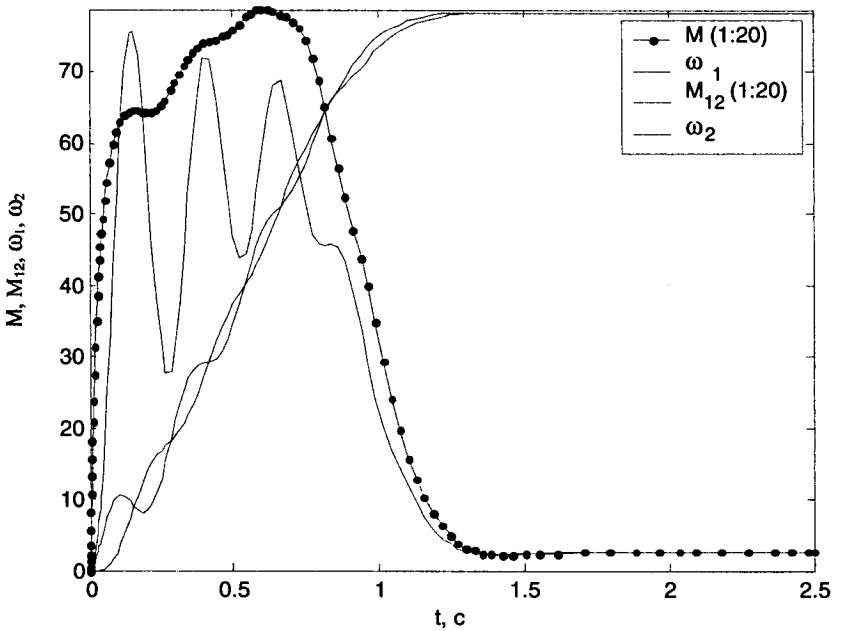


Рис. 4.17. Результати розрахунку запуску АД з короткозамкненим ротором і двомасовою механічною частиною привода в середовищі MATLAB

4.3. Моделювання тягового електропривода за схемою ШІП–ДПС послідовного збудження

У цьому розділі на прикладі показано спосіб отримання комп'ютерної моделі тягового електропривода.

Експлуатаційні режими машин міського електротранспорту (трамваїв, тролейбусів) ставлять низку специфічних вимог як до самого електропривода, так і до системи автоматичного керування. До таких вимог належать: необхід-

ність формування спеціальних тягових характеристик, забезпечення надійних та безпечних умов експлуатації транспортного засобу як у нормальних робочих режимах, так і в аварійних: в режимах буксування, юзу, заносу тощо.

Однією із систем електроприводів, що найкраще задовольняє поставлені вимоги, є система за схемою "широтно-імпульсний перетворювач – двигун постійного струму послідовного збудження" (ШІП–ДПС ПЗ). Використання у замкненій структурі на підставі такої системи формувального зворотного зв'язку за струмом (моментом) двигуна дає змогу отримати необхідні "тягові" механічні характеристики з ефективним обмеженням струму (моменту) на рівні допустимих значень, а додаткове введення у структуру коригувального паралельного зворотного зв'язку за пришвидшенням дає змогу уникнути чи значно обмежити пружне пробуксовування, тобто одержати належне функціонування транспортного засобу в пускогальмівних режимах за нормальних чи погіршених умов зчеплення коліс з дорожнім покриттям, тобто у нештатних режимах буксування чи юзу. Приклад варіанта такого тягового електропривода показано на рис. 4.18.

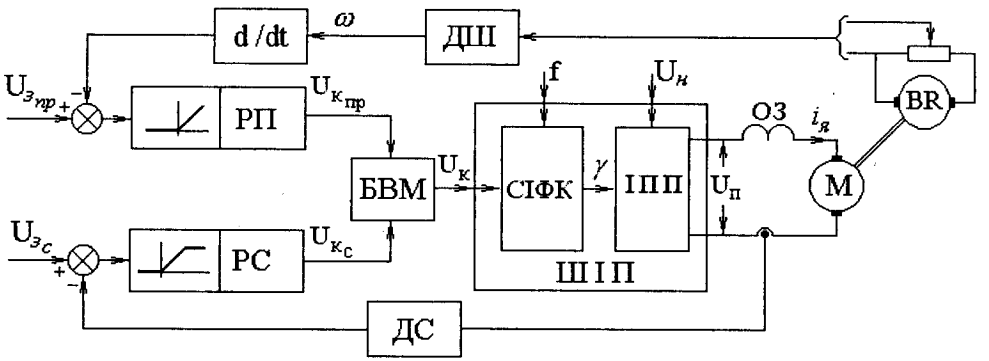


Рис. 4.18. Функціональна блок-схема електропривода за схемою ШІП–ДПС ПЗ з паралельними зворотними зв'язками

Дослідження режимів такої системи за дії різних зовнішніх умов і збурювальних впливів, зокрема й виконання параметричної оптимізації системи керування за критеріями мінімальних відхилень оптимальних та реальних статичних та динамічних характеристик електропривода доцільно здійснити методом математичного моделювання.

Складемо математичну модель описаної та показаної на рис. 4.18 замкненої системи електропривода за схемою ШІП-ДПС ПЗ. Повну математичну модель такої електромеханічної системи подамо послідовним викладенням моделей окремих її композиційних елементарних ланок, а саме:

- широтно-імпульсного перетворювача;
- привідного двигуна постійного струму послідовного збудження;
- механічної системи;
- системи автоматичного керування.

Для регульованих електроприводів загального призначення, що живляться через випростувач від мережі змінного струму і працюють з широтно-імпульсним регулюванням, використовують два різні види модуляції: широтно-імпульсну та частотно-імпульсну, яка застосовується рідше. Порівняно з тиристорними перетворювачами вони мають кращі динамічні показники для діапазону регулювання 2000–6000, краще використання двигуна за струмом і менший негативний вплив на електромережу. У широтно-імпульсному перетворювачі модуляція використовується за незмінної несучої частоти при зміні параметра γ щілинності імпульсів на періоді, а в частотно-імпульсному – за незмінної тривалості імпульсу, щілинність змінюється через зміну їх частоти. Використовують одноплечові та мостові (двоплечові) широтно-імпульсні перетворювачі [Б.6, Б.8]. Одноплечові ШІП вимагають джерела живлення з середньою точкою.

У схемі тягового електропривода тролейбуса за рис. 4.18 регулювання швидкості, а, якщо необхідно, і реверс здійснюється відповідним шунтуванням чи перемиканням обмотки збудження. Тому такий привід вимагає нереверсивного ШІП. На рис. 4.19, *а* показано схему силового кола мостового нереверсивного ШІП з несиметричним алгоритмом керування транзисторними ключами, а на рис. 4.19, *б* – часова діаграма його сигналів.

Регулювання напруги на двигуні у такій схемі ШІП реалізується перемиканням транзисторних ключів VT3 та VT4 при постійно ввімкненому VT1 і постійно закритому VT2 (рис. 4.19, *б*). Транзисторні ключі VT3 та VT4 перемикаються у протифазі. На виході ШІМ формуються імпульси напруги однієї полярності.

Так, при закриванні ключа VT3 і закриванні VT4 в моменти часу t_0, t_2, t_4, ... двигун на інтервалах $t_0 - t_1$, $t_2 - t_3$, $t_4 - t_5$, ... підмикається до джерела напруги

$+U_н$ і струм якоря двигуна $i_я$ зростає від мінімального I_{min} до максимального I_{max} значення. У моменти часу t_1, t_3, t_5, ... перемикання робочих ключів (VT3 відкривається, а VT4 – закривається) двигун відмикається від джерела напруги і його струм $i_я$ внаслідок наявної в колі якоря індуктивності зменшується від максимального I_{max} до мінімального I_{min} значення (t_2, t_4, t_6, ...), бо напруга на якорі двигуна на цих інтервалах $t_1 - t_2$, $t_3 - t_4$, $t_5 - t_6$, ... дорівнює нулеві.

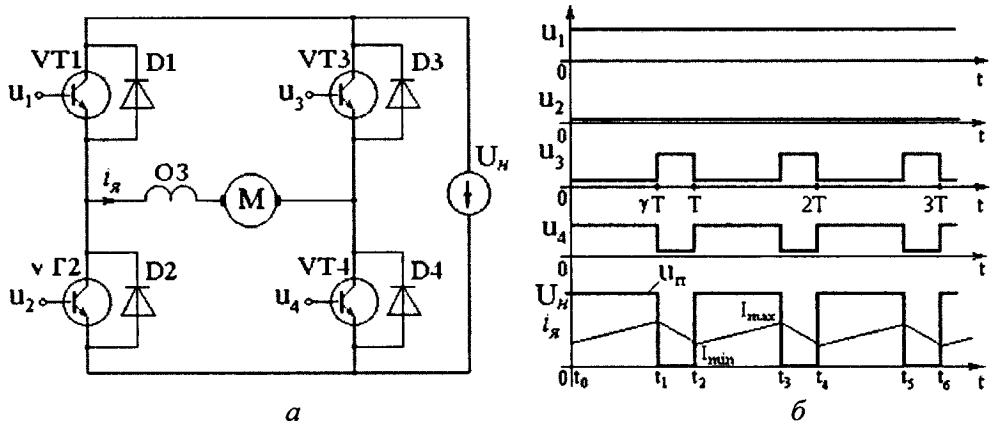


Рис. 4.19. Мостовий ШІП з несиметричним алгоритмом керування (а) та часова діаграма його сигналів (б)

Якщо тривалість першого стану позначити γT, то другий триватиме $(1 - \gamma)T$, де T – період несучої частоти комутації транзисторних ключів; γ – відносна тривалість першого стану, що змінюється від 0 до 1. Частота комутації транзисторів ШІП в електроприводах такого класу становить 3–10 кГц.

Для описання процесів у широтно-імпульсному перетворювачі приймемо такі припущення:

1) транзисторні ключі ідеальні (час відкривання та закривання дорівнює нулеві, опір у відкритому та закритому станах дорівнює нулеві та нескінченності відповідно);
2) напруга на якорі двигуна змінюється миттєво;
3) внутрішнім спадом напруги джерела живлення нехтуємо.

За таких припущень і описаного несиметричного алгоритму комутації ключів на виході ШІП перетворювача формується напруга $U_п$ у вигляді однополярних імпульсів, середнє значення якої розраховують за виразом:

$$U_п = \frac{1}{T}\int_0^{\gamma T} U_н dt = \gamma U_н. \qquad (4.1)$$

Регулювальна характеристика ШІП для такої його схеми та алгоритму комутації силових ключів, яка визначається за (4.1), показана на рис. 4.20.

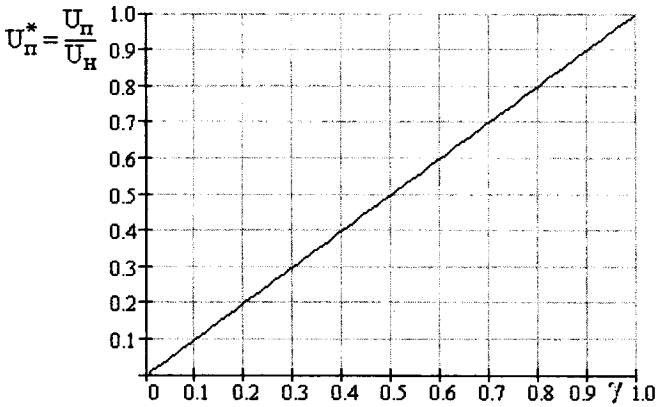


Рис. 4.20. Регулювальна характеристика ШІП для несиметричного керування

Зважаючи на відносно високу частоту комутації порівняно з частотою зрізу замкненої системи електропривода приймаємо, що широтно-імпульсний перетворювач є безінерційною ланкою з коефіцієнтом передачі $k_п = \Delta U_п / \Delta\gamma = U_н / U_{вх\,max}$, де $U_н$ – напруга живлення ШІП, $U_{вх\,max}$ – максимальна вхідна керуюча напруга ШІП, звичайно становить 10 В. У такому разі модель ШІП є найпростішою – це пропорційна ланка з коефіцієнтом підсилення $k_п$, а імпульсний характер роботи ШІП можна імітувати відповідним кроком моделювання.

Якщо необхідна детальніша модель ШІП, моменти відкривання та закривання транзисторних ключів VS3 та VS4 у функції несучої частоти f та сигналу керування γ встановлюються в моделі на підставі розв'язування логічних рівнянь, що описують роботу системи імпульсно-фазового керування (СІФК) ШІП. Керування імпульсним перетворювачем потужності (ІПП), що утворює силове коло ШІП (рис. 4.19, а), відбувається у дискретні моменти відкривання транзисторних ключів, але між цими моментами сигнал керування γ не впливає на вихідну напругу перетворювача. Рівняння, що описують роботу СІФК та ІПП широтно-імпульсного перетворювача, є такими:

$$t_{відкр\ i} = t_{відкр\ i-1} + T; \tag{4.2}$$

$$t_{закр\ i} = t_{відкр\ i} + \gamma T; \tag{4.3}$$

$$u_n = \begin{cases} U_н \ \ для \ \ t_{відкр_i} \le t \le t_{закр_i}; \\ 0 \ \ для \ \ t_{закр_i} < t < t_{відкр_i} + T, \end{cases} \tag{4.4}$$

де $t_{відкр\ i}$, $t_{закр\ i}$ *– моменти відкривання та закривання транзисторних ключів на і-му інтервалі комутації;*

$U_н$ *– номінальна напруга джерела живлення;*

u_n *– миттєва вихідна напруга широтно-імпульсного перетворювача.*

Характер роботи ШІП якнайкраще моделюється за допомогою Z-перетворення, а відповідним вибором кроку моделювання можна відтворити імпульсний характер роботи ШІП. Тому далі для розроблення комп'ютерної моделі використані різницеві рівняння, що отримані на підставі Z-перетворення із застосуванням фіксатора нульового порядку (див. розд. 3.4). Для спрощення побудови моделі використано моделювальні рівняння лише для двох типів динамічних ланок – інтегральної та аперіодичної. Такий підхід дає змогу за невеликої втрати точності забезпечити порівняно просте формування системи моделювальних різницевих рівнянь.

