Міністерство освіти і науки України
Одеський національний політехнічний університет

Андрющенко Олег Андрійович

ОСНОВИ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ ЕЛЕКТРОМЕХАНІЧНИХ ПРИСТРОЇВ І ЕЛЕКТРОМЕХАНІЧНИХ СИСТЕМ

ЧАСТИНА І. ТЕОРЕТИЧНІ ОСНОВИ ФВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ

ЧАСТИНА ІІ. ТЕХНІЧНІ ПРИЙОМИ РОБОТИ В СЕРЕДОВИЩІ SIMULINK СИСТЕМИ MATLAB

КОНСПЕКТ ЛЕКЦІЙ

для студентів напряму підготовки 6.050702 - ЕЛЕКТРОМЕХАНІКА
професійне спрямування 8.092203 – Електромеханічні системи
автоматизації та електропривод

Одеса – 2011

ЗМІСТ

ЧАСТИНА І
ТЕОРЕТИЧНІ ОСНОВИ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ

ЧАСТИНА ІІ
ТЕХНІЧНІ ПРИЙОМИ РОБОТИ В СЕРЕДОВИЩІ SIMULINK СИСТЕМИ MATLAB

ЧАСТИНА І. ТЕОРЕТИЧНІ ОСНОВИ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ

ЛЕКЦІЯ 1. ПРОЦЕСИ ПРОЕКТУВАННЯ ТА ОРГАНІЗАЦІЯ ПРОЕКТНИХ РОБІТ

Студентам, або фахівцям, які вивчають проблеми автоматизації проектування, можна підходити до них із двох позицій. Перша – як користувачу відомих програмних продуктів, і друга – як розробнику нових програмних продуктів. Які б складні та довершені не були програми загального користування, вони не можуть задовольнити всіх професійних потреб користувача. Більш того, обов'язковість самостійної творчості і необхідний для цього інструментарій вже закладені в таких програмах. Згодом досвідчений проектувальник стає власником особистої бібліотеки прикладних програм із власноруч оформленим інтерфейсом. Обрана прикладна програма таким чином, становиться **платформою** для розробки власних програм. А починати створення та поповнення цієї, професійно орієнтованої, бібліотеки потрібно вже зараз, із студентської лави. Є ще одна сторона питання: сьогоднішній студент в недалекому майбутньому може стати одним з керівників підприємства або установи і замовником певних проектів, тому розуміння сутності та процесів проектування стане йому в нагоді. Саме із цих позицій і розглядає автор конспекту мету вивчення дисципліни "Основи автоматизованого проектування електромеханічних пристроїв та електромеханічних систем" та її зміст.

1.1 Актуальність автоматизації проектування

Проект – (англ. progect, design) - сукупність **документів** - розрахунків, креслень, макетів, моделей тощо, необхідних для зведення споруд, виготовлення машин, приладів. Проект створюється внаслідок **проектування** у будь-якій **галузі (електротехніка, електромеханіка, машинобудування, будівництво)**. Проектом може бути план заходів для досягнення якоїсь певної мети. Процес проектування нової техніки є однією із складових **инжинірінгової діяльності** фахівця і безумовно є творчим процесом.

Проектування є складним, трудомістким процесом, який вимагає від фахівця певного обсягу знань, здібностей і творчого підходу і, головне, вимагає великих витрат часу. Тому природно виглядають намагання проектувальників формалізувати процес проектування і перекласти його виконання на плечі комп'ютера, тобто **автоматизувати** проектні роботи. Зробити весь процес проектування від початку до кінця цілком **автоматичним** дуже складно, і, мабуть, не потрібно. Але полегшити роботу над окремими етапами проектування у **автоматизованому** режимі, із участю людини – виконавця, реально і перспективно. Цьому сприяє те, що процес проектування має складові, однакові, або подібні для всіх проектів, які піддаються формалізації. До них можна віднести взагалі всі розрахункові роботи, задачі аналізу властивостей відомих і синтезу параметрів нових пропонованих систем, оптимізацію структур, параметрів та режимів роботи по одному або по багатьом критеріям, математичне моделювання із метою дослідження властивостей автоматизованих систем у статичних та динамічних режимах, перевірку на моделях працездатності систем у штатних та аварійних режимах, побудову графіків та діаграм, а також складання звітної технічної документації.

Таким чином, шлях від ідеї, або концепції до виготовлення приладу або механізму скорочується від тижнів, місяців і навіть років при "ручному" проектуванні та підготовці виробництва до діб, або навіть годин. Декому може здатися, що при автоматизованих процесах створення нової техніки зменшується творча складова участі людини. Навпаки, картина протилежна – від людини, як ніколи раніш, вимагається широка ерудиція, креативність, фантазія щоб перемогти у конкурентному змаганні із виконавцями інших організацій, також "озброєними" не менш ефективними можливостями.

Далі, після виконання проектних робіт, настає час наступних етапів створення нової техніки, які теж підлягають автоматизації: розробки технологічних процесів, підготовки виробництва і, наприкінці, саме виробництво, випуск продукції і її супроводження до користувача, а також у користувача впродовж терміну експлуатації.

1.2 Стадії та етапи проектування

Проектні роботи, незалежно від об’єкту проектування, мають багато спільного і виконуються за єдиними стандартними вимогами. Так, згідно державному стандарту (ГОСТ 34.601.90) для проектних і інших робіт із створення автоматизованих систем (АС) всі проектні роботи розбиваються на ряд стадій та етапів, табл. 1.1.

Таблиця 1.1. Стадії і етапи робіт по створенню АС (ГОСТ 34.601.90 - Автоматизированные системы. Стадии создания)

Стадії	Етапи робіт
1. Формування вимог до АС	1.1. Обстеження об'єкту і обґрунтування необхідності створення АС. 1.2. Формування вимог користувача до АС. 1.3. Оформлення звіту про виконану роботу і заявки на розробку АС (тактико-технічного завдання)
2. Розробка концепції АС.	2.1. Вивчення об'єкту. 2.2. Проведення необхідних науково-дослідних робіт. 2.3. Розробка варіантів концепції АС, що задовольняє вимогам користувача. 2.4. Оформлення звіту про виконану роботу.
3. Технічне завдання.	Розробка і затвердження технічного завдання на створення АС.
4. Ескізний проект.	4.1. Розробка попередніх проектних рішень по системі і її частинам. 4.2. Розробка документації на АС і її частини.
5. Технічний проект.	5.1. Розробка проектних рішень по системі і її частинам. 5.2. Розробка документації на АС і її частини. 5.3. Розробка і оформлення документації на постачання виробів для комплектування АС і (або) технічних вимог (технічних завдань) на їх розробку. 5.4. Розробка завдань на проектування в суміжних частинах проекту об'єкту автоматизації.
6. Робоча документація.	6.1. Розробка робочої документації на систему і її частини. 6.2. Розробка або адаптація програм.
7. Введення в дію.	7.1. Підготовка об'єкту автоматизації до введення в дію.

	7.2. Підготовка персоналу. 7.3. Комплектація АС виробами, що поставляються (програмними і технічними засобами, програмно-технічними комплексами, інформаційними виробами). 7.4. Будівельно-монтажні роботи. 7.5. Пуско-налагоджувальні роботи. 7.6. Проведення попередніх випробувань. 7.7. Проведення дослідної експлуатації. 7.8. Проведення приймальних випробувань.
8. Супровід АС	8.1. Виконання робіт відповідно до гарантійних зобов'язань. 8.2. Післягарантійне обслуговування.

Треба звернути увагу на те, що стадії технічного завдання передує ще дві стадії, які присвячені детальному ознайомленню із об’єктом та розробці концепції системи.

Конкретний зміст і терміни виконання кожного етапу роботи, із описанням результату, що очікується і форми звітності вказують **в календарному плані виконання робіт** - документі, який є складовою частиною **технічного завдання**. В календарному плані вказуються окремі етапи роботи, їх зміст, результати виконання етапу (опис документів та іншої продукції, якими звітують про виконання етапу), терміни виконання етапів. Після виконання кожного етапу складається **Акт здавання – приймання виконаних робіт**. До складання та виконання технічного завдання та календарного плану треба підходити уважно, бо саме ці документи регламентують відповідальність виконавця перед замовником і є документами, на підставі яких виконується фінансування проектних робіт. Якщо ви замовляєте проект, треба уважно підходити до обрання виконавця (виконавців) проекту. Бажано щоб вони мали позитивні рекомендації і авторитет у чинній галузі. Досвідчені менеджери проектних організацій при обговоренні умов та суті проекту практикують презентацію своєї організації із оглядом вже виконаних проектів відомих об’єктів.

1.3 Життєвий цикл проекту

Підвищення якості будь якої продукції, в тому числі і проектної, а також підвищення відповідальності виконавця перед замовником призвело до виникнення такого поняття як **життєвий цикл** виробу, обладнання, проекту, тощо. Життєвий цикл визначає умовно початок і кінець участі виконавця у "житті" певного виробу. Щодо проектування це

послідовність **фаз проекту,** що задається виходячи з потреб управління проектом. Життєвий цикл проекту має 5 фаз:

1. Ініціація
2. Планування
3. Виконання
4. Контроль і моніторинг
5. Завершення

Життєвий цикл проекту може модулюватись за кількома принципами: водопаду, ітеративної моделі, спіральним методом, інкрементним методом.

При моделюванні **за принципом водопаду** робота над проектом рухається лінійно через ряд фаз, таких як:

- аналіз вимог (дослідження середовища);
- проектування;
- розробка і реалізація під проектів;
- перевірка під проектів;
- перевірка проекту в цілому.

Недоліками такого підходу є накопичення можливих на ранніх етапах помилок до моменту закінчення проекту і, як наслідок, зростання ризику провалу проекту, збільшення вартості проекту. Це приклад формального підходу до виконання проекту. Він виправданий, у випадках виконання однотипних проектів, коли проектувальник не очікує якихось відхилень або несподіванок у технічних рішеннях.

Ітеративний підхід - виконання робіт паралельно з безперервним аналізом отриманих результатів і коректуванням попередніх етапів роботи. Проект при цьому підході в кожній фазі розвитку проходить цикл, що повторюється: Планування — Реалізація — Перевірка — Оцінка. Переваги ітеративного підходу:

- зниження дії серйозних ризиків на ранніх стадіях проекту, що веде до мінімізації витрат на їх усунення;
- організація ефективного зворотного зв'язку проектної команди із споживачем (а також замовниками) і створення продукту, що реально відповідає його потребам;
- акцент зусиль на найбільш важливі і критичні напрями проекту;

- безперервне ітеративне тестування, що дозволяє оцінити успішність всього проекту в цілому;
- раннє виявлення конфліктів між вимогами, моделями і реалізацією проекту;
- більш рівномірне завантаження учасників проекту;
- ефективне використання накопиченого досвіду;
- реальна оцінка поточного стану проекту і, як наслідок, велика упевненість замовників і безпосередніх учасників в його успішному завершенні.

Ітеративний підхід, як бачимо з переліку його переваг, найбільш виважений, такий що забезпечує оптимальне проектне рішення і на окремих етапах виконання і в цілому по проекту. Тут можна додати ще одну рису ітеративного підходу, яка є дуже важливою, хоча і не афішується. Важливо кожний підетап роботи доводити до безпосередніх користувачів, обговорювати їх, моделювати типові ситуації, які виникають в експлуатації. Тільки завдяки такому зворотному зв'язку можна виявити похибки та конфліктні ситуації ще на ранніх етапах проектування.

При моделюванні життєвого циклу проекту по **спіральній моделі** розглядається залежність ефективності проекту від його вартості з часом. На кожному витку спіралі виконується створення чергової версії продукту, уточнюються вимоги проекту, визначається його якість і плануються роботи наступного витку. Фактично спіральна модель – це безперервне проектування у напрямі удосконалення об'єкту. Спіральна модель вигідна і навіть необхідна для підтримання у часі конкурентноздатності даного продукту. Прикладами є появлення із часом все нових і нових версій програмного забезпечення різних видів, автомобілів, літаків, в яких назва не змінюється, але змінюється номер серії, версії.

Моделювання життєвого циклу проекту **інкрементним методом** має на увазі розбиття великого об'єму проектно-конструкторських робіт на послідовність більш малих складових частин. Цей метод є аналогом методу декомпозиції, і не виключає використання інших названих методів.

1.4 Особливості та зміст навчального проектування

Форми і зміст навчального проектування (курсові роботи та курсові проекти, дипломні проекти) відрізняються від реального проектування за кількістю і обсягом стадій та етапів і також залежать від галузі знань, напряму, спеціальності і об'єкту проектування.

Нижче пропонується перелік неформальних етапів і зміст навчального проектування системи автоматизованого електроприводу, або системи автоматизації механізму,

технологічного процесу, запропонований професором Ільїнським Н.Ф., професором кафедри електроприводу Московського енергетичного інституту (МЕІ).

Завдання проектування. Найбільш поширені завдання проектування мають приблизно наступні формулювання: замість застарілого електроприводу даної установки розробити сучасний, з кращими технічними і економічними показниками; замість нерегульованого електроприводу агрегату застосувати регульований; розробити електропривод, яким можна замінити імпортний, не забезпечений запасними елементами; розробити електропривод якої-небудь унікальної установки – випробувального стенду, спеціального транспортера і тому подібне. Ці завдання проектування зовсім не прості, оскільки можуть бути вирішені різними, в загальному випадку не рівноцінними способами, а вибір одного рішення, яке і буде потім реалізуватись, повинен бути зроблений на основі ряду критеріїв при урахуванні системи конкретних обмежень. Назвемо основні етапи інженерного проектування.

Формулювання завдання – перший етап проектування. Це точна вказівка того, що є і чим це не влаштовує, і що і в якому сенсі повинне стати краще після реалізації проекту. На цьому етапі не потрібні деталі, потрібні лише найголовніші риси об'єкту до і після проектування. Якщо цей етап виконаний погано, є дуже велика небезпека, що вся подальша праця буде витрачена даремно.

Аналіз завдання – другий етап проектування – виявлення всіх істотних якісних і кількісних ознак створюваного об'єкту в початковому (до проектування) і кінцевому (після проектування) станах, визначення обмежень і призначення критеріїв, по яких оцінюватиметься якість спроектованого об'єкту.

Пошук можливих рішень – це третій етап проектування. Тут в першу чергу необхідні знання, але окрім знань потрібне нестандартне мислення, уміння уникати як консерватизму, так і поспішності. Дуже корисні аналоги, зрозуміло, при критичному до них відношенні, відвідини виставок, читання літератури, пошук в Інтернеті, консультації і тому подібне. Ступінь новизни рішень, що обираються, їх світовий рівень треба оцінювати за результатами патентного пошуку. Навіть у простому випадку доречно запропонувати декілька (багато) рішень, які в принципі відповідають завданню. Коли пропонується багато рішень, зрозуміло, не свідомо непридатних, менше шансів пропустити хороше.

Вибір рішення з множини можливих на основі критеріїв і з урахуванням обмежень. Це четвертий, дуже відповідальний етап. Тут знову не потрібні надмірні деталі, окрім тих, що дозволяють цілеспрямовано, по критеріях, порівнювати рішення. Тут дуже важливі вірні крупні оцінки. У теорії проектування вводиться поняття не гірших рішень, тобто рішень, що

потрапляють в деяку допустиму область по сукупності ознак, і формулюються алгоритми їх пошуку.

Детальна розробка вибраного технічного рішення. Це п'ятий етап – етап остаточного вибору устаткування, розрахунку характеристик, складання алгоритмів управління, конструктивної компоновки вузлів, оцінки основних показників і тому подібне.

П'ятий етап виконується завжди – і в реальних, і в навчальних проектах. Проте якщо йому не передують перші чотири або якщо вони виконані неякісно, не творчо, підсумки можуть бути сумними. Підкреслимо, що, як і всякий творчий процес, конкретне проектування, навіть при дуже жорстких обмеженнях в часі, не розвивається по рівномірно висхідній лінії – неминучі повернення, повтори і тому подібне. У теорії такі дії отримали назву ітерацій. У хороших проектах перші чотири етапи займають не менше 50% всього часу – при цьому створюється або, точніше, може створюватися дійсно нове і дійсно хороше, краще, ніж було, рішення.

1.5 Із історії автоматизації проектних робіт

Історично автоматичні та автоматизовані системи проектування з’явились одночасно із появою комп’ютерів, що цілком природно, адже комп’ютер став потужним інструментом інтелектуальної діяльності людини. Перші автоматизовані системи проектування (50-ті роки минулого сторіччя) були обмежені окремими галузями і використовувались для полегшення графічної роботи, конструювання нескладних деталей, оформлення конструкторської документації, підготовки технологічних процесів. Автоматизовані системи проектування та підготовки технологічного процесу почали широко використовуватись із поширенням у 60-70 р.р. верстатів із програмним керуванням та роботів, а пізніше гнучких автоматизованих ліній та виробництв. Цифрові ЕОМ також почали використовувати в автоматизованих системах управління виробничими процесами, диспетчеризації, обліку та руху деталей та матеріалів. Згодом, із удосконаленням і розширенням функцій комп’ютерів, вони почали використовуватись для автоматизації науково – дослідних робіт.

У більшості західних програм автоматизованого проектування в назві присутня абревіатура **CAD**, яка розшифровується як Computer Aided Design, або Computer Aided Drafting (проектування або конструювання за допомогою ЕОМ, або креслення за допомогою ЕОМ). Інші абревіатури: **CAE (**Computer-aided engineering - загальна назва для програм або програмних пакетів, призначених для інженерних розрахунків, аналізу і симуляції фізичних процесів**); CAM – (**Computer Aided Manufacturing, використовуються у назві програм для

керування виробництвом та рухом матеріалів). В Радянському Союзі стала поширеною абревіатура **САПР**, тобто система автоматизованого проектування.

У теперішній час у світі розроблено багато систем автоматичного проектування та дослідження. Деякі з них мають універсальне призначення, наприклад **MathCAD, AutoCAD, MatLab, Microsoft Excel, Microsoft Access, LABVIEW.** Зараз набули поширення програми конструювання із використанням об'ємного креслення (так звані 3D - системи, які працюють у трьохвимірній Декартовій системі координат), програми з використанням **анімації**, тобто придання властивостей руху різним об'єктам, деталям.

Є також багато прикладних програм більш вузького, галузевого призначення, наприклад програми для креслення та читання електричних та електронних схем, для розробки друкованих плат.

Електротехнічні фірми (спочатку світові гіганти, а далі практично всі) створюють власне програмне забезпечення для автоматизованого проектування систем автоматизації промислового обладнання із використанням електричних машин, напівпровідникових перетворювачів, контактних апаратів та елементної бази виробництва саме цієї фірми.

Із розповсюдженням мікропроцесорних та мікроконтролерних систем керування функції механізмів та апаратів стали настільки складними, що можуть бути спроектовані, проконтрольовані і введені в експлуатацію тільки за допомогою комп'ютерів. Зараз будь - яка електронна техніка, починаючи від побутового до промислового призначення, має фірмовий програмний супровід, розташований безпосередньо у самому приладі або на зовнішньому носії інформації.

ЛЕКЦІЯ 2 КОРОТКИЙ ОГЛЯД ПРОГРАМНИХ ПАКЕТІВ ДЛЯ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ

2.1 Програмні пакети універсального призначення

Є цілий ряд більш – менш універсальних програм для автоматизації обчислювальних, проектних та дослідницьких робіт, які набули популярності у цілому світі і кількість користувачів яких налічує мільйони. Нижче наведений короткий огляд декількох таких

програм, які можуть бути корисними для автоматизації проектних робіт і для навчального процесу.

AUTOCAD (англ. Computer-Aided Design) — двух- і тривимірна система автоматизированного проектирования і креслення, розроблена компанією Autodesk починаючи з 1982 р. AUTOCAD є одною з найбільш розповсюджених САПР в світі, завдяки засобам креслення. Не дивлячись на деяку складність, завдяки вдало зробленому інтерфейсу, AUTOCAD є досить простій у використанні. Ранні версії AUTOCAD оперували елементарними об'єктами, такими як круги, лінії, дуги і інші, з яких складалися складніші об'єкти. Проте на сучасному етапі програма включає повний набір засобів, що забезпечують комплексне тривимірне моделювання, зокрема роботу з довільними формами, створення і редагування 3D-моделей тіл і поверхонь, покращену 3D- навігацію і ефективні засоби випуску робочої документації. Починаючи з версії 2010, в AUTOCAD реалізована підтримка параметричного креслення, тобто можливість накладати на об'єкт геометричні або розмірні залежності. Це гарантує, що при внесенні будь-яких змін до проекту, певні параметри і раніше встановлені між об'єктами зв'язки зберігаються. Інтерфейс користувача підтримує можливість настройки під потреби конкретної галузі. Засоби розробки і адаптації дозволяють створювати спеціалізовані застосування, такі як AUTOCAD Mechanical, AUTOCAD Electric, AUTOCAD Architecture і багато інших. Всього Autodesk зараз випускає порядка ста програмних продуктів.

Maple - програмний пакет - **система комп'ютерної алгебри**. Є продуктом компанії Waterloo Maple Inc., яка з 1984 року випускає програмні продукти, орієнтовані на складні математичні обчислення, візуалізацію даних і моделювання. Система Maple призначена для **символьних обчислень,** хоча має ряд засобів і для чисельного вирішення диференційних рівнянь і знаходження інтегралів. Особливої підготовки для набору формул не потрібно - формули матимуть звичний, аналогічний книжному, або письмовому вигляд. Обчислення з введеними формулами здійснюються за бажанням користувача миттєво, одночасно з набором, або по команді. Звичайні формули обчислюються зліва направо і зверху вниз (подібно до читання тексту). Пакет володіє розвиненими графічними засобами. Має власну мову програмування. **Maple V** є одним з найбільш могутніх математичних пакетів. Його можливості охоплюють достатньо багато розділів математики і можуть з користю застосовуватися на різних рівнях, від вивчення університетського курсу математики до серйозних наукових досліджень. Працювати з ним можна як в режимі інтерактивного діалогу, так і шляхом складання і налагодження програм на спеціальній Maple-мові, орієнтованій на складні математичні обчислення. Основу пакету складає спеціальне ядро -

програма символьних перетворень. Крім того, є декілька тисяч спеціальних функцій, що зберігаються в підвантажуваних до ядра пакетах і бібліотеках. Maple уміє не тільки обчислювати, але і володіє багатими можливостями графічного представлення математичних об'єктів і процесів. Програма корисна не тільки для фахівців, але і для студентів при вивченні розділів вищої математики.

Mathcad — система комп'ютерної алгебри з класу систем автоматизованого проектування орієнтована на підготовку інтерактивних документів з обчисленнями і візуальним супроводом, відрізняється легкістю використання і застосування для колективної роботи. Mathcad має простий і інтуїтивний для використання інтерфейс користувача. Для введення формул і даних можна використовувати як клавіатуру, так і спеціальні панелі інструментів. Містить сотні операторів і вбудованих функцій для вирішення різних технічних завдань. Програма дозволяє виконувати чисельні і символьні обчислення, проводити операції із скалярними величинами векторами і матрицями, автоматично переводити одні одиниці вимірювання до інших. Деякі з математичних можливостей Mathcad (версії до 13.1 включно) засновані на підмножині системи комп'ютерної алгебри Maple. Починаючи з 14 версії — використовує символьне ядро MuPAD. Робота здійснюється в межах робочого листа, на якому рівняння і вирази відображаються графічно, на противагу текстового запису в мовах програмування. При створенні документів-додатків використовується принцип WYSIWYG (What You See Is What You Get — «що бачиш, то і отримуєш»). Не дивлячись на те, що ця програма в основному орієнтована на користувачів-непрограмістів, Mathcad також використовується в складних проектах, щоб візуалізувати результати математичного моделювання шляхом використання розподілених обчислень і традиційних мов програмування. Також Mathcad часто використовується в крупних інженерних проектах, де велике значення має трасуємість і відповідність стандартам. Mathcad достатньо зручно використовувати для навчання, обчислень і інженерних розрахунків. Відкрита архітектура додатку у поєднанні з підтримкою технологій NET і XML дозволяють легко інтегрувати Mathcad практично в будь-які IT-структури і інженерні застосування. Є можливість створення електронних книг (e-Book). За допомогою Mathcad інженери можуть документувати всі обчислення в процесі їх проведення.

Mathematica система комп’ютерної алгебри компанії Wolfram Research. Містить множину функцій як для аналітичних перетворень, так і для чисельних розрахунків, яких може не опинитися в інших пакетах програм. Крім того, програма підтримує роботу з графікою і звуком, включаючи побудову двух- і тривимірних графиків функцій, малювання довільних геометричних фигур, імпорт і экспорт зображень і звуку.

MATLAB (англ. «Matrix Laboratory») термін, що відноситься до пакету прикладних програм для вирішення завдань технічних обчислень, а також до використовуваної в цьому пакеті мови програмування. **MATLAB** працює на більшості сучасних операційних систем, включаючи Linux, Mac OS і Microsoft Windows. MATLAB, як мова програмування, була розроблена в кінці 1970-х років з метою полегшення процесів програмування для студентів (мова розділу Simulink отримала назву візуального проектування). Нова мова була з великим інтересом зустрінута вченими, що працюють в області прикладної математики. Вдосконалений варіант MATLAB на мові C з'явився в 1984 р. Спочатку MATLAB призначався для проектування систем управління, але швидко завоював популярність в багатьох інших наукових і інженерних областях. Він також широко використовувався і в освіті, зокрема, для викладання лінійної алгебри і чисельних методів. Мова MATLAB є високорівневою інтерпретуємою мовою програмування, що включає засновані на матрицях структури даних, широкий спектр функцій, інтегроване середовище розробки, об'єктно-орієнтовані можливості і інтерфейси до програм, написаних на інших мовах програмування. Основною особливістю мови MATLAB є його широкі можливості по роботі з матрицями, які творці мови виразили в гаслі «думай векторно» (англ. Think vectorized). MATLAB надає користувачеві велику кількість (декілька сотень) функцій для аналізу даних, що покривають практично всі області математики. MATLAB надає зручні засоби для розробки алгоритмів, включаючи високорівневі з використанням концепцій об’єктно-орієнтованого програмування. У ньому є всі необхідні засоби інтегрованого середовища розробки, включаючи отладчик і профайлер. Функції для роботи з цілими типами даних полегшують створення алгоритмів для мікро контролерів і інших застосувань, де це необхідно. У складі пакету MATLAB є велика кількість функцій для побудови графіків, зокрема тривимірних, візуального аналізу даних і створення анімованих роликів. Вбудоване середовище розробки дозволяє створювати графічні інтерфейси користувача з різними елементами управління, такими як кнопки, поля введення і іншими. За допомогою компоненту MATLAB Compiler ці графічні інтерфейси можуть бути перетворені в самостійні застосування, для запуску яких на інших комп'ютерах необхідна бути встановлена бібліотека MATLAB Component Runtime. Пакет MATLAB містить функції, які дозволяють йому діставати доступ до інших додатків середовища Windows так само, як і цим застосуванням діставати доступ до даних MATLAB, за допомогою технології динамічного обміну даними (DDE). Інтерфейс для послідовного порту пакету MATLAB забезпечує прямий доступ до периферійних пристроїв, таким як модеми, принтери і наукове устаткування, що підключається до комп'ютера через послідовний порт (COM - порт). Інтерфейс працює шляхом створення об’єкту спеціального класу для послідовного порту. Наявні методи цього класу дозволяють читати і записувати

дані в послідовний порт, використовувати події і обробники подій, а також записувати інформацію на диск комп'ютера в режимі реального часу. Це буває необхідно при проведенні експериментів, симуляції систем реального часу і для інших застосувань. Для MATLAB є можливість створювати спеціальні набори інструментів (англ. toolbox), що розширюють його функціональність. Наборами інструментів є колекції функцій, написаних на мові MATLAB для вирішення певного класу завдань. Компанія Mathworks поставляє набори інструментів, які використовуються в багатьох областях.

LABVIEW (англ. Laboratory Virtual Instrumentation Engineering Workbench**)** це розробки і платформа для виконання програм, створених на графічній мові програмування «G» фірми National Instruments (США). Перша версія LABVIEW була випущена в 1986 році для Apple Macintosh. В даний час існують версії для UNIX Linux, Mac OS і інші, а найбільш розвиненими і популярними є версії для Microsoft Windows. Графічна мова програмування «G», використовувана в LABVIEW, заснована на архітектурі потоків даних. Послідовність виконання операторів в таких мовах визначається не порядком їх проходження (як в імперативних мовах програмування), а наявністю даних на входах цих операторів. Оператори, не зв'язані по даним, виконуються паралельно в довільному порядку. Тому LABVIEW використовується в системах збору і обробки даних, а також для управління технічними об'єктами і технологічними процесами. Програма LABVIEW називається і є **віртуальним приладом** (англ. Virtual Instrument) і складається з двох частин: блокової діаграми, що описує логіку роботи віртуального приладу і лицьової панелі, що описує зовнішній інтерфейс віртуального приладу. Віртуальні прилади можуть використовуватися як складові частини для побудови інших віртуальних приладів. Лицьова панель віртуального приладу містить засоби введення-виводу: кнопки, перемикачі, світло діоди, верньєри, шкали, інформаційні табло і т. п. Вони використовуються людиною для управління віртуальним приладом, а також іншими віртуальними приладами для обміну даними. Блокова діаграма містить функціональні вузли, що є джерелами, приймачами і засобами обробки даних. Також компонентами блокової діаграми є термінали («задні контакти» об'єктів лицьової панелі) і структури, що управляють (що є аналогами таких елементів текстових мов програмування, як умовний оператор «IF», оператори циклу «FOR» і «WHILE» і т. п.). Функціональні вузли і термінали об'єднані в єдину схему лініями зв'язків. LABVIEW підтримує величезний спектр устаткування різних виробників і має в своєму складі (або дозволяє додавати до базового пакету) численні бібліотеки компонентів, що реалізує взаємодію LabVIEW-программы на звичайному персональному комп'ютері з реальними об'єктами, що працюють в реальному

часі. LABVIEW може вдало використовуватися в навчальному процесі у якості віртуальних учбово-дослідницьких стендів, сполучених з реальним устаткуванням.

2.2 Програмні пакети спеціального (галузевого) призначення.

Програми цього виду отримали загальну назву спеціалізованих **САПР – платформ**.

1. nanoCAD САПР. САПР-платформа для різних галузей nanoCAD містить всі необхідні інструменти базового проектування (2D). Пряма підтримка DWG забезпечує сумісність з іншими САПР - рішеннями.

1.1 nanoCAD СПДС Оформлення проектно-конструкторської документації відповідно до стандартів СПДС. Містить функціонал nanoCAD для робіт із створення двовимірних креслень. Вихідна документація зберігається у форматі DWG.

1.2 nanoCAD СКС. Автоматизоване проектування структурованих кабельних систем (СКС) будівель і споруд різного призначення. nanoCAD СКС є робочим інструментом для проектування систем кабельної каналізації і структурованих кабельних систем, телефонії.

1.3 nanoCAD Электро. Автоматизоване виконання проектів в частинах силового електроустаткування (ЕМ) і внутрішнього електроосвітлення (ЕО) промислових і цивільних об'єктів будівництва.

1.4 nanoCAD ЭлектроПроект. Виконання проектів електроустаткування виробів загального машинобудування, приладобудування, верстатобудування, залізничного рухомого складу, може застосовуватися для проектування електроустаткування в енергетиці.

1.5 nanoCAD Схемы. Автоматизована побудова схем в наступних областях проектування промислових і цивільних об'єктів: електротехніка, КипіА, технологічне проектування, а також в інших областях, що вимагають побудови схем.

2. Altium Designer. САПР платформа — комплексна система автоматизованого проектування (САПР) радіоелектронних засобів, розроблена австралійською компанією Altium.

2.1 P-CAD система автоматизованого проектування електроніки. Призначена для проектування багатошарових друкованих плат обчислювальних і радіоелектронних пристроїв. Остання версія системи P-CAD 2006 SP2. У 2006 році компанія Altium офіційно

заявила про припинення розробки даного продукту. Для заміни цієї системи компанія Altium пропонує систему Altium Designer.

2.2 Altium Designer WINTER 09. Це могутня система, що дозволяє реалізовувати проекти електронних засобів на рівні схеми або програмного коду з подальшою передачею інформації в ПЛИС або друковану плату. Відмітною особливістю програми є проектна структура і наскрізна цілісність ведення розробки на різних рівнях проектування. Іншими словами зміни в розробці на рівні плати можуть миттєво бути передані на рівень ПЛИС або схеми і також назад. Бібліотеки програми містять більше 90 тисяч готових компонентів, у багатьох з яких є моделі посадочних місць SPICE і IBIS-моделі, а також тривимірні моделі.

3. EPLAN Software. САПР платформа**.** EPLAN відноситься до провідних фірм по розробці програмного забезпечення систем автоматизованого проектування для галузевих рішень. EPLAN пропонує високоякісні рішення для таких дисциплін як: електротехніка, гідравліка, пневматика, КипіА і механика.

3.1 EPLAN Electric P8. Глобальний стандарт в електротехнічному проектуванні. Завдяки вільному вибору інструментів і об'єктів, а також технічній варіативності і використанню реверсивних технологій EPLAN Electric P8 дає можливість успішної реалізації комплексних глобальних проектів.

3.2 EPLAN Fluid. Програмне забезпечення для проектування пневмогідравлічних рішень змащення, системи охолоджування і кондиціювання і автоматичного створення відповідної проектної документації.

3.3 EPLAN Cabinet. 3D проектування електротехнічних шаф з передачею даних у виробництво. Віртуальне тривимірне моделювання, створення двух- і трьох- мірних креслень, тривимірне зображення дротяних і маршрутних схем, наявність шаблонів для роботи свердлувального устаткування і інтеграція з верстатами ЧПУ.

