Міністерство освіти і науки України

СИСТЕМНИЙ АНАЛІЗ ТА ТЕХНОЛОГІЇ МОДЕЛЮВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ

Методичні вказівки до практичних занять
для здобувачів першого (бакалаврського) рівня вищої освіти
освітньої програми «Інформаційні системи та технології охорони і безпеки»
галузі знань F/12 Інформаційні технології
спеціальності F6/126 Інформаційні системи та технології
денної та заочної форм навчання

Луцьк 2026

УДК 303.732.4:004.94(07)
С 40

Рекомендовано до видання вченою радою факультету КІТ ЛНТУ,
протокол № _____ від « ____ » _____ 20 26 року.

Голова вченої ради факультету КІТ _____ Інна КОНДІУС

Електронна копія друкованого видання передана для внесення в репозитарій ЛНТУ

Директор бібліотеки _____ Наталія ПОЛІЩУК

Розглянуто і схвалено на засіданні кафедри комп'ютерної інженерії та безпеки ЛНТУ, протокол № _____ від « _____» _____ 20 26 року.

Завідувач кафедри КІБ _____ Тарас ТЕРЛЕЦЬКИЙ

Укладач: _____ Олена ЛЮБИМЕНКО, кандидат фізико-математичних наук, доцент кафедри комп'ютерної інженерії та безпеки ЛНТУ

Рецензент: _____ Світлана ЛАВРЕНЧУК, кандидат технічних наук, доцент кафедри комп'ютерної інженерії та безпеки ЛНТУ

Відповідальний за випуск: _____ Тарас ТЕРЛЕЦЬКИЙ, кандидат технічних наук, доцент кафедри комп'ютерної інженерії та безпеки ЛНТУ

С 40 Системний аналіз та технології моделювання інформаційних систем: методичні вказівки до практичних занять для здобувачів першого (бакалаврського) рівня вищої освіти освітньої програми «Інформаційні системи та технології охорони і безпеки» галузі знань F/12 Інформаційні технології спеціальності F6/126 Інформаційні системи та технології денної та заочної форм навчання / уклад. О.М. Любименко. Луцьк: ЛНТУ, 2026. 88 с.

Методичні вказівки до практичних занять з дисципліни «Системний аналіз та технології моделювання інформаційних систем» складено відповідно до діючої програми курсу.

У методичному виданні висвітлено матеріал, який необхідний для опанування принципів проведення системного аналізу та опанування технологій моделювання інформаційних систем для спеціальності F6/126 «Інформаційні системи та технології» для реалізації відповідних проектних рішень.

ЗМІСТ

ВСТУП

Освітній компонент «Системний аналіз та технології моделювання інформаційних систем» присвячено вивченню теоретичних та практичних основ аналізу та технологій моделювання інформаційних систем.

Вивчення курсу «Системний аналіз та технології моделювання інформаційних систем» дасть фахівцям з інформаційних системи та технологій базу знань і практичні навички, які є важливими для ефективного проведення системного аналізу та моделювання інформаційних систем. Зокрема, курс дозволяє:

– сформувати у студентів знання про принципи та методи системного аналізу;

– навчити використовувати технології моделювання для дослідження інформаційних систем;

– розвинути навички постановки, декомпозиції та формалізації задач;

– забезпечити опанування інструментів побудови структурних, функціональних та математичних моделей;

– сформувати вміння застосовувати сучасні програмні засоби аналізу й моделювання;

– виховати здатність оцінювати ефективність та оптимізувати функціонування інформаційних систем

Системний аналіз є напрямом, в якому поєднано методологію і досягнення математичних і прикладних наук. Системний аналіз у технічній галузі орієнтований на вирішення складних проблем аналізу та створення комп'ютерних, комунікаційних, інформаційних та інших технічних систем, і ґрунтується на принципах інженерних наук, імітаційному та інформаційному моделюванні об'єктів і процесів та націлений на застосування в конкретних проектах, розробленнях, прикладних дослідженнях .