Інтегральна ланка

Виконаємо Z-перетворення для фіксатора нульового порядку та інтегральної ланки:

$$Z\left(\left(\frac{1-e^{-sh}}{s}\right)\cdot\left(\frac{1}{Ts}\right)\right) = (1-z^{-1})Z\left(\frac{1}{s^2T}\right) = (1-z^{-1})\cdot\frac{h}{T}\cdot\frac{z}{(z-1)^2} = \frac{h}{T}\cdot\frac{1}{z-1},$$

за дискретною передатною функцією отримуємо рекурентне рівняння для моделювання інтегральної ланки:

$$y_{i+1} = y_i + \frac{h}{T}\cdot x_i.$$

Аперіодична ланка

Здійснимо Z-перетворення для фіксатора нульового порядку і аперіодичної ланки:

$$Z\left(\left(\frac{1-e^{-sh}}{s}\right)\cdot\left(\frac{K}{Ts+1}\right)\right) = (1-z^{-1})Z\left(\frac{K}{s(Ts+1)}\right)$$

$$= (1-z^{-1})\cdot K\cdot\frac{\left(1-e^{-ah}\right)z}{(z-1)\left(z-e^{-ah}\right)} = \frac{\left(1-e^{-\frac{h}{T}}\right)K}{z-e^{-\frac{h}{T}}},$$

за дискретною передатною функцією отримуємо рекурентне рівняння для моделювання аперіодичної ланки:

$$y_{i+1} = y_i\cdot e^{-\frac{h}{T}} + \left(1-e^{-\frac{h}{T}}\right)\cdot K\cdot x_i.$$

Застосування фіксатора першого порядку дає змогу одержати вищу точність, але при цьому отримують складніші вирази. Використання достатньо малого кроку моделювання (порівняно з найменшою сталою часу системи) дає змогу зменшити вплив похибок від застосування фіксатора нульового порядку для отримання моделювальних рівнянь.

Для адекватного відображення в моделі режимів двигуна постійного струму послідовного збудження використаємо його уточнену модель, що враховує такі особливості [Д.15]:

1) наявність вихрових струмів у магнітопроводі (врахуємо в моделі введенням окремої сталої часу T_k вихрових струмів);
2) нелінійність характеристики намагнічування магнітопроводу двигуна (виконаємо її наближення функцією арктангенсу);
3) вплив реакції якоря (подамо в моделі коефіцієнтом K_{ar}).

Розробляючи математичну модель двигуна, приймемо такі припущення:

- індуктивності розсіювання та обмотки якоря не залежать від магнітного потоку;
- реакція якоря пропорційна до струму якоря;
- ефект витіснення струму в магнітопроводі (поверхневий ефект) не враховується.

З урахуванням прийнятого рівня деталізації процесів у двигуні та наведених вище припущень на рис. 4.21 зображено розрахункову схему математичної моделі силового кола двигуна послідовного збудження, а на рис. 4.22 – структурну схему його математичної моделі.

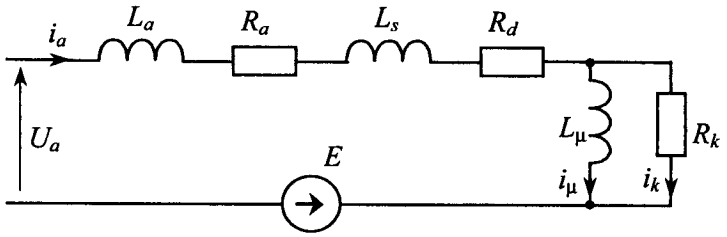


Рис. 4.21. Розрахункова схема математичної моделі силового кола ДПС ПЗ

На цих рисунках використано такі позначення: i_a – струм якірного кола; U_a – напруга на якорі двигуна; i_μ – струм намагнічування; $i_k = i_a - i_\mu$ – вихровий струм; L_a , R_a – індуктивність та активний опір обмотки якоря; $R_{ак}$ – активний опір якірного кола двигуна; L_s – індуктивність розсіювання; L_μ – індуктивність

намагнічування; R_d – активний опір обмотки збудження; R_k – активний опір фіктивного контуру вихрових струмів, K_{ar} – коефіцієнт реакції якоря, E, J_M, K_e – ЕРС, момент інерції та конструктивний коефіцієнт двигуна відповідно.

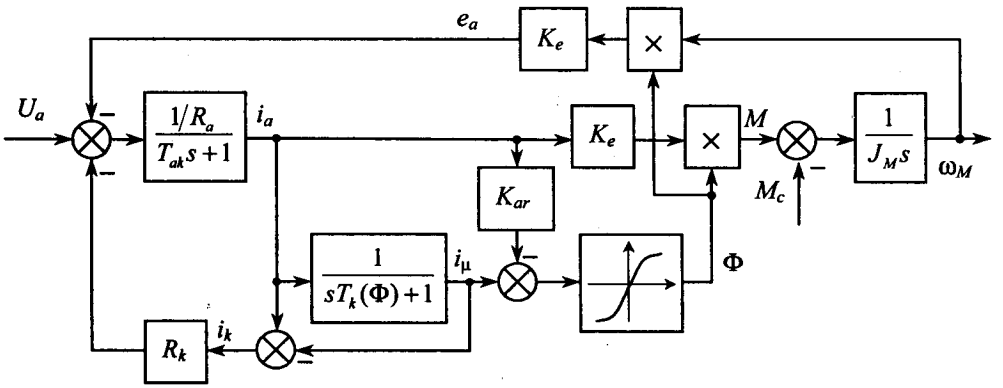


Рис. 4.22. Структурна схема математичної моделі силового кола ДПС ПЗ

Уведемо позначення: $T_{ak} = (L_a + L_s)/(R_a + R_d)$; $T_k = L_\mu / R_k$; $T_\mu = L_\mu / R_d$, A_Φ, B_Φ коефіцієнти аналітичної залежності, що описує (наближує) характеристику намагнічування, Φ – потік двигуна.

З урахуванням цього система рівнянь, що відповідає поданій на рис. 4.22 структурній схемі математичної моделі ДПС послідовного збудження, буде такою:

$$\frac{di_a}{dt} = \frac{\left(U_a - K_e \Phi \omega_M - i_k R_k\right)/R_{ak} - i_a}{T_{ak}}; \tag{4.5}$$

$$\frac{di_\mu}{dt} = \frac{i_a}{sT_k(\Phi)+1}; \tag{4.6}$$

$$L_\mu = \frac{W_\mu A_\Phi B_\Phi}{1+(B_\Phi \cdot (i_\mu - i_a K_{ar}))^2}; \tag{4.7}$$

$$\Phi = A_\Phi \cdot \arctan(B_\Phi \cdot (i_\mu - i_a K_{ar})); \tag{4.8}$$

$$\frac{d\omega_M}{dt} = \frac{K_e \Phi i_a - M_c}{J_M}. \tag{4.9}$$

Для отримання вищої точності моделі двигуна послідовного збудження застосовано наближення її кривої намагнічування функцією арктангенсу $\Phi = A_\Phi \cdot \arctan(B_\Phi \cdot i_\mu)$, де A_Φ, B_Φ – коефіцієнти наближення (апроксимації). Також у виразі (4.8) враховано розмагнічувальну дію реакції якоря.

Вираз (4.7) враховує зміну індуктивності намагнічування L_μ зі зміною величини потоку Ф, що використовується для розрахунку сталої часу T_k. Як відомо, значення індуктивності обмотки збудження може бути знайдене з виразу $L_d = 2 p_n W_d \frac{d\Phi}{di_d}$, де p_n – кількість пар полюсів; W_d – кількість витків на полюс; i_d – струм збудження. Апроксимація арктангенсом (див. розд. 2.2) дає змогу отримати аналітичне значення похідної потоку за струмом намагнічування: $\frac{d\Phi}{di_\mu} = \frac{A_\Phi B_\Phi}{1 + (B_\Phi i_\mu)^2}$, після підстановки якого у вираз для індуктивності та відповідних спрощень і позначення $W_\mu = 2 p_n W_d$ – допоміжного коефіцієнта для обчислення значення індуктивності намагнічування, одержуємо (4.7).

Враховуючи вищесказане і врахувавши реакцію якоря, позначивши $i_\Phi = i_\mu - i_a K_{ar}$, отримуємо систему алгебричних і диференціальних рівнянь, що описують модель двигуна постійного струму послідовного збудження:

$$i_\Phi = i_\mu - i_a K_{ar};$$

$$\frac{di_a}{dt} = \frac{\left(U_a - K_e \Phi \omega_M - (i_a - i_\mu) R_k\right) / R_{ak} - i_a}{T_{ak}}; \tag{4.10}$$

$$\frac{di_\mu}{dt} = \frac{i_a}{s T_k (i_\Phi) + 1}; \tag{4.11}$$

$$L_\mu = \frac{W_\mu A_\Phi B_\Phi}{1 + (B_\Phi \cdot (i_\mu - i_a K_{ar}))^2}; \tag{4.12}$$

$$\Phi = A_\Phi \cdot \arctan(B_\Phi i_\Phi); \tag{4.13}$$

$$\frac{d\omega_M}{dt} = \frac{K_e \Phi i_a - M_c}{J_M}. \tag{4.14}$$

Цій моделі відповідатиме така система алгебричних та різницевих рівнянь, що одержані за допомогою Z-перетворення (із застосуванням раніше виведених моделювальних рівнянь для інтегральної та аперіодичної ланок):

$$i_{\Phi_i} = i_{\mu_i} - i_{a_i} K_{ar};$$

$$\Phi = A_\Phi \cdot \arctan(B_\Phi i_{\Phi i});$$

$$L_\mu = \frac{W_\mu A_\Phi B_\Phi}{1 + (B_\Phi i_{\Phi_i})^2};$$

$$i_{ai+1} = i_{a_i} e^{-\frac{h}{T_a}} + (1 - e^{-\frac{h}{T_a}}) \frac{U_a - K_e \Phi \omega_i - (i_{a_i} - i_{\mu i}) R_k}{R_a};$$

$$i_{\mu i+1} = i_{\mu_i} e^{-\frac{h}{T_k}} + (1 - e^{-\frac{h}{T_k}}) i_a;$$

$$\omega_{i+1} = \omega_i + \frac{h}{J_M}(i_{a_i} K_e \Phi - M_c - K_f \operatorname{sign}(\omega_i)).$$

Для повного відтворення в моделі реальних умов та режимів зчеплення коліс з дорожнім покриттям механічну частину транспортного засобу (у нашому випадку тролейбуса) необхідно подати уточненою моделлю.

Складаючи математичну модель механічної частини привода, приймаємо такі припущення:

1) люфти в редукторі та муфтах відсутні;
2) окремі елементи механічної частини є матеріальними точками;
3) зв'язки між елементами є безінерційними і характеризуються лише сталими пружностями та дисипативними силами;
4) зовнішні сили та моменти прикладають лише до зосереджених мас.

На рис. 4.23 показано структурну схему математичної моделі механічної частини тролейбуса [Д.12]. Вона прийнята як двомасова система з пружним кінематичним зв'язком і такими вузлами зосередження мас:

- момент інерції двигуна і трансмісії J_M;
- момент інерції колеса $J_к$;
- маса тролейбуса $m_{тр}$, що зведена до колеса.

Тут використано позначення: M_M, J_M, ω_M – момент, момент інерції та кутова швидкість двигуна; $M_{пр}$, $C_{пр}$, $J_к$, $\omega_к$, $D_к$ – пружний момент на обводі, пружність, момент інерції, кутова швидкість та діаметр колеса; $M_{тр}$, M_c, $m_{тр}$, $V_{тр}$ – крутний момент, момент статичного опору рухові, маса та лінійна швидкість тролейбуса; K_f – коефіцієнт дисипативних сил.

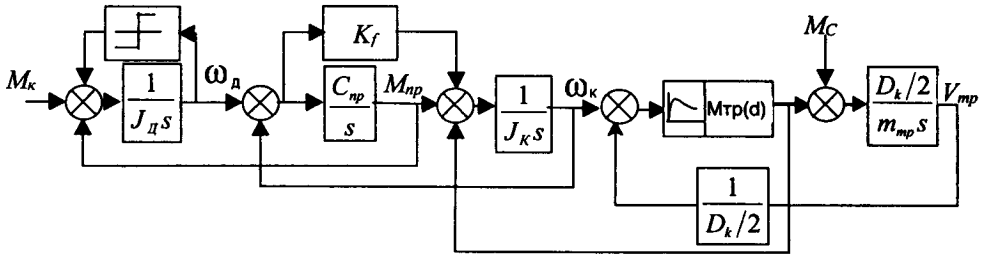


Рис. 4.23. Структурна схема моделі механічної частини привода

Навантаженням тягового електропривода тролейбуса в робочих та аномальних режимах (буксування та юзу) є пара тертя з проковзуванням колеса відносно дорожнього покриття. Залежність рушійного моменту $M_{тр}$ від пробуксовування визначається відомою кривою пробуксовування $\Psi(d)$ [*Ульянов Н.А. Колесные движители строительных и дорожных машин: Теория и расчет. – М.: Машиностроение, 1982. – 279 с.*], яка є залежністю відносного значення крутної сили або моменту $\Psi(d) = M_{тр}/G_{тр}$ від пробуксовування d (криві 1, 2, 3 на рис. 4.24).

Для використання в моделі ці криві з достатньою для практичних розрахунків точністю наближають такою аналітичною залежністю:

$$M_{тр} = f(d)G_{тр} = \frac{A \cdot \arctan(B \cdot d)}{1 + \arctan(C \cdot d)} \cdot G_{тр},$$

де A, B, C – коефіцієнти апроксимації.