3.4 EPLAN PPE. Програмне забезпечення для проектування складних розподілених систем і польового устаткування АСУ ТП

3.5 EPLAN Engineering Center. У цій програмі комплексні системи діляться на мехатронні, функціональні елементи, при цьому дотримуються всі дисципліни і зберігається систематичність даних.

4. Electronic Design Automation. (EDA, автоматизація проектування електронних приладів). Комплекс програмних засобів для полегшення розробки електронних пристроїв, створення мікросхем і друкованих плат. У комплекс входять програми P-cad, OrCAD, Electric, Proteus, KiCad, gEDA, TopoR.

4.1 P-cad, див. п. 2.1

4.2 ORCAD. Пакет комп'ютерных програм, призначених для автоматизації проектування електроніки. Використовується в основному для створення електронних версій друкованих плат для їх виробництва, а так само для виробництва електронних схем і їх моделювання.

4.3 Electric VLSI Design System. САПР, використовувана для розробки електричних схем і проектування топології друкованих плат і інтегральних схем. VLSI система автоматизованого проектування надвеликих інтегральних схем (СБИС). За допомогою Electric можна розробляти інтегральні МОП і біполярні схеми, друковані плати або схеми будь-якого типу. Electric об'єднує в собі безліч різних синтетичних тестів і аналізуючих інструментів.

4.4 PROTEUS VSM. Пакет програм для автоматизованого проектування (САПР) електронних схем. Розробка компанії Labcenter Electronics (Великобритания). Пакет є системою моделювання схемотехніки, що базується на основі моделей електронних компонентів прийнятих в PSpice. Відмінною рисою пакету PROTEUS VSM є можливість моделювання роботи програмованих пристроїв: мікроконтролерів, мікропроцесорів, DSP і інш. Бібліотека компонентів містить довідкові дані. Додатково в пакет PROTEUS VSM входить система проектування друкованих плат.

4.5 KiCad — поширюваний за ліцензією GNU General Public License програмний комплекс класу EDA з відкритими початковими текстами, призначений для розробки електричних схем і друкованих плат.

4.6 gEDA Набір програмного забезпечення для проектування електронних пристроїв (САПР), поширюваний за ліцензією GPL. Включає інструменти для редагування електричних схем, симуляції цифрових і аналогових схем, трасування друкованих плат і підготовки до виробництва.

4.7 TOPOR. (Topological Router) російська система автоматизованого проектування (САПР), призначена для проектування друкованих плат, заздалегідь підготовлених в інших

системах у форматах PCAD, ASCII, PCB, PADS. ASCII, PCB або DSN. У TOPOR є автоматичне розміщення компонентів. Процедура може застосовуватися як до всіх компонентів на платі, так і до компонентів в певному вікні. Система автоматично зменшує ширину провідника, якщо він підходить до контакту, що має меншу ширину (або діаметр контакту менше ширини провідника), і при проході через вузькі місця (наприклад, між контактами компоненту). Є можливість каплеподібного згладжування стиків провідників з контактними майданчиками (teardrops).

2.3. Корпоративні програмні пакети

Цю назву укладач конспекту використовує умовно для позначення спеціального програмного забезпечення продукції, що випускається різними фірмами, а також для проектування електромеханічних систем та систем автоматизації із використанням своєї продукції. Як приклад, назвемо деякі програми німецької корпорації Moeller, обладнанням якої оснащені навчальні лабораторії кафедри електромеханічних систем з комп'ютерним управлінням ОНПУ.

Для проектування систем автоматизації на базі програмованих логічних контролерів різного рівня: PLC Programming, XSoft Professional, XSystem, Sucosoft S40.

Для проектування систем автоматизації на базі електронних програмованих реле типів EASY, MFD-Titan, EASY-Control: easy Soft, easy Soft-CoDeSys. Програмовані реле мають реальні контактні входи та релейні або транзисторі виходи. Все інше виконується у програмній формі. Програмування цих реле є яскравим прикладом візуального проектування. Всі операції – логічні, арифметичні, апаратні програмуються у формі електричної схеми із наочним використанням контактів і котушок. В потрібних випадках контактам та котушкам призначаються певні властивості та характеристики. Після закінчення проектування схема проходить випробування (симуляцію) і автоматично компілюється у програмну форму, що виконується далі.

Для проектування електричних шаф (ввід, ошиновка, контактори, реле, автоматичні вимикачі, устрої вводу - виводу): XEnergy Configurator. При проектуванні використовується

база даних продукції фірми Moeller із автоматичним використанням габаритних та установочних розмірів для кожного елементу обладнання.

Відповідні програми є для промислових систем передачі та обміну інформацією, візуалізації, для перетворювачів частоти із аналоговим та векторним керуванням, для пристроїв плавного пуску асинхронних двигунів (soft - стартерів).

2.4 АРМ – автоматизоване робоче місце.

У широкому сенсі це комп'ютеризоване робоче місце інженера, економіста, управлінця, тощо. Спочатку АРМ почали застосовуватися в автоматичних системах управління технологічними процесами. В даний час АРМ широко застосовуються і для автоматизації проектних робіт. У цьому напрямі можна послатися на розробки залізничників (ІМСАТ ЛІЇЖТА, Росія), початі ще в 1994 р. Ними була розроблена ціла серія АРМ: розробки проектно – технічної документації (АРМ-ПТД), веденню технічної документації (АРМ-ВТД) і ін. Зокрема, АРМ-ПТД зараз стає популярною серед проектувальників – електриків, особливо при розробці систем контролю і автоматизації, принципових і монтажних схем, кабельних розводок. Перспективними напрямами в розвитку АРМ є перехід до мережевих режимів роботи, коли великі масиви інформації (довідкові дані, каталоги) зберігаються на серверах; застосування "живих" каталогів, у яких на одній сторінці присутні декілька джерел інформації (характеристики виробів, суміжні товари і вироби, аксесуари і так далі). У АРМ-ПТД передбачена можливість перекладу старих схем і креслень на машинні носії.

ЛЕКЦІЯ 3. ТЕОРЕТИЧНІ ОСНОВИ ТА ЗАСОБИ ЗАБЕЗПЕЧЕННЯ АВТОМАТИЗАЦІЇ ПРОЕКТНИХ РОБІТ

Автоматизоване проектування передбачає активну співпрацю людини і обчислювальної машини – комп’ютера. Людина, із свого життєвого досвіду звикла приймати рішення, сподіваємось, в більшості вірні, в умовах неповної інформації та різного роду перешкод. Машина, навпаки, не терпить невизначеності. Отже, стає актуальною задача формалізації процесів проектування, чітке і однозначне описання окремих дій, зрозуміле для машини. Формалізація починається з розбивки процесу проектування на проектні процедури та проектні операції.

3.1 Проектні процедури і проектні операції. Проектна операція - дія або формалізована сукупність дій, складова частина проектної процедури, алгоритм якої лишається незмінним для ряду проектних процедур.

Проектна процедура формалізована сукупність дій, виконання яких **закінчується прийняттям рішення.**

Проектне рішення - проміжний або кінцевий опис об'єкта, необхідний і достатній для розгляду і визначення подальшого напрямку або закінчення проектування.

Проектна процедура складається з елементарних проектних операцій з чітко встановленим порядком їх виконання і спрямована на досягнення локальної мети в процесі проектування. Проектна процедура починається після отримання технічного завдання і складається із таких проектних операцій:

- отримання або пошук, розрахунок вхідних даних (початкові структура та параметри системи);
- вибір та обґрунтування необхідних спрощень та обмежень;
- математичне описання об'єкту або процесу проектування;
- вибір або створення математичної моделі, яка вирішує процедуру;
- аналіз об'єкту при заданих структурах та параметрах;
- отримання проектного рішення;
- при необхідності розробка критеріїв оцінки проектного рішення;
- оцінка проектного рішення;

Проектні процедури ґрунтуються на мовах проектування, які служать засобом лінгвістичного чи графічного представлення і перетворення опису при проектуванні.

Проектна процедура називається **типовою**, якщо вона призначена для багаторазового використання при проектуванні багатьох типів об'єктів.

Розрізняють **проектні процедури аналізу і синтезу.**

Аналіз полягає у визначенні властивостей та дослідженні працездатності об'єкту по його опису (найпростіший приклад – розрахунки та побудова статичних та динамічних характеристик електроприводу за заданими параметрами), тобто при аналізі надається оцінка об’єкту. **Синтез** полягає у пошуку структури або параметрів, які надають об’єкту потрібних властивостей (приклад – розробка структури та визначення параметрів регуляторів координат електроприводу, які забезпечують необхідні властивості в статичних та динамічних режимах з урахуванням прийнятих обмежень).

Процедури аналізу діляться на **процедури одноваріантного і багатоваріантного аналізу.** При **одноваріантному аналізі** необхідно визначити вихідні параметри об’єкта за

відомими значеннями керуючих та збурюючих дій і відомими значеннями внутрішніх параметрів об’єкту. **Багатоваріантний аналіз** полягає у дослідженні властивостей об’єктів в деякій області простору внутрішніх параметрів або керуючих та збурюючих дій.

Процедури синтезу поділяються на **структурний і параметричний синтез.** Метою структурного синтез*у* є визначення структури об'єкта - переліку типів елементів, що складають об'єкт та способу зв'язку елементів між собою в складі об'єкта. Параметричний синтез полягає у визначенні числових значень параметрів елементів при заданій структурі та виконанні заданих умов на виході об'єкта.

Треба зазначити, що будь – яке проектування починається з структурного синтезу, навіть якщо структура відома, або предписана. Другим кроком є синтез необхідних параметрів. Після цього проводиться вибір методу аналізу (моделі системи), безпосередньо аналіз і приймається рішення про закінчення процедури проектування, якщо отримані результати відповідають технічному завданню. Алгоритм описаних дій показаний на рис. 3.1. Процес проектування не часто може бути виконаним "за один раз", без повторів окремих операцій або цілком процедури, що наочно відображене на рисунку.

Якщо отримане проектне рішення не відповідає технічному завданню, повторюються етапи параметричного синтезу та аналізу. Використовуються методи спочатку одно варіантного, а далі багатоваріантного синтезу параметрів. Далі, якщо параметричний синтез на дав необхідних результатів, або отримані результати виходять за рамки заданих обмежень, приступають до корекції структури системи, тобто повторюють операцію структурного синтезу (блок "Корекція структури" на схемі). Після корекції структури повторюються операції параметричного синтезу та аналізу до отримання позитивного проектного рішення.

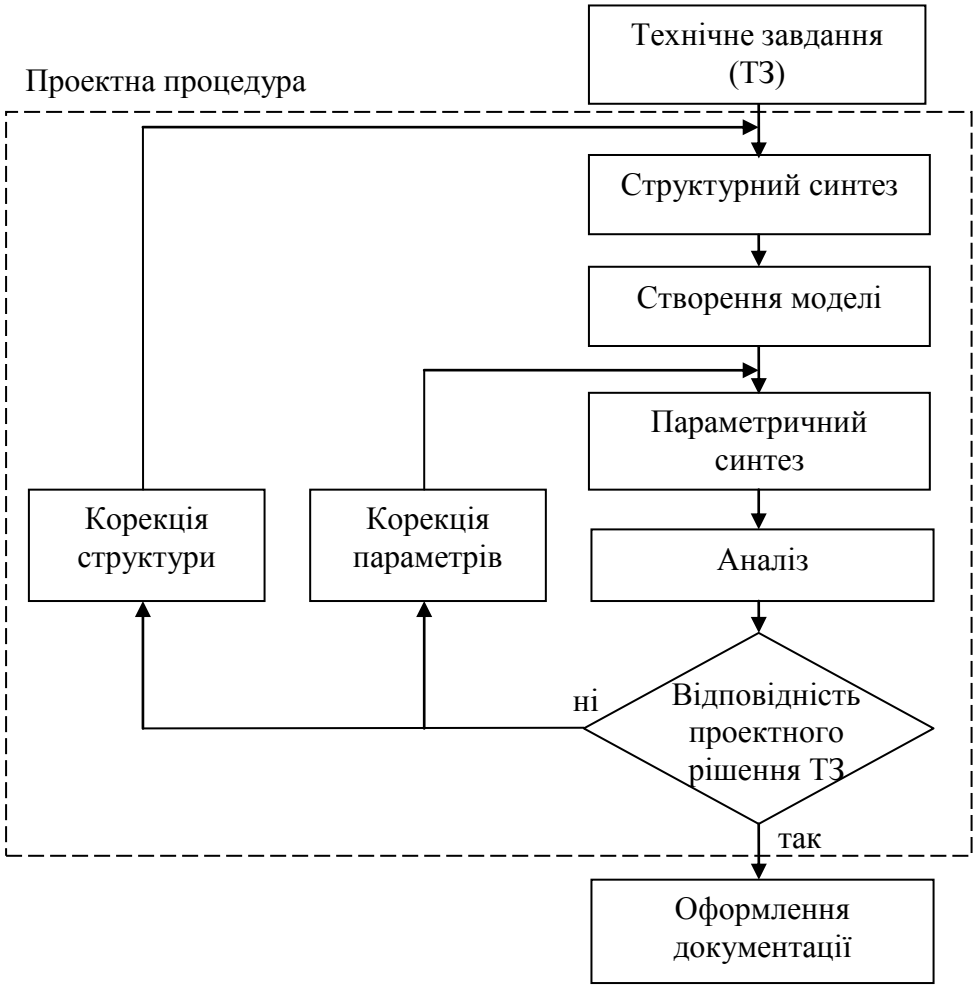


Рис. 3.1 Схема алгоритму типової проектної процедури

В проектній практиці можливі складні випадки, коли а ні корекція структури, а ні корекція параметрів не дають позитивних результатів. Тут можливі висновки, на перший погляд прикрі для замовника та виконавця. Можливі варіанти: завдання надто складне для виконання відомими сучасними методами; завдання принципово може бути виконане, але науково – технічний рівень виконавця не дозволяє це зробити. Виводи і можливі шляхи вирішення досить очевидні: скоректувати технічне завдання (а це зниження загального рівня майбутньої техніки і її конкурентноздатності); зміна виконавця (а це втрата часу і фінансових ресурсів); проведення додаткових науково – дослідних робіт.

Останній шлях найбільш конструктивний, хоча краще це рішення приймати заздалегідь, на рівні розробки концепції та технічного завдання. Чому висновки прикрі "на перший погляд"? Тому, що описана ситуація теж типова, і краще зробити крок назад та створити систему на рівні кращих світових зразків, ніж продовжувати втілення неконкурентних технічних рішень.

3.2 Евристичні і систематичні рішення

Описана вище "тупикова" ситуація може бути подолана при підключенні методів рішення творчих задач. При проектуванні технічних об'єктів ці методи поділяються на:

- систематичні,
- евристичні.

Систематичні рішення отримують в результаті використання методів, «що стимулюють творчу діяльність» (алгоритм рішення винахідницьких задач, метод асоціацій, метафор, інверсії та ін.). Вони ґрунтуються на усвідомленому процесі пошуку і рішення задачі в результаті впорядкованого мислення і застосуванні методів його активації. Методи стимулювання творчої діяльності ґрунтуються на логіці і використовують раніше визначену послідовність дій і операцій (технологію проектування).

Евристичні рішення отримують в результаті такого проектування, коли важлива частина творчого процесу і отримання творчого результату проходить в голові людини і не може бути отримана з попереднього досвіду. Евристичні рішення базуються на застосуванні евристичних методів.

Евристичні методи це послідовність наказів або процедур обробки інформації, що виконується з метою пошуку більш раціонального і конструктивного рішення. Для такої послідовності немає обґрунтованого доведення і немає гарантій отримання найкращого рішення. Евристичні процедури називають евристиками або еврістами, вони спрямовані на рішення задач в умовах дефіциту інформації або часу.

3.3 Види забезпечення САПР

Засоби автоматизації проектування можна згрупувати по видах забезпечення автоматизованого проектування:

- математичне;
- програмне;
- інформаційне;
- технічне;
- лінгвістичне;
- методичне;
- організаційне.

3.3.1 Математичне забезпечення. Основу математичного забезпечення (МЗ) САПР складають алгоритми, по яких розробляється програмне забезпечення САПР. Елементи математичного забезпечення в САПР надзвичайно різноманітні.

Серед них є інваріантні елементи — принципи побудови функціональних моделей, методи чисельного вирішення рівнянь алгебри і диференціальних, постановки екстремальних завдань, пошуки екстремуму. Розробка математичного забезпечення є найскладнішим етапом створення САПР, від якого найбільшою мірою залежать продуктивність і ефективність функціонування САПР в цілому.

За призначенням і способам реалізації МЗ САПР ділиться на дві частини:

- математичні методи і побудовані на їх основі математичні моделі, що описують об'єкти проектування;
- формалізований опис технології автоматизованого проектування.

Способи і засоби реалізації першої частини математичного забезпечення найбільш специфічні в різних САПР і залежать від особливостей об'єктів проектування.

Що стосується другої частини математичного забезпечення, то формалізація процесів автоматизованого проектування в комплексі виявилася складнішим завданням, ніж алгоритмізація і програмування окремих проектних завдань. При рішенні цієї задачі повинні бути формалізована вся логіка технології проектування, зокрема логіка взаємодії проектувальників один з одним на основі використання засобів автоматизації.

Математичне забезпечення САПР повинне описувати у взаємозв'язку об'єкт, процес і засоби автоматизації проектування.

Перспективною у вдосконаленні і типізації технології процесів автоматизованого проектування є централізована розробка математичного апарату моделювання типового процесу проектування і випуск базових програмно-методичних комплексів, що реалізовують такі моделі.

3.3.2 Програмне забезпечення САПР. Програмним забезпеченням (ПЗ) САПР є сукупність всіх програм і експлуатаційної документації до них, необхідних для виконання автоматизованого проектування. Програмне забезпечення ділиться на **загальносистемне і спеціальне (прикладне).**

Загальносистемне ПЗ призначено для організації функціонування технічних засобів, тобто для планування і управління обчислювальним процесом, розподілу наявних ресурсів, і представлено операційними системами ЕОМ і обчислювальних комплексів (ОК). Загальносистемне ПЗ зазвичай створюється для багатьох застосувань і специфіку САПР не відображає.

У спеціальному (прикладному) ПЗ реалізується математичне забезпечення для безпосереднього виконання проектних процедур. Прикладне ПЗ зазвичай має форму пакетів

прикладних програм (ППП), кожен з яких обслуговує певний етап процесу проектування або групу однотипних завдань усередині різних етапів.

Розглянемо принципові особливості ПЗ, що впливають на організацію і ефективність створення і використання САПР. З розвитком і вдосконаленням ЕОМ все більшого значення набуває такий компонент загальносистемного ПЗ, як операційні системи (ОС). Можливості, що надаються користувачам сучасними обчислювальним системами, більшою мірою визначаються їх операційними системами, чим технічними пристроями. ОС організовує одночасне вирішення різних завдань на ЕОМ, динамічний розподіл каналів передачі даних і зовнішніх пристроїв між завданнями, планування потоків завдань і послідовність їх рішення з урахуванням встановлених критеріїв, динамічний розподіл пам'яті обчислювального комплексу. Проте ОС вимагає для своєї роботи певних ресурсів: процесора, зовнішньої і основної пам'яті. Чим великими можливостями володіє ОС, тим більше ресурсів потрібно для неї.

Важливим компонентом загальносистемного ПЗ є базове ПЗ. Базове ПЗ не є об'єктом розробки при створенні програмного забезпечення САПР. Прикладом може служити базове ПО для обробки геометричної і графічної інформації, для формування і використання баз даних (БД).

Використання АРМ, до складу яких включене подібне базове ПЗ, таке, що реалізовує стандартні проектні процедури, істотно понизить трудомісткість створення програмного забезпечення САПР. Проте у всіх випадках за творцями САПР залишиться розробка прикладного ПЗ. З розширенням області застосування обчислювальної техніки і ускладненням завдань автоматизації процесів проектування зростають складність і трудомісткість програмування.

Останнім часом, особливо у зв'язку з широким впровадженням в інженерну практику персональних комп'ютерів, починають використовуватися функціональні і інтегровані пакети програм.

Функціональні пакети програм (ФПП) - це комплекс програмних засобів, орієнтованих на виконання певної функції, більш менш безвідносно до конкретного наочного змісту (обробка текстів - текстові редактори, обробка таблиць, графіки).

Інтегровані пакети програм (ІПП) - це поєднання різних пакетів програм в єдиній технологічній системі.

Інтеграція може бути реалізована з'єднанням основних функціональних пакетів в цілісну монолітну систему, представлену єдиним програмним модулем, або шляхом

створення набору допоміжних засобів інтерфейсного характеру для забезпечення взаємодії пакетів, представлених незалежними модулями.

Прикладне програмне забезпечення повинне задовольняти наступним основним вимогам:

- Правильність - це функціонування відповідно до модельованого об'єкту і повного збігу з вибраним алгоритмом рішення.
- Точність - результати розрахунку мають допустимі відхилення від реальних.
- Сумісність - можливість роботи не тільки в автономному режимі, але і у складі інтегрованих систем, САПР і ін.
- Надійність - за всіх умов забезпечує повторюваність результату.
- Універсальність - робота при будь-яких допустимих початкових даних.
- Захищеність - збереження працездатності при виникненні збоїв.
- Корисність - практична цінність вирішуваних завдань.
- Ефективність - необхідні ресурси (пам'ять, час) невеликі.
- Перевіряємость - можливість демонстрації якості на практиці.
- Можливість адаптування - можливість швидкої модифікації з метою пристосування до умов функціонування, що змінюються.

Останнім часом загальне визнання отримав модульний принцип побудови програмного забезпечення, програм.

Програми доцільно розбивати на модулі, для того, щоб спростити їх розробку і реалізацію; полегшити сприйняття програми; спростити їх відладку і модифікацію; полегшити роботу з даними, що мають складну структуру; уникнути надмірної деталізації алгоритмів; забезпечити вигідніше розміщення програм в пам'яті комп'ютера.

Кожен модуль зазвичай є самостійною програмою, призначеною для розрахунку окремих компонентів, систем або що реалізовує один з методів розрахунку або окрему його процедуру.

Модуль повинен бути, як правило, незалежним від даних варіантів об'єкту, процесу, системи (структури, режиму функціонування і ін.). Він повинен бути тим елементом (компонентом), за допомогою якого можна описати будь-який варіант об'єкту, процесу, системи.

Наявність таких модулів дозволяє звести до мінімуму процес додаткового програмування, скоротити час по підготовці користувача до роботи. В процесі реальної роботи з таким програмним забезпеченням проводиться постійна заміна самих модулів або

окремих компонент, відповідних тому або іншому варіанту об'єкту, процесу, системи, з максимальною уніфікацією введення і виведення даних.

3.3.3 Інформаційне забезпечення САПР. Основу інформаційного забезпечення (ІЗ) САПР складають дані, якими користуються проектувальники в процесі проектування безпосередньо для вироблення проектних рішень. Ці дані можуть бути представлені у вигляді тих або інших документів на різних носіях, що містять відомості довідкового характеру про матеріали, комплектуючі вироби, типові проектні рішення, параметри елементів, зведення про стан поточних розробок у вигляді проміжних і остаточних проектних рішень, структур і параметрів проектованих об'єктів і тому подібне.

При цьому дані, що є результатом одного процесу перетворення, можуть бути початковими для іншого процесу. Сукупність даних, використовуваних всіма компонентами САПР, складає інформаційний фонд САПР. Основна функція ІЗ САПР — ведення інформаційного фонду, тобто забезпечення створення, підтримки і організації доступу до даним. Таким чином, ІЗ САПР є сукупність інформаційного фонду і засобів його ведення.

До складу інформаційного фонду САПР входять*:*

- програмні модулі, які зберігаються у вигляді символічних і об'єктних текстів. Як правило, ці дані мало змінюються протягом життєвого циклу САПР, мають фіксовані розміри і з'являються на етапі створення інформаційного фонду. Споживачами цих даних є монітори різних підсистем САПР;
- початкові і результуючі дані, які необхідні при виконанні програмних модулів в процесі перетворення. Ці дані часто міняються в процесі проектування, проте їх тип постійний і повністю визначається відповідним програмним модулем. При організації проміжних даних можливі конфліктні ситуації в процесі узгодження між собою даних різних типів;
- нормативно-довідкова проектна документація (НДПД), що включає довідкові дані про матеріали, елементи схем, уніфіковані вузли і конструкції. Ці дані, як правило, добре структуровані і можуть бути віднесені до фактографічних. До НДПД відносяться також державні і галузеві стандарти, керівні матеріали і вказівки, типові проектні рішення, що регламентують документи (слабо структуровані документальні дані);
- поточна проектна інформація, що відображає стан і хід виконання проекту. Як правило, ця інформація слабо структурована, часто змінюється в процесі проектування і представляється у формі текстових документів.

При виборі способів ведення інформаційного фонду САПР важливо сформулювати принципи і визначити засоби ведення інформаційного фонду, структуризації даних, вибрати способи управління масивами даних.

Розрізняють наступні способи ведення інформаційного фонду САПР: використання файлової системи; побудова бібліотек; використання банків даних; створення інформаційних програм адаптерів.

Використання файлової системи і побудова бібліотек широко поширене в організації ІЗ обчислювальних систем, оскільки підтримується засобами ОС. У додатках до САПР ці способи застосовують при зберіганні програмних модулів в символічних і об'єктних кодах, діалогових сценаріїв підтримки процесу проектування, початкового введення крупних масивів початкових даних, зберігання текстових документів. Проте вони малопридатні при забезпеченні швидкого доступу до довідкових даних, зберіганні змінних даних, веденні поточної проектної документації, пошуку необхідних текстових документів організації взаємодії між різномовними модулями.

Автоматизовані бази даних є сукупністю баз даних (БД) і систем управління базами даних (СУБД). **База даних** — це спеціальним чином організована сукупність даних і їх описів. Система управління базами даних — це програмний комплекс, що реалізовує функції створення бази даних, її оновлення, зберігання, захисту і вибірки даних.

Основні функції СУБД: створення схеми БД; організація зберігання даних; захист цілісності БД; управління доступом до БД шляхом розмежування доступу; надання користувачам доступу до БД; підтримка завантаження БД і технологічних процесів їх функціонування. Для здійснення цих функцій СУБД повинна мати власне ПЗ, таке, що включає різні компоненти. Для зберігання даних документального типу в САПР використовують СУБД типу інформаційно-пошукових систем.

3.3.4 Технічне забезпечення САПР. Створення і використання ЕОМ дозволило значно понизити трудомісткість і тривалість обчислювальних робіт. Але в процесі проектування виробів і технологічних процесів частка власне обчислювальних робіт не перевищує 15%. Автоматизація проектування зажадала випуску спеціалізованих засобів САПР. Технічним забезпеченням САПР є сукупність взаємозв'язаних і взаємодіючих технічних засобів, призначених для виконання автоматизованого проектування.

Будь-які обчислювальні комплекти САПР, у тому числі і АРМ, повинні включати необхідне число периферійних пристроїв для введення і відображення інформації, зокрема графічні і алфавітно-цифрові дисплеї (ГД і АЦД) з графічними планшетами і електронним

пір'ям, високоточні рулонні і планшетні графічні пристрої різного формату, кодувальники графічної інформації, пристрої зняття твердої копії із зображення на екрані дисплея, інтелектуальні відео термінали з растровими кольоровими дисплеями, накопичувачі на переносних магнітних і оптичних дисках, накопичувачі на дисках типу «Вінчестер», функціональні клавіатури, пристрої виведення інформації на мікрофільми і мікрофіші, пристрої зв'язку з ЕОМ верхнього рівня і модеми.

Останнім часом все більше застосування в САПР знаходять персональні ЕОМ (ПЕВМ), потужності яких можуть нарощуватися залежно від складності вирішуваних завдань. У перспективі передбачається перехід до магістрально-модульної архітектури АРМ, що припускає, зокрема, стандартизацію апаратно-програмних інтерфейсів, в модифікаціях АРМ буде використано стандартна конструктивна база, побудована на міжнародних стандартах.

3.3.5 Лінгвістичне забезпечення САПР. Основу лінгвістичного забезпечення САПР складають спеціальні мовні засоби (мови проектування), призначені для опису процедур автоматизованого проектування і проектних рішень. Основна частина лінгвістичного забезпечення — мови спілкування людини з ЕОМ.

Проблемно-орієнтовані мови (ПОМ) проектування аналогічні алгоритмічним мовам програмування (ФОРТРАН, ПАСКАЛЬ, СІ, АСЕМБЛЕР і ін.). У одних випадках ПОМ будують таким чином, що опис будь-якого завдання або завдання на її рішення в основному містить оригінальні терміни фізичного і функціонального змісту. Перехід від фізичного і функціонального опису завдання до програм для ЕОМ реалізується далі автоматично за допомогою транслятора. У інших випадках, наприклад при вирішенні геометричних завдань інженерного типу, ПОМ сполучає в собі засоби алгоритмічної мови високого рівня для вирішення обчислювальних математичних завдань і спеціальні мовні засоби моделювання геометричних об'єктів. Транслятор алгоритмічної мови високого рівня доповнений необхідними спеціальними програмами.

Очевидно, що ПОМ хоча і називаються мовами, насправді представляють комплекси лінгвістичних і програмних засобів, які повинні включати наступні засоби: набір термінальних символів ПОМ; інтерпретатор з ПОМ; засоби синтаксичного аналізу; засоби пакетування директив; бібліотеки базових функцій ПОМ; інтерфейс для зв'язку СУБД.

Можливості ПОМ мають виключно важливе значення в автоматизованому проектуванні. Вони не тільки впливають на продуктивність і рівень автоматизації проектування, але і визначають складність і характер робіт проектувальників із засобами

САПР; можуть зробити ці роботи привабливішими або навпаки. У останньому випадку проектувальники будуть явно і неявно протидіяти автоматизації. В даний час в світовій і вітчизняній практиці існують спеціальні методики і програмні засоби, створення ПОМ, що значно скорочують трудомісткість. Зокрема, при розробці образотворчих засобів ПОМ може використовуватися метасистема, що дозволяє на підставі заданої формальної граматики отримувати відповідний програмний інтерпретатор. При розробці програмних модулів бібліотеки базових функцій можуть застосовуватися будь-які алгоритмічні мови високого рівня. Проте створення надмірне великої різноманітності ПОМ утруднить обмін засобами САПР між підприємствами і зажадає навчання великого числа фахівців роботі з декількома мовами. Таким чином, розвиток гнучких виробничих систем вимагає особливо ретельного вирішення питань по складу лінгвістичного забезпечення САПР.

3.3.6 Методичне забезпечення САПР. Під методичним забезпеченням САПР розуміють документи, що регламентують порядок її експлуатації. Причому документи, що відносяться до процесу створення САПР, не входять до складу методичного забезпечення. Документи методичного забезпечення носять в основному інструктивний характер.

3.3.7 Організаційне забезпечення САПР

Стандарти по САПР вимагають виділення як самостійного компоненту організаційного забезпечення, яке включає положення, інструкції, накази, штатні розклади, кваліфікаційні вимоги і інші документи, що регламентують організаційну структуру підрозділів проектної організації і взаємодію підрозділів з комплексом засобів автоматизованого проектування.

ЛЕКЦІЯ 4 ОСНОВНІ ПОНЯТТЯ БАЗ ДАНИХ

4.1 Бази даних і системи управління базами даних

База даних – це організована структура, призначена для зберігання інформації. У сучасних базах даних зберігаються не тільки дані, але і інформація. Це твердження легко пояснити, якщо, наприклад, розглянути базу даних машинобудівної САПР. У ній є всі необхідні відомості про окремі типові вузли, про складальні операції, сумісність окремих модулів і функціонально-закінчені пристрої і так далі. Доступ до цієї бази даних є у достатньо великої кількості співробітників, але серед них навряд чи знайдеться така особа, яка має доступ до всієї бази повністю і при цьому здатна одноосібно вносити до неї довільні зміни. Окрім даних, база містить методи і засоби, що дозволяють кожному із співробітників оперувати тільки з тими даними, які входять в його компетенцію. В результаті взаємодії даних, що містяться в базі, з методами, доступними конкретним співробітникам, утворюється інформація, яку вони споживають і на підставі якої в межах власної компетенції проводять введення і редагування даних.

З поняттям бази даних тісно пов'язано поняття **системи управління базою даних** (СУБД)**.** Це комплекс програмних засобів, призначених для створення структури нової бази, наповнення її вмістом, редагування вмісту і візуалізації інформації. Під **візуалізацією інформації** бази розуміється відбір даних, що відображаються, відповідно до заданого критерію, їх впорядкування, оформлення і подальша видача на пристрої виводу або передачі по каналах зв'язку.

У СУБД опис структури інформації прийнято називати схемою. Залежно від рівня представлення інформації розрізняють наступні типи схем:

- концептуальний (загальне уявлення про інформаційну базу наочної області);
- зовнішній (представлення інформації з боку користувачів або завдань; при великому числі завдань їх уявлення, можуть перетинатися). Зовнішніх схем буває декілька;
- внутрішній (представлення інформації в базі даних, тобто на фізичних носіях - дисках).

Серед перерахованих рівнів представлення інформації концептуальний рівень займає особливе місце. Він пов'язує зовнішній рівень з внутрішнім і забезпечує їх відносну незалежність, тобто можливість зміни зовнішньої схеми при незмінній внутрішній і навпаки. Роль концептуального рівня полягає, перш за все, в тому, що на нім відображається та частина загальної інформаційної бази, яка повинна бути представлена у вигляді бази даних. Концептуальний рівень забезпечує незалежність СУБД від конкретного виду ЕОМ. Формалізований опис інформаційної бази на концептуальному рівні, як правило, здійснюється в термінах конкретної СУБД. Але на початковому етапі проектування інформаційної бази ще невідомо, яка СУБД задовольняє вимогам створюваного банку даних. Тому вводиться додатковий рівень, на якому можна було б задати опис наочної області, не

стосуючись питань реалізації, тобто використання конкретної СУБД. Його називають інформаційно - логічним (інфологічним). Загальна схема відображення рівнів інформації представлена на рис. 4.1.