Робиться акцент на глибоких знаннях в області системного аналізу та математичного і комп'ютерного моделювання процесів і систем різної природи, задач прогнозування, оптимізації, та прийняття рішень, а також здатність їхнього застосування для проєктування інформаційних систем.

Курс «Системний аналіз та технології моделювання інформаційних систем» розвиває перспективні напрями комп'ютерного моделювання інформаційних систем та поглиблене вивчення засобів програмування та їх застосування при розробці сучасних IT проєктів.

Метою навчальної дисципліни є сформувати у студентів системне мислення та здатність розглядати інформаційні системи як складні об'єкти, що функціонують у динамічному середовищі; надати знання щодо принципів, методів і засобів системного аналізу, які застосовуються для постановки та розв'язання задач управління, оптимізації й прийняття рішень; навчити методам побудови моделей інформаційних систем та сформувати практичні навички використання сучасних CASE-засобів, середовищ моделювання та аналітичних інструментів для аналізу, проєктування й дослідження інформаційних систем;

Практичні роботи спрямовані на практичне засвоєння базових і просунутих концепцій моделювання інформаційних систем, для забезпечення цих навиків, буде використано Scilab – це математична система для виконання чисельних розрахунків, яка за принципом роботи схожа із відомою математичною системою MATLAB. Засоби пакета дозволяють також виконувати деякі символьні перетворення, наприклад виконувати операції з поліномами. У середовище Scilab інтегрована також програма редагування блочних діаграм і симуляції Xcos, яка є аналогом програми Symulink з MATLAB. Незважаючи на те, що Scilab є клоном MATLAB, зрозуміло, що обидві системи мають різні програмні коди, а Scilab «вміє» конвертувати у свій формат документи з MATLAB.

Математична система Scilab дає змогу вирішувати низку завдань:

- розв'язування завдань лінійної алгебри;
- розв'язування нелінійних рівнянь і систем;
- розв'язування завдань оптимізації;
- диференціювання й інтегрування;
- розв'язування звичайних диференціальних рівнянь і систем;
- оброблення експериментальних даних (інтерполяція й апроксимація, метод найменших квадратів).

Програма належить до класу freeware, тобто безкоштовних, вона є відкритою системою, що дає змогу кожному бажаючому одержати доступ до кодів системи і внести до неї зміни, додавши нові функції, типи даних або просто настроїти систему «під себе». Підтримку системи забезпечує консорціум Scilab, до складу якого входять більше 20 учасників, у тому числі французькі компанії INRIA та ENPC, а також ESI Group та Scilab Enterprises.

У роботі розглядається українська локалізація версії 6.0.2. Ця версія містить у своєму складі додатки ATOMS (AutomaTic mOdules Management for Scilab) для керування зовнішніми модулями та Xcos для моделювання й емуляції гібридних динамічних систем.

Курс практичних робіт забезпечує формування практичних навичок програмування, проектування та аналізу алгоритмів, необхідних для вирішення прикладних задач комп'ютерної інженерії.

На практичних роботах кожен здобувач опрацьовує теоретичні відомості, що охоплюють перелік питань, винесених на обговорення, а потім виконує завдання, оформляє звіт до практичної роботи та дає відповіді на питання викладача (усно, з переліку контрольних питань). Виконання наступних робіт може опиратися на результати попередніх, тому не рекомендовано їх виконувати в довільному порядку.

Оцінювання практичних робіт здійснюється за критеріями:

1. Теоретична підготовка (розуміння основних понять та вміння пояснити теоретичні концепції).

2. Практичне виконання завдань (повнота та точність).

3. Технічна реалізація (правильно обрані структури відповідно до теми практичної роботи та конкретної задачі, коректна організація коду, логічна організація коду тощо).