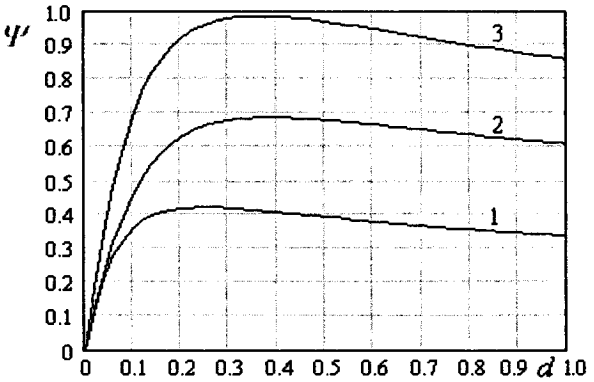


Рис. 4.24. Характеристики пробуксовування колісного рушія

Враховуючи це, математичну модель механічної частини тролейбуса подають такими рівняннями:

$$\frac{d\omega_M}{dt} = \frac{M_M - M_{np} - M_c \cdot \text{sign}(\omega_M)}{J_M}; \tag{4.15}$$

$$\frac{dM_{np}}{dt} = C_{np}(\omega_M - \omega_к); \tag{4.16}$$

$$\frac{d\omega_к}{dt} = \frac{M_{np} - M_{mp} - K_f(\omega_M - \omega_к)}{J_M + J_k}; \tag{4.17}$$

$$d = (\omega_k - 2V_{mp} / D_к)/\omega_к; \tag{4.18}$$

$$M_{mp} = f(d)G_{mp} = \frac{A \cdot \arctan(B \cdot d)}{1 + \arctan(C \cdot d)} \cdot G_{mp}; \tag{4.19}$$

$$\frac{dV_{mp}}{dt} = \frac{2}{D_k \cdot m_{mp}}(M_{mp} - M_c). \tag{4.20}$$

Цій моделі механічної частини відповідатиме така система алгебричних та різницевих рівнянь, що отримані за допомогою Z-перетворення (із застосуванням раніше виведених моделювальних рівнянь для інтегральної та аперіодичної ланок):

$$M_{np} = C_{12}\varphi_i;$$

$$\omega_{ki+1} = \omega_{ki} + \frac{h}{J_M + J_k} \cdot (M_{Mi} - M_{npi} - K_f \cdot (\omega_{Mi} - \omega_{ki}));$$

$$\varphi_{i+1} = \varphi_i + h \cdot (\omega_{Mi} - \omega_{ki});$$

$$d = (\omega_{ki} - 2V_{mp}/D_k)/\omega_{ki};$$

$$M_{mp} = \frac{A \cdot \arctan(B \cdot d)}{1 + \arctan(C \cdot d)} \cdot G_{mp};$$

$$V_{mpi+1} = V_{mpi} + \frac{2h}{D_k m_{mp}} \cdot (M_{mp} - M_C).$$

У разі використання від'ємного зворотного зв'язку за струмом якоря двигуна вихідний сигнал регулятора $U_{к_c}$ струму РС отримаємо за виразами:

$$U_{к_c} = \begin{cases} (U_{зс} - k_{зс} i_a) \cdot k_{рс} & для \quad U_{к_c} < U_{к_{c.max}}; \\ U_{к_{c.max}} & для \quad U_{к_c} > U_{к_{c.max}}. \end{cases} \tag{4.21}$$

де $U_{зс}$ – сигнал завдання регулятора струму РС, що пропорційний до максимального струму $I_{a\,max}$ якоря двигуна;

$k_{зс}$, $k_{рс}$ – коефіцієнт зворотного зв'язку за струмом якоря та коефіцієнт підсилення регулятора струму РС відповідно;

$U_{к_{c.max}}$ – сигнал обмеження виходу регулятора струму.

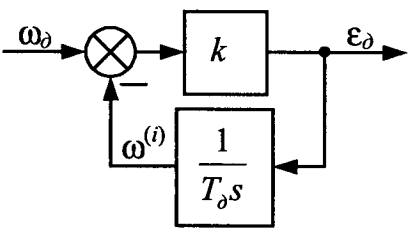


Рис. 4.25. Структурна схема реальної диференціювальної ланки

Для реалізації зворотного зв'язку за пришвидшенням необхідно виконувати операцію диференціювання. Оцінка поточного пришвидшення отримується за допомогою давача швидкості – тахогенератора BR та диференціювальної ланки *d/dt*. Зважаючи на наявність колекторних пульсацій у вихідному сигналі тахогенератора, диференціюван-

ня цього сигналу виконується реальною диференціювальною ланкою (4.17) (рис. 4.25):

$$W_{диф}(s) = \frac{T_\partial s}{sT_\partial / k + 1} \tag{4.22}$$

яка у моделі системи електропривода подається такими рівняннями:

$$\frac{d\omega^{(i)}}{dt} = \frac{\varepsilon_\partial}{T_\partial}; \tag{4.23}$$

$$\varepsilon_\partial = (\omega - \omega^{(i)}) \cdot k, \tag{4.24}$$

де T_∂ – стала часу диференціювання;

k – параметр реального диференціатора;

ε_∂, ω_i – пришвидшення двигуна та вихідна координата інтегрувальної ланки відповідно.

У разі переходу до різницевих рівнянь реальну диференціювальну ланку можна реалізувати простіше за відомою формулою скінченних різниць першого порядку:

$$\varepsilon_\partial = \frac{\omega_{i+1} - \omega_i}{h} \cdot T_\partial,$$

де ω_i, ω_{i+1} – значення швидкості в i-й та i+1-й моменти часу;

h – крок моделювання.

Вихідний сигнал регулятора пришвидшення формується у функції різниці сигналу максимально допустимого пришвидшення $U_{з.пр}$ та його поточного значення. У разі використання від'ємного нелінійного зворотного зв'язку за пришвидшенням двигуна вихідний $U_{к_{пр}}$ сигнал регулятора пришвидшення РП отримаємо за виразом:

$$U_{к_{пр}} = \begin{cases} (U_{з.пр} - \varepsilon_\partial \cdot k_{з_{пр}}) \cdot k_{р_{пр}} & для\ U_{к_{пр}} > 0; \\ 0 \quad для\ U_{к_{пр}} \le 0, \end{cases} \tag{4.25}$$

де $U_{з.пр}$ – сигнал завдання регулятора пришвидшення РП, що пропорційний до максимального допустимого пришвидшення $\varepsilon_{\partial.max}$ двигуна;

$k_{з_{пр}}, k_{р_{пр}}$ – коефіцієнт зворотного зв'язку за пришвидшенням та коефіцієнт підсилення регулятора пришвидшення РП відповідно.

Сигнал керування $U_к$, що надходить на вхід широтно-імпульсного перетворювача, формується за законом вибору меншого з двох вихідних сигналів керування $U_{к_с}$ та $U_{к_{пр}}$ регулятора струму та пришвидшення відповідно, що реалізується блоком вибору меншого БВМ.

Отримана система рівнянь (4.5)–(4.21) подає математичну модель системи тягового електропривода з двома паралельними зворотними зв'язками: від'ємним за струмом якоря двигуна та за пришвидшенням двигуна. Ця математична модель описується системою алгебричних та різницевих рівнянь (частина з них виведена раніше).

$$U_{к_с} = \begin{cases} (U_{зс} - k_{зс} i_a) \cdot k_{рс} & \text{для } U_{к_с} < U_{к_{с.\max}}; \\ U_{к_{с.\max}} & \text{для } U_{к_с} > U_{к_{с.\max}}. \end{cases}$$

$$U_{к_{пр}} = \begin{cases} (U_{з.пр} - \varepsilon_{\partial i} \cdot k_{з_{пр}}) \cdot k_{р_{пр}} & \text{для } U_{к_{пр}} > 0; \\ 0 \quad \text{для } U_{к_{пр}} \le 0, \end{cases}$$

$$U_п = k_п \cdot \min(U_{К_С}, U_{К_{ПР}}); \qquad i_{\Phi_i} = i_{\mu_i} - i_{a_i} K_{ar};$$

$$\Phi = A_\Phi \cdot \arctan(B_\Phi i_{\Phi i}); \qquad L_\mu = \frac{W_\mu A_\Phi B_\Phi}{1 + (B_\Phi i_{\Phi_i})^2};$$

$$i_{ai+1} = i_{a_i} e^{-\frac{h}{T_a}} + (1 - e^{-\frac{h}{T_a}}) \frac{U_a - K_e \Phi \omega_{Mi} - (i_{a_i} - i_{\mu i}) R_k}{R_a};$$

$$T_k = L_\mu / R_k; \qquad i_{\mu i+1} = i_{\mu_i} e^{-\frac{h}{T_k}} + (1 - e^{-\frac{h}{T_k}}) i_a;$$

$$\omega_{Mi+1} = \omega_{Mi} + \frac{h}{J_M} (i_{a_i} K_e \Phi - M_c - K_f \operatorname{sign}(\omega_{Mi})).$$

$$M_{пр} = C_{12} \varphi_i;$$

$$\omega_{ki+1} = \omega_{ki} + \frac{h}{J_M + J_k} \cdot (M_{Mi} - M_{npi} - K_f \cdot (\omega_{Mi} - \omega_{ki}));$$

$$\varphi_{i+1} = \varphi_i + h \cdot (\omega_{Mi} - \omega_{ki}); \qquad d = (\omega_{ki} - 2V_{mp}/D_k)/\omega_{ki};$$

$$M_{mp} = \frac{A \cdot \arctan(B \cdot d)}{1 + \arctan(C \cdot d)}; \qquad V_{mpi+1} = V_{mpi} + \frac{2h}{D_k m_{mp}} \cdot (M_{mp} - M_C).$$

Зміни координат електропривода тролейбуса з двома паралельними зворотними зв'язками у режимі розганяння, що отримані на складеній цифровій моделі тягового електропривода ШІП–ДПС ПЗ, складеної на підставі описаної вище математичної моделі, показано на рис. 4.26.

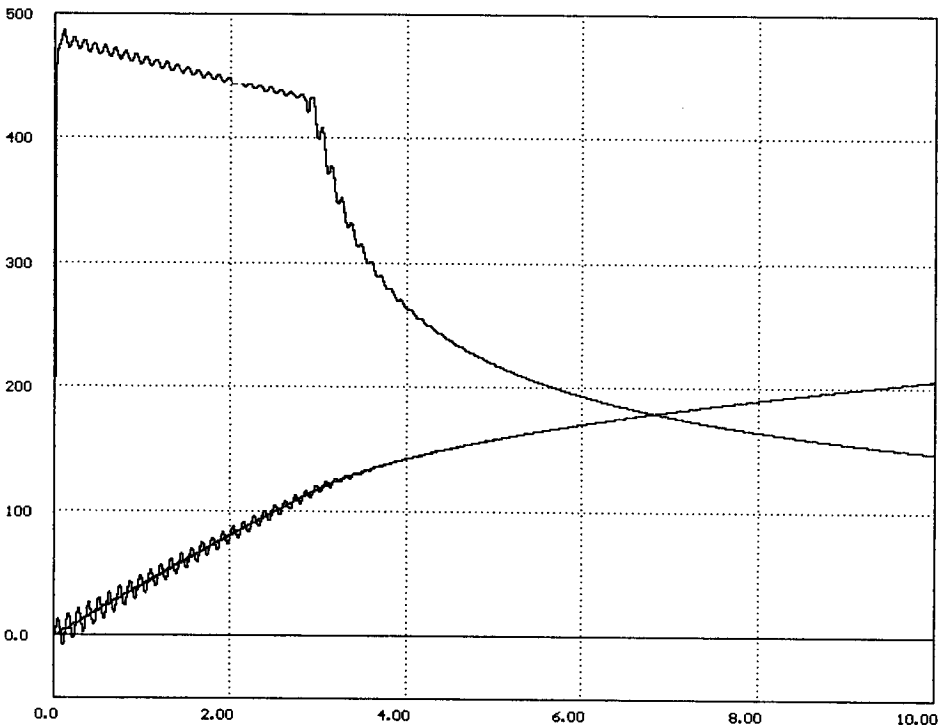


Рис. 4.26. Часові залежності координат (а) та статичні характеристики (б) системи електропривода ШІП-ДПС ПЗ з паралельними зворотними зв'язками за струмом якоря та пришвидшенням двигуна у режимі розганяння