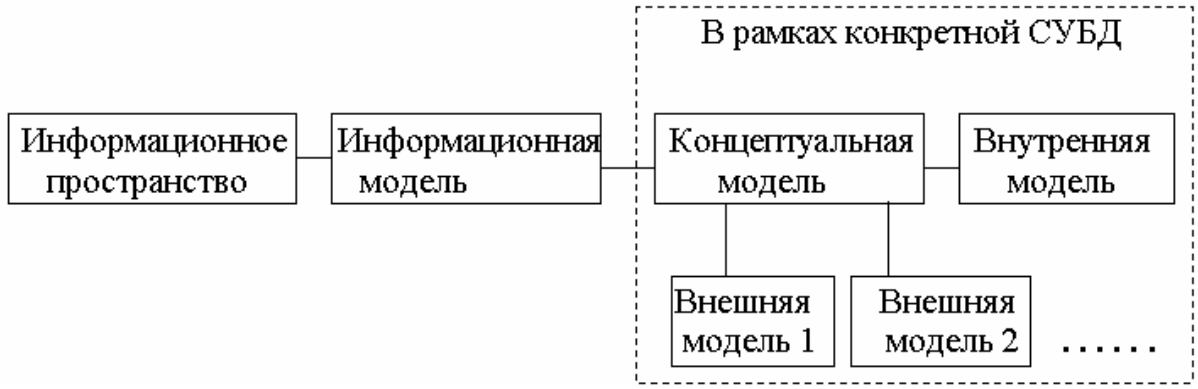


Рис. 4.1. Схема відображення рівнів інформації при проектуванні бази даних

Інформаційно-логічна модель визначає інформаційні потреби проектованої системи і характеристики інформаційної бази.

СУБД виконує наступні основні функції:

- визначення баз даних (тобто опис концептуального, зовнішнього і внутрішнього рівнів схем);
- запис даних в базу;
- організацію зберігання даних (зміна, доповнення, реорганізація даних);
- представлення доступу до даних (пошук і видача даних).

Додаткові функції (діалог, багатопользовательскій режим и т. д.) можуть бути реалізовані у вигляді пакетів програм оточення СУБД.

Для визначення даних і доступу до них в СУБД є мовні засоби (спеціальні мови). Так, визначення даних (опис концептуальної, внутрішньої і зовнішньої структур) забезпечується за допомогою мови визначення даних. Функції доступу до даних реалізуються за допомогою мови маніпулювання даними і мови запитів.

За типом підтримуваних структур розрізняють наступні види СУБД: ієрархічний, мережевий і реляційний.

Ієрархічна структура бази даних — це така структура, в якій існує впорядкований по рівнях запис елементів об'єкту. У кожній групі записів один елемент вважається головним, а інші елементи носять підлеглий характер по відношенню до головного. Групи записів упорядковуються по рівнях в певній послідовності.

Мережева структура бази даних — це така структура, в якій елементарні дані і відносини між ними представляються у вигляді орієнтованої мережі вершини, - дані, дуги -

відносини, зв'язки. Така структура дозволяє користувачеві дістати доступ до потрібного файлу без звернення до всіх інших файлів більш високого рівня інтеграції.

Реляційна база даних — це така база, в якій елементарні дані і відносини, взаємозв'язки між ними представляються у вигляді таблиць. Стовпці таблиці — це елементи даних, а рядки — записи. Основними перевагами реляційної бази даних є простота, велика гнучкість і доступність; недоліком є менша продуктивність в порівнянні з ієрархічною і мережевою структурами бази даних.

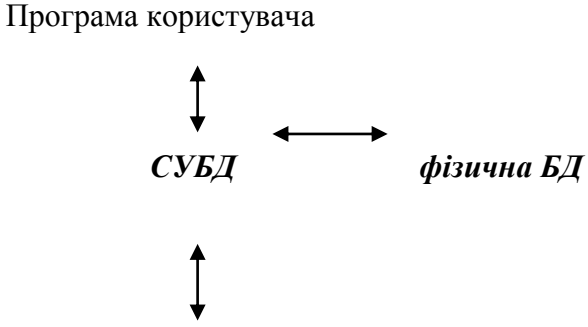
В світі існує багато систем управління базами даних. Не дивлячись на те, що вони можуть по-різному працювати з різними об'єктами і надають користувачеві різні функції і засоби, більшість СУБД спираються на єдиний сталий комплекс основних понять. Це дає нам можливість розглянути одну систему і узагальнити її поняття, прийоми і методи на весь клас СУБД. Як такий учбовий об'єкт, ми виберемо СУБД Microsoft Access, що входить в пакет Microsoft Office.

4.2 Склад СУБД

Архітектура СУБД представлена на рис. 4.2.

Мова опису даних (МОД) – засоби опису даних в БД і зв'язків між ними. Засобами цієї мови описується структура БД, формати записів, паролі, що захищають дані. Мова маніпулювання даними (ММД) – мова для виконання операцій над даними, що дозволяє міняти їх будову.

Для різних СУБД реалізація цих рівнів мов може бути різною. У одних випадках МОД і ММД вимагає складання користувачем програми повністю “уручну”, в інших (що відображає сучасну тенденцію) в СУБД присутні засоби візуальної (зримої, наочної) розробки програм. Для цього в сучасних СУБД є редактори екранних форм, звітів. “Цеглинками” (інструментами) таких редакторів є поля різних видів (поля введення, поля виводу, обчислювані поля), процедури обробки різних типів (форми введення, таблиці, звіти, запити). На підставі створених користувачем об'єктів програми – генератори формують програмний код на мові конкретної машини або на проміжній мові.



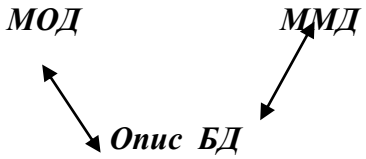


Рис. 4.2 Архітектура СУБД

4.3 Структура нескладної бази даних

Одразу пояснимо, якщо в базі немає ніяких даних (порожня база), то це все одно повноцінна база даних. Цей факт має методичне значення. Хоча даних в базі і немає, але інформація в ній все-таки є – це структура бази. Вона визначає методи занесення даних і зберігання їх в базі. Простий «некомп'ютерний» варіант бази даних – діловий щоденник, в якому кожному календарному дню виділено по сторінці. Навіть якщо в ньому не записано ні рядка, він не перестає бути щоденником, оскільки має структуру, що чітко відрізняє його від записників, робочих зошитів і іншої паперової продукції.

Бази даних можуть містити різні об'єкти. Основними об'єктами будь-якої бази даних є її таблиці. Проста база даних має хоч би одну таблицю. Відповідно, структура простої бази даних тотожно рівна структурі її таблиці.

Структуру двовимірної таблиці утворюють стовпці і рядки. Їх аналогами в простій базі даних є **поля і записи.** Якщо записів в таблиці поки немає, значить, її структура утворена тільки набором полів. Змінивши склад полів базової таблиці (або їх властивості), ми змінюємо структуру бази даних і, відповідно, отримуємо нову базу даних.

4.4 Властивості полів бази даних

Поля бази даних не просто визначають структуру бази – вони ще визначають групові властивості даних, записаних в осередки, що належать кожному з полів. Нижче перераховані основні властивості полів таблиць баз даних на прикладі СУБД Microsoft Access.

- **Ім'я поля** – визначає, як слід звертатися до даних цього поля при автоматичних операціях з базою (за умовчанням імена полів використовуються як заголовки стовпців таблиць).
- **Тип поля** – визначає тип даних, які можуть міститися в даному полі.
- **Розмір поля** – визначає граничну довжину (у символах) даних, які можуть розміщуватися в даному полі.
- **Формат поля** – визначає спосіб форматування даних в осередках, що належать полю.

- **Маска введення** – визначає форму, в якій вводяться дані в поле (засіб автоматизації введення даних).

- **Підпис** – визначає заголовок стовпця таблиці для даного поля (якщо підпис не вказаний, то як заголовок стовпця використовується властивість Ім'я поля).

- **Значення за умовчанням** – те значення, яке вводиться в осередки поля автоматично (засіб автоматизації введення даних).

- **Умова на значення** – обмеження, використовуване для перевірки правильності введення даних (засіб автоматизації введення, яке використовується, як правило, для даних, що мають числовий тип, грошовий тип або тип дати).

- **Повідомлення про помилку** – текстове повідомлення, яке видається автоматично при спробі введення в полі помилкових даних.

- **Обов'язкове поле** – властивість, що визначає обов'язковість заповнення даного поля при наповненні бази.

- **Порожні рядки** – властивість, що вирішує введення порожніх рядкових даних (від властивості "Обов'язкове поле" відрізняються тим, що відноситься не до всіх типів даних, а лише до деяких, наприклад до текстових).

- **Індексоване поле** – якщо поле володіє цією властивістю, всі операції, пов'язані з пошуком або сортуванням записів за значенням, що зберігається в даному полі, істотно прискорюються. Крім того, для індексованих полів можна зробити так, що значення в записах перевірятимуться по цьому полю на наявність повторів, що дозволяє автоматично виключити дублювання даних.

Оскільки в різних полях можуть міститися дані різного типу, то і властивості у полів можуть розрізнятися залежно від типу даних. Так, наприклад, список вищезгаданих властивостей полів відноситься в основному до полів текстового типу. Поля інших типів можуть мати або не мати цих властивостей, але можуть додавати до них і свої. Наприклад, для даних, що представляють дійсні числа, важливою властивістю є кількість знаків після десяткової коми. З іншого боку, для полів, використовуваних для зберігання малюнків, звукозаписів, відео кліпів і інших об'єктів OLE, більшості вищезгаданих властивостей не мають сенсу.

4.5 Типи даних

Таблиці баз даних, як правило, допускають роботу з великою кількістю різних типів даних. Так, наприклад, бази даних Microsoft Access працюють з наступними типами даних.

- **Текстовий** – тип даних, використовуваний для зберігання звичайного неформатованого тексту обмеженого розміру (до 255 символів).

- **Числовий** – тип даних для зберігання дійсних чисел.
- **Поле Мемо** – спеціальний тип даних для зберігання великих об'ємів тексту (до 65 535 символів). Фізично текст не зберігається в полі. Він зберігатися у іншому місці бази даних, а в полі зберігатися покажчик на нього, але для користувача таке розділення помітне не завжди.
- **Дата/час** – тип даних для зберігання календарних дат і поточного часу.
- **Грошовий** - тип даних для зберігання грошових сум. Теоретично, для їх запису можна було б користуватися і полями числового типу, але для грошових сум є деякі особливості (наприклад, пов'язані з правилами округлення), які роблять зручнішим використання спеціального типу даних, а не настройку числового типу.
- **Лічильник** – спеціальний тип даних для унікальних (що не повторюються в полі) натуральних чисел з автоматичним нарощуванням. Природне використання – для порядкової нумерації записів.
- **Логічний** - тип для зберігання логічних даних (можуть приймати тільки два значення, наприклад "так" чи "ні").
- **Гіперпосилання** – спеціальне поле для зберігання адрес URL Web-объектов Інтернету. При кліканні на посиланні автоматично відбувається запуск браузера і відтворення об'єкту в його вікні.
- **Майстер підстановок** – це не спеціальний тип даних. Це об'єкт, настройкою якого можна автоматизувати введення даних в полі так, щоб не вводити їх уручну, а вибирати їх із списку, що розкривається.

4.6 Безпека баз даних

Бази даних – це теж файли, але робота з ними відрізняється від роботи з файлами інших типів, що створюються іншими застосуваннями. Вище ми бачили, що всю роботу по обслуговуванню файлової структури бере на себе операційна система. Для бази даних пред'являються особливі вимоги з погляду безпеки, тому в них реалізований інший підхід до збереження даних.

Бази даних – це особливі структури. Інформація, яка в них міститься, дуже часто має суспільну цінність. Нерідко з однією і тією ж базою працюють тисячі людей по всій країні. Від інформації, яка міститься в деяких базах, може залежати благополуччя безлічі людей. Тому цілісність вмісту бази не може і не повинна залежати ні від конкретних дій якогось користувача, що забув зберегти файли перед виключенням комп'ютера, ні від перебоїв в електромережі.

Проблема безпеки баз даних вирішується тим, що в СУБД для збереження інформації використовується подвійний підхід. У частині операцій, як завжди, бере участь операційна система комп'ютера, але деякі операції збереження відбуваються в обхід операційної системи.

4.7 Проектування баз даних

4.7.1 Режими роботи з базами даних. Зазвичай з базами даних працюють дві категорії осіб. Перша категорія – проектувальники. Їх завдання полягає в розробці структури таблиць бази даних і узгодження її із замовником. Окрім таблиць проектувальники розробляють і інші об'єкти бази даних, призначені, з одного боку, для автоматизації роботи з базою, а з іншого боку – для обмеження функціональних можливостей роботи з базою (якщо це необхідно з міркувань безпеки). Проектувальники не наповнюють базу конкретними даними (замовник може вважати їх конфіденційним і не надавати стороннім особам). Виняток становить експериментальне наповнення модельними даними на етапі налаштування об'єктів бази.

Друга категорія - користувачі, що працюють з базами даних. Вони отримують початкову базу даних від проектувальників і займаються її наповненням і обслуговуванням. У загальному випадку, користувачі не мають засобів доступу до управління структурою бази – тільки до даних, та і то не до всіх, а до тих, робота з якими передбачена на конкретному робочому місці.

Відповідно СУБД має два режими роботи: **проектувальний** і **призначений для користувача.** Перший режим призначений для створення або зміни структури бази і створення її об'єктів. У другому режимі відбувається використання раніше підготовлених об'єктів для наповнення бази або отримання даних з неї.

4.7.2 Об'єкти бази даних

Таблиці – це основні об'єкти будь-якої бази даних. По-перше, в таблицях зберігаються всі дані, наявні в базі, а по-друге, таблиці зберігають і структуру бази (поля, їх типи і властивості).

Запити - ці об'єкти слугують для отримання даних з таблиць і надання їх користувачеві в зручному вигляді. За допомогою запитів виконують такі операції як відбір даних, їх сортування і фільтрацію. За допомогою запитів можна виконувати перетворення даних по заданому алгоритму, створювати нові таблиці, виконувати автоматичне наповнення таблиць даними, імпортованими з інших джерел, виконувати прості обчислення в таблицях і багато що інше.

Форми. Якщо запити – це спеціальні засоби для відбору і аналізу даних, то форми – це засоби для введення даних. Сенс їх той же – надати користувачеві засоби для заповнення

тільки тих полів, які йому заповнювати належить. Одночасно з цим у формі можна розмістити спеціальні елементи управління (лічильники, списки, що розкриваються, перемикачі, прапорці і інше) для автоматизації введення. Переваги форм розкриваються особливо наочно, коли відбувається введення даних із заповнених бланків. В цьому випадку форму роблять графічними засобами так, щоб вона повторювала оформлення бланка – це помітно спрощує роботу складача, знижує його стомлення і запобігає появі друкарських помилок.

Звіти

По своїх властивостях і структурі звіти багато в чому схожі на форми, але призначені тільки для виведення даних, причому для виводу не на екран, а на принтер. У зв'язку з цим звіти відрізняються тим, що в них прийняті спеціальні заходи для групування даних, що виводяться, і для виведення спеціальних елементів оформлення, характерних для друкарських документів.

Сторінки

Це спеціальні об'єкти баз даних, реалізованих в СУБД Microsoft Access. Правда, більш корректно їх називати **сторінками доступу до даних.** Фізично це особливий об'єкт, виконаний в коді HTML, що розміщується на Web-сторінці і переданий клієнтові разом з нею. Сам по собі цей об'єкт не є базою даних, але містить компоненти, через які здійснюється зв'язок переданої Web-страницы з базою даних, що залишається на сервері. Користуючись цими компонентами, відвідувач Web-узла може проглядати записи бази в полях сторінки доступу. Таким чином, сторінки доступу до даних здійснюють інтерфейс між клієнтом, сервером і базою даних, розміщеною на сервері. Ця база даних не обов'язково повинна бути базою даних Microsoft Access. Сторінки доступу, створені засобами Microsoft Access, дозволяють працювати також з базами даних Microsoft SQL Server.

Макроси і модулі

Ці категорії об'єктів призначені як для автоматизації операцій, що повторюються, при роботі з СУБД, так і для створення нових функцій шляхом програмування. У СУБД Microsoft Access **макроси** складаються з послідовності внутрішніх команд СУБД і є одним із засобів автоматизації роботи з базою. **Модулі** створюються засобами зовнішньої мови програмування, в даному випадку мови Visual Basic for Applications. Це один із засобів, за допомогою яких розробник бази може закласти в неї нестандартні функціональні можливості, задовольнити специфічну вимогу замовника, підвищити швидкодію системи управління, а також рівень її захищеності.

4.7.3 Основи проектування баз даних

Методично правильно починати роботу з олівцем і листом паперу в руках, не використовуючи комп'ютер. На даному етапі він просто не потрібний. Неоптимальні рішення і прямі помилки, закладені на етапі проектування, згодом дуже важко усуваються, тому цей етап є основоположним.

Розробка технічного завдання. Технічне завдання на проектування бази даних повинен надати замовник. Проте для цього він повинен володіти відповідною термінологією і знати, хоч би у загальних рисах, технічні можливості основних СУБД. На практиці таке положення зустрічається не завжди. Тому зазвичай використовують наступні підходи:

- Демонструють замовникові роботу аналогічної бази даних, після чого погоджують специфікацію відмінностей;
- Якщо аналога немає, з'ясовують круг завдань і потреб замовника, після чого допомагають йому підготувати технічне завдання.

При підготовці технічного завдання складають:

- Список початкових даних, з якими працює замовник;
- Список вихідних даних, які необхідні замовникові для управління структурою свого підприємства;
- Список вихідних даних, які не є необхідними для замовника, але які він повинен надати в інші організації (у вищестоящі структури, в органи статистичного обліку, інші адміністративні і контролюючі організації).

При цьому дуже важливо не обмежуватися взаємодією з головним підрозділом замовника, а провести обговорення зі всіма службами і підрозділами, які можуть надалі виявитися постачальниками даних в базу або їх споживачами.

Розробка структури бази даних. З'ясувавши основну частину даних, які замовник споживає або поставляє, можна приступати до створення структури бази, тобто структури її основних таблиць.

1. Робота починається з складання генерального списку полів – він може налічувати десятки і навіть сотні позицій.
2. Відповідно до типу даних, що розміщуються в кожному полі, визначають найбільш відповідний тип для кожного поля.
3. Далі розподіляють поля генерального списку по базових таблицях. На першому етапі розподіл проводять за функціональною ознакою. Мета – забезпечити, щоб введення даних в одну таблицю проводилося, по можливості, в рамках одного підрозділу, а ще краще – на одному робочому місці.
4. У кожній з таблиць намічають **ключове поле**. Як таке, вибирають поле, дані в якому повторюватися не можуть. Наприклад, для таблиці даних про студентів таким полем

може служити індивідуальний шифр студента. Для таблиці, в якій міститися розклад занять, такого поля можна і не знайти, але його можна створити штучним комбінуванням полів «Час заняття» і «Номер аудиторії». Ця комбінація не повторювана, оскільки в одній аудиторії в один і той же час не прийнято проводити два різні заняття. Якщо в таблиці взагалі немає ніяких полів, які можна було б використовувати, як ключові, завжди можна ввести додаткове поле типу Лічильник – воно не може містити даних, що повторюються, за визначенням.

5. За допомогою олівця і паперу розкреслюють зв'язки між таблицями. Таке креслення називається **схемою даних.** Існує декілька типів можливих зв'язків між таблицями. Найбільш поширеними є зв'язки «один до багатьох» і «один до одного». Зв'язок між таблицями організовується на основі загального поля, причому в одній з таблиць воно обов'язково повинне бути ключовим, тобто на стороні «один» повинне виступати ключове поле, що містить унікальні значення, що не повторюються. Значення на стороні «багато хто» може повторюватися.

6. Розробкою схеми даних закінчується «паперовий» етап роботи над технічною пропозицією. Цю схему можна погоджувати із замовником, після чого приступати до безпосереднього створення бази даних.

Слід пам'ятати, що по ходу розробки проекту замовникові неодмінно приходитимуть в голову нові ідеї. На всіх етапах проектування він прагне охопити єдиною системою все нові і нові підрозділи і служби підприємства. Можливість гнучкого використання його побажань багато в чому визначається кваліфікацією розробника бази даних. Якщо схема даних складена правильно, підключати до бази нові таблиці неважко. Якщо структура бази нераціональна, розробник може зазнавати серйозні труднощі і увійти у суперечність із замовником. Суперечності виконавця із замовником завжди свідчать про недостатню кваліфікацію виконавця. Саме по цьому етап попереднього проектування бази даних слід вважати основним. Від його успіху залежить, наскільки база даних стане зручною, і чи з нею працюватимуть користувачі. Якщо наголошується, що користувачі бази «саботують» її експлуатацію і вважають за краще працювати традиційними методами, це говорить не про низьку кваліфікацію користувачів, а про недостатню кваліфікацію розробника бази.

На цьому етапі завершується попереднє проектування бази даних, і на наступному етапі починається її безпосередня розробка. З цієї миті слід почати роботу з СУБД.

ЛЕКЦІЯ 5. ОПТИМІЗАЦІЯ ПРОЕКТНИХ РІШЕНЬ.

АНАЛІТИЧНІ МЕТОДИ

5.1 Визначення та терміни у галузі оптимізації.

Оптимізація - процес надання будь-чому найвигідніших характеристик, співвідношень (наприклад, оптимізація руху електроприводу, оптимізація енергетичних показників). **Задача оптимізації** сформульована, якщо задані: **критерій оптимальності** (швидкодія, точність, економічність); параметри, що варіюються (наприклад, форма керуючих дій, параметри електричної або механічної частини електроприводу, спосіб електромеханічного перетворення енергії), зміна яких дозволяє впливати на ефективність процесу; математична модель процесу; обмеження, пов'язані з конструктивними та економічними умовами, можливостями апаратури, вимогами безпеки та ін. Оптимізація полягає в знаходженні найкращого варіанта. Методи оптимізації застосовуються до пошуку розрахунку оптимальної траєкторії руху електроприводу, оптимального часу перехідного процесу, оптимального обсягу електроенергії, що споживана із мережі, оптимальних втрат потужності, отже оптимального ККД та оптимального нагріву електричної машини. **Практична цінність використання процесу оптимізації полягає в тому, що вона дозволяє вибирати найкращий варіант рішення без перевірки всіх можливих варіантів.**

Якщо проектне рішення обирається на підставі деяких міркувань, власного досвіду проектанта, і без повного аналізу згідно теорії оптимізації, то таке рішення має назву **раціонального рішення.** Це рішення теж має право на життя, але не виключає, що є інші, більш вигідні рішення.

Крите́рій оптима́льності - основний показник якості роботи системи. Критерій оптимальності — фундаментальне поняття системи оптимального функціонування об'єктів (машин, процесів, підприємства, галузі, економіки у цілому). У системах керування технологічними процесами критерієм оптимальності може бути один з показників якості перехідного процесу (скажімо, його тривалість), точність у номінальному режимі, собівартість продукції тощо. Наприклад, надзвичайно важливою є оптимізація параметрів статичних та динамічних режимів транспортних засобів, електроприводи яких живляться від акумуляторних батарей, таких як електромобілі, електрокари, інвалідні візки, тощо. Тут важливішим критерієм оптимізації виступає максимум механічної роботи, що виконана при обмеженні обсягу уживаної електричної енергії. **Частковий критерій** - критерій оптимальності, в якому як цільова функція приймається один з вихідних параметрів, що якнайповніше відображає якість досліджуваного об'єкту. При цьому інші параметри або не змінюються, або не повинні виходити за обговорені межі.

Крите́рій оптима́льного проектува́ння - економічний показник, який є оцінкою альтернативних варіантів проекту (плану). У більшості випадків критерій оптимального проектування — це приведені витрати при впровадженні того чи іншого варіанту проекту, детальніше див. п. 4. Іноді критерієм є собівартість, питомі капіталовкладення, трудоємність технологічного процесу на дільниці або підприємстві.

Цільова функція — функція, що зв'язує мету (змінну, що оптимізується) з керованими змінними в задачі оптимізації. У широкому сенсі **цільова функція** є математичний вираз деякого критерію якості одного об'єкту (рішення, процесу і т. д.) в порівнянні з іншим. Мета — знайти такі оцінки, при яких цільова функція досягає мінімуму. Важливо, що критерій завжди привноситься ззовні, і тільки після цього шукається правило рішення, що мінімізує або максимізує цільову функцію. Цільова функція може бути **одно параметричною** та **багато параметричною**, тобто залежати від одного, або кількох параметрів. Одно параметрична функція може мати один. або більше екстремумів. **Унімодальна цільова функція** – така, що має тільки один екстремум на заданій ділянці змінної.

Зада́ча оптимізáції — задача знаходження точки (точок) мінімуму, або декількох мінімумів заданої цільової функції. Формальне визначення задачі оптимізації: нехай задано деяку множину **X** із *n*-вимірного евклідового простору і функцію f(x), визначену на **X**. Необхідно знайти точки мінімуму значень функції f(x) на **X**. Або:

$$f(x) \to \min, x \in \mathbf{X}$$

тут f(x) — цільова функція, **X** — допустима множина, кожна точка x цієї множини - допустима точка задачі. Також, задачу оптимізації можна сформулювати як пошук максимуму (максимумів) цільової функції:

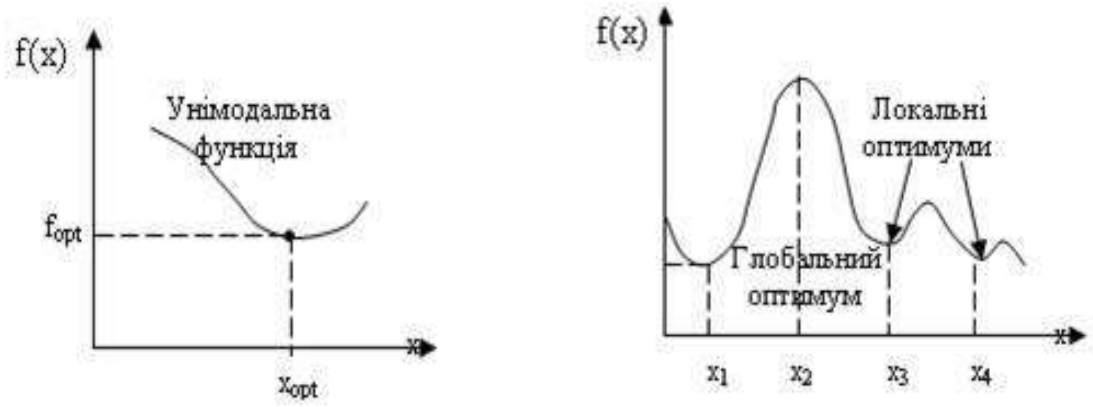
$$f(x) \to \max, x \in \mathbf{X}$$

ця задача еквівалентна попередній задачі мінімізації цільової функції із знаком мінус, в тому сенсі, що множини їхніх розв'язків збігаються.

Багатокритеріальна оптимізація - це процес одночасної оптимізації двох або більше конфліктуючих цільових функцій в заданій області визначення. Задача багатокритеріальної оптимізації зустрічаються в багатьох галузях науки та техніки. Задача багатокритеріальної оптимізації полягає у пошуку вектора цільових змінних, який задовольняє накладеним

обмеженням та оптимізує векторну функцію, елементи якої відповідають цільовим функціям. Ці функції утворюють математичне описання критерію задовільності та, зазвичай, взаємно конфліктують. Звідси, «оптимізувати» означає знайти такий розв'язок, за якого значення цільових функцій були б прийнятними для постановника задачі. Строго кажучи, кожний проект є багатокритеріальним. Адже для прийняття проектного рішення треба, окрім оптимізації технічних показників технічними же засобами, врахувати економічні показники, складність технології виробництва та експлуатації, екологічні і соціальні наслідки впровадження проекту. При вирішенні багатокритеріальних задач, які важко, або неможливо точно описати, можуть використовуватися методики бальних оцінок із залученням фахівців – експертів. Звісно, ці методи не вільні від суб'єктивізму, але загалом відображають колективну думку, або узагальнене ставлення до проблеми, що вирішується. Прикладом такої методики є **кваліметрія.**

Оптимум локальний та глобальний. Якщо цільова функція унімодальна, то цей оптимум буде одночасно і локальним і глобальним, рис. 5.1,а. Якщо цільова функція не унімодальна, тобто має багато екстремумів то і оптимумів може бути багато. Такі оптимуми є локальними. Серед локальних оптимумів можна виділити глобальний оптимум, який забезпечує найбільший значення екстремуму, рис. 5.1 ,б.



а. б.

Рис. 5.1 Приклади цільових функцій:
а. – унімодальна; б. з локальним та глобальним оптимумом

Методи оптимізації – розподіляються на аналітичні, чисельні (пошукові), та графічні. Аналітичні методи оптимізації застосовуються для вирішення завдань автоматизованого проектування у тому випадку, коли функція мети і обмеження задані аналітично - у вигляді безперервних функцій, що диференціюються. Сучасні програмні продукти, орієнтовані на

автоматизоване проектування, дозволяють використовувати різні методи оптимізації для систем і пристроїв, що описуються в рамках даного середовища і не тільки аналітично, але і у вигляді структурних схем з нелінійними елементами. Пошукові методи оптимізації засновані на поступовому переміщенні в просторі змінних у напрямі до найбільшого значення функції мети. Головна особливість пошукових методів полягає в тому, що найбільше значення функції визначається (відшукується) не відразу, а в результаті процесу пересування, що розгортається в часі, в результаті певної послідовності дій. Чисельні методи мають на увазі обов'язкове використання ЕОМ, тому їх ще називають **методами програмування.** Графічний метод полягає в побудові графіку цільової функції, наочному знаходженні екстремуму та його подальшому уточненні. Графічний метод зручно використовувати, якщо є сумніви в вірності використання інших методів або в вірності отриманих результатів.

Геометричне представлення цільової функції. Якщо цільова функція F(x) одно параметрична, то її графічним зображенням є лінія у площині із координатними осями F(x); x. Приклади ліній як функції різного порядку показані на рис. 5.1.

Двох параметрична функція F(x1,x2) може бути відображена як об'ємна у трьохвимірній Декартовій системі координат із осями F(x1,x2); x1; x2, як показано на рис. 5.2. У наведеному прикладі об'ємна фігура, рис. 5.2,а сприймається природно, або наочно, на ній можна побачити три вершини (максимуми) і три впадини (мінімуми). Якщо всю фігуру розрізати на окремі прошарки за допомогою горизонтальних площин із заданим шагом по вертикали ε, то отримаємо лінії однакового рівня, яки можна зобразити на площині, рис. 5.2,б. У такий спосіб зображують гори та впадини на географічних мапах. Кожна лінія на рис. 5.2,б відповідає якомусь рівню у вертикальній площині. Якщо переміщуватись у площині по координаті x1 або x2 від лінії до лінії, то завжди є інформація чи ми підіймаємось, чи опускаємось. На цих особливостях базуються деякі методи пошуку екстремуму.