4. Документація (наявність коментарів до програм, якість оформлення звіту – чіткий та детальний звіт, в якому студент пояснює, які завдання виконувалися, які труднощі виникали, як були вирішені проблеми. Звіт має бути структурованим і зрозумілим).

5. Вміння захистити роботу – студент повинен продемонструвати здатність чітко і впевнено відповісти на питання щодо виконаних завдань, пояснити вибір рішень і відповісти на технічні запитання.

Практична робота №1

Введення у систему Scilab. Найпростіші математичні операції з числами.

Мета – ознайомлення з інтерфейсом Scilab, базовими типами даних та виконанням елементарних обчислень.

Завдання: ознайомитися з робочим середовищем Scilab; дослідити типи змінних та їх властивості. Виконати арифметичні операції над числами.

Література: [1, 3, 5, 9, 10].

Теоретичні відомості

Інтерфейс системи.

Основою інтерфейсу системи є вікно команд, яке називається консоль. Саме у ньому вводяться дані та виводяться результати Вікно програми структуроване у чотирьох вікнах – областях, які називають також компонентами.

1. Перегляд файлів. Область для вибору файлів на зразок провідника.
2. Вікно команд. Основна область, призначена для введення команд та одержання результатів. Ця область називається «консоль».
3. Перегляд змінних. Відображення усіх змінних, що використовуються у поточній сесії.
4. Журнал команд. Список команд, що були введені у поточній сесії.

Будь-яку область можна вилучити, натиснувши кнопку закриття у заголовку вікна-області. Але в будь-який момент початкове (повне) розташування вікон (компонування) можна відновити. Для цього потрібно:

1. звернутися до настройок програми;
2. відкрити групу «загальні», з якої вибрати пункт «компонування»;
3. натиснути кнопку «відновити початкове компонування»;

Крім цього, вікно програми містить звичайні елементи Windows: заголовок, рядок меню, панель інструментів і рядок стану.

Рядок стану зокрема відображає номер рядка доку- мента та позицію, де знаходиться курсор.

Вікно команд (консоль).

Інтерфейс вікна має текстовий вигляд. До тексту за замовчуванням застосоване певне форматування. Але його можна змінити. Для цього слід виконати команду «Змінні ▸ Налаштування» або натиснути на панелі стандартних інструментів кнопку налаштування. Після цього з’явиться вікно вибору параметрів налаштування, в якому на вкладці «Шрифти» можна змінити тип, розмір і накреслення шрифту.

Для очищення вікна команд слід виконати команду clc, натиснути функціональну клавішу <F2> або виконати команду Змінні ▸ Спорожніти консоль.

Вікно перегляду змінних.

Усі змінні, що використовуються в поточній сесії, відображаються в окремому вікні, яке за замовчуванням є одним з елементів інтерфейсу системи. У вікні відображається перелік змінних і їх атрибути: розмірність змінної або її

значення, тип даних і область видимості. Вікно перегляду змінних за замовчуванням відображає тільки змінні, визначені користувачем, і не показує системні змінні (наприклад, змінна *%e*, у якій зберігається наближене значення числа *e*). За необхідності відображення у вікні перегляду системних змінних слід клацнути в області вікна перегляду змінних, що призведе до появи меню цього вікна. У ньому слід вибрати команду Фільтр і натиснути на пункті Приховати змінні Scilab і тим самим зняти прапорець з цього пункту (рис.1.1).