У такій системі реалізується ефективне обмеження максимального струму якоря та пришвидшення електропривода, що виключає виникнення небажаних режимів пробуксовування за несприятливих умов зчеплення з дорожнім покриттям в автоматичному режимі.

```
{ ------------------------- }
{   Приклад до розд. 4.3    }
{ ------------------------- }
uses
   Crt, Plot;
const
   Tmax = 15.0;        { Максимальний час розрахунку }
   Ymin = 0.0;         { Нижня межа графіка          }
   Ymax = 500.0;       { Верхня межа графіка         }
   h = 0.001;          { Крок моделювання            }

   Uzc = 10.0;         { Напруга завдання струму             }
   Kpc = 9.486;        { Коефіцієнт підсилення рег-ра струму }
   Kzc = 0.0207;       { Коефіцієнт зв. зв'зку за струмом    }
   Uze = 10.0;         { Напруга завдання ЕРС                }
   Kpe = 1.403;        { Коефіцієнт підсилення рег-ра ЕРС    }
   Kze = 0.00522;      { Коефіцієнт зв. зв'зку за ЕРС}
   Kd = 55.0;          { Коефіцієнт підсилення ШІМ           }
   Ke = 85.944;        { Електромеханічна стала двигуна      }
   Kar = 0.14;         { Коефіцієнт реакції якоря            }
   Ra = 0.0634;        { Сумарний опір якірного кола         }
   Rd = 0.029;         { Опір серієсної ОЗД                  }
   La = 0.00552;       { Індуктивність якоря                 }
   Ls = 0.00063;       { Індуктивність розсіювання           }
   Af = 0.04589;       { Коефіцієнти апроксимації            }
   Bf = 0.00619;       { кривої намагнічування               }
   Wm = 60.0;          { Коеф. для обчис.інд-сті ОЗД         }
   Rk = 2.0;           { Опір фіктивного контуру вихр. струмів }
   J1 = 7.0;           { Сумарний момент інерції двигун+колесо }
   Mc = 875.4;         { Статичний момент опору              }
   C12 = 8.1e3;        { Коефіцієнт жорсткості механізму}
   J2 = 46.55;         { Зведений момент інерції тролейбуса  }
var
   t, Ua, Ia, Im, w1 : double;
   I_f, Ce, w2, fi, Tk : double;
   Lm, F, Ta, M12 : double;
```

```
   Up, Upe, Upc : double;
begin
   Ia := 0.0;              { "Обнулювання" змінних:}
   Im := 0.0;              { Струм намагнічування  }
   w1 := 0.0;              { швидкість двигуна     }
   wk := 0.0;              { швидкість механізму   }
   fi := 0.0;              { кут скручування валу  }
   M12 := 0.0;             { пружний момент        }
   t := 0.0;
   Ta := (La+Ls)/(Ra+Rd);  { Стала часу якоря }
   { ======================================== }
   {         Підготовка до виводу графіка     }
   { ======================================== }
    InitGraphic(0, Tmax, Ymin, Ymax, 1.0, 20.0);
   { -------   Основний цикл програми   ------- }
   repeat
      { Використовується підпрограма виводу графіка              }
      Graphic(12, t, Ia); { Вивід графіка струму якоря         }
      Graphic(9, t, w1);  { Вивід графіка швидкості двигуна    }
      Graphic(1, t, w2);  { Вивід графіка швидкості механізму }

      { Розрахунок допоміжних величин }
      I_f := Im - Kar*Ia;
      { Розрахунок потоку }
      F := Af*ArcTan(Bf*I_f);
      { Розрахунок індуктивності намагнічування }
      Lm := Wm*Af*Bf/(1.0 + sqr(Bf*I_f));
      Tk := Lm/Rk;    { Стала часу вихрових струмів }

      { Регулятор ЕРС }
      Upe := (Uze - w1*Ke*F*Kze)*Kpe;
      { Обмеження виходу регулятора }
      if Upe > 10.0 then Upe := 10.0;
      if Upe < -10.0 then Upe := -10.0;

      { Регулятор струму }
      Upc := (Uzc - Ia*Kzc)*Kpc;
      { Обмеження виходу регулятора }
      if Upc > 10.0 then Upc := 10.0;
      if Upc < -10.0 then Upc := -10.0;

      { Вибір мінімальної напруги }
      if abs(Upc) < abs(Upe) then
```

```
        Up := Upc
      else
        Up := Upe;

      { Розрахунок якірного струму }
 Ia := Ia*exp(-h/Ta)+(1-exp(-h/Ta))*(Up*Kd-Ke*F*w1-Rk*(Ia-Im))/Ra;
      { Розрахунок струму намагнічування }
      Im := Im*exp(-h/Tk) + (1-exp(-h/Tk))*Ia;

      { Розрахунок швидкості двигуна }
      w1 := w1 + h *(Ke*F*Ia - M12)/J1;
      { Розрахунок швидкості механізму }
      w2 := w2 + h *(M12 - Mc)/J2;

      { Розрахунок кута скручування валу }
      fi := fi + h*(w1 - w2);
      { Пружний момент }
      M12 := C12*fi;

      t := t + h;
   until t > Tmax;
   repeat until KeyPressed;
end.
```

СПИСОК ЛІТЕРАТУРИ

А. ПРОГРАМНІ ЗАСОБИ

BASIC

1. Башмакова Е.С., Виттенберг И.М., Либеров А.Б., Пашков А.Л. Программирование микроЭВМ на языке Бейсик. – М.: Радио и связь, 1991. – 240 с.
2. Введение в программирование на языке Microsoft BASIC: Учеб. пособие / Ю.Я. Максимов, С.В. Осипов, А.В. Потемкин и др. – М.: ДИАЛОГ-МИФИ, 1991. – 176 с.
3. Височанський В.С. та ін. Елементи інформатики / В.С. Височанський, А.І. Кардаш, О.В. Костів, В.В. Черняхівський; За ред. А.І. Кардаша: Довідник. – Львів: Світ, 1990. – 192 с.
4. Вчіться спілкуватися з персональним комп'ютером: Посібник для вчителя / І.Ф. Следзінський, А.М. Ломакович, В.Г. Габрусєв та ін. – К.: Рад. шк., 1990. – 143 с.
5. Дьяконов В.П. Справочник по алгоритмам и программам на языке БЕЙСИК для персональных ЭВМ: Справочник. – М.: Наука, 1987. – 240 с.
6. Жалдак М.І., Рамський Ю.С. Інформатика: Навч. посібник / За ред. М.І. Шкіля. – К.: Вища шк., 1991. – 319 с.
7. Зельднер Г.А. Microsoft BASIC Professional Development System 7.1. Руководство программиста. – М.: ABF, 1994. – 400 с.
8. Кетков Ю.Л. GW-, Turbo- и Quick Basic для IBM PC. – М.: Финансы и статистика, 1992. – 240 с.
9. Лозинський А., Мороз В., Паранчук Р., Паранчук Я. Microsoft QBasic: програмування та розрахунок динаміки електроприводів: Навч. посібник. – Львів: Вид-во Нац. ун-ту "Львівська політехніка", 2001. – 274 с.

10. Основы информатики и вычислительной техники: Проб. учеб. пособие для сред. учеб. заведений: В 2 ч. Ч. 1 / А.П. Ершов, В.М. Монахов, А.А. Кузнецов и др.; Под ред. А.П. Ершова и В.М. Монахова. – К.: Рад. шк., 1985. – 95 с.
11. Основы информатики и вычислительной техники: Проб. учеб. пособие для сред. учеб. заведений: В 2 ч. Ч. 2 / А.П. Ершов, В.М. Монахов, А.А. Кузнецов и др. Под ред. А.П. Ершова и В.М. Монахова. – К.: Рад. шк., 1986. – 143 с.
12. Очков В.Ф., Пухначев Ю.В. 128 советов начинающему программисту. – М.: Энергоатомиздат, 1991. – 256 с.
13. Очков В., Рахаев М. Этюды на языках QBasic, Quick Basic и Basic Compiler. – М.: Финансы и статистика, 1995.
14. Паранчук Я.С., Глинський Я.М., Мороз В.І. Бейсик: програмування в середовищах QBasic, Turbo Basic, Power Basic: Навч. посібник. – К.: ІЗМН, 1998. – 300 с.
15. Радер Дж., Миллсап К. Бейсик для персонального компьютера фирмы IBM: Пер. с англ. – М.: Радио и связь, 1991. – 412 с.
16. Райманс Г.-Г. Qbasic: Основы языка программирования (Вводный курс) / Пер. с нем. – К.: Торгово-издательское бюро BHV, 1992. – 171с.
17. Светозарова Г.И., Мельников А.А., Козловский А.В. Практикум по программированию на языке Бейсик: Учебн. пособие для вузов. – М.: Наука, 1988. – 368 с.
18. Сибирцев В.Г., Ткачев И.И. Бейсик для персональных ЭВМ: Справ. пособие. – К.: СП "ГУТ", 1992. – 320 с.
19. Трэктон К. Программирование на Бейсике для инженерно-технических расчетов: Пер. с англ. – М.: Радио и связь, 1985. – 96 с.
20. Уолш Б. Программирование на Бейсике: Пер. с англ. – М.: Радио и связь, 1988. – 335 с.
21. Теннант-Смит Дж. Бейсик для статистиков: Пер. с англ. – М.: Мир, 1988. – 208 с.
22. Фокс Д., Фокс А. Бейсик для всех. – М.: Энергоатомиздат, 1986. – 176 с.

Turbo Pascal

1. Абрамов В.Г., Трифонов Н.П., Трифонова Г.Н. Введение в язык Паскаль: Учеб. пособие. – М.: Наука, 1988.

2. Абрамов С.А., Зима В.С. Начала программирования на языке ПАСКАЛЬ. – М.: Наука, 1987.
3. Абрамов С.А., Зима Е.В. Начала информатики. – М.: Наука, 1989.
4. Белецкий Я. Турбо Паскаль с графикой для персональных компьютеров. – М., 1991.
5. Боон К. ПАСКАЛЬ для всех. – М.: Энергоатомиздат, 1988.
6. Бородич Ю.С., Вальвачев А.Н., Кузьмич А.И. Паскаль для персональных компьютеров: Справ. пособие. – Минск: Вышэйш. шк., 1991. – 365 с.
7. Бутомо И.Д., Самочадин А.В., Усанова Д.В. Программирование на алгоритмическом языке Паскаль для микроЭВМ: Учеб. пособие. – Л.: Изд-во Ленингр. ун-та, 1985.
8. Васюкова Н.Д., Тюляева В.В. Практикум по основам программирования. Язык Паскаль : Учеб. пособие для учащихся сред. спец. учеб. заведений. – М.: Высшая школа, 1991.
9. Вирт Н. Язык программирования ПАСКАЛЬ / Алгоритмы и организация решения экономических задач. – М.: Статистика, 1974.
10. Глинський Я.М. Основи інформатики та обчислювальної техніки: У 4 ч. Ч. IV: Паскаль: Експеримент. підручник для 11 класу середн. шк. – Львів: СП "БаК", 1996. – 96 с.
11. Грогоно П. Программирование на языке Паскаль / Пер. с англ. – М.: Мир, 1982.
12. Грэхем Р. Практический курс языка Паскаль для микро-ЭВМ. – М.: Радио и связь, 1986.
13. Джонс Ж., Харроу К. Решение задач в системе Турбо Паскаль / Пер. с англ. – М.: Финансы и статистика, 1991.
14. Довгаль С.И., Литвинов Б.Ю., Сбитнев А.И. Персональные ЭВМ: ТурбоПаскаль V 6.0, Объектное программирование, Локальные сети: Учеб. пособие. – К.: Информсистема сервис, 1993.
15. Довгаль С.И., Сбитнев А.И. Персональные ЭВМ: ТурбоПаскаль V 7.0, Объектное программирование: Учеб. пособие. – К.: Информсистема сервис, 1995.
16. Епанешников А., Епанешников В. Программирование в среде Turbo Pascal 7.0. – М.: Диалог-МИФИ, 1993.
17. Зуев Е.А. Система программирования Turbo Pascal. – М.: Радио и связь, 1991.

18. Зуев Е.А. Язык программирования Turbo Pascal 6.0. – М.: Унитех, 1992. – 298 с.
19. Зуев Е.А. Программирование на языке Turbo Pascal 6.0, 7.0. – М.: Унитех, 1993.
20. Йенсен К., Вирт Н. Паскаль. Руководство для пользователя и описание языка / Пер. с англ. – М.: Финансы и статистика,1982.
21. Климов Ю.С., Касаткин А.И., Мороз С.М. Программирование в среде Turbo Pascal 6.0: Справ. пособие. – Минск: Вышэйш. шк., 1992.
22. Марченко А.И., Марченко Л.А. Программирование в среде Turbo Pascal 7.0. – М.: Бином Универсал, К.: ЮНИОР, 1997. – 496 с.
23. Мизрохи С.В. Turbo Pascal и объектно-ориентированное программирование. – М.: Финансы и статистика, 1992.
24. Новичков В.С. и др. Паскаль: Учеб. пособие для сред. спец. учеб. заведений. – М.: Высш. шк., 1990.
25. Перминов О.Н. Программирование на языке Паскаль. – М.: Радио и связь, 1988.
26. Перминов О.Н. Язык программирования Паскаль. – М.: Радио и связь, 1983.
27. Поляков Д.Б., Круглов Ю.И. Программирование в среде Турбо Паскаль (версия 5.5): Справ.-метод. пособие. – М.: Изд-во МАИ, 1992. – 576 с.
28. Прайс Д. Программирование на языке Паскаль / Пер. с англ. – М.: Мир, 1987.
29. Рюттен Т., Франкен Г. Турбо Паскаль 6.0. – К.: Торгово-издательское бюро BHV, 1992.
30. Семашко Г.Л., Салтыков А.И. Программирование на языке Паскаль. – М.: Наука, 1988.
31. Сердюченко В.Я. Розробка алгоритмів та програмування на мові Turbo Pascal: Навч. посібник для техн. вузів. – Харків, 1995.
32. Уилсон И., Эддиман А. Практическое введение в ПАСКАЛЬ / Пер. с англ. под ред. Л.Д.Райкова. – М.: Радио и связь, 1983.
33. Фаронов В.В. Программирование на персональных ЭВМ в среде ТУРБО-ПАСКАЛЬ. – М.: Изд-во МГТУ, 1990.
34. Фаронов В.В. Турбо Паскаль 7.0. Начальный курс: Учеб. пособие. – М.: Нолидж, 1997. – 616 с.

35. Фаронов В.В. Турбо Паскаль 7.0. Практика программирования: Учеб. пособие. – М.: Нолидж, 1997. – 432 с.
36. Форсайт Р. Паскаль для всех / Пер. с англ. под ред. Ю.И. Топчеева. – М.: Машиностроение, 1986.
37. Шаньгин В.Ф., Поддубная Л.М. Программирование на языке ПАСКАЛЬ: Учеб. пособие для ПТУ / Под ред. В.Ф. Шаньгина. – 2-е изд., перераб. и доп: Кн. 7: Программное обеспечение микроЭВМ: В 11 кн. – М.: Высшая школа, 1991.
38. Эрбс Х.-Э., Штольц О. Введение в программирование на языке Паскаль. – М.: Мир, 1989.
39. Borland Pascal with Objects. Version 7.0. User's Guide / Borland International, Inc., 1992.
40. Borland Pascal with Objects. Version 7.0. Language Guide / Borland International, Inc., 1992.
41. Borland Pascal with Objects. Version 7.0. Programmer's Guide / Borland International, Inc., 1992.