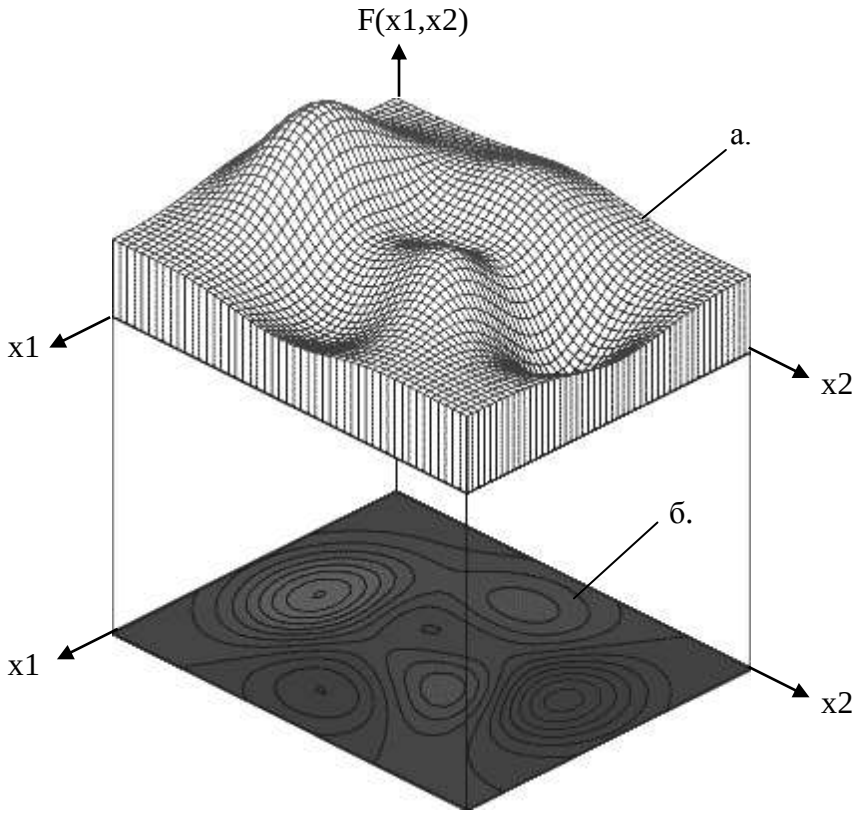


Рис. 5.2 Геометрична інтерпретація двох параметричної цільової функції.
а. об’ємне зображення у координатах F(x1,x2); x1; x2;
б. умовне зображення на площині у координатах x1; x2

5.2 Аналітичні методи оптимізації

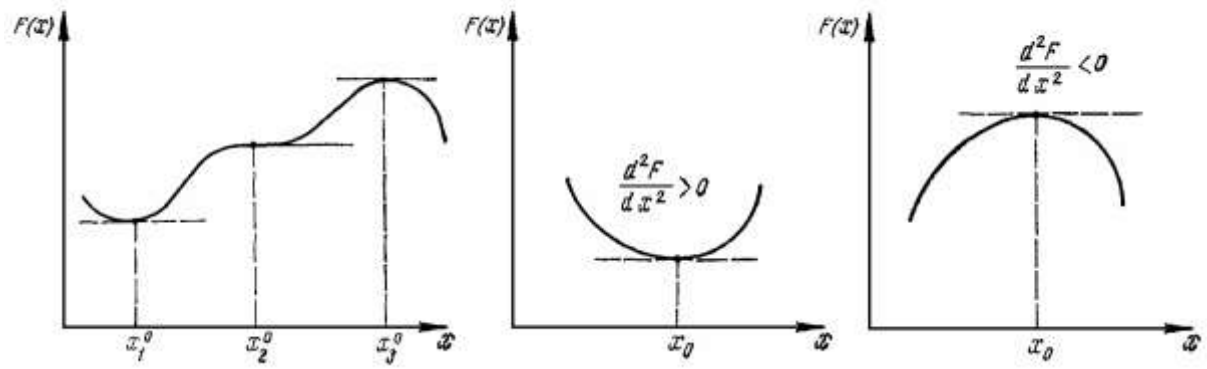
Знаходження екстремуму функції однієї змінної без обмежень. Нехай цільова функція, тобто функція критерію від змінної *x* задана безперервною функцією, що диференціюється

$$F=F(x). \tag{5.1}$$

Як відомо з математичного аналізу, необхідна умова екстремуму функції *F(x)* є рівність нулю її першої похідної

$$\frac{dF}{dx}=0. \tag{5.2}$$

Точка x^0, в якій похідна стає рівною нулю (дотична до функції *F(x)* паралельна осі абсцис), має назву **стаціонарної** точки функції *F(x).* Стаціонарна точка може бути максимумом, мінімумом або точкою перегину (рис. 1,а). Тип стаціонарної точки визначається достатніми умовами оптимальності.



а. б. в.

Рис. 5.3 Приклади екстремумів цільових функцій

Якщо друга похідна *F(x)* в точці x^0 позитивна, то в цій крапці є мінімум (мал. 1,б)

$$\frac{d^2F}{dx^2} > 0. \tag{5.3}$$

Якщо друга похідна негативна — в точці x^0 є максимум (мал. 1,в)

$$\frac{d^2F}{dx^2} < 0 \tag{5.4}$$

Приклад: функція задана аналітичним виразом $F(x)=3x^2—2x— 1$;

$$\frac{dF}{dx} = 6x - 2 = 0; \quad x_0 = \frac{1}{3}; \quad \frac{d^2F}{dx^2} = 6 > 0.$$

У точці $x^0 = \frac{1}{3}$ є мінімум.

Екстремум функції багатьох змінних. Нехай цільова функція задається безперервною функцією, що диференціюється

$$F = F(x_1, x_2, ..., x_n). \tag{5.5}$$

Необхідна умова екстремуму — рівність нулю часткових похідних функції *F* по всіх змінних.

$$\left.\begin{array}{l} \dfrac{\partial F}{\partial x_1} = 0; \\ \dfrac{\partial F}{\partial x_2} = 0; \\ \ldots \\ \dfrac{\partial F}{\partial x_n} = 0. \end{array}\right\} \qquad (5.6)$$

Вирішенням системи рівнянь (6) визначаємо стаціонарну точку.

Щоб сформулювати достатні умови, розглянемо матрицю других похідних функції *F:*

$$M = \begin{pmatrix} \dfrac{\partial^2 F}{\partial x_1^2} & \dfrac{\partial^2 F}{\partial x_1 \partial x_2} & \cdot & \dfrac{\partial^2 F}{\partial x_1 \partial x_n} \\ \dfrac{\partial^2 F}{\partial x_1 \partial x_2} & \dfrac{\partial^2 F}{\partial x_2^2} & \cdot & \dfrac{\partial^2 F}{\partial x_2 \partial x_n} \\ \ldots & \ldots & \cdot & \ldots\ldots\ldots \\ \dfrac{\partial^2 F}{\partial x_n \partial x_1} & \dfrac{\partial^2 F}{\partial x_n \partial x_2} & \cdot & \dfrac{\partial^2 F}{\partial x_n^2} \end{pmatrix} \qquad (5.7)$$

Позначимо визначників діагонального мінору цієї матриці Δ_1, $\Delta_{2\ldots}\Delta_n$:

$$\Delta_1 = \frac{\partial^2 F}{\partial x_1^2}; \qquad \Delta_2 = \begin{vmatrix} \dfrac{\partial^2 F}{\partial x_1^2} & \dfrac{\partial^2 F}{\partial x_1 \partial x_2} \\ \dfrac{\partial^2 F}{\partial x_1 \partial x_2} & \dfrac{\partial^2 F}{\partial x_2^2} \end{vmatrix} = \frac{\partial^2 F}{\partial x_1^2} \cdot \frac{\partial^2 F}{\partial x_2^2} - \left(\frac{\partial^2 F}{\partial x_1 \partial x_2} \right)^2 \qquad (5.8)$$

і так далі.

Щоб в стаціонарній точці був мінімум, достатньо, щоб визначники всього діагонального мінору були більше нуля:

$$\Delta 1>0;\ \Delta 2>0;\ 3>0 \ldots,\ n>0. \qquad (5.9)$$

Для того, щоб в стаціонарній точці був максимум, достатньо, щоб визначники всього парного мінору були більше нуля, а всього непарного мінору — менше нуля:

$$\Delta 1<0;\ \Delta 2>0;\ .,\ 2k>0 \ldots,\ 2k+1<0. \qquad (5.10)$$

Приклад: задана функція двох змінних $F = x_1^4 + 2x_2^2 - 4x_1$;

$$\frac{\partial F}{\partial x_1} = 4x_1^3 - 4 = 0 \qquad \frac{\partial F}{\partial x_2} = 4x_2 = 0.$$

Стаціонарна точка $x1^0$ =l; $x2^0$ =0.

Визначники діагонального мінору:

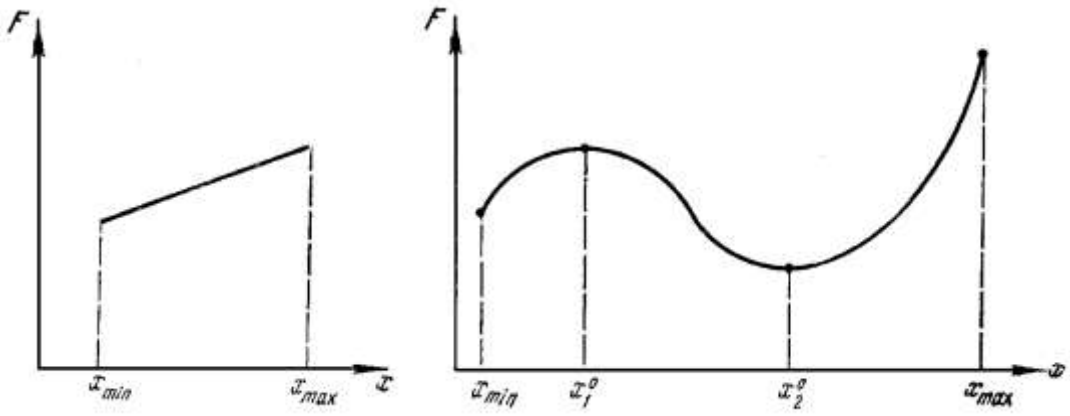
$$\Delta_1 = \frac{\partial^2 F}{\partial x_1^2} = 12x_1^2 = 12 > 0; \qquad \Delta_2 = \frac{\partial^2 F}{\partial x_1^2} \cdot \frac{\partial^2 F}{\partial x_2^2} - \left(\frac{\partial^2 F}{\partial x_1 \partial x_2} \right)^2 = 12 \cdot 4 = 48 > 0.$$

Згідно умові (9) маємо висновок: у точці , *x1=1, x2=0* є мінімум.

Існує достатньо багато різних аналітичних методів пошуку екстремумів функцій, заданих аналітично (з обмеженнями у вигляді рівності і нерівностей, на основі множників Лагранжа та інші). З цими методами можна ознайомитися в численній літературі.

Недоліки аналітичних методів оптимізації. Аналітичні методи оптимізації, засновані на необхідних умовах екстремуму, не завжди придатні для вирішення завдань управління технологічними установками і процесами по наступних причинах:

- Для вирішення завдань аналітичними методами функція мети і модель об'єкту повинні описуватися безперервними гладкими функціями, що диференціюються, заданими в



а. б.

Рис. 5.4 а – лінійна функція мети; б - максимум розташований на межі області визначення и в стороні від локальних максимумів

аналітичній формі. Реальні моделі об'єктів часто описуються нерівними функціями, що не диференціюються, наприклад кусочно-лінійними, або бувають задані у вигляді таблиць і тому подібне. В цих випадках, перед рішенням, необхідно заздалегідь апроксимувати ці залежності гладкими функціями.

- При використанні аналітичних методів максимальне значення функції визначається в її стаціонарній точці. Між тим, стаціонарних точок може бути кілька. Тоді ми змушені проводити перевірку у всіх стаціонарних точках, відшукуючи найбільше значення (глобальний максимум) функції. Максимуму взагалі може не бути, наприклад у випадку лінійної функції, рис. 5.4,а. Крім того, найбільше значення функції цілі може бути досягнутим не в стаціонарній точці, а на межі області визначення функції мети, рис. 5.4, б.
- Велика розмірність, наявність обмежень у вигляді рівності і особливо у вигляді нерівностей істотно ускладнюють рішення задачі оптимізації.

- Аналітичні методи часто погано пристосовані для вирішення завдань оптимізації за допомогою ЕОМ.

ЛЕКЦІЯ 6. ОПТИМІЗАЦІЯ ПРОЕКТНИХ РІШЕНЬ. ЧИСЕЛЬНІ МЕТОДИ.

6.1 Чисельні методи оптимізації

Всі чисельні методи є пошуковими і їх можна розділити на три класи:

- **методи прямого пошуку**, що базуються на обчисленні тільки значень цільової функції;

- **градієнтні методи**, в яких використовуються точні значення перших похідних f(x) ;

- **методи другого порядку**, в яких поряд з першими похідними використовуються також другі похідні функції f(x).

Методи прямого пошуку використовуються для визначення оптимуму цільової функці частіш за інші. Найпростіший випадок коли **цільова функція** є **унімодальною**, тобто має на інтервалі дослідження, який розглядається, тільки один оптимум. Навіть якщо цільова функція має багато екстремумів, то можна графічно знайти глобальний екстремум, визначити границі його існування і надалі провести дослідження функції на екстремум, як унімодальної на заданому інтервалі. У цьому випадку задача одновимірної оптимізації ставиться таким чином: значення параметра X цільової функції f(x), який називають **проектним параметром**, знаходяться на інтервалі дослідження [a, b]. В процесі пошуку оптимуму цільової функції цей інтервал, який називається **інтервалом невизначеності**, постійно зменшується (звужується), тому методи одновимірної оптимізації іноді називають **методами звуження інтервалу невизначеності**. Деякі алгоритми оптимізації пристосовані до пошуку максимума, а інші – для пошуку мінімуму. Однак, незалежно від типу задачі, яка розв’язується на екстремум (оптимум) можливо користуватись одним і тим же алгоритмом, так як задачу мінімізації можливо легко переробити в задачу на пошук максимуму, змінивши знак цільової функції на протилежний. Вибір чисельного методу в першу чергу залежить від виду цільової функції, яка може бути однопараметричною і багатопараметричною.

Більшість чисельних методів одновимірної оптимізації - це методи звуження відрізка, а саме: метод розділення відрізку навпіл, метод дихотомії, метод золотого перерізу, метод Фібоначчі.

6.2 Метод загального пошуку

Природнім способом звуження інтервалу невизначеності для одновимірної унімодальної функції є ділення його на декілька рівних частин з наступним обчисленням значень цільової функції в вузлах отриманої сітки, рис. 6.1.

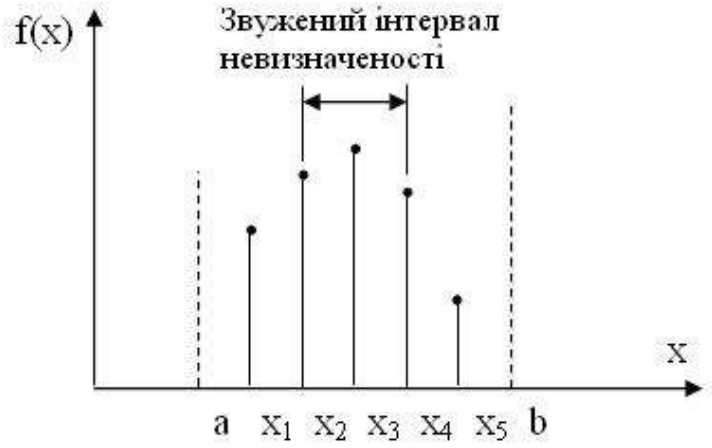


Рис. 6.1 Метод загального пошуку

В результаті інтервал невизначеності звужується до двох кроків сітки. Звичайно говорять про дроблення інтервалу невизначеності, яке характеризується коефіцієнтом f. Розділивши інтервал невизначеності на N частин, отримаємо N+1 вузол, і тоді

$$f = \frac{2}{N+1} \tag{6.1}$$

Щоб отримати значення *f = 0,01*, необхідно обчислити цільову функцію в 199 точках, а при *f = 0,001 N=1999*. Звідси видно, що ефективність цього методу при зменшенні інтервалу невизначеності швидко падає. Напрошується інший варіант: щоб отримати *f=0,01*, необхідно обчислити спочатку функцію в 19 точках і отримати *f = 0,1*, а потім обчислити ще 19 значень функції на скороченому інтервалі невизначеності, отримати *f = 0,01*, зробивши при цьому всього 38, а не 199 обчислень. Таким чином, при деякій винахідливості ефективність пошуку можна різко збільшити.

6.3 Метод половинного ділення (розділення відрізка навпіл)

Суть метода полягає в постійному діленні відрізка дослідження цільової функції [a, b] навпіл і визначенні на ньому координат трьох точок x_1, x_2, x_m. При чому значення їх визначаються як:

$$x_m = \frac{a+b}{2}; \quad L = b - a; \quad x_1 = a + \frac{L}{4}; \quad x_2 = b - \frac{L}{4} \tag{6.2}$$

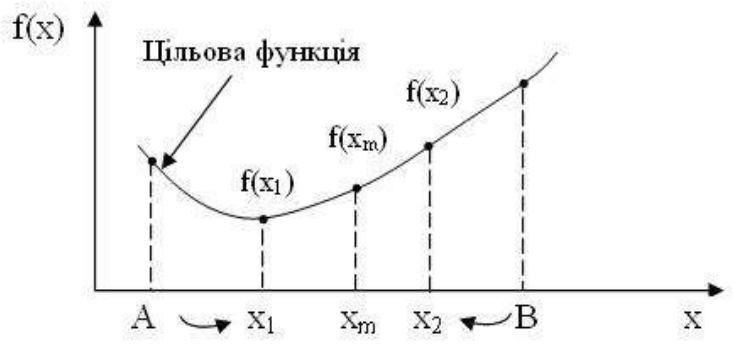


Рис. 6.2 Геометрична інтерпретація методу половинного ділення

Точка x_m є серединою відрізка AB осі, точки x_1, x_2 і x_m ділять відрізок на чотири рівні частини (рис. 6.2). Обчислюємо значення цільової функції $f(x_1); f(x_m); f(x_2)$, потім порівнюємо значення $f(x_1)$ і $f(x_m)$, якщо $f(x_1) < f(x_m)$, то виключаємо з дослідження відрізок $[x_m, b]$ та покладемо $b = x_m$. Тоді середньою точкою нового відрізка [a, b] стає x_1 $(x_m = x_1)$. Але якщо $f(x_1) \geq f(x_m)$, то порівнюємо значення цільової функції $f(x_2)$ і $f(x_m)$. Якщо $f(x_2) < f(x_m)$, то виключаємо відрізок $[a, x_m]$, покладемо $a = x_m; \; x_m = x_2$; якщо $f(x_2) \geq f(x_m)$, то виключаємо відрізок $[a, x_1]$та $[x_2, b]$, покладемо $a = x_1; b = x_2$, тобто формуємо новий відрізок дослідження. Обчислюємо нове значення L = b – a. Даний алгоритм ітераційний, тому зазвичай в якості умови закінчення ітераційного процесу обирають умову $|a - b| < \varepsilon$, тобто звуження відрізку виконується до тих пір, поки його величина не зменшиться до заданої обчислювальної похибки ε. Схема алгоритму методу половинного ділення показана на рис. 6.3 і є типовою для ітераційних процесів.

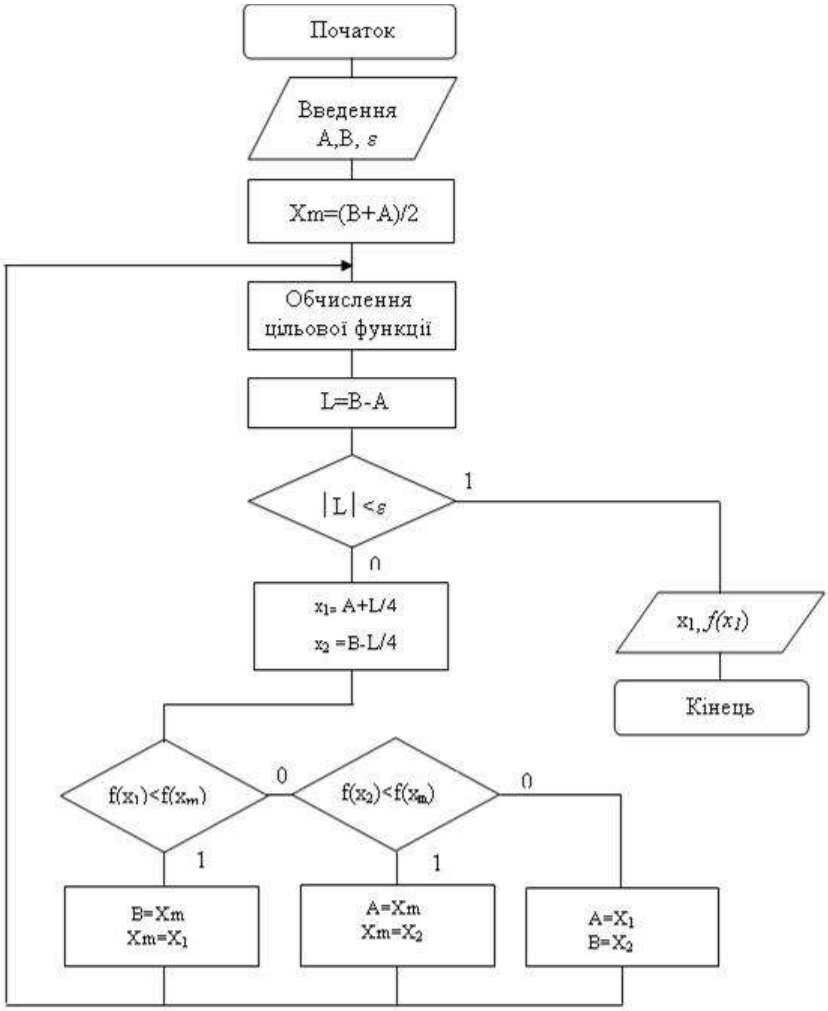
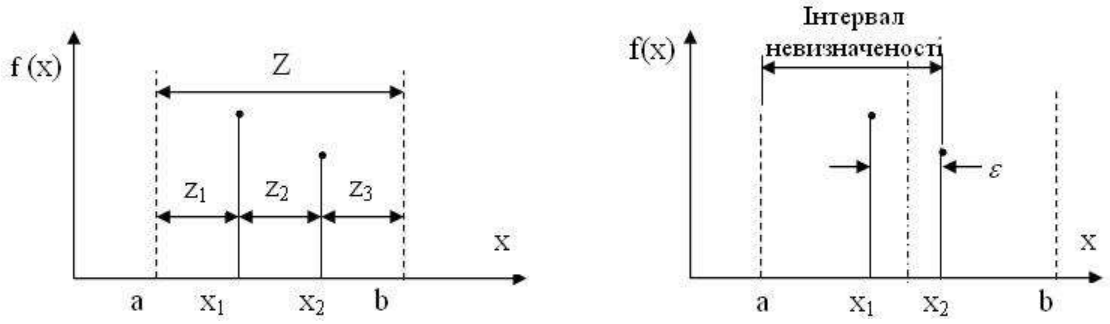


Рис. 6.3 Схема алгоритму методу половинного ділення

6.4 Метод дихотомії

Обчислення цільової функції в двох точках інтервалу невизначеності дозволяє його звузити. Можна ці точки обрати таким чином, що інтервал невизначеності буде мінімальним. На рис. 6.4,а показані позначення, які використовуються в цій схемі.



а. б.

Рис. 6.4 Геометрична інтерпретація методу дихотомії

Якщо значення цільової функції при x_1 більше, ніж при x_2, то новий інтервал невизначеності дорівнює $Z_1 = z_1 + z_2$. В протилежному випадку він визначається виразом $Z_2 = z_2 + z_3$. Задача полягає в тому, щоб одночасно мінімізувати Z_1 і Z_2, задовольнивши умовам

$z1+z2+z3 = Z,\ z_1 > 0,\ z2 > 0,\ z_3 > 0.$

Із рівності можна виключити z_2. Тоді

$Z - z_3 = min,\ Z - z_1 = min.$

Так як величина Z задана, то праві частини цих рівнянь будуть тим менше, чим більше z_1 і z_3. Отже, оптимум відповідає умові

$z1 = z3 = 0{,}5Z.$

Але тоді $z_2 = 0$, що суперечить умові $z_2 > 0$.

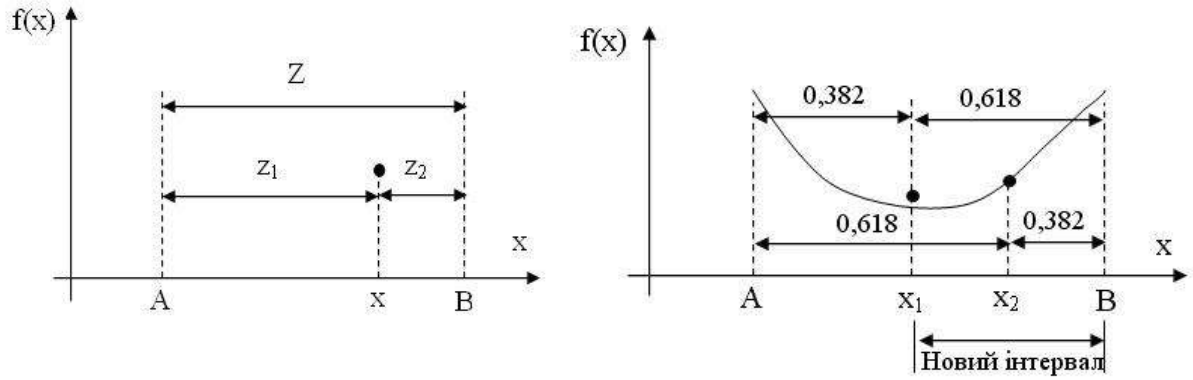
Нехай z_2 має деяке дуже маленьке значення ε. Тоді із z_1 і z_3 віднімемо по ε/2. В результаті після обчислення першої пари значень цільової функції при близьких значеннях x інтервал невизначеності звузиться, як показано на рис. ,б і коефіцієнт ділення буде дорівнювати

$$f = \frac{l}{2} + \frac{\varepsilon}{2} \qquad (6.3)$$

В межах, при $\varepsilon \to 0 \quad f \to 1/2$. Надалі, при використанні метода дихотомії виконуються ті ж ітераційні операції, що і при використанні методу ділення відрізка навпіл.

6.5 Метод “золотого перетину”

З кожних трьох значень цільової функції, які були обчислені в інтервалі невизначеності в подальшому використовуються лише два, а третє не дає додаткової інформації і в подальшому не використовується. В методі золотого перетину цільова функція обчислюється в точках інтервалу невизначеності, які розташовані таким чином, щоб кожне обчислене значення цільової функції давало нову корисну інформацію.



а. б.

Рис. 6.5 Геометрична інтерпретація методу золотого перетину

Суть цього методу полягає в наступному. Інтервал невизначеності ділиться на дві нерівні частини, відношення довжини більшого відрізку до довжини всього інтервалу дорівнює відношенню довжини меншого відрізку до довжини більшого відрізку. На рис. 6.5,а показаний інтервал невизначеності Z, який складається з відрізків z_1 і z_2, відношення довжин яких визначається правилом золотого перетину $\frac{z_1}{Z} = \frac{z_2}{z_1}$.

Крім того, $z_1 + z_2 = Z$. Із першого рівняння витікає $z_1^2 = Z \cdot z_2$. Підставивши значення Z з другого рівняння і поділивши обидві частини на z_1^2, отримаємо

$$\left(\frac{z_2}{z_1}\right)^2 + \frac{z_2}{z_1} - 1 = 0 \qquad (6.4)$$

Розв’язуючи це квадратне рівняння, знаходимо для додатнього кореня значення

$$\frac{z_2}{z_1} = \frac{-1+\sqrt{5}}{2} = 0{,}618 \qquad (6.5)$$

На рис. 6.5,б показано ділення інтервалу невизначеності в цьому відношенні і нанесені відповідні значення цільової функції, які дозволяють зменшити інтервал невизначеності в 1/0,618 раз (Число 1,618 отримало назву **золотого числа**). На цій стадії ще не видно переваг методу золотого перетину в порівнянні з методом дихотомії, однак їх добре видно при подальшому діленні інтервалу, так як виявляється, що одне із значень цільової функції, яке необхідно обчислити на наступному кроці, вже відомо. Тому, щоб зменшити невизначеність ще в 1/0,618 раз, потрібно додатково обчислити тільки одне значення цільової функції в точці, яка визначається правилом золотого перетину.

На рис. 6.6, рис. 6.7 представлені алгоритми вибору наступної точки пошуку. Задана точність може, звичайно, змінюватися вибором значення ε. Для функції $f(x) = -e^{-x}\ln(x)$, пошук відбувався в інтервалі (0, 2).

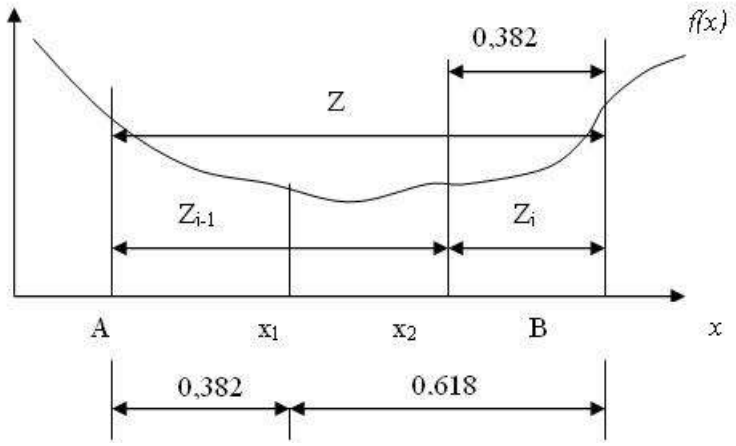


Рис. 6.6 Послідовність етапів вибору наступної точки пошуку
у методі золотого перетину

Істинний мінімум знаходиться в точці 1,76322211, де значення функції дорівнює

-0, 0972601313.

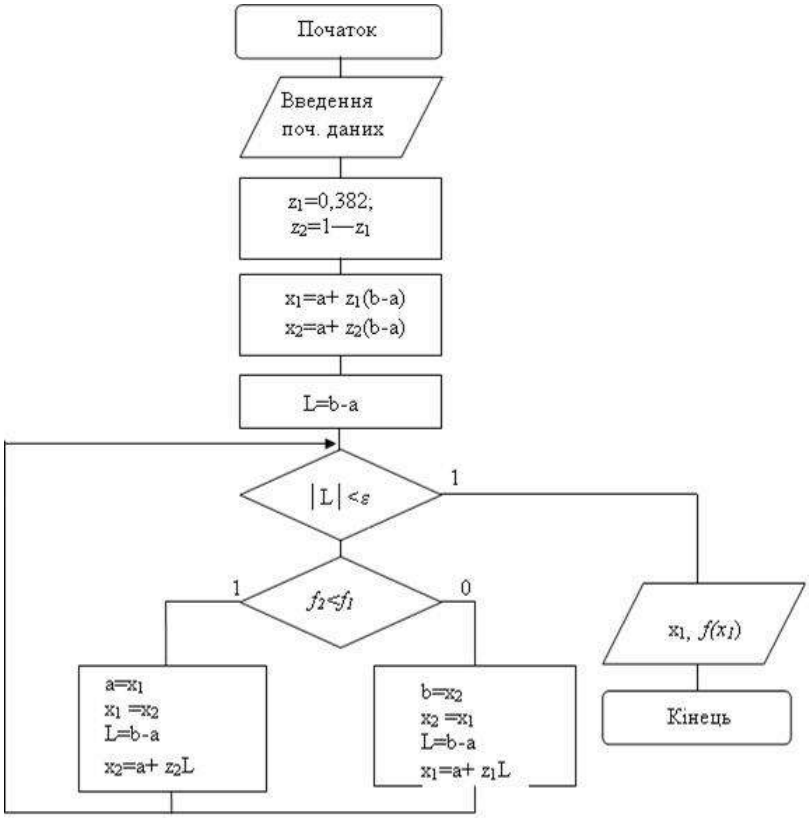


Рис. 6.7 Схема алгоритму методу золотого перетину

При розробці програм для рішення задач однопараметричної оптмізації використовують наступні співвідношення:

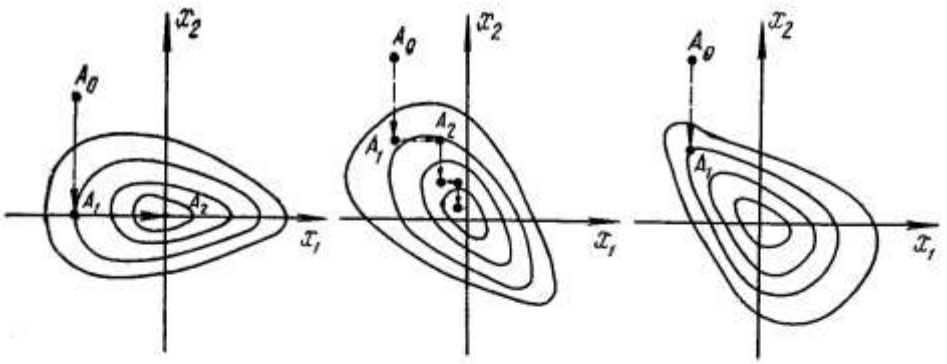
$$z_1 = 0{,}382; \quad z_2 = 1 - z_1 = 0{,}618; \quad x_1 = A + z_1(b-a); \quad x_2 = A + z_2(b-a)$$

6.6 Метод сканування.

Якщо цільова функція не є унімодальною, або вона багато параметрична, застосовують більш універсальні методи, відносно до яких декотрі із розглянутих методів є частковими випадками. Так, наприклад, метод сканування – це той же метод **загального пошуку**, узагальнений для n – змінних цільової функції. Для двох параметричної цільової функції метод сканування (або метод повного перебору) полягає в тому, що обчислюють всі значення функції $F(x_i)$ у вузлах квадратної сітки з кроком ε. З обчислених значень F_i вибирають найбільше F_0. Координати відповідного вузла сітки x_{10} і x_{20} — це координати екстремуму, визначені з точністю до $\pm\varepsilon$. Чим менше крок сітки, тим вище точність обчислення, проте при зменшенні кроку швидко росте необхідна кількість обчислень. Дійсно, якщо інтервал зміни кожної змінної розбитий на 10 рівних частин, то при двох змінних необхідна кількість обчислень функції F дорівнює 100, при трьох змінних — 1000,

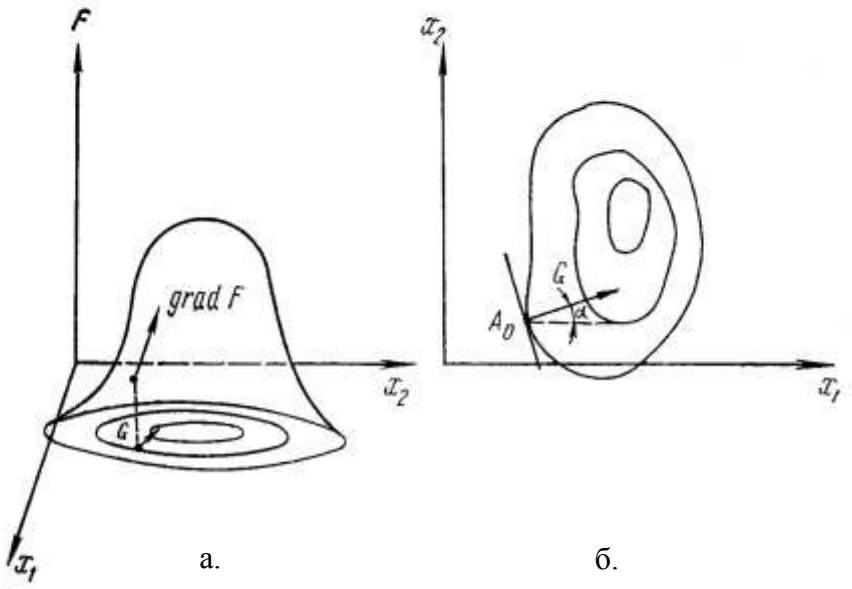
при *n* змінних — 10n. Тому метод повного перебору застосовується тільки при малому числі змінних (у разі одновимірного або двовимірного об'єкту).