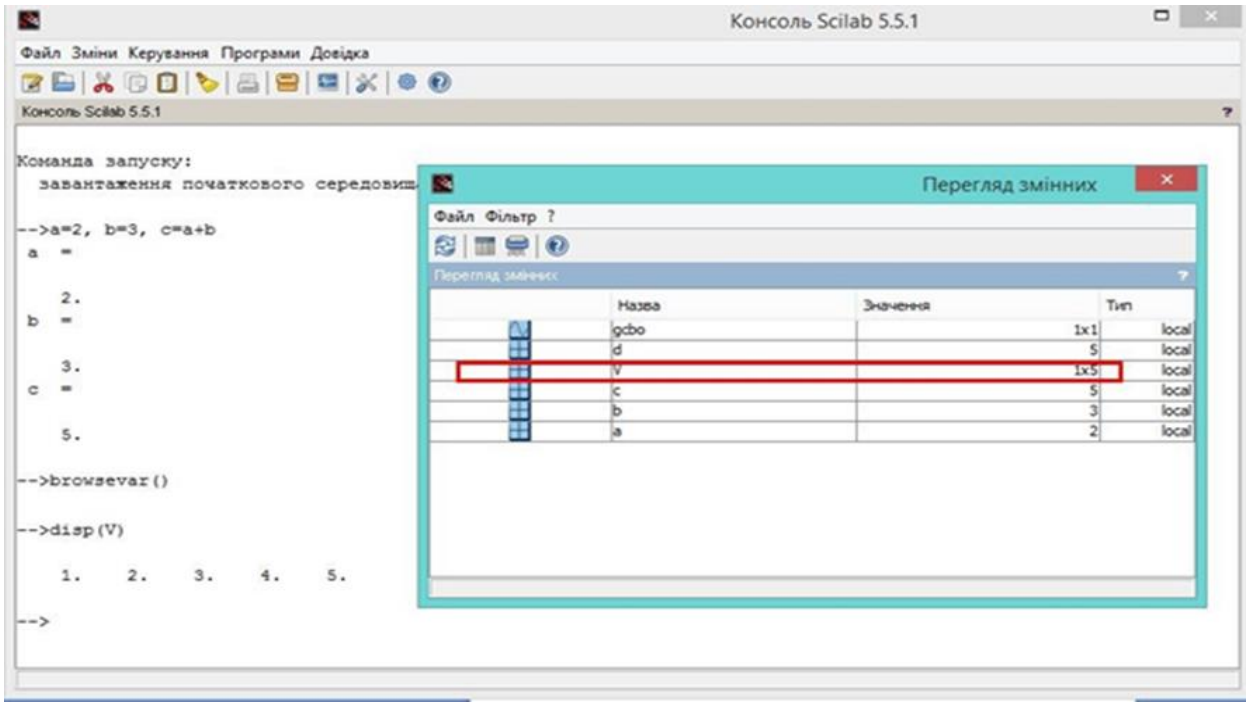


Рисунок 1.1 – Фільтр

Так саме і відображення тих чи інших груп системних змінних регулюється командою Фільтр. Список змінних у вікні оновлюється автоматично, але оновлення можна ініціювати і вручну натиснувши у вікні перегляду змінних кнопку «Освіжити змінну».

У вікні перегляду змінних не відображуються значення тих масивів, які містять достатньо багато елементів. Одержати їх можна функцією disp, яка має такий синтаксис:

disp (ім’я змінної 1, ім’я змінної 2, …).

Переглянути значення будь-якої змінної можна ще простіше, для чого потрібно у вікні перегляду змінних двічі натиснути на рядку з її назвою. Це спричиніть появу вікна редактора змінних в якому можна не тільки переглянути значення змінної, але й змінити їх не використовуючи для цього відповідні команди (рис.1.2).

Якщо область перегляду змінних була вилучена, то для відображення потрібно виконати команду Програми ▸ Перегляд змінних або ввести у рядку введення функцію browsevar() і натиснути клавішу <Enter>.