MathCAD

1. Аладьев В.З., Гершгорн Н.А. Вычислительные задачи на персональном компьютере. – К.: Техніка, 1991. – 245 с.
2. Використання пакета MathCAD для дослідження електромеханічних систем автоматичного керування: Навч. посібник / А.В. Маляр, В.І. Мороз, І.З. Щур. – Львів: Вид-во Нац. ун-ту "Львівська політехніка", 2003. – 72 с.
3. Дьяконов В. Система MathCAD: Справочник. – М.: Радио и связь, 1993. – 128 с.
4. Дьяконов В. Справочник по MathCAD Plus 6 Pro. – М.: СК Пресс, 1997. – 336 с.
5. Дьяконов В. Справочник по MathCAD Plus 7.0 Pro. – М.: СК Пресс, 1998. – 352 с.
6. Дьяконов В. MATHCAD 8/2000: Спец. справочник. – СПб: Питер, 2001. – 592 с.
7. Дьяконов В.П., Абраменкова И.В. MathCAD 7 PRO в математике, физике и в Internet. – М.: Нолидж, 1999. – 352 с.

8. Дьяконов В.П., Абраменкова И.В. Mathcad 8 в математике, в физике и в Internet. – М.: Нолидж, 1999. – 512 с.
9. Лозинський А., Мороз В., Паранчук Я. Розв'язування задач електромеханіки в середовищах пакетів MathCAD і MATLAB: Навч. посібник. – Львів: Вид-во Держ. ун-ту "Львівська політехніка", 2000. – 166 с.
10. MathCad 6.0 Plus. Финансовые, инженерные и научные расчеты в среде Windows 95: Пер. с англ. – М.: Филинъ, 1996. – 712 с.
11. Очков В.Ф. MathCad PLUS 6.0 для студентов и инженеров. – М.: ТОО фирма "КомпьютерПресс", 1996. – 238 с.
12. Очков В.Ф. MathCad 7 Pro для студентов и инженеров. – М.: ТОО фирма "КомпьютерПресс", 1998. – 384 с.
13. Очков В.Ф. MathCAD 8 Pro для студентов и инженеров. – М.: "КомпьютерПресс", 1999. – 523 с.
14. Плис А.И., Сливина Н.А. MathCAD: математический практикум. – М: Финансы и Статистика. – 1999.

MATLAB

1. Герман-Галкин С.Г. Компьютерное моделирование полупроводниковых систем в MATLAB 6.0: Учеб. пособие. – СПб.: КОРОНА принт, 2001. – 320 с.
2. Герман-Галкин С.Г. Силовая электроника: Лабораторные работы на ПК. – СПб.: КОРОНА принт, 2002. – 304 с.
3. Гультяев А.К. MATLAB 5.2. Имитационное моделирование в среде Windows: Практ. пособие. – СПб.: КОРОНА принт, 1999. – 288 с.
4. Дьяконов В.П., Абраменкова И.В. Математическая система MATLAB 5.0/5.3. – М.: Нолидж, 1999. – 640 с.
5. Лозинський А., Мороз В., Паранчук Я. Розв'язування задач електромеханіки в середовищах пакетів MathCAD і MATLAB: Навчальний посібник. – Львів: Вид-во Держ. ун-ту "Львівська політехніка", 2000. – 166 с.
6. Потемкин В.Г. MATLAB 5 для студентов. Справ. пособие. – М.: Диалог-МИФИ, 1998. – 314 с.
7. Потемкин В.Г. Система MATLAB: Справ. пособие. М: Диалог-МИФИ, 1997. – 350 с.

Методики програмування

1. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на С++. – 2-е изд. / Пер. с англ. – М.: Изд-во Бином, СПб: Невский диалект, 1998. – 560 с.
2. Ван Тассел Д. Стиль, разработка, эффективность, отладка и испытание программ: Пер. с англ. – М.: Мир, 1981. – 320 с.
3. Зиглер К. Методы проектирования программных систем: Пер. с англ. – М.: Мир, 1985. – 328 с.
4. Йодан Э. Структурное проектирование и конструирование программ: Пер. с англ. – М.: Мир, 1979. – 416 с.
5. Хьюз Дж., Мичтом Дж. Структурный подход к программированию: Пер. с англ. – М.: Мир, 1980. – 278 с.

Б. Загальна

1. Ажогин В.В., Згуровский М.З. Моделирование на цифровых, аналоговых и гибридных ЭВМ. – К., Вища школа, 1983. – 280 с.
2. Башарин А.В., Постников Ю.В. Примеры расчета автоматизированного электропривода на ЭВМ: Учеб. пособие для вузов. – 3-е изд. – Л.: Энергоатомиздат, 1990.
3. Белова Д.А., Кузин Р.Е. Применение ЭВМ для анализа и синтеза автоматических систем управления. – М.: Энергия, 1979. – 264 с.
4. Бронштейн И.Н., Семендяев К.А. Справочник по математике для инженеров и учащихся втузов. – М., Наука, 1981. – 720 с.
5. Герман-Галкин С.Г. Компьютерное моделирование полупроводниковых систем в MATLAB 6.0: Учеб. пособие. – СПб.: КОРОНА принт, 2001. – 320 с.
6. Герман-Галкин С.Г. Силовая электроника: Лабораторные работы на ПК. – СПб.: КОРОНА принт, 2002. – 304 с.
7. Егоров В.Н., Корженевский-Яковлев О.В. Цифровое моделирование систем электропривода. – Л.: Энергоатомиздат, 1986. – 168 с.
8. Зеленов А.Б., Шевченко І.С., Андрєєва Н.І. Синтез та цифрове моделювання систем управління електроприводів постійного струму з вентиль-

ними перетворювачами: Навч. посіб. для студ. вузів. – Алчевськ: ДГМІ, 2002. – 400 с.
9. Кубрак А.І., Ярощук Л.Д. Програмування та розрахунок автоматичних систем. – К.: Вища шк., 1992. – 366 с.
10. Лозинський А., Мороз В., Паранчук Р., Паранчук Я. Microsoft QBasic: програмування та розрахунок динаміки електроприводів: Навч. посібник. – Львів: Вид-во Нац. ун-ту "Львівська політехніка", 2001. – 274 с.
11. Макаров И.М., Менский Б.М. Линейные автоматические системы (элементы терии, методы расчета и справочный материал). – М.: Машиностроение, 1982. – 564 с.
12. Моделювання електромеханічних систем: Підручник / О.П. Чорний, А.П. Луговой, Д.Й. Родькін, Г.Ю. Сисюк, О.В. Садовой. – Кременчук, 2001. – 376 с.

В. Числові методи

1. Ажогин В.В., Згуровский М.З. Моделирование на цифровых, аналоговых и гибридных ЭВМ. – К.: Вища шк., 1983. – 280 с.
2. Бабушка И., Витасек Э., Прагер М. Численные процессы решения дифференциальных уравнений: Пер. с англ. – М.: Мир, 1969.
3. Дёч Г. Руководство к практическому применению преобразования Лапласа и Z-преобразования / Пер. с нем. – М.: Наука, 1971. – 288 с.
4. Егоров В.Н., Корженевский-Яковлев О.В. Цифровое моделирование систем электропривода. – Л.: Энергоатомиздат, 1986. – 168 с.
5. Мак-Кракен Д., Дорн У. Численные методы и программирование на ФОРТРАНе / Пер. с англ. – 2-е изд., стереотип. – М.: Мир, 1977. – 584 с.
6. Марчук Г.И. Методы вычислительной математики. – М.: Наука, 1989. – 608 с.
7. Мэтьюз Дж., Финк К. Численные методы. Использование MATLAB. – 3-е изд. / Пер. с англ. – М.: Издательский дом "Вильямс", 2001. – 720 с.
8. Смит Дж. М. Математическое и цифровое моделирование для инженеров и исследователей: Пер. с англ. – М.: Машиностроение, 1980. – 271 с.
9. Современные численные методы решения обыкновенных дифференциальных уравнений / Под ред. Дж. Холла и Дж. Уатта: Пер. с англ. – М.: Мир, 1979.
10. Форсайт Дж., Малькольм М., Моулер К. Машинные методы математических вычислений: Пер. с англ. – М.: Мир, 1980. – 279 с.

11. Хайрер Э., Нерсетт С., Ваннер Г. Решение обыкновенных дифференциальных уравнений. Нежесткие задачи: Пер. с англ. – М.: Мир, 1990.
12. Хемминг Р.В. Численные методы (для научных работников и инженеров) / Пер. с англ. – М.: Наука, 1972. – 400 с.
13. Чуа Л.О., Лин Пен-Мин. Машинный анализ электронных схем: Алгоритмы и вычислительные процедуры: Пер. с англ. – М.: Энергия, 1980. – 640 с.

Г. Використання Z-перетворення

1. Ажогин В.В., Згуровский М.З. Моделирование на цифровых, аналоговых и гибридных ЭВМ. – К.: Вища шк., 1983. – 280 с.
2. Дёч Г. Руководство к практическому применению преобразования Лапласа и Z-преобразования / Пер. с нем. – М.: Наука, 1971. – 288 с.
3. Джури Э. Импульсные системы автоматического регулирования. – М.: Физматгиз, 1963. – 456 с.
4. Егоров В.Н., Корженевский-Яковлев О.В. Цифровое моделирование систем электропривода. – Л.: Энергоатомиздат, 1986. – 168 с.
5. Макаров И.М., Менский Б.М. Таблицы обратных преобразований Лапласа и обратных z-преобразований: Дробно-рациональные изображения: Учеб. пособие для втузов. – М.: Высшая школа, 1978. – 247 с.
6. Смит Дж. М. Математическое и цифровое моделирование для инженеров и исследователей: Пер. с англ. – М.: Машиностроение, 1980. – 271 с.
7. Ту Ю. Цифровые и импульсные системы автоматического управления. – М.: Машиностроение, 1964.
8. Шипилло В.П. Исследование процессов в замкнутых вентильных системах методом Z-преобразования // Электричество. – 1969. – № 11. – С. 63–68.
9. Шипилло В.П. Операторно-рекуррентный анализ электрических цепей и систем. – М.: Энергоатомиздат, 1991. – 312 с.

Д. Моделі електроприводів та їх елементів

1. Автоматизированное проектирование силовых электронных схем / В.Я. Жуйков, В.Е.Сучик, П.Д.Андриенко, М.А.Еременко. – К.: Техніка, 1988. – 184 с.

2. Архангельский Б.И. Аналитическое выражение кривой намагничивания электрических машин // Электричество. – 1950. – № 3. – С. 30–32.
3. Башарин А.В., Новиков В.А., Соколовский Г.Г. Управление электроприводами: Учеб. пособие для вузов. – Л.: Энергоиздат, 1982. – 392 с.
4. Борцов Ю.А., Соколовский Г.Г. Автоматизированный электропривод с упругими связями. – 2-е изд., перераб. и доп. – СПб.: Энергоатомиздат, 1992.
5. Борцов Ю.А., Суворов Г.В. Методы исследования динамики сложных систем электропривода: Библиотека по автоматике. – Л.: Энергия, 1966. – Вып. 208.
6. Вешеневский С.Н. Характеристики двигателей в электроприводе. – 6-е изд., исправ. – М.: Энергия, 1977. – 432 с.
7. Герман-Галкин С.Г. Компьютерное моделирование полупроводниковых систем в MATLAB 6.0: Учеб. пособие. – СПб.: КОРОНА принт, 2001. – 320 с.
8. Герман-Галкин С.Г. Силовая электроника: Лабораторные работы на ПК. – СПб.: КОРОНА принт, 2002. – 304 с.
9. Дунаевский С.Я., Крылов О.А., Мазия Л.В. Моделирование элементов электромеханических систем. – 2-е изд. – М.: Энергия, 1971.
10. Егоров В.Н., Корженевский-Яковлев О.В. Цифровое моделирование систем электропривода. – Л.: Энергоатомиздат, 1986. – 168 с.
11. Зайцев Г.Ф., Стеклов В.К. Компенсация естественных нелинейностей автоматических систем. – М.: Энергоиздат, 1982. – 96 с.
12. Л. Карплюк, О. Лозинський, В. Мороз, Б. Панченко Дослідження режимів розгону тягового електроприводу тролейбуса // Вісн. Держ. ун-ту "Львівська політехніка". – 2000. – № 400. – С. 49–54.
13. Ключев В.И. Теория электропривода: Учеб. для вузов. – М.: Энергоатомиздат, 1985. – 560 с.
14. Маляр В.С., Фильц Р.В. Аппроксимация характеристик намагничивания сплайнами // Изв. вузов. Энергетика. – 1977. – № 11. – С. 119–121.
15. Мороз В., Карплюк Л. Уточнення моделі двигуна постійного струму послідовного збудження // Вісн. Держ. ун-ту "Львівська політехніка". – 1998. – № 347. – С. 118–123.
16. Піцан Р., Бардачевський В., Бойчук Б. Збірник задач до курсу "Електропривід". Ч. 1: Розімкнені системи електропривода: Навч. посібник. – Львів: Вид-во Держ. ун-ту "Львівська політехніка". – 1999. – 426 с.