6.7 Метод релаксації (метод Гауса). Назва методу походить від слова релаксація, тобто ослаблення (розслаблення). Мається на увазі, що на кожному етапі «розслаблюється» одна з координат і точка дістає можливість рухатися по цій координаті. Наприклад, починаємо пошук від початкової точки *A0* з координатами *x10* і *x20*, рис. 6.8,б.



а. б. в.

Рис. 6.8 Залежність числа кроків від положення цільової функції відносно осей координат (розворот функції)



а. б.

Рис. 6.9 Приклад градієнту цільової функції

Переміщення продовжується до тих пір, поки функція *F* зростає до точки *A1*(*x11, x21*). У цій точці напрям руху змінюємо. Тепер друга змінна залишається постійною і рівною *x21*, а перша змінюється у напрямі зростання функції *F* до точки *A2*(*x12, x22*). У цій точці знову змінюється напрям руху. Змінна *x1* залишається постійною і рівною *x12,* а змінна *x2* змінюється. Процедура повторюється до тих пір, поки при русі по обох координатах функція

F перестане збільшуватися. Метод релаксації пошуку дуже простий і його легко автоматизувати. Основний недолік методу полягає в тому, що число кроків в схемі релаксації сильно залежить від положення лінії рівня щодо осей координат. На рис. 6.8 показана одна і та ж поверхня в трьох різних положеннях відносно початки координат. У випадку, зображеному на рис. 6.8,а пошук екстремуму закінчується за два кроки (у термінах нарисної геометрії ми попали на лінію найбільшого скату) у випадку на рис. 6.8,б число кроків дуже велике (теоретично може бути нескінченним); у випадку на рис. 6.8,в взагалі не потрапляємо в екстремум: рух з точки *A1* в обох напрямах приводить до зменшення функції *F*, хоча ця точка і не є точкою максимуму.

6.8 Градієнтний метод.

Градієнтом функції називається вектор, проекціями якого на координатні осі є часткові похідні функції по координатах

$$grad\ \mathrm{F} = \left(\frac{\partial \mathrm{F}}{\partial \mathrm{x}_1}, \frac{\partial \mathrm{F}}{\partial \mathrm{x}_2}, ..., \frac{\partial \mathrm{F}}{\partial \mathrm{x}_\mathrm{n}} \right). \tag{6.6}$$

Напрям градиента—это напрям найбільш швидкого зростання функції, найбільш «крутого схилу» поверхні цільової функції (рис. 6.9,а). Проекція градієнта *G* на площину змінних *x1* і *x2* перпендикулярна дотичною до лінії рівня функції *F* (рис. 6.9,б). Напрям вектора *G* визначається кутом α, складеним вектором *G* з позитивним напрямом осі *x1*.

При пошуку екстремуму градієнтним методом в початковій точці *A0* визначають похідні $\frac{\partial \mathrm{F}}{\partial \mathrm{x}_1}$ потім роблять крок у напрямі градієнта. Координати нової точки *A1*, обчислюють по формулах:

$$\begin{aligned} \mathrm{x}_{11} &= \mathrm{x}_{10} + \mathrm{h} \frac{\partial \mathrm{F}}{\partial \mathrm{x}_1} \Bigg/ \sqrt{\left(\frac{\partial \mathrm{F}}{\partial \mathrm{x}_1} \right)^2 + \left(\frac{\partial \mathrm{F}}{\partial \mathrm{x}_2} \right)^2}; \\ \mathrm{x}_{12} &= \mathrm{x}_{20} + \mathrm{h} \frac{\partial \mathrm{F}}{\partial \mathrm{x}_2} \Bigg/ \sqrt{\left(\frac{\partial \mathrm{F}}{\partial \mathrm{x}_1} \right)^2 + \left(\frac{\partial \mathrm{F}}{\partial \mathrm{x}_2} \right)^2} \end{aligned} \tag{6.7}$$

Розмір кроку обирають достатньо малим. В точці А1 знову обчислюють похідні і визначають нове направлення градієнту. Звичайно крок обирають так, щоб при переході від одної точки до другої напрям градієнту змінювався б на 10…15°.

Переміщення продовжується до тих пір, поки похідні $\frac{\partial F}{\partial x_1}$ і $\frac{\partial F}{\partial x_2}$ не стануть близькі до нуля (що говорить про досягнення стаціонарної точки) або поки не буде досягнута межа області завдання змінних *x1, x2.* На рис. показана траєкторія переміщення від початкової точки *A0* до максимуму функції *F*.

6.9 Метод най скорішого підйому (найскорішого спуску).

Недолік градієнтного методу полягає в тому, що на кожному кроці треба обчислювати

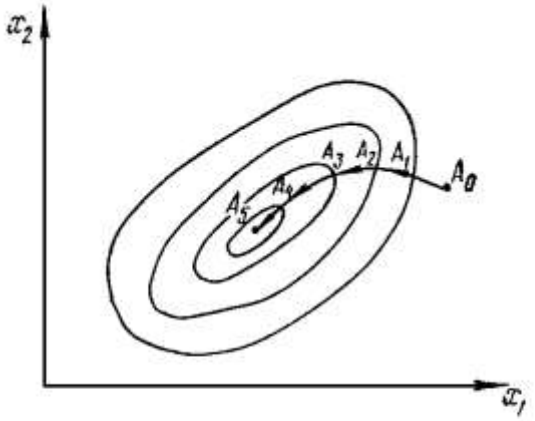


Рис. 6.10 Пошук екстремуму градієнтним методом

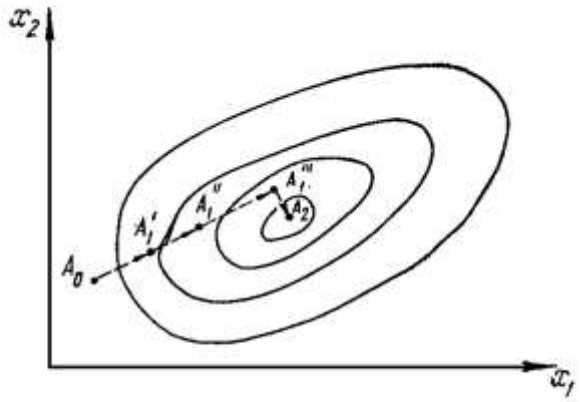


Рис. 6.11 Пошук екстремуму методом най скорішого спуску

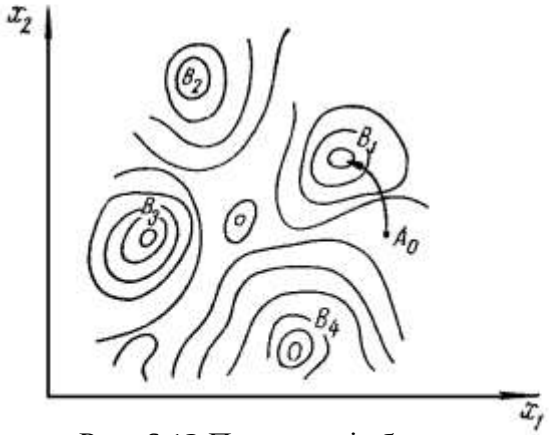


Рис. 6.12 Поверхня із багатьма екстремумами

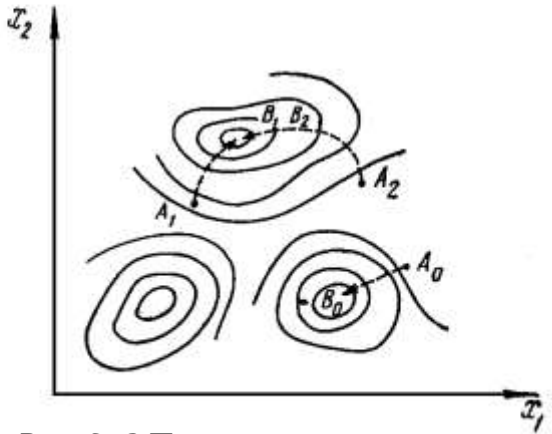


Рис. 6.13 Пошук екстремуму випадковим методом із підйомом по градієнту.

всі похідні функції *F* і визначати напрям градієнта (при великому числі змінних *x* це трудомістка операція).

Від цього недоліку вільний метод най скорішого підйому, також відомий під назвою методу най скорішого спуску, оскільки він розроблявся для завдання мінімізації.

Так само як в градієнтному методі, в початковій точці *A0* визначають напрям градієнта і переміщаються в цьому напрямі, рис. 6.11. Проте переміщаються не на один крок, а на декілька кроків до тих пір, поки функція *F* зростатиме. Коли функція *F* починає зменшуватись (у точці A1'"), знов обчислюють часткові похідні і визначають новий напрям градієнта. На рисунку показана траєкторія переміщення до оптимуму. При використанні цього методу, як правило, забезпечується найшвидший пошук екстремуму.

6.10 Методи випадкового пошуку.

Методи най скорішого спуску, градієнтний і релаксації придатні для відшукання екстремуму тільки унимодальных функцій. Якщо ж максимумів декілька, то залежно від положення початкової точки ці методи приводять в ту або іншу точку локального екстремуму, тоді як функція мети може мати найбільше значення (глобальний екстремум) в іншій точці. На рис. 6.12 показана траєкторія пошуку з точки *A0,* яка приводить у вершину *B1*, тоді як найбільшого значення функція *F* отримає у вершині *B4*. Тому всі ці методи носять назву методів локального пошуку.

Для відшукання глобального екстремуму можна скористатися методом повного перебору або яким-небудь варіантом випадкового пошуку. У простому варіанті випадкового пошуку початкова точка *A0* вибирається випадково. У ній вимірюється значення функції *F* і це значення запам'ятовується. Наступна точка *A1* і все подальші вибираються також абсолютно випадково. Зміряне в кожній точці нове значення функції *F* порівнюється з тим, яке зберігається в пам'яті. Якщо нове значення більше старого, воно заноситься в пам'ять замість старого; якщо нове значення менше, воно просто відкидається.

Така проста схема випадкового пошуку застосовується рідко, зазвичай використовуються складніші модифікації. Так, наприклад, один з можливих варіантів — це випадковий пошук з проміжними підйомами. У цьому варіанті вибирають випадкову точку *A0,* проводять з неї під'їм по градієнту до найближчої локальної вершини *B0* і запам'ятовують значення *F* в цій вершині. Потім вибирають нову випадкову крапку і з неї піднімаються до іншої локальної вершини *B1* (рис. 6.13). Значення функції в цій вершині запам'ятовують, якщо воно більше, ніж в попередній і так далі. Кожен етап складається з випадкового кроку і градієнтного пошуку.

Порівняння пошукових методів оптимізації.

Різні методи пошуку мають свої переваги і недоліки. Вибір методу, в першу чергу, залежить від виду цільової функції. Якщо відомо, що функція має один екстремум, можна використовувати один з локальних методів пошуку: релаксації, градієнтний, най скорішого спуску. При цьому, якщо функція мети має вигляд

$$F(x_1, x_2, ..., x_n) = f(x_1) + f(x_2) + ... + f(x_n) \quad (6.8)$$

або мало відрізняється від такої функції, то для відшукання її екстремуму краще всього застосувати метод Гауса. Якщо всі змінні тісно зв'язані і фізично однорідні, то краще всього застосовувати метод най скорішого спуску. Поблизу оптимуму краще застосовувати градієнтний метод. При великій розмірності завдання найбільш ефективний метод випадкового пошуку.

Якщо ж функція має декілька екстремумів, необхідно застосовувати повний перебір або які-небудь варіанти випадкового пошуку.

Для функцій, що мають гострі круті гребені або яри, застосовуються в поєднанні випадковий пошук і градієнтний метод. До теперішнього часу вибір найбільш ефективного методу пошуку у великій мірі визначається досвідом, мистецтвом і інтуїцією проектувальника. Останніми роками активно розвивається метод пошуку екстремумів, заснований на генетичних алгоритмах. Зараз він є найбільш ефективним в тих випадках, коли про функцію і її екстремуми немає ніякої інформації. У поєднанні з описаними вище алгоритмами, метод генетичних алгоритмів – найефективніший, але такий, що вимагає значних обчислювальних ресурсів.

ЛЕКЦІЯ 7. ВИЗНАЧЕННЯ ЗАГАЛЬНОГО ТЕХНІКО-ЕКОНОМІЧНОГО РІВНЯ ПРОЕКТУ

При прийнятті рішення про доцільність впровадження проекту бажано (а у особливо відповідальних випадках обов'язково), щоб це робилось після співставлення декількох, хоча б двох, варіантів проекту. Бажано також, щоб ці варіанти були розроблені різними незалежними виконавцями, а оцінку проектів виконував сам замовник, або якась третя сторона. І тут постає проблема – якими мірилами вимірювати показники якості різних проектів, які мають різні шкали та одиниці виміру, або деякі із яких не піддається строгому обчисленню. Або якась властивість, наприклад енергетична ефективність виробу, може бути відображена у числах, але напевне невідомо, наскільки вагомим є цей показник серед інших. Або деякі властивості принципово можна обчислити, але для цього бракує інформації, потрібно багато часу і так далі. Проблема вирішується за допомогою визначення інтегрованих показників якості із залученням методів аналітичних та експертних оцінок.

7.1 Економічний критерій оптимальності проектного рішення.

Проектування закінчується прийняттям проектного рішення. Часто вимоги технічного завдання на проектування можуть бути задовільнені різними технічними засобами. Типовий приклад – конкуренція електроприводів постійного та змінного струму. Оптимальність проектного рішення, якщо технічні вимоги виконуються однаково успішно різними системами електроприводів, оцінюється із залученням економічних показників. Виправдав себе метод **приведених витрат**, тобто всіх витрат, які бере на себе користувач, умовно приведених до одного року. Витрати поділяються на дві частини – капітальні і експлуатаційні. Капітальні витрати ($К_в$) витрачаються у відносно короткий термін і йдуть на придбання обладнання, його транспортування, монтаж, налагодження і ввід в експлуатацію. Експлуатаційні витрати ($Е_в$) відносно постійні і складаються із оплати обслуговуючого персоналу, витрат всіх видів енергії, обслуговування та ремонту, а також так званих накладних витрат. Експлуатаційні витрати зручно рахувати як річні, тобто за один рік експлуатації. Капітальні витрати теж треба розбити на кожний рік експлуатації, звідси термін **приведені** (до одного року) **капітальні витрати**. Проблема полягає в визначені кількості років, до яких приводяться капітальні витрати. Якщо взяти короткий термін, то капітальні витрати, що приведені до одного року, виявляться надто великими, і можуть стати

економічно невигідними, або невиконаними. Якщо взяти повний термін експлуатації (термін життя техніки), а він може вимірюватись десятками років, то приведені капітальні витрати знижуються, але зніжується і мотивація до оновлення техніки (збільшується термін очікування чистого прибутку). Реально обирають термін, який отримав назву **термін окупності** (Т), тобто термін, впродовж котрого користувач отримує економічну (фінансову) вигоду від нової техніки, яка окупить зроблені капітальні витрати. У сучасній термінології термін окупності це є термін часу, який потрібен для того, щоб доходи, що генеруються інвестиціями, перекрили витрати на інвестиції. Після терміну окупності користувач починає отримувати чистий прибуток. Таким чином, приведені витрати (ПВ) розраховуються за виразом

$$ПВ = E_в + \frac{K_в}{T} \qquad (7.1)$$

Серед конкуруючих проектів перемагає той, хто має менше значення приведених витрат. Ще краще, якщо конкуруючий варіант має не тільки зменшені експлуатаційні витрати, але і підвищену продуктивність та якість виробництва продукції. У такому випадку термін окупності капітальних витрат, навіть великих, значно скорочується. Терміни окупності різних видів техніки та виробництв слугують інструментом проведення економічної політики у межах окремого виробництва, галузі або держави. Короткі терміни окупності стимулюють швидкий підйом виробництва, але потребують більших приведених капіталовкладень. Вони характерні саме для нашої сучасності - умов ринкової економіки та економічної нестабільності. В таких умовах власник може наважитись навіть на великі витрати, але тільки із коротким терміном окупності. Прикладом, що став вже класичним, є впровадження частотно керованого асинхронного електроприводу замість релейно – контакторних схем на підйомно – транспортних механізмах, а також у вентиляційних та насосних установках замість некерованих систем електроприводу. Незважаючи на велику вартість напівпровідникових перетворювачів частоти, за рахунок зменшення витрат електроенергії, оптимізації показників динамічних режимів, що призводить до значного зменшення аварій та відмов, і підвищення загального рівня культури експлуатації, терміни окупності іноді становили до 0,5 року. Великі терміни окупності використовуються при стабільному плановому розвитку а також для техніки із довгим терміном експлуатації. В Радянському Союзі, для якого була характерною планова економіка, терміни окупності були нормованими. Для техніки і технологій загально промислового значення регламентувались терміни окупності від 5 до 7 років. Для великих об’єктів, наприклад енергетичних, нормативні терміни були значно більшими. Якщо проектне рішення не забезпечувало регламентований термін окупності, воно не приймалось до впровадження.

7.2 **Визначення технічного рівня проектного рішення методами кваліметрії**

Кваліметрія (від латинського "qualis" - який за якістю і грецького "метрео" - міряти, вимірювати) - наукова дисципліна, в рамках якої вивчаються методологія і проблематика комплексної, кількісної оцінки якості об'єктів будь-якої природи: одушевлених або неживих, предметів або процесів, продуктів праці або продуктів природи, що мають матеріальний або духовний характер.

Термін "**якість"** позначає сукупність властивостей будь-якого об'єкту, що виявляються в процесі споживання (експлуатації, використання, застосування) об'єкту і характеризують результати (позитивні і негативні), що досягаються при споживанні, але не витрати на його виробництво і споживання. Якість, наприклад продукції, визначають три чинники:

- якість проекту;
- якість матеріалів (сировини, напівфабрикатів, що комплектують);
- якість роботи (дотримання проекту і норм, тобто недопущення браку).

На якість готової продукції ці три чинники впливають не однаковою мірою. У переважній більшості випадків **найбільш важливим (до 70%) є перший чинник – якість проекту.**

У Кваліметрії розроблено декілька підходів до кількісного оцінювання якості. Найбільш розповсюджений з них базується на наступних принципах:

- якістю є сукупність тільки тих **властивостей об'єкту**, які пов'язані з результатом (але не з понесеними при цьому витратами), що досягається при його допомозі, і які виявляються в процесі споживання (експлуатації, використання) об'єкту відповідно до його призначення;
- деякі складні і будь-які прості властивості можуть бути зміряні за допомогою **абсолютного показника властивості** Q_i (i = 1, n; n - кількість властивостей оцінюваного об'єкту). Отримані в результаті цього значення показника Q_i виражаються в специфічних для кожної властивості одиницях. Для вимірювань можуть використовуватися метрологічні, експертні, аналітичні методи.
- всі властивості, що формують якість, утворюють ієрархічну структуру у вигляді дерева властивостей. Нижчий ярус цього дерева (корінь дерева) представляє найскладнішу

властивість - якість об'єкту, а гілки вищого ярусу представляють прості і квазіпрості властивості;

Методичне зауваження. Ствердження про те, що треба враховувати тільки ті властивості, які виявляються в процесі експлуатації об'єкту, може викликати суперечності. Наприклад, якщо йдеться про прилад, механізм, електромеханічну систему, як продукцію виробництва, то дійсно, треба враховувати властивості, які проявляються в їх експлуатації, "за воротами" підприємства. Якщо розглядаємо проект, то в поняття його "експлуатації" входять і питання виробництва (матеріалоємність, уніфікація приладів та блоків, технологічність виготовлення тощо) і можливості експорту майбутньої продукції і, мабуть ще деякі властивості, які не вписуються у поняття експлуатації готового виробу.

При оцінюванні властивостей можливе застосування декількох шкал:

- шкала відносин ("в скільки разів");
- шкала інтервалів ("на скільки");
- шкала порядку, тобто інформація про те, хто краще за якістю, але не на скільки краще і не в скільки разів краще.

Для зіставлення різних властивостей, вимірюваних в різних по розмаху і розмірності шкалах, використовується **відносний безрозмірний показник властивості Ki,** що відображає ступінь наближення **абсолютного показника властивості Qi**, до **еталонного показника Qiет** і **показника бракування Qiбр**, що характеризують найвищий і найнижчий рівні оцінки. Відносний показник описується залежністю Ki=f(Qi,Qiет,Qiбр), яка у разі застосування спрощеного методу кваліметрії може бути представлена **нормуючою функцією**

$$K_i = \frac{Q_i - Q_i^{бр}}{Q_i^{ет} - Q_i^{бр}} \qquad (7.2)$$

Для зіставлення по відносній важливості всіх властивостей, що входять в "дерево властивостей", використовуються безрозмірні **коефіцієнти вагомості властивостей Gi**. Для зручності зазвичай приймається 0<Gi<1, а $\sum_{i=1}^{n} G_i = 1$. Значення коефіцієнтів вагомості визначаються із залученням різновидів експертного і не експертного (аналітичного) методів.

Кількісна оцінка якості об’єкту в цілому виражається за допомогою **показника якості**

$$K_{jя}=\varphi(K_i,G_i,K_{jеф}), \qquad (7.3)$$

де $K_{jеф}$ - коефіцієнт збереження ефективності j -го об'єкту ($0\le K_{jеф}\le 1$).

Функція φ може виражатися різними поліномами, середніми і так далі. При застосуванні **спрощеного методу Кваліметрії** ця функція дуже часто може бути виражена формулою

$$K_{jя} = K_{jее} \sum_{i=1}^{n} K_i G_i \qquad (7.4)$$

Якщо, окрім якості об'єкту необхідно враховувати витрати на його виробництво і споживання (використання, експлуатацію) - так звані сукупні витрати (окремим випадком яких є використовувані в теорії економічної ефективності **приведені витрати**), то замість показника якості $K_к$ використовується показник інтегральної якості, визначення значень якого ґрунтується на тих же принципах.

Приведемо деякі рекомендації, якими слід керуватися при виконанні окремих елементів Кваліметрії.

Побудова дерева властивостей. Оцінки якості в дуже сильному ступені залежать від тих показників (критеріїв) властивостей, сукупність яких і утворює модель якості оцінюваного об'єкту. Ця залежність настільки велика, що цілком можлива наступна ситуація: при одному наборі показників об'єкт А буде краще за якістю, чим об'єкт Б. А при іншому наборі - може бути навпаки: Б краще А. Ясно, що набір показників, по яких оцінюється якість, повинен бути однозначно представлений (якщо не стандартизований). Більш того, цей набір повинен бути якось впорядкований (декомпозированій) в деяку ієрархічну структуру (дерево властивостей). Іноді, порівнюючи за якістю два (або більше) варіанти об'єкту одного типу, для зменшення трудомісткості розрахунків виключають з розгляду ті властивості, які однаковою мірою виражені в порівнюваних варіантах. При цьому трактують результати оцінювання таким чином, неначе вони виражені в шкалі відносин (тобто дозволяють отримати інформацію не тільки на **скільки** один варіант відрізняється за якістю від іншого, але і в **скільки разів** виявляється ця відмінність). Насправді ж, при подібному виключенні однаково виражених властивостей зазвичай неможливо отримати результати оцінювання не тільки в шкалі відносин, але і в шкалі інтервалів. В цьому випадку оцінки якості можуть бути виражені тільки в шкалі порядку, тобто давати інформацію про той, хто краще за якістю, але не на скільки краще і, тим більше - не в скільки разів краще.

Визначення еталонних значень і значень бракування показників властивостей. Еталонні значення показників найчастіше приймаються за допомогою одного з двох алгоритмів.

- Вибирається декілька об'єктів - аналогів по відношенню до оцінюваного об'єкту; з них визначається кращий за якістю; значення показників окремих властивостей цього кращого об'єкту приймаються як еталонні значення.
- Для вибраних об'єктів-аналогів, визначаються кращі для всієї сукупності цих аналогів значення показників кожної властивості. Ці значення і приймаються як еталонні.

У теоретичній кваліметрії доведені теореми, з яких виходить, що використання обох алгоритмів може привести до крупних помилок в підсумкових результатах (хоча вірогідність такої помилки при використанні першого алгоритму більша, ніж при використанні другого). Там же показано, що правильний принцип призначення еталонних значень може бути тільки такий: **значення повинні вибиратися як кращі в світі** (на момент оцінювання якості) **значення показника відповідної властивості**. Сказане відноситься і до значень бракувань показників властивостей. Визначення показників властивостей. У дуже багатьох методиках оцінювання якості застосовується не цифрова, а вербальна технологія виразу градацій значень **абсолютних показників властивостей**. Наприклад, часто використовується вербальна шкала з п'ятьма градаціями: дуже добре, добре, задовільно, незадовільно, дуже незадовільно. Іноді замість такої вербальної використовують еквівалентну їй цифрову п’ятибальну шкалу. Але вже за рахунок подібного невеликого числа градацій відносна погрішність збільшується до ±20%. Щоб зменшити величину відносної погрішності, потрібно, за інших рівних умов, збільшити число градацій. Але не в будь-яких, тобто максимальних розмірах, а в тих, які відповідають психологічним можливостям людини. А ці можливості зумовлюють, що оптимальне число градацій повинне бути в межах 10-12, тобто повинна використовуватися знайома всім п’ятибальна шкала, доповнена проміжними значеннями "+" або "-". Схожих результатів можна добитися, якщо використовувати не п’ятибальною (з "+" і "-"), а 100%-ною шкалою з градаціями через 10% (окрім початку і кінця шкали, де можливі і дрібніші градації). Зрозуміло, все сказане відноситься до тих властивостей, для показників яких або дуже важко, або по якихось причинах небажано використовувати для виразу їх значень звичайні фізичні одиниці вимірювання.

7.3 Приклад розрахунку показника якості

Розглядаємо приклади проектів електромеханічного пристрою (електромеханічної системи, системи автоматизації технологічного процесу, тощо) із залученням методик Кваліметрії. **Використовуємо описану вище спрощену методику, та експертні оцінки показників властивостей.** При цьому треба виконати кілька обов'язкових кроків.

Методичне зауваження. В даному прикладі, за для скорочення, розглянуто розрахунок показника якості тільки одного проекту (об'єкту) j=1, тобто $К_{1я}$ по (7.4), оскільки розрахунки показників інших проектів $К_{2я}$, $К_{3я}$ виконуються за тими же алгоритмами, а найкращий (оптимальний) із проектів визначається простим співставленням цих коефіцієнтів.

1. **Відбір та призначення експертів.** Від особистих рис, спеціальності, досвіду експерта залежить наближення до об'єктивної оцінки окремих складових якості проекту. Для отримання більш достовірної інформації тут треба дотримуватись таких принципів:

- експерт повинен мати певний стаж та досвід роботи;
- оптимальна кількість експертів повинна складати 7-10 осіб;
- треба прагнути, щоб у групі експертів були особи з різних спеціальностей, які представляють різні служби та відділи підприємства;
- серед експертів не повинно бути екстремально налаштованих людей (затяти консерватори, або їх антиподи).

2. Визначення показників (критеріїв) якості (побудова дерева якості). Кількість показників обговорюється і затверджується попередньо. Треба прагнути до визначення найбільш повно характеризуючих показників, але уникати їх надто великої кількості. Для прикладу пропонується такий список критеріїв (їх черга у переліку на перших порах не має значення).

2.1 **Патентоспроможність** – кількість захищених патентами технічних та технологічних рішень (замовника, виконавця, галузі або держави) патентів та повнота охоплення ними пропонованих в проекті технічних рішень.

2.2 **Енергетична економічність** виробу.

2.3 **Матеріалоємність** – у кількісному та якісному вираженні, особливо для дефіцитних матеріалів (дорогоцінні та кольорові метали, рідкоземельні елементи, тощо).

2.4 **Технологічність виготовлення** виробу – ступінь уніфікації окремих складових, наявність відповідних технологій на виробництві або навпаки потреба створення унікальної технології.

2.5 **Ступінь охоплення** виробу та процесу його експлуатації сучасними **ІТ технологіями.**

2.6 **Надійність**, тобто спроможність відпрацювати нормовані терміни експлуатації із мінімумом обсягів простоїв та ремонтних робіт.

2.7 **Екологічність**, тобто вплив на середовище, що оточує на етапах виготовлення та експлуатації.

2.8 **Ергономічність, естетичність**, тобто максимальне наближення до фізіологічних можливостей та естетичних уподобань людини.

2.9 **Функціональність** – тобто у виробі все пристосовано для виконання головної мети, нема нічого зайвого, надто складного і т.д.

Обмежмось тільки 6 –ю першими показниками і надалі, при розрахунках, будемо використовувати їх номери для позначення коефіцієнтів вагомості G_i, абсолютних та відносних показників властивостей Q_i та K_i . Іншими трьома показниками нехтуємо, ради скорочення викладок, хоча деякі із них, наприклад екологічність та ергономічність теж можуть мати значну вагу.

3. Визначення вагомості показників якості. Тут у повній мірі проявляються індивідуальні досвід, нахили та уподобання окремих експертів. Наприклад, технолог підприємства може віддавати перевагу, як більш важливим, номерам 3- матеріалоємність та 4- технологічність; досвідчений, але більш консервативний спеціаліст може недооцінити вагомість використання ІТ технологій. Більше того, керівник, що затверджує склад експертів вільно або не вільно може вплинути на висновки, змінюючи саме склад експертної групи.

Методика полягає в тому, що кожному експерту (Ек.1 - Ек.7) пропонується заповнити стовпчик таблиці, в якому проти кожного показника проставляється номер від 6 до 1 по порядку зниження його вагомості. Цифра 6 відповідає найвагомішому показнику, а 1 найменш вагомому, таблиця 7.1

Таблиця 7.1 Розрахунок показників вагомості властивостей виробу (продукції)

Показник	Ранжировка показників експертами							Сума оцінок	Вагомість G_i
	Ек. 1	Ек. 2	Ек. 3	Ек. 4	Ек. 5	Ек. 6	Ек. 7		
1.Патентоспроможність	1	3	6	1	2	1	3	17	**0,116**
2.Енергетична економічність	5	2	5	6	4	4	6	32	**0,218**
3.Матеріалоємність	3	5	2	5	3	3	5	26	**0,177**
4.Технологічність	2	6	1	2	1	2	2	16	**0,108**
5.ІТ технології	4	1	3	4	6	5	1	24	**0,163**
6.Надійність	6	4	4	3	5	6	4	32	**0,218**
Сума	**21**	**21**	**21**	**21**	**21**	**21**	**21**	**147**	**1**

Методичне зауваження. Таблиця ранжировки показників різними експертами заповнювалась автором, і звісно, може мати характер випадковості. Але, для прикладу, звертаю увагу на роботу експертів №1, №2 та №3. Перший експерт, мабуть, більше за все цінує експлуатаційні якості майбутньої продукції. Для нього важливішими є показники надійності (6), економічності (2), зручності в експлуатації (5). В останню чергу його турбує патентоспроможність. Другий експерт, припустимо технолог, більш опікується проблемами виробництва, ніж експлуатації (вагоміші показники 3 і 4). Для третього експерта, можливо представника керівного складу, важливими є властивості, вигідні для експортних поставок (патентоспроможність (1), економічність (2) та надійність (6)) – все інше зробимо.

Із таблиці 7.1 випливає, що вагомість окремих показників загалом не відповідає черзі у попередньому списку, і це не суттєво. На першому місці по вагомості економічність та надійність, далі віддається перевага ІТ технологіям, зменшенню матеріалоємності. На останніх місцях патентоспроможність та технологічність. Головне тут те, що кожний із показників отримав конкретний коефіцієнт вагомості, що відображає узагальнені уявлення фахівців про прогресивні напрями розвитку техніки.

4. Визначення абсолютних показників властивостей. Використовуємо 100% або 100 – бальну шкалу оцінювання. Тут можливі ситуації, коли експерти, що дуже цінують деякі властивості і підвищують їх вагомість, при визначенні властивостей об’єкту саме за цими показниками будуть дуже суворо і прискіпливо їх судити.

Методична рекомендація. Для збільшення об’єктивності оцінок рекомендовано використовувати шкалу оцінок, подібну до шкали оцінювання знань студентів. Позитивна оцінка роботи починається від 60 балів, тобто цінка від 60 до 74 балів відповідає оцінці "задовільно". Нижче 60 балів це вже незадовільна робота, причому, якщо оцінка становить від 30 до 59 балів, то можна сподіватись на виправлення, доробку і т.д. Якщо оцінка становить нижче 30 балів, то робота цілком бракується. Відповідно, оцінка від 75 до 89 балів характеризується як "добре", а 90 і вище як "відмінно". Приклад показників властивостей по одному із конкуруючих проектів приведений у таблиці 7.2.

Таблиця 7.2 Розрахунок абсолютних показників властивостей згідно до 1-го проекту

Показник	Абсолютні показники властивостей							Узагальнені показники			
	Ек.1	Ек.2	Ек.3	Ек.4	Ек.5	Ек.6	Ек.7	Q_i	$Q_i^{ет}$	$Q_i^{бр}$	K_i
1.Патентоспроможність	50	90	30	40	50	80	70	58,6	100	30	0,41
2.Енергетична економічність	60	70	80	60	40	70	80	65,7	100	40	0,43
3.Матеріалоємність	80	70	70	50	60	80	70	68,6	100	50	0,37
4.Технологічність	80	60	50	40	50	70	60	58,6	100	40	0,31
5.ІТ технології	60	80	90	70	60	70	80	72,8	100	60	0,32
6.Надійність	70	60	70	50	60	70	80	65,7	100	50	0,31

Абсолютні показники, вказані в таблиці, кожний із експертів формує відносно до якогось уявного ідеального зразку, якість якого оцінюється у 100% або 100 балів. Тут доречно обговорити проблему визначення еталонних показників властивостей $Q_i^{ет}$. В теорії Кваліметрії не рекомендується обирати у якості еталону найбільший із показників, що є в таблиці. Рекомендовано обирати за еталон абсолютний показник об’єкту кращого світового рівня. Мабуть, треба щоб експерти ретельно ознайомились із таким взірцем і визначили його показники, заповнюючи таблицю, аналогічну табл. 7.2. Чи реально це? Ризикнемо взяти за еталонні значення кожного з показників 100%. У цьому випадку нормовані показники K_i по (7.2) будуть нижчими порівняно із визначеними по світовим зразкам. Показник бракування $Q_i^{бр}$ оберемо як найнижчий показник з цієї ж таблиці 7.2. Абсолютні показники властивостей Q_i розраховуємо як середньоарифметичні між сьома експертами.

Методичне зауваження. У прикладі наведена таблиця 7.2 із показниками тільки одного варіанту. Але реально треба заповнювати таблицю із двома, трьома варіантами.

Бажано, щоб колонки із різними варіантами по кожній із властивостей були рядом, для наочного співставлення всіх варіантів.

Судячи по абсолютним показникам властивостей Q_i експерти більш задоволені застосуванням ІТ технологій, менш за все їх влаштовує технологічність та патентоспроможність. Але це не остаточні висновки, оскільки ще треба перевести показники у відносні одиниці та врахувати ступінь відхилення від еталонних та бракувальних значень показників за виразом (7.2). Після розрахунків коефіцієнтів K_i виявилсь, що найбільші відносні значення мають показники економічності та патентоспроможності, останні три місця займають технологічність, ІТ технології та надійність.

Тепер можна визначити загальний коефіцієнт якості проекту по (7.4). Оскільки ми не маємо відомостей про збереження окремих властивостей проектів (у часі?), обираємо для всіх проектів $K_{i\ еф}=1$. Отримуємо коефіцієнт якості першого проекту $K_{1я}$ = 0,360 тобто проект, що розглядається, відповідає на 36% найкращім світовим зразкам. (з урахуванням того, що ми обрали $Q_i^{ет}$ =100%, можна сказати, що проект відповідає на 36% деякому ідеалізованому варіанту). Після розрахунків коефіцієнтів якості інших проектів можна зробити висновки про найкращий або найгірший із них.

СЛОВНИК

термінів у галузі автоматизованого проектування

автоматизоване робоче місце – (АРМ) комплекс технічних засобів, що забезпечують проектувальникові оперативний і легкий доступ до ЕОМ, що допомагає реалізації ітераційних циклів проектування в рамках діалогових режимів роботи, дозволяє обмінюватися з ЕОМ інформацією в графічній формі. Ядром системи є ЕОМ.

адекватність математичної моделі - відповідність математичної моделі (ММ) об'єкту відносно віддзеркалення заданих властивостей об'єкту. Зазвичай ММ вважають адекватними, якщо погрішності розрахунків, при застосуванні випробовуваної моделі, не перевищують обумовлених граничних значень.

алгоритм – точне зібрання інструкцій, що описують порядок дій виконувача для досягнення результату у вирішенні задачі за кінцевий відрізок часу.

анімація (фр. animation – оживлення, одушевлення) – процес надання здібності рухатись, або видимості життя неживим об’єктам.

база даних - інформаційний фонд, що дозволяє оптимізувати процес вводу, виводу і обробки інформації.

банк даних - сукупність баз даних і системи управління базами даних, а також технічних, мовних і організаційних засобів, призначених для централізованого накопичення і колективного багатоаспектного використання даних.

візуальне проектування (програмування) – спосіб створення програми для ЕОМ шляхом маніпулювання графічними об’єктами замість написання її тексту.

візуальне середовище розробки – частковий випадок інтегрованого середовища розробки. Середовище розробки програмного забезпечення, в якому окремі блоки програми представлені у вигляді графічних об'єктів. Приклад - MatLab Simulink.

віртуальний прилад – концепція, згідно якої організовані програмно керовані системи сбіру даних та керування технічними об’єктами і технологічними процесами. Система працює у вигляді моделі реально існуючого або гіпотетичного приладу, причому програмно реалізуються не тільки засоби управління (рукоятки, кнопки, лампочки) але і логіка роботи приладу.

декомпозиція – науковий метод, що використовує структуру задачі, який дозволяє замінити рішення одної великої задачі на рішення серії менших задач.

діалоговий режим - режим взаємодії користувача з обчислювальною системою (ОС), при якому людина і ОС обмінюються даними в темпі, призвичаєному до темпу обробки даних людиною.

ДКР - дослідно-конструкторські роботи.

етап проектування - умовно виділена частина процесу проектування, що складається з однієї або декількох проектних процедур. Зазвичай етап включає процедури, пов'язані з отриманням описів в рамках одного аспекту і одного або декількох сусідніх рівнів абстрагування.

ідентифікація - встановлення відповідності між об'єктом, представленим деякою сукупністю експериментальних даних про його властивості, і одним з описів із заданої безлічі описів (моделей) об'єкту.

інтегроване середовище розробки – (ICP або англ. IDE) система програмних засобів, що використовується для розробки програмного забезпечення. ICP звичайно призначена для однієї певної мови програмування.

інтерактивність – ступінь взаємодії між об’єктами, маючи на увазі під одним з об’єктів людину. **Ступінь інтерктивності** характеризує наскільки швидко і зручно користувач може досягнути своїх цілей. **Елементами інтерактивності** є спеціалізовані програмні модулі.

інтерпретація – пооператорна, покомандна **обробка і виконання** ісходної програми (на відміну від **компіляції** – трансляції програми без її виконання).

інтерфейс (interface) – Засіб стандартного з’єднання пристроїв, що відрізняється уніфікацією засобів і процедур встановлення зв’язку, обміну інформацією і завершення зв’язку. В залежності від рівня спілкування із комп’ютером розрізняють рівень користувача, програмний, апаратний. Приклади: клавіатура і миша – інтерфейс спілкування між користувачем та ЕОМ, графічний інтерфейс – набір засобів для графічних робіт.

інтерфейс інтуїтивний – очевидний за своїм видом та структурою, такий, що не потребує спеціальної підготовки для його використання, або потребує мінімум досвіду.

інтерфейс користувача –сукупність засобів, за допомогою яких користувач взаємодіє із різними програмами, додатками та пристроями.

інформаційне забезпечення - сукупність представлених в заданій формі відомостей, необхідних для виконання автоматизованого проектування, зокрема опису стандартних проектних процедур, типових проектних рішень, типових елементів, комплектуючих виробів, матеріалів і тому подібне. Основною частиною інформаційного забезпечення є бази даних.

IT технології - інформаційні технології (англ. information technology, IT). Згідно до визначення, що прийнято ЮНЕСКО, IT – це комплекс взаємно пов’язаних наукових, технологічних, інженерних дисциплін, які вивчають методи ефективної організації праці людей, що зайняті обробкою та зберіганням інформації, обчислювальну техніку та методи організації взаємодії з людьми та виробничим обладнанням, їх практичні застосування, а також пов’язані із всім цим соціальні, економічні та культурні проблеми.

ітерація – в широкому сенсі означає повторювання дій; у вузькому сенсі це поетапний процес, у котрому результати виконання групи операцій в рамках одного етапу використовуються наступним етапом, окрім останнього, оскільки він є кінцевим результатом.

компоновка - проектна процедура, що полягає у визначенні складу конструктивів проектованої системи.

конструкторське проектування - вибір форми, компоновка і розміщення конструктивів, трасування міжз'єднань, виготовлення конструкторської документації.

критерій – ознака, прикмета, основа, мірило для оцінювання будь – чого. Розподіляються на логічні і емпіричні.

лінгвістичне забезпечення - сукупність мов, термінів, визначень, необхідних для виконання автоматизованого проектування.

математичне забезпечення - сукупність математичних методів, алгоритмів і моделей, необхідних для виконання автоматизованого проектування.

методичне забезпечення - документи в яких відображені склад, правила відбору і експлуатації засобів автоматизації проектування.

моделювання математичне - дослідження фізичного об'єкту шляхом створення його математичної моделі і операції нею з метою отримання корисної інформації про фізичний об'єкт.

НДДКР - науково-дослідні і дослідно-конструкторські роботи.

НДР - науково-дослідні роботи.

об’єктно орієнтоване програмування – парадигма проектування, в котрому головними концепціями є поняття об’єктів і класів. Об’єкт – дещо, що має певну поведінку та засоби її представлення. Об’єкти належать до якоїсь предметної галузі, але можуть бути і віртуальними.

оболонка операційної системи – інтерпретатор команд операційної системи, що забезпечує інтерфейс для взаємодії користувача з функціями системи. Розрізняють два типи інтерфейсу: текстовий інтерфейс користувача (англ. Text user interface – TUI або Character user interface - CUI) і графічний інтерфейс користувача (англ. Graphical user interface - GUI).

операційна система – (скорочено ОС) – система, яка виступає з одного боку як інтерфейс між пристроями обчислення та прикладними програмами, а з другого боку для розподілу обчислювальних ресурсів та керування процесами обчислення. Приклади ОС: MS-DOS, Windows.

оптимальне рішення - вектор керованих параметрів X*=(x1*,xn*), що задовольняє всім обмеженням, доставляє екстремальне значення цільової функції в оптимальний точці, тобто F(X*).

оптимізація - (параметрична оптимізація) - досягнення екстремуму деякої цільової функції.

організаційне забезпечення - сукупність документів, що встановлюють склад проектної організації і її підрозділів, їх функції, зв'язки між ними і комплексом засобів автоматизації проектування, форму, склад і перелік проектної документації.

отладчик або дебаггер (від debugger) – модуль середовища розробки або окремий додаток, призначений для пошуку помилок у програмі. Отладчик дозволяє виконати покрокову трасировку, відстежити та відшукувати значення змінних у процесі виконання програми, встановлювати та видаляти контрольні точки, або умови і т.д.

парадигма – сукупність ідей і понять в певній галузі знань а бо діяльності.

параметр - величина, що характеризує деяку властивість об'єкту або режим його функціонування.

параметрична оптимізація - проектна процедура, направлена на вибір критерію оптимальності і визначення значень параметрів елементів проектованого об'єкту, якнайкращих з позицій вибраного критерію, за умови дотримання всіх обмежень і при заданій структурі об'єкту.

параметричний синтез - визначає чисельні значення параметрів елементів при заданих структурі об'єкту і діапазоні можливої зміни зовнішніх змінних.

периферійний пристрій – зовнішній по відношенню до комп'ютера пристрій разом із з'єднуючим їх інтерфейсом. Периферійний пристрій розширює можливості комп'ютера, але не змінює його архітектуру.

послідовний порт (COM – порт) – двонаправлений послідовний інтерфейс, призначений для обміну бітової інформації. Послідовний тому, що інформація передається послідовно побітово, біт за бітом. COM – порт має стандарт RS-232C.

програмне забезпечення – сукупність програм, процедур та правил, а також документації, які мають відношення до функціонування системи обробки даних. Є одним із видів забезпечення системи обчислення (крім технічного, математичного, інформаційного, лінгвістичного, організаційного, методичного). Як сленг використовується слово "софт", від англ. software.

програмний пакет – набір комп'ютерних програм із одної предметної області, які мають схожі інтерфейси користувача і які можуть легко обмінюватись даними між собою.

програмний продукт – програмний пакет, програмне забезпечення як товар, як предмет комерції.

проектна операція - дія або сукупність дій, складові частини проектної процедури, алгоритм яких залишається незмінним для ряду проектних процедур.

проектна процедура - формалізована сукупність дій, виконання яких закінчується **проектним рішенням.**

проектне рішення - проміжний або остаточний опис об'єкту проектування, необхідний і достатній для розгляду і визначення подальшого напряму або закінчення проектування.

проектування схемотехніки - розробка функціональних і принципових схем.

результат проектування - **проектне рішення** (сукупність проектних рішень), що задовольняє заданим вимогам, необхідне для створення об'єкту проектування.

робоча програма - програма, безпосередньо використовувана при рішенні конкретної проектної задачі.

робочий проект - повний проект конструкторської документації, достатній для виготовлення технічних систем в заданих умовах (в умовах конкретного виробництва).

системне проектування - аналіз тактико-технічних вимог на проектований комплекс, визначення основних принципів функціонування, розробка структурних схем.

структура об'єкту - склад елементів об'єкту і їх зв'язків між собою.

структурний синтез - синтез, при якому визначається структура проектованого об'єкту.

термінал – кінцева частина будь якої системи, яка забезпечує зв’язок системи із зовнішнім оточенням. Комп’ютерний термінал це пристрій вводу - виводу, монітор разом із клавіатурою, взагалі робоче місце користувача.

технічне забезпечення - сукупність взаємозв'язаних і взаємодіючих технічних засобів для введення і зберігання, переробки, передачі програм і даних, організації спілкування людини з ЕОМ, виготовлення проектної документації.

технологічне проектування - розробка маршрутної і операційної технології, вибір оснащення, визначення технологічних баз.

частковий критерій - критерій оптимальності, в якому як цільова функція приймається один з вихідних параметрів, що якнайповніше відображає якість досліджуваного об'єкту. При цьому умови працездатності решти вихідних параметрів входять в обмеження завдання оптимізації.

ПИТАННЯ ДЛЯ САМОПЕРЕВІРКИ

ЧАСТИНА І
ТЕОРЕТИЧНІ ОСНОВИ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ

1. Дайте визначення терміну "проект", або що таке проект?
2. Поясніть відмінність проектних робіт автоматизованих та автоматичних.
3. Обгрунтуйте актуальність автомтаизції проектних робіт.
4. Основні тенденції розвитку обчислювальної техніки й систем автоматизації проектування.
5. Назвіть стадії та етапи проектування.
6. Зміст і занчення календарного плану виконання проектних робіт.
7. Розкрийте зміст поняття життєвого циклу проекту.
8. Дайте характеристику життєвого циклу проекту згідно моделі "водопаду".
9. Дайте характеристику життєвого циклу проекту згідно "ітеративної" моделі.
10. Дайте характеристику життєвого циклу проекту згідно "спіральній" моделі.
11. Особливості і зміст навчального проектування.
12. Розшифруйте і назовіть призначення програм автоматизованого проетування, які мають у назві абревіатури **CAD**, **CAE, CAM.**
13. Приклади програмних пакетів універсального призначення, їх коротка характеристика.
14. Дайте коротку характеристику універсальної програми автоматизованого проектування **AUTOCAD** (англ. Computer-Aided Design).
15. Дайте коротку характеристику універсальної програми автоматизованого проектування **Maple.**
16. Дайте коротку характеристику універсальної програми автоматизованого проектування **Mathcad.**
17. Дайте коротку характеристику універсальної програми автоматизованого проектування **MATLAB** (англ. «Matrix Laboratory»), зокрема середовища Simulink.
18. Дайте коротку характеристику універсальної програми автоматизованого проектування **LABVIEW** (англ. Laboratory Virtual Instrumentation Engineering Workbench**).**
19. Надайте коротку характеристику прикладних пакетів спецального (галузевого) призначення, наведить приклади прикладних програм.
20. Корпоративні прикладни пакети, їх призначення.
21. АРМ – автоматизоване робоче місце - призначення, склад, режими роботи.
22. Що таке проектна операція, наведить приклади проектних операцій.

23. Що таке проектна процедура, наведіть схему алгоритму типової проектної процедури.
24. Розкрийте сутність проектних процедур аналізу і синтезу.
25. Що таке структурний та параметричний аналіз та синтез?
26. Дайте характеристику методів рішення творчих задач – систематичних і евристичних.
27. Назвіть види забезпечення САПР.
28. Дайте коротку характеристику математичного забезпечення САПР.
29. Дайте коротку характеристику лінгвістичного забезпечення САПР.
30. Дайте коротку характеристику програмного забезпечення САПР.
31. Дайте коротку характеристику інформаційного забезпечення САПР.
32. Дайте коротку характеристику технічного забезпечення САПР.
33. Дайте коротку характеристику методичного забезпечення САПР.
34. Дайте коротку характеристику організаційного забезпечення САПР.
35. Дайте визначення бази (БД) даних та системи управління базами даних (СУБД)
36. Розкрийте сутність принципів побудови баз данних – ієрархічного, мережевого, реляційного.
37. Надайте короткі відомості щодо структури бази даних (поля і записи).
38. Опишіть можливі для використання типи даних в базах даних.
39. Опишить об’єкти баз даних (таблиці, запити, форми, звіти).
40. Сутність процесу оптимізації проектного рішення. Задача оптимізації та критерії оптимальності.
41. Оптимум глобальний та частковий, методи їх дослідження.
42. Дайте визначення цільової функції, зокрема унімодальної цільової функції.
43. Сутність багатокритеріальної оптимізації.
44. Геометричне представлення цільової функції на прикладі одно- та двохпараметричних функцій.
45. Аналітичні методи оптимізації, їх переваги та недоліки.
46. Загальна характеристика чисельних методів оптимізації.
47. Розкрийте сутність пошукових методів оптимізації.
48. Економічний критерій оптимальності проектного рішення. Метод приведених витрат.
49. Сутність метода Кваліметрії при визначенні загального рівня проектних робіт. Показник якості.

ЧАСТИНА ІІ

ТЕХНІЧНІ ПРИЙОМИ РОБОТИ В СЕРЕДОВИЩІ SIMULINK СИСТЕМИ MATLAB

ЛЕКЦІЯ 1. ОСНОВИ ВІЗУАЛЬНОГО ПРОГРАМУВАННЯ.

Десять типових кроків при створенні S – моделі та роботі из нею

1.1 Коротка характеристика системи MATLAB та пакету Simulink.

Назва системи скорочена від Matrix Laboratory – Матрична лабораторія. Саме це є відмінністю системи – орієнтація на роботу з векторами і масивами даних у матричній і комплексній формах. Система сумісна із програмами, що написані на мовах FORTRAN, C та C^{++}.

Система, окрім використання загальних математичних функцій, надає багато можливостей з інженерних розрахунків, проектних та дослідницьких робіт: розв'язання систем алгебраїчних та диференційних рівнянь, роботу із аналоговими та цифровими сигналами, проектування аналогових та цифрових фільтрів, дослідження частотних, імпульсних та перехідних характеристик, виконання спектрального аналізу та синтезу, зокрема прямого та зворотного перетворення Фур'є, розв'язання задач оптимізації і багато інших. Система MATLAB є відкритою і постійно удосконалюється.

Система має особисту М - мову. Позитивними її рисами є порівняно невелика кількість операторів і необов'язковість обирання типів та розмірів змінних. На відміну від деяких мов, в MATLAB відсутні глобальні змінні, дія котрих розповсюджувалась би на всі програмні одиниці. Але при цьому MATLAB має цінну можливість, яка відсутня в інших програмах – дозволяє в одному і тому ж сеансі роботи виконувати кілька самостійних програм, причому всі змінні цих програм становляться спільними і утворюють єдиний робочий простір (WorkSpace).

Система MATLAB забезпечує отримання якісної двох- та трьохвимірної графіки і надає різноманітні можливості редагування цих графіків.

Одною із найбільш привабливих особливостей системи MATLAB є наявність в неї наочного та ефективного засобу створення програмних моделей – пакету візуального програмування Simulink. Пакет дозволяє проводити дослідження (моделювання у часі) поведінки динамічних лінійних та нелінійних систем. Програма створюється в діалоговому режимі, шляхом зборки на екрані схеми з'єднань елементарних ланок (блоків). В результаті отримуємо так звану S - модель (файл із розширенням *.mdl). Такий процес створення програми має назву візуального програмування. В процесі моделювання є можливість

спостерігати процес за допомогою спеціальних засобів вимірювання, які виглядають наочно, як наприклад, осцилограф або цифровий дисплей.

Складання моделі в пакеті Simulink виконується із використанням окремих, тематично згрупованих у бібліотеці блоків, які можна "перетягувати" мишею у робочій простір програми.

1.2 Вікно MATLAB

Робота починається із відкриття вікна MATLAB після запуску програми, рис 1.1. В цьому вікні є кілька інших вікон. Розмір кожного вікна можна змінювати, тому вигляд вікна MATLAB може відрізнятися від наведеного на рисунку.

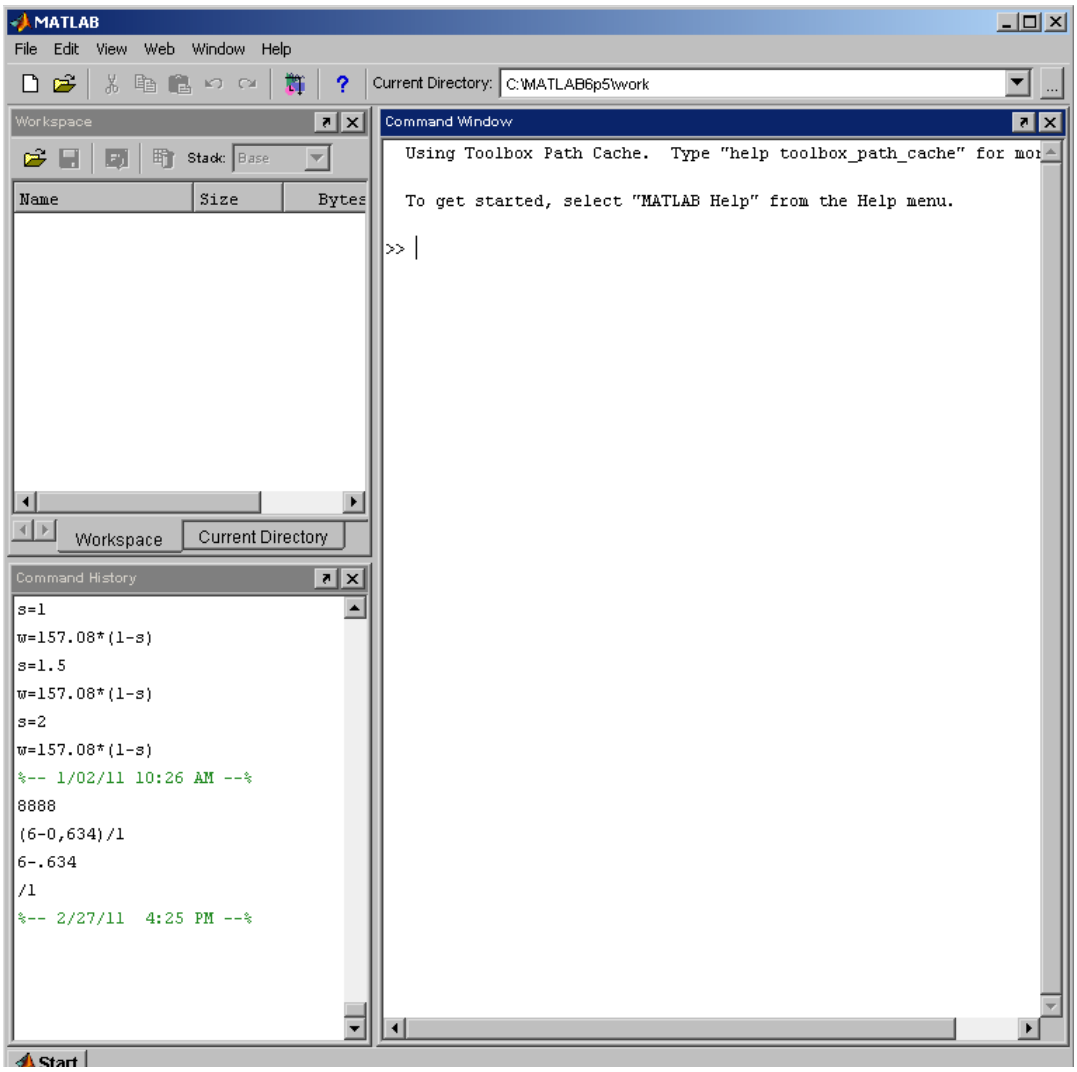


Рис 1.1 – Вікно MATLAB

Головне з них - Командне вікно (Command Window), яке розташовано з права. Якщо поставити курсор після >>, програма готова до роботи. В командному вікні з'являються символи команд, що набираються з клавіатури, результати виконання команд, повідомлення про помилки. Командне вікно можна також використовувати як потужний калькулятор для

біжучих обчислень досить складних формул. В верхній частині вікна MATLAB є дві строки – меню та панель інструментів. Зміст меню File та Edit зрозумілий навіть початковому користувачу. В меню View за допомогою міток (прапорців) встановлюється кількість вікон, з якими потрібно працювати. Це, зазвичай, Робоча область, або Робочий простір (Workspace) та Біжучий каталог (Current Directory) – вони перемикаються натиском на відповідний підпис внизу вікна, а також вікно Переліку або історії команд, що були виконані (Command History). За допомогою меню Edit можна очистити потрібні для нової роботи вікна а через меню File – Close закрити непотрібні вікна. Кожне з вікон можна також закрити натиснувши на хрестик у правому верхньому куті вікна. При натисканні на хрестик у вікні MATLAB закривається вся програма. Безпосередньо з вікна MATLAB можна працювати із M- файлами: відкривати, закривати, створювати нові. M – файл може створюватись як самостійна програма, або як програма, що обслуговує модель Simulink: для вводу числових даних параметрів моделі, побудови графіків тощо.

1.3 Вікно Simulink.

Для переходу до пакету Simulink треба натиснути на панелі інструментів MATLAB на значок  , після чого відкривається вікно Браузера бібліотеки Simulink (Simulink Library Browser), рис. 1.2. З ліва у вікні показаний перелік бібліотек, від Continuous до User-Defined Functions. Кожну з цих бібліотек можна відкрити, клацнувши по відповідному значку. Зміст бібліотеки відкривається у правому полі вікна. Ще нижче розташовані додаткові пакети прикладних програм за різними галузями знань (від аерокосмічної до економіки і біології - зараз їх вже понад півсотні), але у галузі електротехніки та електромеханіки нас буде цікавити трохи пізніше пакет SimPowerSystems. Ця бібліотека пропонує не набір елементарних блоків, а цілком завершені об’єкти в одно- або трьохфазному виконанні: електричні машини різного типу, трансформатори, лінії електропередач, різного типу напівпровідникові елементи та перетворювачі, тощо.

Із цього вікна можна створити новий S – файл або відкрити вже існуючий  . Через меню File можна також закрити файл або зберегти його під іншим ім’ям. Треба зауважити, що зберігати файл можна там, де це зручно, поряд з іншими файлами на дану тематику, але для запобігання непорозумінь рекомендовано в імені файлу використовувати латиницю. При створенні нового файлу відкривається робоче вікно – пусте вікно із вже привласненим ім’ям Untitled ("Безіменний", із розширенням *.mld). Це ім’я треба змінити на інше за сенсом роботи. При необхідності можна відкрити декілька робочих вікон.

1.4 Створення S-моделі і робота із нею.

Продемонструємо створення S-моделі на дуже простому прикладі. Тут важливим є не сутність прикладу, а порядок дій при моделюванні. Спочатку розглянемо у скороченому вигляді обов'язкові елементарні дії для демонстрації можливостей програми, а пізніш розглянемо подробиці та особливості цих дій для виконання більш складних завдань. У підзаголовку до лекції ці дії названі як **десять типових кроків при створенні S – моделі та роботі из нею.**

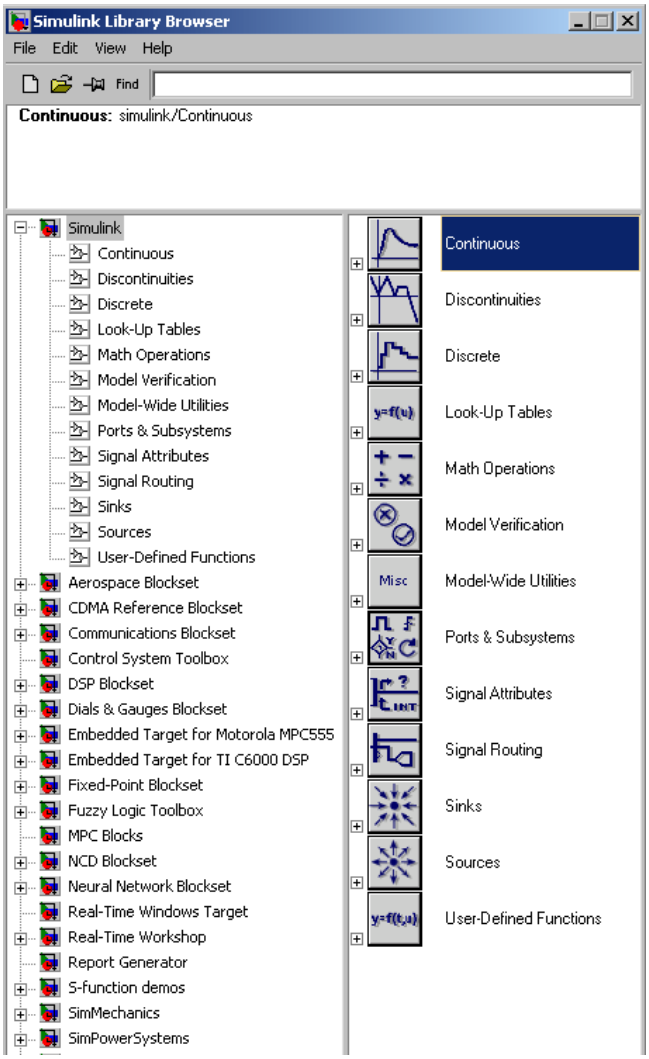


Рис. 1.2 – Вікно бібліотеки Simulink

1.4.1 Створення нового S – файлу. На панелі інструментів Simulink Library Browser натискаємо на значок  - створити новий файл, після чого відкривається порожнє вікно, або для роботи з новою моделлю. В меню File обираємо Save as… і присвоюємо новому файлу ім'я.

1.4.2 Вибір необхідних для задачі блоків та їх розташування. Набір блоків та їх кількість залежать від задачі, що поставлена. Наприклад, я хочу продемонструвати роботу генератора гармонійного сигналу. Із бібліотеки Simulink лівою кнопкою миші перетягуємо в робочий простір необхідні блоки. Наприклад, із розділу Джерела (Sources) обираємо генератор гармонійних сигналів 

, а із розділу Приймачі (Sinks) обираємо осцилограф 

та графопобудовувач 

, і розташовуємо їх у зручному вигляді, рис. 1.3. Генератор гармонійних сигналів або перетягуємо двічі, або копіюємо його вже в робочому вікні. Кожен із типових блоків має особисту назву. Якщо однотипних блоків більше одного, програма автоматично присвоює їм номери, див. Sine Wave та Sine Wave 1.

1.4.3 Створення схеми з'єднання блоків. Лівою кнопкою миші з'єднуємо вихід генераторів сигналів із входами осцилографів. Відгалуження лінії униз утворюємо правою кнопкою. На цьому, власне, закінчується побудова такої простої моделі.