17. Плахтина О.Г., Мороз В.І. Уточнення моделі генератора постійного струму при моделюванні електроприводів // Вісн. Держ. ун-ту "Львівська політехніка". – 1995. – № 288. – С. 80–83.
18. Сен П. Тиристорные электроприводы постоянного тока: Пер. с англ. – М.: Энергоатомиздат, 1985. – 232 с.
19. Сипайлов Г.А., Лоос А.В. Математическое моделирование электрических машин (АВМ): Учеб. пособие для студентов вузов. – М.: Высшая шк., 1980. – 176 с.
20. Слежановский О.В. Об учете и компенсации влияния вихревых токов в системах управления потоком возбуждения электрических машин // Электричество. – 1962. – № 9. – С. 23–27.
21. Теорія електропривода: Підручник / За ред. М.Г. Поповича. – К.: Вища шк., 1993. – 494 с.
22. Ткачук В. Електромеханотроніка: Навч. посібник. – Львів, Вид-во Нац. ун-ту "Львівська політехніка". – 2001. – 404 с.
23. Управляемый выпрямитель в системах автоматического управления / Н.В. Донской, А.Г. Иванов, В.М. Никитин, А.Д. Поздеев; Под ред. А.Д. Поздеева. – М.: Энергоатомиздат, 1984. – 352 с.
24. Фёдоров Н.Н. Основы электродинамики: Учеб. пособие для вузов. – М.: Высшая шк., 1980. – 399 с.
25. Фрер Ф., Орттенбургер Ф. Основные звенья регулируемого привода постоянного тока / Пер. с нем. Б.М. Лакса. – М.: Энергия, 1977.
26. Частотно-керовані асинхронні та синхронні електроприводи: Навч. посібник / О.Г. Плахтина, С.С. Мазепа, А.С. Куцик. – Львів: Вид-во Нац. ун-ту "Львівська політехніка", 2002. – 228 с.
27. Чиликин М.Г., Ключев В.И., Сандлер А.С. Теория автоматизированного электропривода: Учеб. пособие для вузов. – М: Энергия, 1979. – 616 с.
28. Шенфельд Р., Хабигер Э. Автоматизированные электроприводы: Пер. с нем. – Л.: Энергоатомиздат, 1985. – 464 с.
29. Шипилло В.П. Исследование процессов в замкнутых вентильных системах методом Z-преобразования // Электричество. – 1969. – № 11. – С. 63–68.
30. Шипилло В.П. Операторно-рекуррентный анализ электрических цепей и систем. – М.: Энергоатомиздат, 1991. – 312 с.
31. Ягуп В.Г. Автоматизированный расчет тиристорных схем. – Харків: Вища шк. Вид-во при Харк. ун-ті, 1986. – 160 с.

ДОДАТКИ

Нижче наведено текст підпрограми для виведення на екран до 15 графіків одночасно, її можна використовувати як стандартну для моделювання автоматизованих електроприводів.

Quick Basic

```
SUB Graphic (N%, x, Y) STATIC
' ------------------------------------------------------
'   Підпрограма виведення графіків (до 15) на екран дисплея
'
' У головній програмі повинні бути задані:
'    Xmin - мінімальне значення за віссю  X
'    Xmax - максимальне значення за віссю  X
'    Ymin - мінімальне значення за віссю  Y
'    Ymax - максимальне значення за віссю  Y
'
' Вхідні змінні підпрограми:
'     N% - номер графіка ( 1...15 )
'     x  - координата точки  X
'     y  - координата точки  Y
' ------------------------------------------------------
DIM Xold(15), Yold(15)
DIM first%(15)
' ======================================================
'        Для  дисплея  VGA  ( 640 x 480 , 16 кольорів )
' ======================================================
CONST Xleft% = 55          ' Ліва   межа графіка
CONST Xright% = 635        ' Права  межа графіка
CONST Xgrid% = 104         ' Ширина розбиття за віссю  X
CONST Yupper% = 10         ' Верхня межа графіка
CONST Ylower% = 415        ' Нижня  межа графіка
CONST LocatX = 54          ' Положення цифрових позначок осі  X
' Задання фону графіка
```

```
CONST White = 0              ' White=0    - чорний фон
                             ' White=1    - білий фон
' ========================================================
'
'   Перевірка першого виклику за кожним графіком
'
IF first%(N%) = 0 THEN
'
'   Перевірка найпершого виклику підпрограми  Graphic
'
   IF FirstCALL% = 0 THEN
      FirstCALL% = 1
'
' Встановлення кольорового режиму дисплея  VGA  640 x 480
'
      SCREEN 12
      WIDTH 80, 60
      ' Якщо треба, то білий фон
      IF White = 1 THEN LINE (0, 0)-(639, 479), 15, BF
'
'   Визначення ширини графіка за  X  та  Y
'
      XD = (Xmax - Xmin) / 10!
      YD = (Ymax - Ymin) / 10!
'
'   Виведення цифрових значень  Y
'
      IF White = 1 THEN COLOR 15 ELSE COLOR 7
'
      FOR k% = 0 TO 10
         LOCATE 2 + 5 * k%, 1
         S = Ymax - YD * k%
         SELECT CASE ABS(S)
            CASE IS < .9995
               Form$ = "#.###"
            CASE .9995 TO 9.994999
               Form$ = "##.##"
            CASE 9.994999 TO 99.999
               Form$ = "###.#"
            CASE 99.999 TO 1000
               Form$ = "#####"
            CASE ELSE
               Form$ = "##.#^^^^"
```

```
         END SELECT
         PRINT USING Form$; S;
      NEXT k%
'
'    Виведення цифрових значень  X
'
      FOR k% = 0 TO 10
         LOCATE LocatX, 6 + 7 * k%
         S = Xmin + XD * k%
         SELECT CASE ABS(S)
            CASE IS < .9995
               Form$ = "#.###"
            CASE .9995 TO 9.994999
               Form$ = "##.##"
            CASE 9.994999 TO 99.999
               Form$ = "###.#"
            CASE 99.999 TO 1000
               Form$ = "#####"
            CASE ELSE
               Form$ = "##.#^^^^"
         END SELECT
         PRINT USING Form$; S;
      NEXT k%

      IF White = 1 THEN
         '  Білий колір фону
         VIEW (Xleft%, Yupper%)-(Xright%, Ylower%), 15
         GridColor = 0
      ELSE
         '  Чорний колір фону
         VIEW (Xleft%, Yupper%)-(Xright%, Ylower%)
         GridColor = 7
      END IF
      WINDOW (Xmin, Ymin)-(Xmax, Ymax)
'
'    Формування горизонтальної розбивки графіка
'
      FOR y = Ymin TO Ymax + YD / 10! STEP YD
         LINE (Xmin, y)-(Xmax, y), GridColor, , &H1111
      NEXT y
'
'    Формування вертикальної розбивки графіка
'
```

```
      FOR x = Xmin TO Xmax + XD / 10! STEP XD
         LINE (x, Ymin)-(x, Ymax), GridColor, , &H1111
      NEXT x

'  Якщо можливо, рисуємо осі
     ' Вісь  Y
     IF SGN(Ymin)<>SGN(Ymax) THEN LINE (Xmin,0)-(Xmax,0),GridColor
      ' Вісь  X
     IF SGN(Xmin)<>SGN(Xmax) THEN LINE (0,Ymin)-(0,Ymax),GridColor
   END IF

   PSET (Xin, Yin), N%
   first%(N%) = 1
ELSE
'
'   Якщо виклик підпрограми не перший, то до цієї точки
'   проводять лінію від попередньої
'
   LINE (Xold(N%), Yold(N%))-(Xin, Yin), N%
END IF
'
'   Точка запам'ятовується як попередня для наступного кроку
'
   Xold(N%) = Xin
   Yold(N%) = Yin

END SUB
```

Для функціонування цієї підпрограми у головній програмі мають бути такі рядки:

```
CONST Xmin = ....        ' Ліва межа графіка
CONST Xmax = ....        ' Права межа графіка
CONST Ymin = ....        ' Нижня межа графіка
CONST Ymax = ....        ' Верхня межа графіка

   . . .
```

```
' ------------------------------------------------------
'  Виклик підпрограми виведення графіка:
'  1-ша цифра - номер кольору змінної, що виводиться
' ------------------------------------------------------
   CALL Graphic(1, t, y(1))        ' Виведення на графік змінної
```

Деякі пояснення до підпрограми:

- у підпрограмі для реалізації перевірки першого звертання до неї використана особливість середовища Quick Basic: під час запуску програми всі змінні "обнулюються", тому, якщо змінній ще не присвоєно жодного значення, вона дорівнює нулю;
- для задання меж графіка за осями X та Y можна застосувати не тільки глобальні константи (ті, що задані в головному модулі), але й глобальні змінні;
- якщо на персональному комп'ютері встановлено інший тип дисплея, необхідно підпрограму трохи підкоригувати; наприклад, нижче показано зміни в підпрограмі для дисплеїв EGA і CGA.

```
' ========================================================
'       Для  дисплея  EGA  ( 640 x 350 , 16 кольорів )
' ========================================================
CONST Xleft% = 55          ' Ліва   межа графіка
CONST Xright% = 635        ' Права  межа графіка
CONST Yupper% = 3          ' Верхня межа графіка
CONST Ylower% = 323        ' Нижня  межа графіка
CONST LocatX = 42          ' Положення оцифровки осі  X

CONST White = 0            ' White=0   - чорний фон
                           ' White=1   - білий фон
' ========================================================

   . . .

' Встановлення кольорового режиму дисплея  VGA  640 x 350
     SCREEN 9
```

```
      WIDTH 80, 43
      ' Якщо треба, то білий фон
      IF White = 1 THEN LINE (0, 0)-(639, 349), 15, BF

   . . .
```

```
' ============================================================
'       Для  дисплея  CGA  ( 640 x 200 , 1 колір )
' ============================================================
CONST Xleft% = 55          ' Ліва   межа графіка
CONST Xright% = 635        ' Права  межа графіка
CONST Yupper% = 3          ' Верхня межа графіка
CONST Ylower% = 165        ' Нижня  межа графіка
CONST LocatX = 22          ' Положення оцифровки осі  X
' ============================================================

   . . .

' Встановлення режиму дисплея  CGA  640 x 200
      SCREEN 2
'
'   Визначення ширини графіка за  X  та  Y
'
      XD = (Xmax - Xmin) / 10!
      YD = (Ymax - Ymin) / 10!
'
'   Виведення цифрових значень  Y
'
      FOR k% = 0 TO 10
         LOCATE 1 + 2 * k%, 1

   . . .

      VIEW (Xleft%, Yupper%)-(Xright%, Ylower%)
      WINDOW (Xmin, Ymin)-(Xmax, Ymax)
'
'   Формування горизонтальної розбивки графіка
'
      FOR y = Ymin TO Ymax + YD / 10! STEP YD
         LINE (Xmin, y)-(Xmax, y), , , &H1111
      NEXT y
```

```
'
'   Формування вертикальної розбивки графіка
'
      FOR x = Xmin TO Xmax + XD / 10! STEP XD
         LINE (x, Ymin)-(x, Ymax), , , &H1111
      NEXT x
'  Якщо можливо, рисуємо осі
      ' Вісь  Y
      IF SGN(Ymin) <> SGN(Ymax) THEN LINE (Xmin, 0)-(Xmax, 0)
      ' Вісь  X
      IF SGN(Xmin) <> SGN(Xmax) THEN LINE (0, Ymin)-(0, Ymax)
   END IF

   . . .
```

Іноді виникає потреба результати розрахунків у вигляді графіка вставити у текст статті чи звіту, який набрано в середовищі текстового редактора у середовищі Windows (наприклад, Microsoft Word), а потім роздрукувати документ на чорно-білому принтері ("друкарці"). У такому разі найзручніше використати монохромний режим VGA для того, щоб отримати найвищу роздільову здатність без проблем з перетворенням кольору під час вставляння графіків у текст. Для вставляння графіків, що одержані у середовищі Quick Basic, у документ Microsoft Word (чи іншого текстового редактора) можна скористатися описаним нижче алгоритмом для середовища Windows 95/98 і вище (аналогічні дії можна виконувати і в інших середовищах програмування, наприклад, Turbo Pascal).

- Запустити графічний редактор Paint, який є в середовищі Windows;
- запустити програмне середовище Quick Basic (чи Turbo Pascal) і отримати на екрані графік;
- одержаний графік скопіювати до буфера Windows натисканням комбінації клавіш **Alt + PrintScreen (Alt + PrtScr)**;
- натисканням комбінації клавіш **Alt + Tab** перейти до графічного редактора Paint і вставити з буфера графік як рисунок (можна скористати поєднанням клавіш **Ctrl + V** чи **Shift + Ins**);

- інвертувати кольори графіка (**Ctrl + I**) для отримання білого фону і чорних ліній; встановити атрибути (**Ctrl + E**) для чорно-білого (монохромного) рисунка графічного редактора Paint;
- одержане зображення можна зберегти як файл у графічному форматі BMP (монохромний) на диску або вставити у текст.

Для підтримання монохромного режиму VGA підпрограму виведення графіків потрібно змінити так (зверніть увагу, що колір ліній не вказується):

Quick Basic

```
' ============================================================
'        Для  дисплея  VGA  ( 640 x 480 , 2 кольори )
' ============================================================
CONST Xleft% = 55          ' Ліва   межа графіка
CONST Xright% = 635        ' Права  межа графіка
CONST Xgrid% = 104         ' Ширина розбиття за віссю  X
CONST Yupper% = 10         ' Верхня межа графіка
CONST Ylower% = 415        ' Нижня  межа графіка
CONST LocatX = 54          ' Положення цифрових позначок осі  X
' ============================================================
'   Перевірка першого виклику за кожним графіком
'
IF first%(N%) = 0 THEN
'   Перевірка найпершого виклику підпрограми  Graphic
   IF FirstCALL% = 0 THEN
      FirstCALL% = 1
'
' Встановлення монохромного режиму дисплея  VGA  640 x 480
'
      SCREEN 11
      WIDTH 80, 60
'
'   Визначення ширини графіка за  X  та  Y
      XD = (Xmax - Xmin) / 10!
      YD = (Ymax - Ymin) / 10!
'   Виведення цифрових значень  Y

   . . .   (без змін)
```

```
'    Виведення цифрових значень  X
   . . .    (без змін)
      NEXT k%
      VIEW (Xleft%, Yupper%)-(Xright%, Ylower%)

      WINDOW (Xmin, Ymin)-(Xmax, Ymax)
'
'    Формування горизонтальної розбивки графіка
'
      FOR y = Ymin TO Ymax + YD / 10! STEP YD
         LINE (Xmin, y)-(Xmax, y), , , &H1111
      NEXT y
'
'    Формування вертикальної розбивки графіка
'
      FOR x = Xmin TO Xmax + XD / 10! STEP XD
         LINE (x, Ymin)-(x, Ymax), , , &H1111
      NEXT x

'   Якщо можливо, рисуємо осі
       ' Вісь  Y
      IF SGN(Ymin)<>SGN(Ymax) THEN LINE (Xmin,0)-(Xmax,0)
       ' Вісь  X
      IF SGN(Xmin)<>SGN(Xmax) THEN LINE (0,Ymin)-(0,Ymax)
   END IF

   PSET (Xin, Yin)
   first%(N%) = 1
ELSE
'
'    Якщо виклик підпрограми не перший, то до цієї точки
'    проводиться лінія від попередньої
'
   LINE (Xold(N%), Yold(N%))-(Xin, Yin)
END IF
'    Точка запам'ятовується як попередня для наступного кроку
   Xold(N%) = Xin
   Yold(N%) = Yin

END SUB
```

Наведений далі текст програми і отриманий графік ілюструють використання розглянутої підпрограми.

```
' ===========================================
' Потрібно для виведення графіка
' ===========================================
CONST Xmin = 0              ' Ліва межа графіка
CONST Xmax = 10             ' Права межа графіка
CONST Ymin = -1             ' Нижня межа графіка
CONST Ymax = 1              ' Верхня межа графіка

' -------   Основний цикл програми   -------
FOR x = Xmin TO Xmax STEP .1
   y = EXP(-.3 * x) * COS(3 * x)     ' Розраховуємо коливання
   CALL Graphic(1, x, y)             ' Будуємо графік коливань
NEXT x
END
```

На рис. Д.1 показано отриманий графік коливань.