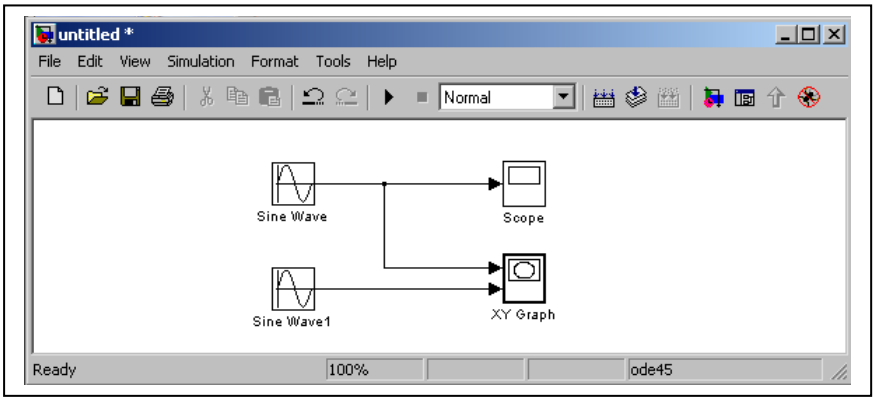
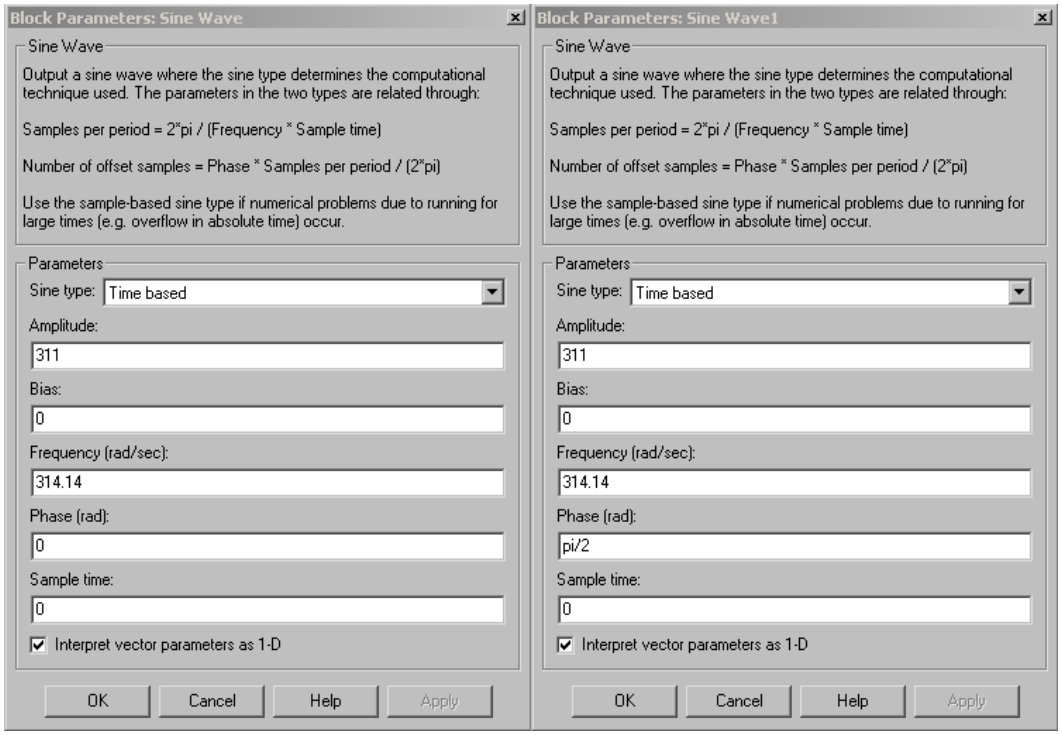


Рис. 1.3 – Встановлення обраних блоків в робочому просторі та їх з'єднання

1.4.4 Завдання параметрів блоків. Кожен із блоків можна виділити клацанням лівої кнопки, при цьому на кутах блоку з'являються чорні квадратики. Після другого клацання на блоці відкривається вікно налагодження параметрів, рис. 1.4. Відповідно до поставленої задачі перший блок Sine Wave налаштовуємо на генерацію синусоїдального сигналу, наприклад із амплітудою 311 В, і кутовою частотою 314,14 рад/с (це амплітуда напруги мережі 220 В, частотою 50 Гц.). Другий – на генерацію косинусоїди із тими же параметрами (косинусоїда відрізняється від синусоїди наявністю початкової фази $\pi/2$).

Зокрема в генераторі синусоїдального сигналу встановлюємо Amplitude: 311; Bias: 0; Frequency: 314,14; Phase: 0, а в генераторі косинусоїдального сигналу встановлюємо Amplitude: 311; Bias: 0; Frequency: 314,14; Phase: pi/2.

Настроювання осцилографу в даному випадку не потрібне, оскільки маштабування відбувається автоматично. Блок XY Graph треба відкрити і встановити мінімальні та максимальні значення по осям X та Y (встановлюємо на кожній осі ±350 В).



1.4 Вікна настроювання параметрів блоків Sine Wave та Sine Wave 1

1.4.5 Встановлення параметрів моделювання. Параметри моделювання встановлюємо за допомогою меню Simulation→Simulation parameters (або Configuration Parametrs) і встановлюємо необхідні параметри, рис. 1.5. У розділі Solver (Решатель) встановлюємо початок (Start time) та кінець (Stop time) часу моделювання, обрання програми для чисельного рішення задачі, постійний або змінний крок розрахунків, абсолютна (Absolute tolerance) або відносна (Relative Tolerance) припустима помилка розрахунків. За умовчанням встановлена відносна похибка 0,001, тобто похибка не перевищує 0,1% від біжучого значення змінної. Параметри обираються в залежності від поставленої задачі.

Розрахунок із змінним кроком використовується для безперервних систем. Значення кроку залежить від швидкості зміни результатів. Якщо змінна змінюється повільно, крок збільшується, і навпаки. За умовчанням встановлюється метод Дормунда – Принса (ode45). При моделюванні безперервних електромеханічних систем найкращі результати досягаються при використанні методу ode23tb.

Розрахунок із фіксованим кроком обирається при моделюванні дискретних систем. Найбільш поширеним серед них є метод Рунге – Кутта (ode4).

Ідея прикладу, що розглядається, полягає в демонструванні трьох періодів синусоїди (час моделювання 0,06 с) та кругового годографу вектору напруги (як відомо, він утворюється, якщо на одну ось подавати синусоїдальний сигнал, а на другу - косинусоїдальний).

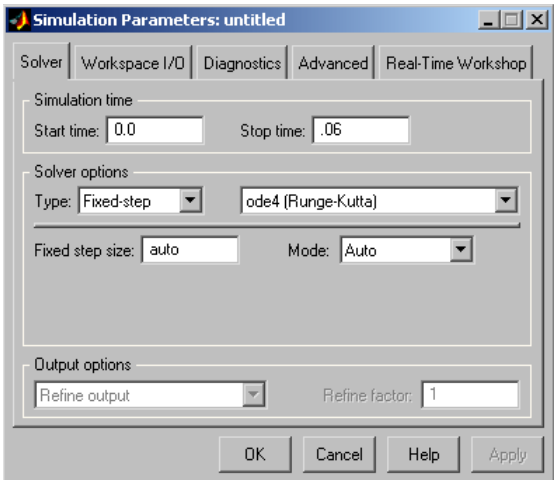


Рис. 1.5 Вікно встановлення параметрів моделювання

1.4.6 Процес моделювання. Повертаємось до моделі рис. 1.3 і натискаємо на кнопку меню  , Start Simulation. Внизу вікна моделі є строчка стану моделі, на який відображається хід моделювання та його умови. Оскільки приклад дуже простий, то моделювання закінчується практично миттєво.

1.4.7 Спостерігання результатів моделювання на приладах, що регіструють. Вікно приладу XY Graph відкривається автоматично. Вікно осцилографу Scope відкривається після двохкратного клацання на ньому. Зображення на екрані осцилографу може бути спотвореним (поданим у якомусь довільному масштабі), тому в меню приладу треба натиснути на  , - команда встановити автоматичне маштабування (Autoscale). Загальний вигляд вікон обох приладів показаний на рис. 1.6.

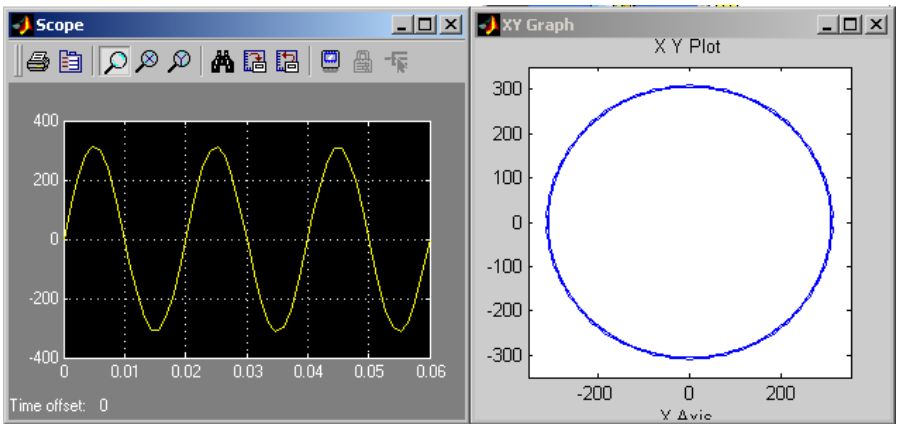


Рис. 1.6 Вікна приладів, що регіструють: Scope та XY Graph

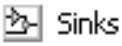
1.4.8 Редагування отриманих графіків процесів. Графіки процесів, що отримані, можуть бути відредаговані, а саме: зроблені підписи на осях, змінена товщина, тип та колір ліній, змінений масштаб по осям, шрифти, тощо.

1.4.9 Спостерігання результатів моделювання у вікні Workspace. Повертаємось до вікна MATLAB і звертаємо увагу на вікно Workspace. В ньому є тільки один запис "tout", тобто вихідний час моделі. Клацнувши два рази на "tout" відкриваємо вікно із масивом часу. Оскільки ми не задавали конкретних умов моделювання, програма це зробила сама, автоматично. Тобто заданий відрізок часу 0,06 с був промодельований за 51 крок із змінним шагом інтегрування. Якщо ми в моделі застосували б блок To Workspace, подавши на нього будь яку змінну, то ми побачили б також масив із 51 значення цієї змінної. Дані із робочого простору можна переправити в М- файл, можна також по цим даним побудувати інші графіки.

1.4.10 Зберігання результатів моделювання. Результати моделювання (схему моделі, отримані осцилограми, масиви даних в робочому просторі) можна зберігати як окремі файли. Схеми моделей, програми М –файлів, а також відредаговані графіки можна вставити в документ Word і далі користуватись ними вже як документами.

ЛЕКЦІЯ 2. ЗАГАЛЬНА ХАРАКТЕРИСТИКА ЕЛЕМЕНТАРНИХ БЛОКІВ БІБЛІОТЕКИ SIMULSNK

Окремі елементи або блоки бібліотеки позначені відповідними значками, вигляд яких та відповідний підпис розкривають сутність елементу. Дамо коротку характеристику окремим розділам та блокам бібліотеки. Розглядаємо тільки ті, що найбільш частіше зустрічаються у практиці проектування та дослідження електромеханічних пристроїв та систем. Зауважимо, що подальша робота проектувальника передбачає більш детальне описання блоків, ніж це зроблено тут. Тут обмежимось тільки словесним описанням призначення та особливостей блоків, яке буде слугувати підказкою при налаштуванні параметрів блоків і більш детальному самостійному знайомству із ними. Деякі із блоків будуть описані більш детально у наведених прикладах.

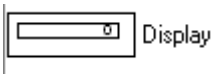
2.1 Розділ Sinks (Приймачі). 

Блоки розділу призначені для візуалізації результатів моделювання, налагодження моделі та контролю проходження сигналів. Блоки розподілені на три групи.

Група Data Viewers – блоки обзору (візуалізації).

Блок із одним входом, дає картину поданого на нього сигналу, розгорнуту у модельному часі. Його можна розглядати як аналог осцилографа. Один вхід не означає, що можна спостерігати одну змінну. На вхід можна подати вектор із кількох змінних. Кілька змінних можна спостерігати на одному екрані, а можна екран розбити на кілька поверхів, для відображення кожного сигналу окремо. Масштабування змінних виконується автоматично. Масштаби можна зменшувати або збільшувати, уточнюючи окремі деталі осцилограми.

Блок із двома входами, призначений для спостерігання залежності одної змінної від другої. Корисний для спостерігання динамічних механічних та електромеханічних характеристик, годографів векторів, тощо.

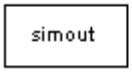


Блок із одним входом, призначений для відображення числових значень вхідної величини. В перехідному процесі показує змінну на кожному кроці розрахунку. Дані блоку можна зафіксувати візуально після закінчення моделювання, або після спеціальної

зупинки моделювання. На блок, також, як і на Scope, можна подавати вектор, тоді в блоці утворюється кілька шкал, відповідно до розміру вектору.

Група Model & Subsystem Outputs - вихідні блоки для пересилання та збереження результатів.

Вихідний порт для виводу результатів на зовні моделі (в інші моделі або субмоделі - підсистеми).



Блок, який зберігає результати в робочому просторі. На нього також можна подавати скалярні сигнали або вектор сигналів. Замість "simout" на блоці треба записати ім’я змінної, під котрим вона буде зберігатись.

Порт для виводу результатів "в нікуди". Якщо в моделі є якийсь вихід, який тимчасово ніде не використовується, система сповіщає про помилку. Використання цього блоку "заспокоює" систему.

Група керування моделюванням.

Перериває моделювання при виконанні тих або інших умов. Блок спрацьовує при поданні на нього ненульового сигналу.

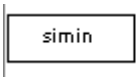
2.2 Розділ Sources (Джерела). 

Блоки використовуються як джерела різноманітних сигналів, необхідних для роботи моделі. Джерела поділені на дві групи.

Група Model & Subsystem Inputs - Входи моделей та субмоделей (підсистем).

Вхідний порт моделі або підсистеми, забезпечує прийом сигналу від іншої моделі більш високого порядку.

Забезпечує нульовий сигнал в точці встановлення. Аналог короткого замикання в заданій точці моделі.



Забезпечує ввід до моделі даних безпосередньо із робочого простру.

Група Signal Generators - це генератори сигналів різного виду, розгорнуті у модельному часі.

Блок не має входу, на виході формує постійну величину (скаляр, вектор або матрицю).

 Signal Generator

Формує безперервний сигнал заданого виду (синусоїдальний, трикутний, прямокутний, випадковий).

 Pulse Generator

Генератор безперервних прямокутних імпульсів.

 Ramp

Створює лінійно наростаючий або спадаючий сигнал.

 Step

Генерує сигнал у вигляді ступеню із заданою висотою (від початкового до кінцевого значення) у заданий час.

 Sine Wave

Генерує гармонійні сигнали. Налаштовуються амплітуда, частота, фаза та постійна складова сигналу (Bias). Можливе завдання на виході блоку кількох гармонійних сигналів. В такому випадку параметри задаються в квадратних дужках через кому, наприклад амплітуда [311,200].

 Repeating Sequence

Генерує періодичну послідовність пилкоподібних сигналів із заданою амплітудою та періодом.

 Chirp Signal

Генерує гармонійні сигнали, частота яких лінійно змінюється у часі.

 Clock

Джерело безперервного сигналу, пропорційного модельному часу.

2.3 Розділ Continuous (Безперервні елементи).  Блоки, що моделюють лінійні стаціонарні динамічні ланки. Розподіляються на дві групи.

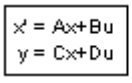
Група Continuous – Time Linear Systems – Лінійні безперервні елементи.

 Integrator

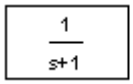
Ідеальна інтегруюча ланка.

 Derivative

Ідеальна диференціююча ланка.

 State-Space

Блок, що дозволяє описати лінійну ланку чотирма матрицями його простору стану.

 Transfer Fcn

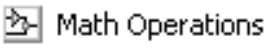
Блок, що задає лінійну ланку вводом її передавальної функції.

Група Continuous – Time Delays – Лінійні блоки затримки сигналу.

 Transport Delay Забезпечує затримку сигналу на задану кількість кроків модельного часу, причому не обов'язково на ціле число.

 Variable Transport Delay Забезпечує керовану із зовні величину затримки сигналу.

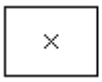
2.4 Розділ Math Operations (Математичні операції).



Реалізують вбудовані математичні функції. Об'єднують чотири групи блоків.

Група Math Operations – блоки, що забезпечують математичні перетворення вхідних величин.

 Sum Сумує сигнали, що поступають до його входів. Призначається потрібна кількість входів і їх знаків. Форма суматора може бути круглою або прямокутною.

 Product Виконує множення та ділення сигналів. Кількість входів та їх функція призначається.

 Gain Лінійна підсилююча ланка (підсилювач).

 Math Function Математичні функції – квадрати, корені, логарифми тощо, залежно від вибору. Позначка на блоці – умовний символ математичної операції.

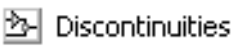
 Trigonometric Function Тригонометричні функції, обираються із запропонованого списку.

 MinMax Визначає мінімальну або максимальну величину змінної. Кількість змінних, або входів блоку задається.

 Sign Розв'язує сігнум – функцію, тобто визначає знак вхідної змінної. Якщо на вході позитивна величина, то на виході +1, якщо негативна, то -1, якщо нульова, то нуль. Використовується, наприклад для відображення в моделі реактивного характеру моменту.

 Abs На виході блоку формується абсолютна величина від вхідної величини.

До групи входять також блоки, що виконують ***векторні операції, логічні операції, та перетворення комплексних величин.***

2.5 Розділ Discontinuities (Нелінійні, або розривні елементи). 

Блоки цього розділу реалізують типові нелінійні (кусково – лінійні) залежності вихідної величини від вхідної.

Saturation

Блок реалізує лінійну залежність із насиченням (із обмеженням). До межі насичення вихідна величина співпадає із вхідною, тобто коєфіціент підсилювання в цього блоку дорівнює 1. Якщо вхідна величина більше заданої, то на виході блока вона обмежена цим завданням.

Dead Zone

Блок реалізує мертву зону, або зону нечутливості. Блок має дві настройки: початок та кінець зони нечутливості. Якщо вхідна величина в середині зони нечутливості, то на виході нуль, якщо вхідна величина знаходиться зовні зони нечутливості, то вихідна величина рівна різниці вхідної величини і зони нечутливості.

Rate Limiter

Обмежувач швидкості. У вікні настроювання встановлюються два параметра: Rising slew rate (R - обмеження збільшення швидкості) та Falling slew rate (F - обмеження зменшення швидкості). Швидкість зміни вхідного сигналу розраховується за виразом

$$\text{rate} = \frac{u_i - y_{i-1}}{t_i - t_{i-1}},$$

де u_i – сигнал на вході в момент часу t_i , y_{i-1} – сигнал на виході в момент часу t_{i-1}.

Якщо швидкість зміни сигналу у часі знаходиться у межах заданого (між R та F) то швидкість на виході співпадає із швидкістю на вході $y_i = u_i$. Якщо швидкість зміни сигналу на вході перевищує задану, то на виході формується сигнал із дозволеною максимальною швидкістю $y_i = y_{i-1} + R\Delta t$. Якщо швидкість сигналу на вході менша за встановлену, то на виході формується сигнал із дозволеною мінімальною швидкістю $y_i = y_{i-1} + F\Delta t$.

Backlash

Нелінійність типу зазору. У вікні налагодження встановлюються два параметри: Deadband widt (величина люфту) та Initial output (початкове значення вихідної величини).

Relay

Блок – аналог звичайного електромагнітного реле. Має чотири параметра, що налаштовуються: Switch on point (точка включення - рівень сигналу, при котрому реле включається); Switch off point (точка відключення – рівень сигналу, при котрому реле відключається)) Output when on (вихід при включеному стані – рівень сигналу на виході реле

при включенні); Output when off (вихід при виключеному стані – рівень вихідного сигналу при відключенні реле).

 Quantizer Квантувач сигналу за рівнем. Має єдиний параметр налаштування – Quantization interval (інтервал квантування). Перетворює безперервний сигнал в дискретний із заданим рівнем дискретності.

 Hit Crossing Визначає момент перетину вхідним сигналом деякого встановленого значення. В мить перетину на виході блоку формується одиничний сигнал. Має наступні параметри, що можуть налаштовуватися:

Hit crossing offset – установлення значення вхідного сигналу, яке потрібно зафіксувати;

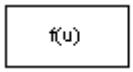
Hit crossing direction - установлення напряму перетину: rising (наростаючий сигнал), falling (сигнал, що зменшується), either (у будь якому напрямі).

Show output port – вказати порт виходу (встановлюється прапорцем);

Enable zero crossing – дозволити перетин нуля (встановлюється прапорцем).

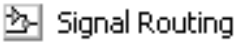
2.6 Розділ User Defined Functions (Функції користувача).

 Призначення та зміст цих блоків визначається користувачем. Створений користувачем блок може бути записаний у бібліотеку Simulink, як один із S – блоків.

 В цей блок можна ввести будь яку скалярну функцію від одного вхідного аргументу (наприклад формулу механічної або електромеханічної характеристики двигуна). Функція вводиться у вікні налагодження, причому вхідна змінна записується як "u", незалежно від того, як вона була названа в моделі.

Інші блоки цього розділу дозволяють формувати векторний вихід (MATLAB Fnc), записувати не тільки функцію, а досить складну програму обробки вхідного сигналу (S-function), а також будувати S – функцію за допомогою майстра в діалоговому режимі.

2.7 Розділ Signals Routing (Пересилання сигналів).

 Блоки цього розділу забезпечують збір, об’єднання в одну шину, розподіл та переключення сигналів.

 Мультиплексор – об’єднує кілька сигналів в одну шину (один вектор). Вхідні величини можуть бути як скалярними, так і векторними. Порядок елементів в векторі визначається порядком входів (зверху вниз). Блок має два параметра, що налагоджуються:

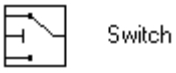
Number of inputs (кількість входів) та Display option (форма зображення блоку). Мультиплексор зручно використовувати для подачі кількох сигналів на один вхід осцилографу (Scope).

Демультиплексор – виконує зворотну функцію від Mux, тобто розділяє вхідний вектор на задане число вихідних сигналів. Налагодження такі ж, як і в блоці Mux: Number of outputs (кількість виходів) та Display option (форма зображення блоку). Бажано, щоб кількість встановлених виходів співпадала із розміром вхідного вектору, інакше вихідні вектори розподіляються по наявним виходам у досить складний спосіб.

Блоки "Побудова шини" та "Розділення шини" виконують ті ж функції, що і Mux та Demux, але надають можливості перерозподілу сигналів в середині шини.

Селектор – обирає на вхідному векторі та передає на вихід тільки ті елементи, номера яких вказані в полі вводу Elements вікна налагодження. Обрані зв’язки показуються на зображенні блоку.

Ручний перемикач. Не має параметрів, що налагоджуються. Два входи перемикаються при подвійному клацанні миші на зображенні блоку. Такий блок зручно використовувати, наприклад для вибору моменту навантаження активного, або реактивного характеру.



На відміну від попереднього перемикача має керуючий (середній або другий вхід). В полі налагодження встановлюється поріг перемикання (Threshold). Якщо на інформаційному вході поданий сигнал рівний або більше порога, то перемкнутий перший вхід, якщо менше, то третій.

Багато портовий перемикач. Має не менше трьох входів, один з яких (верхній) є керуючим, а інші інформаційні. В вікні налагодження встановлюється кількість інформаційних входів Number of inputs. Входи нумеруються зверху вниз. Для переключення заданого входу на керуючий вхід треба подати ціле число, яке співпадає із номером входу. Якщо число менше одиниці, перемикається перший вхід; якщо число більше кількості входів, перемикається останній вхід; якщо на інформаційний вхід подати дробне число, то

воно буде округлено за правилами і перемкнеться відповідний вхід, згідно до округленого числа.

Data Store Memory

Data Store Read

Data Store Write

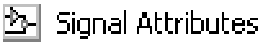
Блоки використовуються сумісно і забезпечують зчитування, запис та збереження даних на протязі моделювання.

From

Goto

Goto Tag Visibility

Блоки "Прийняти із", "Відправити", "Признак видимості" використовуються сумісно для обміну даними між окремими частинами моделі з урахування їх досяжності.

2.8 Розділ Signals Attributes (Атрибути сигналів). 

Вміщує блоки, які можуть визначити атрибути сигналів (тип даних, кількість елементів в векторі або в матриці, початкові умови), а також перетворювати цілі 1-, 2- та 4- байтові числа.

2.9 Розділ Ports & Subsystems (Порти та Підсистеми).

Пустий блок підсистеми – заготовка для її створення. При відкритті підсистеми бачимо один вхідний і один вихідний порти, з’єднані між собою. Як мінімум, між двома цими портами треба щось встановити. Далі в відкритому блоці створюється модель. Необхідна кількість входів та виходів набувається копіюванням або перетягуванням із бібліотеки.

Вхідний та вихідний порт для зв’язку підсистеми із іншими частинами головної моделі.

Вже згадані у інших розділах блоки 

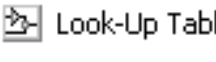
та 

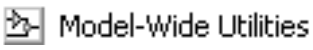
використовуються як "заглушки" для тих портів, які з якихось причин не були задіяними. Причому Термінатор встановлюється на вихідні порти, а Заземлення – на вхідні.

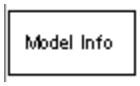
Блоки 

-Дозволити, та 

-Засувка призначені для логічного керування роботою підсистем.

2.10 Розділ Look-Up Tables (Табличні функції).  Блоки, що формують вихідний сигнал, залежність якого від вхідного сигналу задається за допомогою таблиці відповідності із подальшою лінійною інтерполяцією. Використовуються, наприклад для відображення кривої намагнічення.

2.11 Розділ Model-Wide Utilities (Утиліти розширення моделі).  Блоки, що сприяють оформленню документації на модель.

 Дозволяє зробити і розташувати у бажаному місці заголовок та пояснюючи надписи щодо моделі, обрати шрифт, тощо. Текст вводиться у вікні, яке відкривається після подвійного клацання на блоці. Після повернення до моделі, в блоці можна бачити ту частину тексту, яка вміщується в рамці. Розмір рамки можна регулювати. Після подвійного клацання на рамці відкривається вікно із повним текстом.

 Цей блок можна використовувати також, як і блок Model Info. При подвійному клацанні на ньому, відкривається вікно, в якому можна записати будь – які подробиці і дані щодо створеної моделі. На відміну від блоку Model Info після **збереження запису** на блоці в моделі зостається назва DOC, а подробиці можна читати після відкривання блоку. При збереженні введення спеціального імені не потребується.

2.12 Розділ Model Verification (Верифікація моделі). Блоки цього розділу призначені для верифікації (підтвердження) того, що вхідний сигнал на протязі всього часу моделювання не виходить за межі, що встановлені, або відповідає іншим певним умовам. При невиконанні умов верифікації можливі два варіанти дій – зупинка моделювання, або генерація на виході блоку сигналу, який може бути використаний як керуючий.

ЛЕКЦІЯ 3. ТЕХНІКА СТВОРЕННЯ БЛОК-СХЕМ S – МОДЕЛІ

Окремі операції, що тут описуються, настільки прості, що не потребують додаткових ілюстрацій. На відміну від багато ілюстрованих підручників та посібників, які можна вивчати без комп'ютера, рекомендовано чинний матеріал (як і попередній) засвоювати у ігровому вигляді, повторюючи дії, що описуються, у самостійно створеній тимчасовій моделі. Нагадаємо, що після запуску програми відкривається вікно MATLAB, в цьому вікні треба відкрити вікно Simulink та створити новий файл, п.п. 1.4.1.

3.1 Операції із блоками.

До операцій із блоками відносимо: виділення, копіювання, видалення, переміщення, зміну розміру та орієнтації, вилучення блоку із схеми, присвоєння та зміну імені, встановлення параметрів блоку.

Виділення блоку або кілька блоків. Блок виділяється клацанням на ньому лівою кнопкою миші. Для виділення кількох блоків треба їх обвести рамкою при натиснутій лівій кнопці. У всіх виділених блоків, а також ліній з'єднання, що опинились у рамці, з'являються маленькі чорні квадратики – свідоцтво виділення. Щоб зняти виділення достатньо клацнути в будь якому пустому місці схеми. Кілька блоків можна також виділити послідовним клацанням на них миші при натиснутій клавіші Shift.

Копіювання блоків. Копіювання методом перетягування застосовується при обранні блоків із бібліотеки Simulink. Копіювати блок (дублювати) можна, взявши прототип із вже створеної схеми. Це робиться перетягуванням при натиснутій клавіші Ctrl, або правою кнопкою миші. Новому блоку автоматично присвоюється новий порядковий номер. Налаштування блоку зберігається таким, як у прототипу. Так можна копіювати не тільки в межах одного вікна, але і перетягувати в інші вікна. Можна відкрити інший файл, виділити в ньому частину схеми разом із з'єднаннями і перетягнути її в нове вікно.

Копіювати і вставляти блоки також можна, використовуючи меню Edit→Copy та Edit→Paste. Копія знімається після виділення, а вставлення - після клацання мишею у місці, де потрібно розташувати об'єкт. Скорочений варіант – використання команд Copy та Paste після натискання правої кнопки миші.

Видалення (удаление – рос.) блоків. Видалення непотрібних блоків або групи блоків із схеми відбувається після їх виділення та натискання клавіші Delete, або командою Clear після натискання правої кнопки миші.

Переміщення блоків. Для переміщення блоку, або частини схеми, їх виділяють, а потім перетягують мишею у потрібне місце. При цьому всі з'єднання між блоками зберігаються і "тягнуться" за ними. На це треба звертати увагу і, якщо потрібно, відредагувати з'єднання. Більш точне переміщення блоків можна робити за допомогою клавіш управління курсором, звичайно, виділивши перед тим об'єкт.

Зміна розміру блоків. Для зміни розміру треба виділити блок і встановити курсор на один із квадратиків. Якщо тягнути тільки по горизонталі або тільки по вертикалі, то розмір відповідно змінюється в одному напрямі. Якщо тягнути під кутом 45˚, то розмір по вертикалі та горизонталі змінюється пропорційно. Питання актуально при створенні складних схем із великою кількістю блоків, або якщо блок має багато входів або виходів.

Зміна орієнтації блоків. Потреба виникає для зручного з'єднання із іншими блоками, для оптимізації прокладення ліній з'єднання. Спочатку виділяємо потрібний блок, потім відкриваємо меню Format і використовуємо команди Flip Block (обертання на 180˚) або Rotate Block (обертання за стрілкою годинника на 90˚).

Вилучення блоку із схеми. Якщо блок треба видалити, то його виділяють і видаляють командою Delete. При цьому лінії з'єднання зберігаються. Можна блок просто витягнути із схеми, і десь тимчасово встановити, не порушуючи ліній зв'язків. Це робиться мишею при натиснутій клавіші Shift.

Зміна імені блоку та маніпуляції із ним. Як було сказано, ім'я та номер блоку присвоюється автоматично. За умовчанням розташування імені блоку таке:

- під блоком, якщо блок орієнтований зліва на право;
- над блоком, якщо блок орієнтований з права на ліво;
- з права, якщо блок орієнтований вертикально.

При видалені якогось із однойменних блоків нумерація інших блоків не змінюється.

Для зміни шрифту, треба виділити блок, а потім змінити його, визвавши меню Format→Font.

Змінити місце імені можна, перетягнувши його на протилежний бік мишею, або скориставшись командою Format→Flip Name. Скрити ім'я, або відновити його можна командами Format→Hide Name або Format→Show Name.

Для зміни імені треба його виділити (відображається рамка тексту), і відредагувати його відомим чином. Нове ім'я повинне складатись хоча б з одного символу.

Встановлення параметрів блоку. Параметри блоку встановлюються в діалоговому вікні налаштування блоку. Для відкриття діалогового вікна треба двічі клацнути на блоці. Рекомендуємо спочатку уважно ознайомитись із інструкцією, яка міститься в верхній частині вікна параметрів. Там же надаються приклади їх встановлення. В вікнах багатьох параметрів є списки, що розкриваються для уточнення деяких параметрів. Звертаємо увагу на коректне вказання класу або типу змінних. Array - обробка змінних класу об'єктів – масивів (зустрічається найчастіше), Char – обробка строк символів, Double – обробка комплексних чисел, Struct – доступ до змісту полів. При невірному вказанні класу об'єкту програма видає сповіщення про похибку. Якщо є сумніви щодо вірності обраних параметрів, треба провести "випробування" блоку у якійсь простій тимчасовій моделі, а вже після цього вставляти блок у головну модель.

3.2 Проведення з'єднувальних ліній.

Сигнали в моделі передаються по лініям. Лінії з'єднують вихідні порти одного блоку із вхідними портами інших блоків. Одна лінія може розгалужуватись і обслуговувати вхідні порти кількох блоків. Лінії можуть передавати скалярний або векторний сигнал. У останньому випадку лінію зв'язку можна зробити більш широкою, див. п. 3.5. Звернемо увагу на те, що до того, як проведені лінії, на зображенні кожного блоку видно позначки вхідних та вихідних портів. Після проведення ліній позначки портів зникають. Якщо якась лінія з'єднання не проведена, то це можна завжди встановити по незадіяним портам. Згідно до проведених ліній зв'язку між окремими блоками, програма автоматично створює систему алгебраїчних, диференційних, або логічних рівнянь для подальшого їх моделювання.

Створення ліній зв'язку між блоками. Простіший випадок – проведення горизонтальних або вертикальних прямих ліній. Ставимо курсор на вихідний порт (при цьому курсор перетворюється на хрестик) і лівою кнопкою миші проводимо пунктирну лінію до входу іншого блоку. Після "стиковки" блоків кнопка миші відпускається і утворюється суцільна лінія із стрілкою, яка показує напрям передачі сигналу. Якщо лінія не доведена до кінця, або, навпаки "переведена", вона позначається червоним кольором, як похибка. Лінію можна також проводити у зворотному напрямі – від входу одного блоку до виходу іншого.

Створення ліній зв'язку із окремих сегментів. Прямі лінії вдається проводити не завжди. За мірою ускладнення схеми, доводиться проводити ломані лінії, тобто лінії із окремих сегментів. Дозволяється, не відриваючи пальця від лівої кнопки миші зробити один перший поворот лінії під прямим кутом. Подальші повороти можна робити тільки після відпускання кнопки в потрібному місці і повторного її натискання. Таким чином можна провести досить складну лінію з'єднання, обходячи інші блоки.

Автоматичне створення ліній зв'язку. Лінія, навіть досить складної форми, може бути створена автоматично. Для цього треба виділити блок – джерело сигналу, натиснути на Ctrl і клацнути на блоці, до якого треба провести лінію.

Створення розгалуження лінії. Для створення розгалуження (відпайки) лінії встановлюють курсор у потрібному місці і після натискання правої кнопки миші відводять нову лінію.

Виділення лінії. Лінія стає виділеною, якщо на будь якій її ділянці (сегменті) натиснути кнопкою миші. Незалежно від складності лінії, вона виділяється цілком. При натисканні правої кнопки одночасно із виділенням з'являється вікно з командами для редагування.

Видалення лінії. При похибці або невдалій спробі проведення лінії її можна вилучити шляхом виділення та використання команд Delete або Clear.

Переміщення сегментів лінії. Часто після проведення складної лінії із сегментами потрібне деяке її редагування (лінія проходить занадто близько від блоку, або навпаки, пройшла десь далеко). Лінія виділяється на тому сегменті, де потрібні зміни. Після виділення ліва кнопка миші утримується натиснутою, а курсор перетворюється на чотири направлену стрілку. Не відпускаючи кнопки переміщують сегмент у потрібному напрямі. Такі дії можна повторювати до отримання бажаного результату.

Розподіл прямої лінії на сегменти або створення зламаної лінії. Порядок дій: виділити лінію, натиснути клавішу Shift, встановити курсор у потрібне місце на лінії. Натиснути на ліву кнопку, при цьому курсор отримує форму кружка, потягнути мишею у потрібну сторону. Місце зламу лінії можна змінювати, переміщуючи курсор уздовж лінії.

3.3 Створення міток сигналів та маніпулювання ними.

Створення міток. Для наочного оформлення схеми, лінії зв'язку можна помічати спеціальними пояснюючими мітками (назва змінної, наприклад). Мітка розміщується зверху або знизу від горизонтальної лінії, та справа або зліва від вертикальної лінії. Мітку можна створювати у будь який частині лінії, зокрема зручно це зробити у місті виходу із блоку та входу лінії у блок. Для створення мітки треба подвійно клацнути **на потрібному сегменті лінії** і ввести запис в утвореній рамці як звичайно у текстовому редакторі.

Маніпулювання мітками. Мітку можна перетягувати на другу сторону лінії, або уздовж лінії. Мітки можна копіювати та видаляти таким чином, як і блоки. Якщо рамка мітки остається не заповненою, вона зникає.

Створення коментарів. У будь якому місці схеми можна створити коментар шляхом подвійного клацання. В утворену рамку вписується текст. Рамку можна перетягувати на інше

місце. Текст після виділення можна редагувати. Для зміни шрифту використовуємо команди Format→Font.

3.4 Створення підсистем.

Підсистеми використовуються для створення складних, великого розміру моделей, які складаються із окремих моделей нижчого рівня. Необхідність у створенні підсистеми виникає тоді, коли модель стає великою і її розміщення на екрані утруднено. Або такий приклад – модель або її частина вже опрацьовані, налагоджені і добре працюють, а на подальше бажано зобразити їх у більш компактному вигляді, без подробиць, але зберігши потрібні входи та виходи. До речі, подробиці завжди можна побачити, і провести необхідні зміни, "відкривши" підсистему. Підсистему можна створити двома способами. Якщо модель вже створена, виділяється мишею потрібна її частина і натиском правої кнопки обирається Edit→Create Subsystem (Створити підсистему). З'являється єдиний блок підсистеми із автоматично створеними входами і виходами для зв'язку із зовні. Бажано, щоб обрана для підсистеми частина моделі була логічно та тематично завершеною і мала мінімум входів та виходів. Другим способом є перетягування із бібліотеки у робочий простір пустого блоку Subsystem і подальше наповнення його змістом п.п. 2.9.

3.5 Редагування загального вигляду блок – схеми.

Програма дає можливості змінювати такі параметри щодо "художнього оформлення" схеми:

- Format→Screen color - обрання кольору екрану;
- Format→Foreground color - обрання кольору ліній з'єднання та контурів блоків (після виділення);
- Format→Background color – обрання кольору заповнення блоків (після виділення);
- Format→Wide nonscalar lines – встановлюється прапорець для того, що б векторні лінії зв'язку були більш широкими, порівняно із скалярними.
- View→Zoom in або Zoom out – збільшення або зменшення розміру схеми (введення маштабу);
- View→Normal (100%) – відміна маштабування, повернення до попередніх розмірів.

3.6 Використання M – файлів для обслуговування S – моделей.

В кожному блоці схеми S – моделі потрібно встановити якісь конкретні дані (напруги, опори, моменти навантаження, тощо). В простих моделях зручно встановлювати в блоках цифрові значення параметру. В схемах великого обсягу, або в таких, де одні й ті ж параметри

використовуються в різних блоках та часто змінюються, зручніше вказувати лише ім'я параметру. Таким чином S – модель будується із параметрами, позначеними у загальному вигляді. Конкретні цифрові дані параметрів вказуються в окремому M – файлі. Новий M – файл створюється у вікні MATLAB командами File→New→Model. Відкривається нове вікно, в якому вводяться дані параметрів, формули для обчислення змінних, текстові коментарі тощо. Наприклад, зручно ввести до M – файлу каталожні дані двигуна, ввести формули всіх проміжних розрахунків ($\omega_н$; ω_0; $c_н$; R_0; J_Σ), розрахувати та задати умови перехідного процесу (момент навантаження, опори ступенів резисторів, умови переключень в блок – схемі моделі). Після запуску програми на виконання всі дії програми відображаються у командному вікні, в якому можна проконтролювати правильність її виконання. Для запобігання виводу результату в командному вікні після кожної строки із присвоєнням даних або обчисленнями встановлюється знак ";". Строка коментарів починається з знаку "%". Коментарі автоматично виділяються зеленим кольором. Для зручності подальшого використання бажано, щоб ім'я цього файлу співпадало із ім'ям S – моделі, а сам M - файл зберігався поряд із S – файлом. Нижче наведений фрагмент M – моделі, створений для моделювання одної із систем електроприводу.

```
%Параметри моделі системи електроприводу Г-Д.
%Сумісно із моделлю "EP_GD_Dynamika"
udn=220; %Номінальна напруга двигуна
udg=230; %Номінальна напруга генератора
rjd=0.05; % опір якоря двигуна
rjg=0.04; % Опір якоря генератора
```

Для роботи відкриваються обидва файли, але першим на виконання запускається M – файл. Всі дані цього файлу з'являються у робочому просторі (Workspace) і в подальшому використовуються у блоках S – моделі. У пізніших версіях MATLAB є можливість автоматичного запуску M – файлу при запуску S – моделі.

Цей же, або інший M – файл може використовуватись надалі для обробки результатів, отриманих у S – моделі, для побудови графіків тощо.

M – файли надають можливість у зручній формі змінювати параметри моделі і запобігти при цьому помилкам.

3.7 Роздрукування S – моделі.

Щоб роздрукувати схему моделі на принтері, треба надати команду Fule→Print (Файл→Друк). Щоб перенести схему у документ Word треба спочатку її скопіювати Edit→Copy model to clipboard, а потім вставити в документ.

3.8 Збереження моделей.

Збереження моделей відбувається відомим чином (File→Save) або (File→Save As). При збереженні кожній моделі присвоюється унікальне ім'я. Імена M – файлів та S – файлів однієї моделі можуть співпадати.

ЛЕКЦІЯ 4. ГРАФІЧНЕ ОФОРМЛЕННЯ РЕЗУЛЬТАТІВ МОДЕЛЮВАННЯ

Під графічним оформленням маємо на увазі побудову, редагування, збереження та експорт графіків залежності одних змінних від інших, або графіків зміни їх у часі – осцилограм.

Для обговорення цього питання спочатку створимо приклад S – моделі.

4.1 Приклад для ілюстрації способів отримання та редагування графіків.

Обираємо приклад дослідження перехідного процесу пуску двигуна постійного струму незалежного збудження (ДПСНЗ).

Дані двигуна: $P_н$=2,5 кВт; $U_н$=220 В; $n_н$=1000 об/хв.; $I_н$=17 А; $R_я$=1,0 Ом, індуктивністю кола якорю нехтуємо.

Сумарний приведений момент інерції $J_Σ$=0,05 кгм^2.

Умови пуску: пуск із реактивним характером моменту, M_c=10 Нм, максимальна величина пускового моменту $3M_н$.

Розрахункові параметри:

$\omega_н = \frac{\pi n}{30} = \frac{\pi 1000}{30} = 104{,}7 \ 1/с$ - номінальна швидкість;

$c_н = \frac{U_н - I_н R_я}{\omega_н} = \frac{220 - 17 \cdot 1}{104{,}7} = 1{,}94$ Вб - номінальний коефіцієнт ЕРС;

$\omega_0 = \frac{U_н}{c_н} = \frac{220}{1{,}94} = 113{,}4 \ 1/с$ = швидкість неробочого ходу;

$M_н = I_н c_н = 17 \cdot 1{,}94 = 33$ Нм - номінальний електромагнітний момент;

$M_{кз} = 3M_н = 3 \cdot 33 = 99$ Нм - Момент короткого замикання, або початковий пусковий момент;

$R_{0п} = \frac{U_н}{3I_н} = \frac{220}{3 \cdot 17} = 4{,}31$ Ом - повний опір якорю при пуску;

$T_{мп} = J_{\Sigma} \frac{R_{0п}}{c_н^2} = 0{,}05 \frac{4{,}31}{1{,}94^2} = 0{,}057$ с - електромеханічна постійна часу при пуску.

Моделюємо рівняння рівноваги напруг у колі якорю, рівняння електромагнітного моменту та рівноваги моментів на валу двигуна. Для полегшення "зборки" схеми, алгебраїчні рівняння записуємо відносно невідомих змінних, а диференційні рівняння у формі Коші, тобто відносно похідної. Схема моделі показана на рис. 4.1,а. В вікні параметрів моделювання Simulation→Simulatiun parameters…Solver встановлюємо час моделювання 0,3с (це становить 4-5$T_{мп}$), обираємо метод інтегрування із фіксованим кроком (Ode4 – Runge-Kutta) та крок інтегрування 0,001 с.

Вихідні рівняння	Перетворені рівняння для моделі
$U = c_н\omega + IR_{0п}$	$I = \frac{U - c_н\omega}{R_{0п}}$
$M = c_н I$	$M = c_н I$
$M - M_c = J_{\Sigma} \frac{d\omega}{dt}$	$\frac{d\omega}{dt} = \frac{M - M_c}{J_{\Sigma}}$

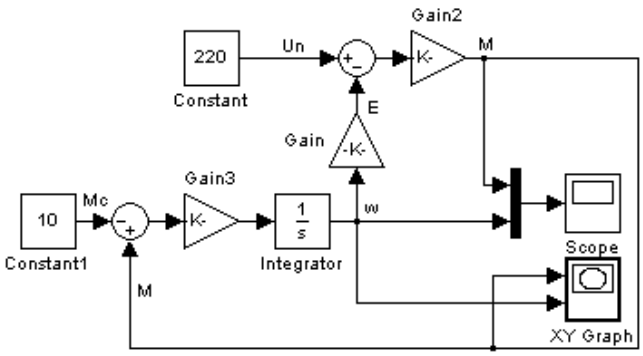


Рис. 4.1 – Схема моделі процесу пуску ДПСНЗ

Для спостереження динамічної механічної характеристики і графіків перехідних процесів в схемі моделі встановлені осцилограф Scope та графопобудовувач XY Graph, зображення на яких, після моделювання, показані на рис. 4.2.

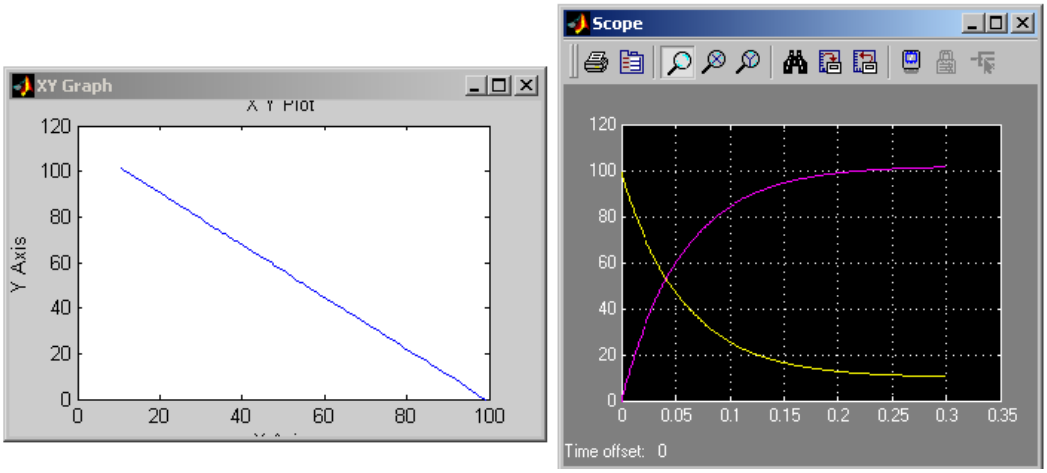


Рис. 4.2 Вікна вимірювальних приладів Scope та XY Graph

Маніпуляції із цими зображеннями досить обмежені і надають зручності саме для спостереження результатів моделювання. Наприклад, можна довільно змінювати розмір вікон, в XY Graph можна змінювати масштаби на осях, в Scope можна також змінювати масштаби осей, можна збільшувати або зменшувати масштаби зображення окремо по кожній з осей, можна екран осцилографу розбити на кілька частин і кожний графік показувати окремо.

Відображення цих вікон у якості документів можливо у найпростішому (краще сказати, примітивному) вигляді – як копії екрану, знятої натиском на клавішу Print Screen. Після копіювання це зображення "вирізається" із кадру за допомогою будь якої програми редагування зображень, наприклад Paint, і копіюється у документ Word, як це зроблено тут. Як ілюстрація для наукової статті, звіту або як елемент технічної документації таке зображення не принадне.

4.2 Порядок підготовки, отримання та редагування графічних матеріалів.

Для отримання графічної інформації високої якості, яку можна використовувати як документ, треба виконати кілька обов’язкових дій:

- В S – моделі передбачити збір необхідної інформації для направлення її в робочий простір MATLAB.
- Направити інформацію у робочий простір (Workspace).
- Отримати графічну інформацію у вигляді графіків безпосередньо із Workspace, або після передачі її у M – файл.
- Редагувати графіки.

4.3 Підготовка інформації і направлення її в Workspace.

Перед усім треба підкреслити важливу особливість робочого простору – до нього потрапляють тільки ті змінні (скалярні або векторні), яким привласнені особисті імена. Наприклад, при старті M – файлу всі параметри, що записані в ньому, потрапляють в Workspace і ці параметри використовуються в S – моделі. Дані, введені у блоки S – моделі у формі чисел до робочого простору не потрапляють. Дані, отримані в результаті обчислень, можуть потрапити у Workspace тільки після присвоєння їм імені та використання певного способу їх передачі. Таких способів є декілька.

4.3.1 Передача даних через блок To Workspace. Інформація може передаватись як у скалярному вигляді (по одній змінній), так і у векторній формі (кілька змінних). Умовне ім'я Simout в вікні настроювання блоку To Workspace, треба замінити на інше, за сенсом задачі, рис. 4.3.

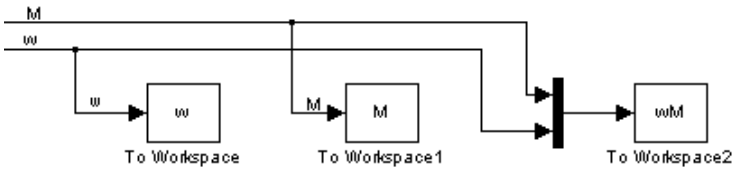
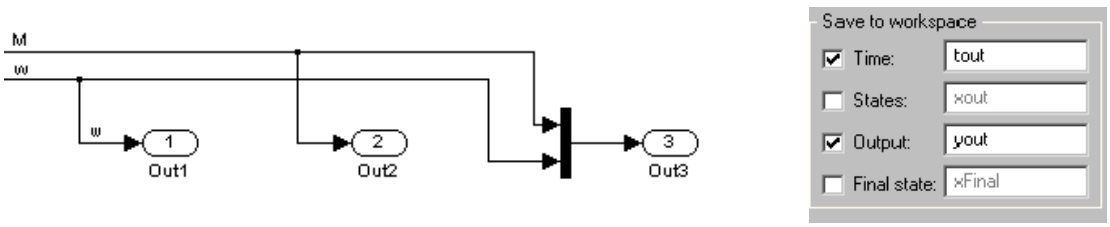


Рис. 4.3 Передавання інформації у Workspace через блок To Workspace

Зазначимо, що позначення ω або M на лінії з'єднання в програмі Simulink – це мітка для уваги користувача, вона співпадає із ім'ям змінної, але сама не є іменем. Робочий простір "бачить" ім'я тільки після його присвоєння в блоці To workspace.

4.3.2 Передача даних через блок (блоки) виходу Out. Для наочності на рис. 4.4 повторений той же самий фрагмент схеми, але з передачею інформації через блоки Out, рис. 4.4,а.



а б.

Рис. 4.4 Передавання інформації у Workspace через блоки Out
а. Фрагмент S – моделі, б. Фрагмент вікна налагодження параметрів моделювання.

Інформація потрапить до робочого простору, якщо заздалегідь встановити прапорці в вікні налагодження параметрів моделювання: Simulation→Simulation parameters…Workspace I/O, Save to workspace, рис. 4.4,б. Якщо встановити прапорець також біля Time, то до

робочого простору потрапляє і масив модельного часу під іменем tout. Особливістю даного способу є те, що незалежно від кількості блоків Out, всі змінні в робочому просторі об'єднуються в один вектор під іменем yout, оскільки блок Out не передбачає передачі імені змінної. В прикладі, наведеному на рис. 4.4,а, в робочому просторі автоматично сформований чотирьохвимірний вектор (за чергою - ω, M, M, ω) під ім'ям yout. Після чого стає ясно, що або перші два виходи, або третій вихід тут зайві, бо вони дублюють один одного.

4.3.3 Використання блоку Scope. Названий блок може слугувати одночасно і для спостереження за змінними і для передачі їх до робочого простору. Для цього у вікні налагодження Scope відкриваємо Parameters→Data history, ставимо прапорець на Save data to workspace, записуємо ім'я змінної в віконце Variable name та обираємо формат змінної Array. В робочому просторі буде записаний вектор змінної плюс біжучий час моделювання.

4.4 Побудова графіків.

Графіки будуються безпосередньо із робочого простору або із попереднім передаванням інформації у M – файл.

4.4.1 Побудова графіків із робочого простору. Для цього треба виділити в робочому просторі потрібний вектор, і правою кнопкою миші відкрити вікно вибору подальших дій. Обрати Graph→plot, після чого з'явиться фігура із зображенням графіків. У всіх прикладах ми посилались на дві змінні: швидкість ω та момент M. При формуванні вектору у робочому просторі через блок Scope, до вектору додається ще одна складова – модельний час.

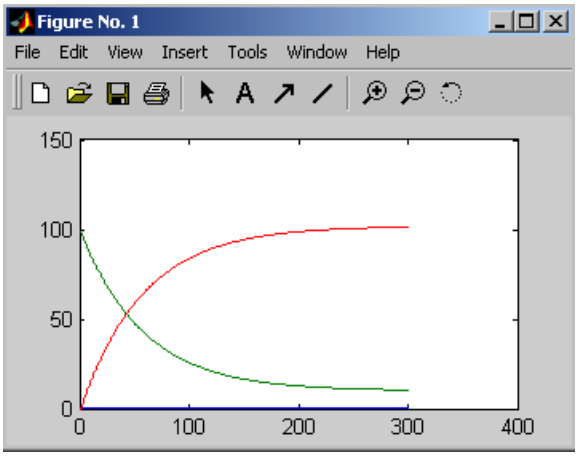


Рис. 4.5 Графіки перехідного процесу, отримані із робочого простору.

Якщо придивитись уважніше, внизу рис. 4.5 є ще один графік синього кольору – це графік часу моделювання, (див. п. 4.1, де заявлено час моделювання 0,3с). Тоді настає питання: у функції якої змінної будувався вектор змінних і, відповідно, графік? На осі абсцис

бачимо не модельний час (0,3с), а кількість зроблених за цей час кроків (300, враховуючи заданий крок 0,001с). Зараз головне те, що ми отримали графік, відкритий для редагування. В подальшому непотрібну лінію часу вилучимо, а вісь абсцис у кроках замінимо на вісь у часі.

4.4.2 Побудова графіків із М – файлу. Згідно прикладу, що розглядається, після моделювання у робочому просторі сформовано два вектори, або масиви tout та Mw, перший одномірний, другий двомірний, рис. 4.6.

Всі рядки векторів пронумеровані згідно номеру кроку інтегрування "i" від 1 до 301. Звертатися до змінних можна таким чином t_i=tout(i); M_i=Mw(i,1); w_i=Mw(i,2). У створеному М – файлі формуємо цикл вводу даних із робочого простору у файл (i=1:1:301). Графіки будуємо, використовуючи функцію plot. Графіки можна виводити по одному, а можна відразу кілька графіків, використовуючи команду Subplot. Текст М – файлу показаний нижче.

Array Editor: tout

Numeric for

	1
1	0
2	0.001
3	0.002
4	0.003
5	0.004
6	0.005

а.

Array Editor: Mw

Numeric format: shortG

	1	2
1	99.026	0
2	97.484	1.7651
3	95.97	3.4995
4	94.481	5.204
5	93.019	6.879
6	91.581	8.5249

б.

Рис. 4.6 Фрагменти значень масивів векторів tout, а., та Mw, б.

```
%Формування циклу вивода даних із робочого простору
i=1:1:301; %Початкова строка 1, крок по строкам 1, кінцевий крок 301
%Побудова графіків перехідного процесу
subplot(1,2,1)
plot(Mw(i,1),Mw(i,2),'k-'); %Динамічна механічна характеристика
grid on;
subplot(1,2,2)
plot(tout(i),Mw(i,1),'k-',tout(i),Mw(i,2),'k-'); %Осцилограми M(t); w(t)
grid on;
```

Тут функція Subplot (row,col,cur) для отримання блоку графіків розшифровується так: row- кількість рядків графіків, col – кількість стовпчиків (колонок) графіків, cur – порядковий

номер поточного графіку. У прикладі M – файлу заявлено row = 1, col = 2. Для всіх графіків заявлено 'k-', що означає суцільну лінію чорного кольору.

Отриманий невідредагований графік показаний на рис. 4.7. Додамо, що це не саме графік, а його зображення на екрані, отримане через Print Screen.

4.5 Редагування графіків.

На рис. 4.5 і на рис. 4.7 є строчка меню та панель інструментів, що дозволяє редагувати їх за єдиними правилами. Доступними є такі дії:

- Редагування графіку 
- Вставка тексту 
- Вставка стрілки 
- Вставка лінії 

Останні три функції досить прості і зрозумілі. Зупинимось детальніше на першій – редагування графіку. Після однократного клацання на графіку, вікно графіку стає виділеним і можна коректувати графіки за контекстним меню, клацанням правої кнопки миші. Після виділення конкретної кривої можна змінити товщину лінії, її колір та тип (суцільна, пунктирна, штрих - пунктирна).

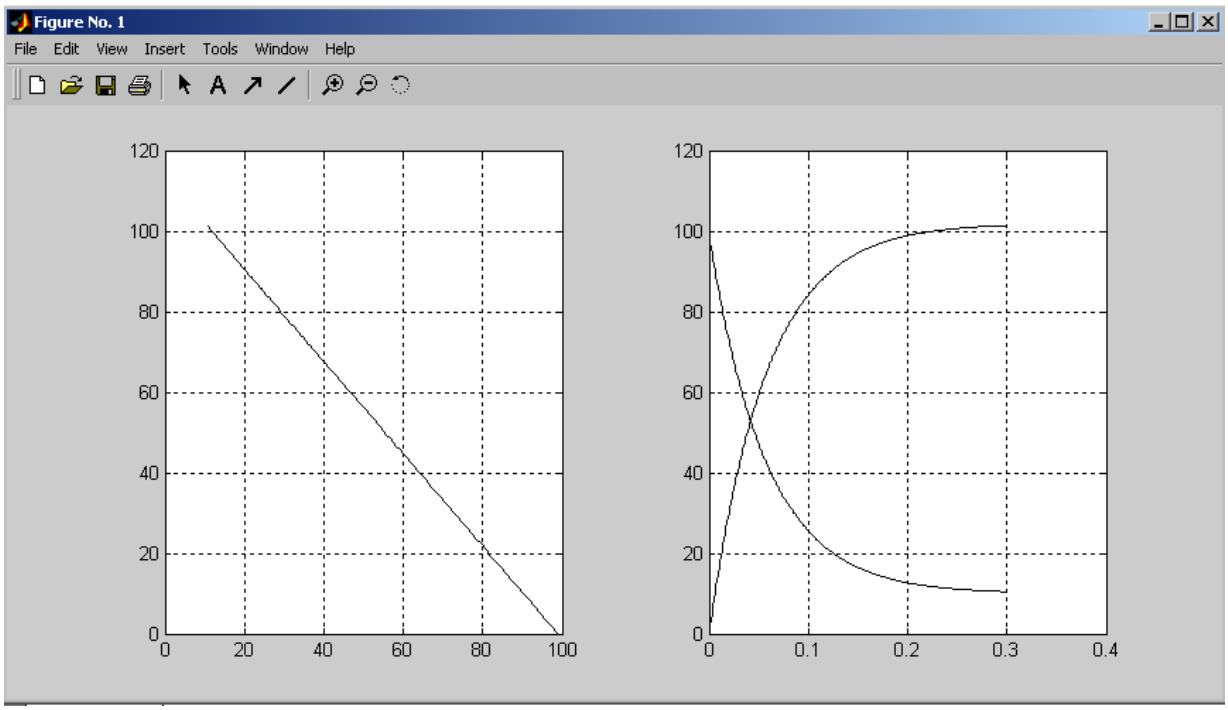


Рис. 4.7 Невідредаговані динамічна механічна характеристика та графіки перехідного процесу пуску

При подвійному клацанні на рисунку відкривається вікно Property Editor – Axes (Редактор властивостей осей).

Для кожної із обраних осей (X, Y, Z) можна виконати такі дії:

- Присвоїти номер осі (Label)
- Задати колір сітки та надпису на цієї осі (Color).
- Змінити положення надпису на осях (Location) – для осі X знизу або зверху, для осі Y з права або зліва.
- Встановити або відмінити сітку на графіку (Grid).
- Скорегувати маштабування за осями. За умовчанням всі масштаби встановлені автоматично. Якщо зняти відповідний прапорець, то можна виконати такі дії:
    - Змінити початок і кінець шкали відліку по осі (Limits). Це корисно зробити, наприклад, щоб убрати незаповнену частину графіку при t > 0,3 с.
    - Додати або зменшити кількість позначок на осі (Ticks), наприклад по осі X проставити позначки не через 20 Нм, як на рис. 4.7, а через 10 Нм. Відповідно буде змінена і сітка.
    - Замінити позначки на осях (Labels). Наприклад 0, 20, 40, 60... на будь які інші. Саме цією послугою ми скористаємось, щоб замінити на рис. 4.5 по осі X позначки кроку розрахунку (0, 100, 200, 300) на потрібні нам значення моменту (0, 20, 40…).

Можна також встановлювати форму відліку на осі лінійну від лінійної (Linear) до логаріфмічної (Log), від нормальної (Normal) до зворотної, наприклад на осі X зправа на ліво (Reverse). Якщо ви заплутались із редагуванням осей, або зроблені редагування вам не підходять, можна все повернути до початкового стану поставивши всюди прапорці Auto.

В вікні редагування властивостей осей можна змінювати загальний вигляд графіку або його стиль, якщо перейти до Style.

- Позначити підписом осі (Title).
- Обвести графік рамкою, якщо поставити прапорець на Axes box.
- Змінити колір фону графіку (Background).
- Змінити товщину лінії осей або рамки (Axes line widht).
- Змінити тип шрифту (Font name), товщину ліній шрифту (Font weight), уклін шрифту (Font angle), розмір шрифту (Font size).

На розмір шрифту осей та надписів треба звертати увагу, оскільки при переносі графіку у документ Word часто розмір графіку змінюється (частіше зменшується) і стає "молочитабельним".

Після закінчення редагування отримані графіки перенесені у чинний документ командами Edit→Copy Figure→Paste, рис. 4.8. При редагуванні графіків нашого прикладу були виконані наступні зміни.

- Змінена товщина ліній графіків і осей.
- Підписані осі координат.
- Введені позначення кривих.
- Дорисовані лінії механічної характеристики механізму (Ms), асимптоти швидкості ωs та моменту Ms, продовжена штрих пунктирною лінією динамічна механічна характеристика до швидкості ідеального неробочого ходу.
- Весь текст виконаний жирним шрифтом 12 розміру (за умовчанням встановлюється тонкий шрифт 10 розміру).

На рис. 4.8. зберігся сірий відтінок фону графіків. Його можна позбутись вже в документі Word, якщо виділити рисунок, а на панелі інструментів Рисування клацнути Меню "Зображення" → Чорно – біле, рис. 4.9. Тепер можна погодитись, що останній рисунок виглядає значно привабливіше, ніж початковий рис. 4.7.

Аналогічне редагування можна зробити і для рис. 4.5, отриманого безпосередньо із робочого середовища. На відміну від розглянутого прикладу, тут додатково треба замінити кольорові графіки на чорно – білі та змінити шкалу осі X з кроків розрахунку на час у секундах.

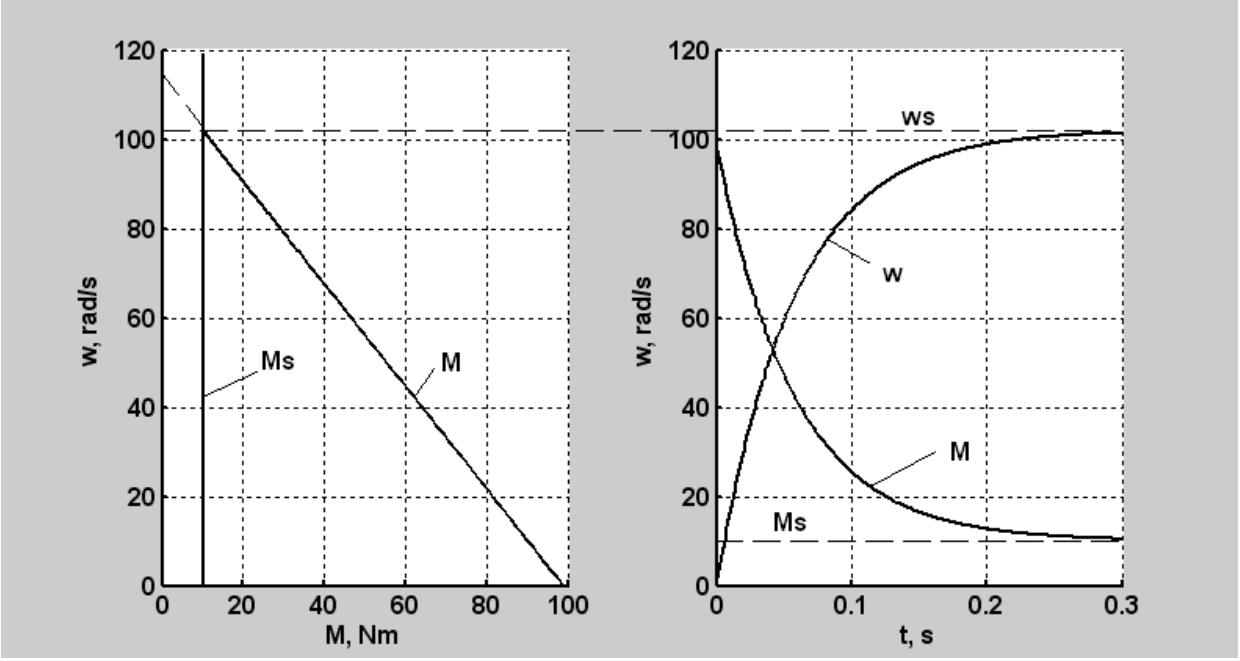


Рис. 4.8 Відредаговані динамічна механічна характеристика та графіки перехідного процесу пуску

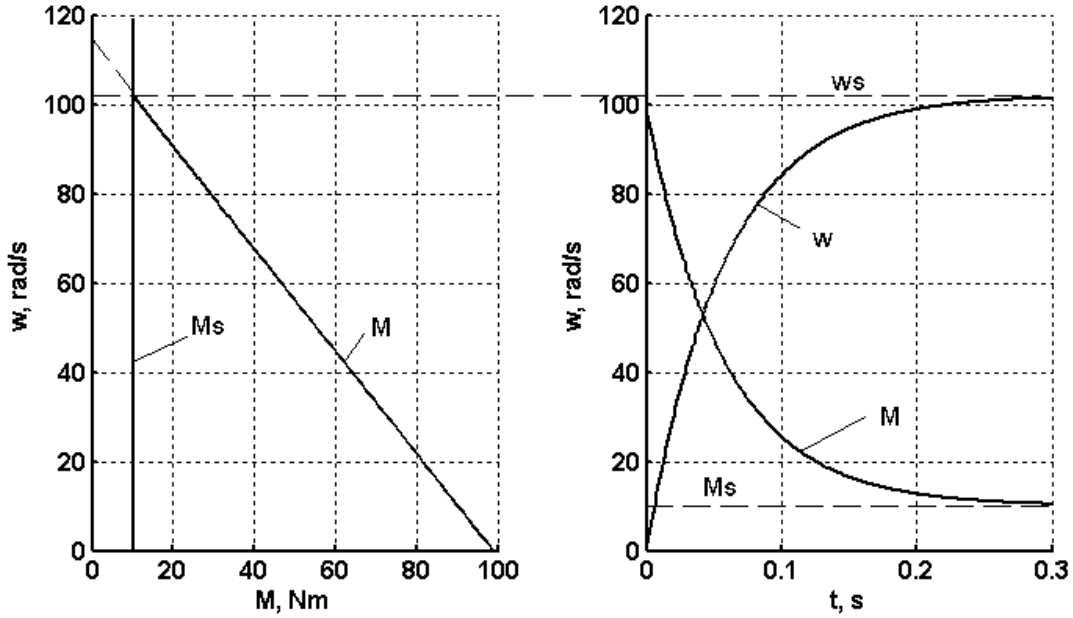


Рис. 4.9 Відредаговані динамічна механічна характеристика та графіки перехідного процесу пуску у чорно-білому зображенні