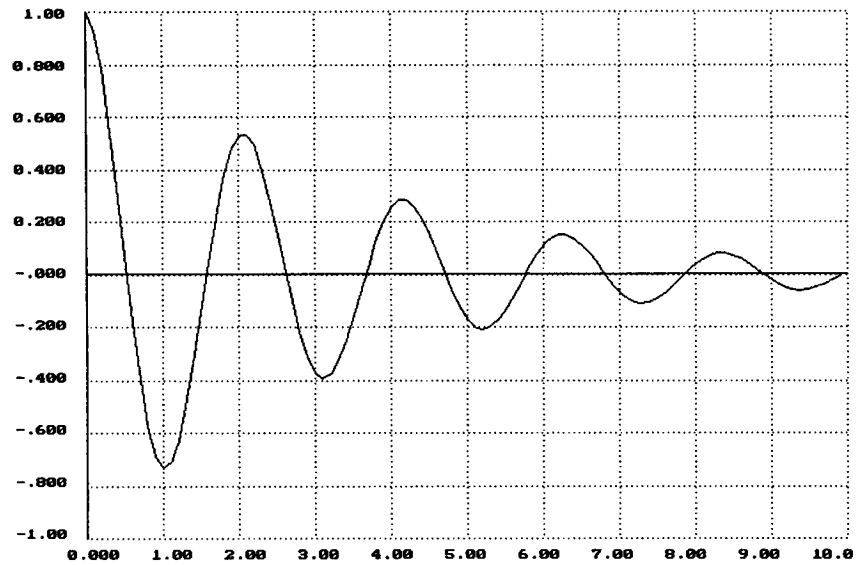


Рис. Д.1. Графік коливань, що виведений підпрограмою Graphic (Quick Basic)

Turbo Pascal

Для увімкнення монохромного (чорно-білого) режиму виведення на дисплей в модулі виведення графіка в середовищі Turbo Pascal можна спробувати задати відповідні змінні графічного режиму (можливо, для різних відеокарт доведеться поекспериментувати):

для дисплея EGA – `GraphDriver := EGAmono;`

для дисплея VGA – `GraphDriver := MCGA; GraphMode := MCGAHi;`

```
{$E+,N-   }
UNIT  Plot ;
{ ------------------------------------------------------- }
{               Модуль для виведення графіків             }
{ ------------------------------------------------------- }
{    Виконувані  функції :                                }
{  Виведення до 15 графіків функцій у графічне вікно      }
{                                                         }
{   Для ініціалізації графіка викликається процедура:     }
{                                                         }
{     InitGraphic( Xmin : REAL - мінімальне  значення  X  }
{                  Xmax : REAL - максимальне значення  X  }
{                  Ymin : REAL - мінімальне  значення  Y  }
{                  Ymax : REAL - максимальне значення  Y  }
{                  divX : REAL - крок сітки за віссю   X  }
{                  divY : REAL - крок сітки за віссю   Y  }
{                ) ;                                      }
{                                                         }
{   Для виведення графіка використовується процедура:     }
{                                                         }
{    Graphic( Np : Number_of_Plot - номер  графіка        }
{             X : REAL - координата  X                    }
{             Y : REAL - координата  Y                    }
{           ) ;                                           }
{ ------------------------------------------------------- }
INTERFACE
```

```
USES
  Graph ;
CONST
   MaxPlotNum  = 15 ;    { Максимальна кількість графіків }
TYPE
  Number_of_Plot  =  1 .. MaxPlotNum ;
PROCEDURE  InitGraphic(Xmin, Xmax ,    { Межі графіка по  X }
                       Ymin, Ymax ,       { Межі графіка по  Y }
                       divX, divY : REAL { Крок сітки по X та Y }
                           ) ;
PROCEDURE
  Graphic(Np : Number_of_Plot ;  {Номер кривої на графіку}
          X, Y : REAL) ;      { Координати поточної точки }

{ ========================================== }
IMPLEMENTATION
{ ========================================== }

VAR
   GraphDriver, GraphMode : INTEGER ;
   MaxX, MaxY          : INTEGER ;
   ReadyGraph          : BOOLEAN ;
   X_Bord, Y_Bord      : INTEGER ;
   Xleft, Xright       : INTEGER ;    { Ліва / права межі     }
                                      { графічного вікна      }
   Yupper, Ylower      : INTEGER ;    { Верхня/нижня межі     }
                                      { графічного вікна      }
   X_min, X_max        : REAL ;       { Діапазон по осі   X   }
   Y_min, Y_max        : REAL ;       { Діапазон по осі   Y   }
   dx, dy              : REAL ;       { Розбивка по X та Y    }
   Xold, Yold          : ARRAY  [1..MaxPlotNum]  OF  INTEGER ;
   PlotColor           : ARRAY  [1..MaxPlotNum]  OF  WORD ;
   StartPlot           : ARRAY  [1..MaxPlotNum]  OF  BOOLEAN ;
   i                   : Number_of_Plot ;

{ ========================================================== }
{ Процедура переведення декартових координат у фізичні екрана }
{ ========================================================== }
PROCEDURE Location(X, Y : REAL ; VAR Xplot, Yplot : INTEGER) ;
BEGIN
  Xplot := Xleft + round((X-X_min)/dx) ;
  Yplot := Yupper + round((Y_max-Y)/dy) ;
END ;
```

```
{ ========================================================= }
{            Процедура ініціалізації графіка                }
{ ========================================================= }
PROCEDURE
  InitGraphic(Xmin, Xmax ,          { Межі графіка по  X  }
              Ymin, Ymax ,          { Межі графіка по  Y  }
              divX, divY: REAL ) ; { Крок сітки по X та Y}
CONST
  XleftBord  : INTEGER = 50 ; { Ліва межа графіка на екрані }
  XrightBord : INTEGER = 5 ; { Права межа графіка на екрані }
  YupperBord  : INTEGER = 5 ; { Верхня межа графіка на екрані }
  YlowerBord  : INTEGER = 15 ; { Нижня межа графіка на екрані }
  Y_TextShift : INTEGER = 5 ;{ Зміщення позиції "оцифрування" }
  X_TextShift : INTEGER =  30 ;{                              }
{ ========================================================= }
{ Функція переведення числа у зручну для сприйняття форму    }
{ ========================================================= }
FUNCTION  Out3RealDig( RealValue : REAL ) : STRING ;
VAR
   DigString   : STRING ;
BEGIN
   IF (abs(RealValue) > 1000.0) OR (abs(RealValue) < 0.1) THEN
      IF RealValue = 0.0 THEN
         Str(RealValue:6:1,DigString)
      ELSE
         Str(RealValue:8,DigString)
   ELSE
      IF abs(RealValue) < 100.0 THEN
         Str(RealValue:6:2,DigString)
      ELSE
         Str(RealValue:6:0,DigString);
   Out3RealDig := DigString ;
END ;

VAR
   nX, nY, i          : INTEGER ;
   x1, x2, y1, y2     : INTEGER ;
   y_value, x_value   : REAL ;
   value              : STRING ;
BEGIN
   { ------------------------------------------- }
   {   Ініціалізація  графічного  режиму         }
   { ------------------------------------------- }
```

```
X_min := Xmin ;
X_max := Xmax ;
Y_min := Ymin ;
Y_max := Ymax ;
IF NOT ReadyGraph THEN
   BEGIN
      InitGraph(GraphDriver,GraphMode,'C:\BP\BGI');
      ReadyGraph := TRUE ;
   END ;
{ ------------------------------------------- }
SetViewPort(0, 0, MaxX, MaxY, ClipOn) ;
ClearViewPort ;
Xleft  := XleftBord ;          { Задання    }
Xright := MaxX - XrightBord ;  { розмірів   }
Yupper := YupperBord ;         { рамки      }
Ylower := MaxY - YlowerBord ;  { графіка    }
{ Визначення коеф. зведення декартових координат до фізичних }
dx := (Xmax - Xmin) / (Xright - Xleft) ;
dy := (Ymax - Ymin) / (Ylower - Yupper) ;
{ ================================================== }
{          Формування координатної сітки             }
{ ================================================== }
{ Задається шрифт виведення "оцифрування" }
SetTextStyle(SmallFont, 0, 0) ;
SetUserCharSize(1, 1, 1, 1) ;
SetColor(LightGray) ;
SetLineStyle(UserBitLn, $1111, NormWidth) ;
{ ---------------------------------- }
{      Побудова сітки за віссю  Y    }
{ ---------------------------------- }
nY := 1 + Trunc(Ymax/divY) - Trunc(Ymin/divY) ;
FOR  i := nY  DOWNTO  1   DO
   BEGIN
      { ------------------------------------- }
      { Виведення "оцифрування" сітки за віссю  Y }
      { ------------------------------------- }
      { Визначається значення числа  }
      y_value := divY*Trunc(Ymax/divY) - divY*(i-1) ;
      value := Out3RealDig( y_value ) ;
      { Визначається положення лінії сітки } .
      Location(Xmin, y_value, x1, y1) ;
      Location(Xmax, y_value, x2, y2) ;
      { Виведення значення числа     }
```

```
        OutTextXY(Xleft-XleftBord, y1 - Y_TextShift, value) ;
        { Проведення лінії сітки        }
        IF  y_value > Ymin  THEN  Line(x1, y1, x2, y2);
     END ;

  { ------------------------------------ }
  {      Побудова сітки за віссю  X       }
  { ------------------------------------ }
  nX := 1 + Trunc(Xmax/divX) - Trunc(Xmin/divX) ;
  FOR  i := 1  TO  nX   DO
     BEGIN
        { ---------------------------------------- }
        { Виведення "оцифрування" сітки за віссю  X }
        { ---------------------------------------- }
        { Визначається значення числа  }
        x_value := divX*Trunc(Xmin/divX) + divX*(i-1) ;
        value := Out3RealDig(x_value) ;
        { Визначається положення лінії сітки }
        x1 := Xleft + round((x_value-Xmin)/dx) ;
        OutTextXY(x1-X_TextShift, Ylower + Y_TextShift, value);
        { Проведення лінії сітки        }
        Line(x1, Yupper, x1, Ylower) ;
     END ;
  { ------------------------------------ }
  {   Побудова рамки і осей координат     }
  { ------------------------------------ }
  SetLineStyle(SolidLn, 0, NormWidth) ;
  Rectangle(Xleft, Yupper, Xright, Ylower) ;
  IF  Xmin * Xmax < 0.0  THEN
     BEGIN  { Вісь  Y }
        Location(0.0, Ymin, x1, y1) ;
        Location(0.0, Ymax, x2, y2) ;
        Line(x1, y1, x2, y2) ;
     END ;
  IF  Ymin * Ymax < 0.0   THEN
     BEGIN  { Вісь  X }
        Location(Xmin, 0.0, x1, y1) ;
        Location(Xmax, 0.0, x2, y2) ;
        Line(x1, y1, x2, y2) ;
     END ;
END ;
```

```
{ ========================================================= }
{                Процедура побудови графіка                 }
{ ========================================================= }
PROCEDURE  Graphic(Np : Number_of_Plot ; X, Y : REAL) ;
VAR
   x1, x2, y1, y2     : INTEGER ;
BEGIN
   { ------------------------------------------- }
   {     Якщо потрібно, ініціалізація ...        }
   { ------------------------------------------- }
   IF  NOT StartPlot[Np]   THEN
       BEGIN
          Location(X, Y, Xold[Np], Yold[Np]) ;
          StartPlot[Np] := TRUE ;
       END ;
   { ------------------------------------------- }
   {      Побудова графіка за поточною точкою    }
   { ------------------------------------------- }
   x1 := Xold[Np] ;
   y1 := Yold[Np] ;
   Location(X, Y, x2, y2) ;
   SetColor(PlotColor[Np]) ;
   Line(x1, y1, x2, y2) ;
   Xold[Np] := x2 ;     { Запам'ятовуємо точку  }
   Yold[Np] := y2 ;     { як попередню          }
END ;

{ ===================================================== }
{                      Реалізація                       }
{ ===================================================== }
BEGIN
  { Ініціалізація кольорів та готовності }
  { за кожним з 15 графіків              }
  FOR  i := 1  TO  MaxPlotNum  DO
     BEGIN
        PlotColor[i] := i ;
        StartPlot[i] := FALSE ;
     END ;
  ReadyGraph := FALSE ;
  { ------------------------------------- }
  { Встановлення параметрів графіка залежно }
  { від типу адаптера дисплея               }
  { ------------------------------------- }
```

```
  GraphDriver := Detect ;
  DetectGraph(GraphDriver, GraphMode) ;
  CASE  GraphDriver   OF
     1    : BEGIN
              GraphMode:=4;    { -------------------------- }
              MaxX:=639;       {         CGA - дисплей      | }
              MaxY:=199;       { -------------------------- }
              X_Bord:=75;
              Y_Bord:=15;
            END ;
     3,4  : BEGIN
              GraphMode:=1;    { -------------------------- }
              MaxX:=639;       {         EGA - дисплей      | }
              MaxY:=349;       { -------------------------- }
              X_Bord:=75;
              Y_Bord:=15;
            END ;
     5    : BEGIN
              GraphMode:=3;    { -------------------------- }
              MaxX:=639;       {    EGAMono - дисплей       }
              MaxY:=349;       { -------------------------- }
              X_Bord:=75;
              Y_Bord:=15;
            END ;
     7    : BEGIN
              GraphMode:=0;    { -------------------------- }
              MaxX:=719;       {     Hercules - дисплей     }
              MaxY:=347;       { -------------------------- }
              X_Bord:=75;
              Y_Bord:=15;
            END ;
     9    : BEGIN
              GraphMode:=2;    { -------------------------- }
              MaxX:=639;       {         VGA - дисплей      }
              MaxY:=479;       { -------------------------- }
              X_Bord:= 75 ;
              Y_Bord:= 15 ;
            END ;
  END ;     {   ***   OF  CASE   ***   }

END .
```

Наведений далі текст програми ілюструє використання пропонованого модуля.

```
program  Plot_Demo;

uses
   Crt, Plot;
const
   Xmin : real = 0;       { Ліва межа графіка    }
   Xmax : real = 10;      { Права межа графіка   }
   Ymin : real = -1;      { Нижня межа графіка   }
   Ymax : real = 1;       { Верхня межа графіка  }
   dX   : real = 1;       { Крок сітки за X      }
   dY   : real = 0.2;     { Крок сітки за Y      }
var
   x, y1, y2, y3, h : real;
begin
   x := Xmin;
   h := 0.025;
{ ------------------------------------------ }
{            Визначити графік                }
{ ------------------------------------------ }
   InitGraphic(Xmin, Xmax, Ymin, Ymax, dX, dY);
{                                            }
   while  x <= Xmax  do
      begin
         y1 := exp(-0.3*x) * cos(3*x);
         y2 := exp(-0.3*x);
         y3 := cos(3*x);
         Graphic(12, x, y1);
         Graphic(2, x, y2);
         Graphic(1, x, y3);
         x := x + h;
      end;
   repeat  until  KeyPressed;
end.
```

MathCAD

Графіки в пакеті MathCAD вирізняються різноманітністю і простотою використання, до того ж просто копіюються і вставляються (як і в будь-якій іншій програмі для Windows) в іншу програму. Для вставляння графіків і формул MathCAD у Microsoft Word можна рекомендувати використовувати пункт меню "**Правка → Спеціальна вставка → Рисунок**" (для української версії редактора Word) – це дасть змогу отримати графіку вищої якості та уможливить її обробку в середовищі Microsoft Word без встановленого пакета MathCAD. Детальніше з можливостями MathCAD щодо виведення графіків можна ознайомитися у відповідній літературі [розд. А].

MATLAB

Графіки пакета MATLAB прості у використанні та мають багато можливостей, для вивчення яких слід звернутися до відповідної літератури чи вбудованої допомоги пакета. З екрана виведені графіки зручно копіювати за допомогою пункту меню графіка "**Edit → Copy figure**" у будь-який текстовий редактор, наприклад, Microsoft Word, причому у векторному форматі, що дає змогу отримати рисунки високої якості. Для забезпечення такої можливості необхідно перевірити, чи встановлений режим "**Preserve information (metafile if possible)**", що досягається використанням пунктів меню середовища "**File → Preferences → Figure Copy Template → Copy Option**".

Таблиця Д.1

Технічні дані двигунів постійного струму з незалежним збудженням серії 2П

№	Тип	$P_н$, кВт	$U_н$, В	$I_н$, А	$n_н$, об./хв	$n_{макс.}$, об./хв	η	$R_я$, Ом	$R_{ДП}$, Ом	$R_{ОЗД}$, Ом	$L_я$, мГ	$J_Д$, кг·м^2
1	2ПН132L	8,5	220	46,00	2200	4000	0,84	0,167	0,124	89	3,5	0,048
2		8,5	440	22,86	2240	4000	0,845	0,67	0,445	25	14	
3		14,0	220	74,00	3150	4000	0,86	0,08	0,066	76	1,8	
4		14,0	440	36,78	3150	4000	0,865	0,322	0,27	20,6	7,0	
5	2ПО200L	7,1	110	78,24	750	3000	0,825	0,055	0,037	102	2,4	0,3
6		7,1	220	38,65	750	2500	0,835	0,22	0,15	23,7	9,4	
7		11	220	57,80	1000	3000	0,865	0,125	0,08	102	5,3	
8		17	220	86,82	1500	3500	0,89	0,055	0,037	102	2,4	
9		17	440	43,41	1500	3500	0,89	0,22	0,15	23,7	9,4	
10		24	220	121,2	2360	3500	0,90	0,031	0,037	102	1,3	
11		24	440	60,27	2120	3500	0,905	0,125	0,15	23,7	5,3	
12	2ПФ200М	20	440	50,50	2200	3500	0,90	0,143	0,073	96	5,6	0,25
13		22	220	114,3	1600	3500	0,875	0,047	0,029	46	1,6	
14		22	440	56,82	1600	3500	0,88	0,188	0,116	13,1	6,4	
15		30	440	75,75	2200	3500	0,90	0,106	0,061	46	3,6	
16		40	440	100,5	3000	3500	0,905	0,071	0,041	96	2,5	
17	2ПФ200L	15	110	166,3	750	3300	0,82	0,031	0,02	42	1,2	0,3
18		15	220	82,64	750	2500	0,825	0,125	0,08	10,6	4,6	
19		20	220	106,1	1000	3300	0,855	0,083	0,053	55	3,2	
20		30	220	154,1	1500	3500	0,885	0,031	0,02	31,7	1,2	

Примітка. Опори обмоток збудження наведено для кожного типорозміру для номінальної напруги збудження 220 В.

Таблиця Д.2

Основні технічні дані асинхронних електродвигунів серії 4А з фазним ротором (220/380 В)

№	Тип	$P_н$, кВт	$I_н$ А	ккд %	$\cos\varphi_н$	$E_{р0}$ В	λ	$S_н$ %	$S_к$ %	r_1 Ом	x_1 Ом	r_2' Ом	x_2' Ом	$J_д$ кг·м^2
	Синхронна швидкість обертання 1500 об./хв													
1	4АК160S4	11,0	22	86,5	0,86	305	3,0	4,4	33,0	0,038	0,068	0,051	0,086	0,10
2	4АК160М4	14,0	29	88,5	0,87	300	3,5	3,7	32,1	0,032	0,060	0,042	0,078	0,13
3	4АК180М4	18,5	38	89,0	0,88	295	4,0	2,9	31,1	0,022	0,042	0,034	0,063	0,23
4	4АК200М4	22,0	45	90,0	0,87	340	4,0	2,5	22,0	0,024	0,050	0,026	0,075	0,37
5	4АК200L4	30,0	55	90,5	0,87	350	4,0	2,5	22,0	0,026	0,057	0,030	0,087	0,45
6	4АК225М4	37,0	160	90,0	0,87	160	3,0	3,5	20,0	0,023	0,061	0,027	0,069	0,64
7	4АК250SA4	45,0	170	91,0	0,88	230	3,0	3,0	20,5	0,020	0,067	0,030	0,080	1,0
8	4АК250SB4	55,0	170	90,5	0,90	200	3,0	2,3	19,6	0,017	0,061	0,025	0,073	1,1
9	4АК250М4	71,0	170	91,5	0,86	250	3,0	2,5	19,5	0,015	0,053	0,021	0,064	1,2
	Синхронна швидкість обертання 1000 об./хв													
10	4АК160S6	7,5	18	82,5	0,77	300	3,5	5,0	30,1	0,054	0,079	0,068	0,12	0,14
11	4АК160М6	10,0	20	84,5	0,76	310	3,8	4,3	27,1	0,043	0,071	0,058	0,13	0,18
12	4АК180М6	13,0	25	85,5	0,80	325	4,0	4,4	29,1	0,035	0,065	0,057	0,11	0,22
13	4АК200М6	18,5	35	88,0	0,81	360	3,5	3,5	27,5	0,030	0,060	0,038	0,078	0,40
14	4АК200L6	22,0	45	88,0	0,80	330	3,5	3,5	21,0	0,032	0,066	0,041	0,080	0,45
15	4АК225М6	30,0	150	89,0	0,85	140	2,5	3,3	19,5	0,029	0,073	0,030	0,091	0,74
16	4АК250S6	37,0	165	89,0	0,84	150	2,5	3,5	18,0	0,026	0,063	0,024	0,078	1,16
17	4АК250М6	45,0	160	90,5	0,87	180	2,5	2,5	17,0	0,029	0,062	0,024	0,092	1,26
	Синхронна швидкість обертання 750 об./хв													
18	4АК160S8	5,5	14	80,0	0,70	300	2,5	6,4	29,0	0,060	0,112	0,094	0,175	0,14
19	4АК160М8	7,1	6	82,0	0,70	290	3,0	5,5	23,3	0,053	0,11	0,079	0,208	0,18
20	4АК180М8	11,0	25	85,5	0,72	270	3,5	4,4	22,7	0,041	0,086	0,062	0,167	0,25
21	4АК200М8	15,0	28	86,0	0,70	360	3,0	3,5	23,0	0,040	0,081	0,048	0,12	0,40
22	4АК200L8	18,5	40	86,0	0,73	300	3,0	3,6	21,5	0,038	0,089	0,046	0,2	0,61
23	4АК225М8	22,0	140	87,0	0,82	102	2,2	4,5	19,5	0,039	0,10	0,043	0,13	0,74
24	4АК250S8	30,0	155	88,5	0,81	125	2,2	4,0	20,0	0,033	0,081	0,034	0,10	1,20
25	4АК250М8	37,0	155	89,0	0,80	148	2,2	3,5	18,5	0,031	0,078	0,031	0,10	1,40

Примітка. $\lambda = M_к/M_н$ – перевантажувальна здатність АД за моментом.

Таблиця Д.3

Таблиця найчастіше вживаних Z-перетворень

$F(s)$	$f(t)$	$F(z)$
1	$\delta(t)$	1
$\frac{1}{s}$	$1(t)$	$\frac{z}{z-1}$
$\frac{1}{s^2}$	t	$\frac{hz}{(z-1)^2}$
$\frac{1}{s^3}$	$\frac{1}{2!}t^2$	$\frac{h^2 z(z+1)}{2(z-1)^3}$
$\frac{1}{s^4}$	$\frac{1}{3!}t^3$	$\frac{h^3\left(z^2+4z+1\right)}{6(z-1)^4}$
$\frac{1}{s+a}$	e^{-at}	$\frac{z}{z-e^{-ah}}$
$\frac{1}{(s+a)^2}$	te^{-at}	$\frac{hze^{-ah}}{\left(z-e^{-ah}\right)^2}$
$\frac{a}{s(s+a)}$	$1-e^{-at}$	$\frac{\left(1-e^{-ah}\right)z}{(z-1)\left(z-e^{-ah}\right)}$
$\frac{a}{s^2(s+a)}$	$t-\frac{1-e^{-at}}{a}$	$\frac{hz}{(z-1)^2}-\frac{\left(1-e^{-ah}\right)z}{a(z-1)\left(z-e^{-ah}\right)}$

Продовження табл. Д3

$F(s)$	$f(t)$	$F(z)$
$\frac{a}{s^3(s+a)}$	$\frac{1}{2}\left(t^2-\frac{2}{a}t+\frac{2}{a^2}-\frac{2}{a^2}e^{-ah}\right)$	$\frac{h^2z}{(z-1)^3}+\frac{hz(ah-2)}{2a(z-1)^2}+$ $+\frac{z}{a^2(z-1)}-\frac{z}{a^2\left(z-e^{-ah}\right)}$
$\frac{\omega}{s^2+\omega^2}$	$\sin\ \omega t$	$\frac{z\ \sin\ \omega h}{z^2-2z\ \cos\ \omega h+1}$
$\frac{s}{s^2+\omega^2}$	$\cos\ \omega t$	$\frac{z\ (z-\cos\ \omega h)}{z^2-2z\ \cos\ \omega h+1}$
$\frac{\omega}{(s+a)^2+\omega^2}$	$e^{-at}\sin\omega t$	$\frac{ze^{-ah}\sin\ \omega h}{z^2-2ze^{-ah}\ \cos\ \omega h+e^{-2ah}}$
$\frac{s+a}{(s+a)^2+\omega^2}$	$e^{-at}\cos\omega t$	$\frac{z^2-ze^{-ah}\cos\ \omega h}{z^2-2ze^{-ah}\ \cos\ \omega h+e^{-2ah}}$

ЗМІСТ

НАВЧАЛЬНЕ ВИДАННЯ

Костинюк Лев Дмитрович
Паранчук Ярослав Степанович
Мороз Володимир Іванович

МОДЕЛЮВАННЯ ЕЛЕКТРОПРИВОДІВ

Редактор Оксана Чернигевич
Технічний редактор Лілія Саламін
Комп'ютерний набір Володимир Мороз
Комп'ютерне верстання Володимира Мороза,
Галини Сукмановської
Художник-дизайнер Уляна Келеман

Здано у видавництво 24.09.2004. Підписано до друку 28.12.2004.
Формат 70×90/16. Папір офсетний. Друк офсетний.
Умовн. друк. арк. 29,5. Обл.-вид. арк. 21,70.
Наклад 300 прим. Зам. 40683.

Видавництво Національного університету "Львівська політехніка"
Реєстраційне свідоцтво серії ДК № 751 від 27.12.2001 р.

Поліграфічний центр Видавництва
Національного університету "Львівська політехніка"

вул. Ф. Колесси, 2, Львів, 79000