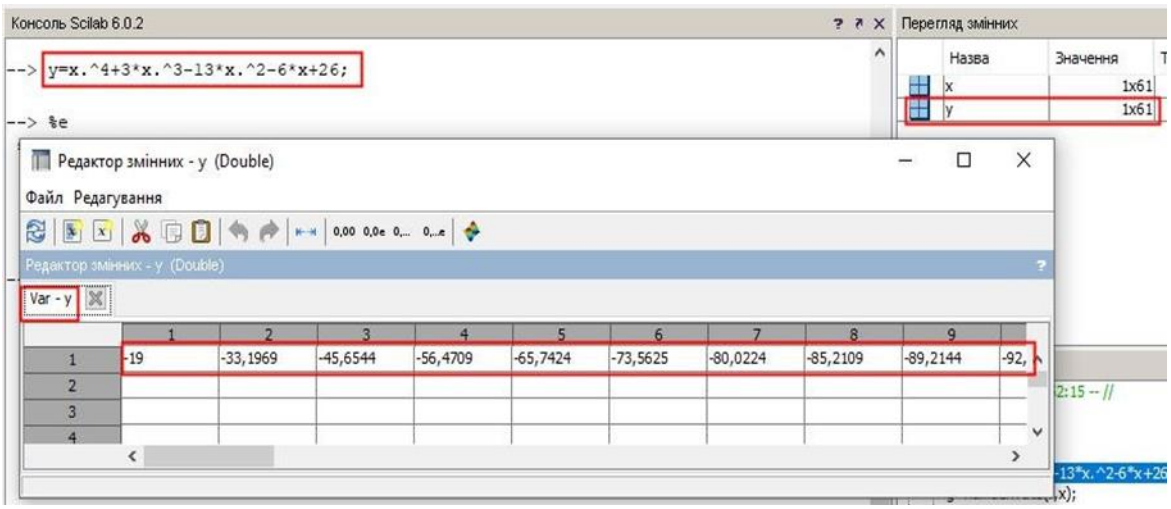


Рисунок 1.2 – Редактор змінних

Журнал команд.

У журналі команд відображаються усі команди, що були введені останнім часом під час роботи з системою, у тому числі і команди з попередніх сесій. Вікно за замовчуванням є одним з компонентів (елементів інтерфейсу) системи.

Будь-яку команду з журналу можна виконати у вікні консолі, для чого потрібно просто двічі клацнути на рядку з її назвою.

Якщо ця область була вилучена, то для її відображення слід виконати команду Програми ▸ Журнал команд, після чого в окремому вікні з’явиться вікно з переліком команд.

Сесія.

Сеанс роботи із системою називають сесією (session). Сесія – це поточний документ, що містить роботу користувача, тобто рядки введення даних, команд та функцій, виведення результатів, повідомлення про помилки тощо. Визначення змінних і функцій, які розташовані в робочій області пам’яті, (але не саму сесію) можна записати на диск за командою Файл ▸ Зберегти середовище… Сам файл зберігається з розширенням SOD.

Слід зазначити, що можливості збереження всього тексту сесії команда Зберегти середовище не дає. Це зроблено навмисно: сесія є результатом проб і помилок, її текст разом із правильними визначеннями може містити повідомлення про помилки, непотрібні виведення і т. ін. Необхідності зберігати все це, як правило, немає. Проте це не значить, що відсутня можливість записати раціональне зерно, створене під час роботи над розв’язком задачі.

Слід просто скористатися редактором і відлагоджувачем, які дозволяють (після налагодження програми) одержати документ у коректній формі без синтаксичних і інших помилок. Завантаження з диска даних робочої області відбувається за командою Файл ▸ Завантажити середовище….

Якщо все ж таки є потреба повністю зберегти сесію, то це можна зробити за допомогою оператора diary, що призначений для ведення щоденника сесії. Оператор має два формати, за допомогою яких починається і припиняється процес запису сесії:

– diary (filename) – запис на диск всі введені команди й отримані результати до файлу з ім’ям filename; ім’я файлу подається в апострофах і може включати шлях;

– diary(0) – припиняє запис у файл.

Після закінчення роботи файл автоматично не зберігається.

За замовчуванням система після її завантаження шукає документи в системній папці документів користувача. Для зміни папки у поточній сесії слід виконати команду Файл ▸ Змінити поточний каталог…, вибрати потрібну папку і натиснути «ОК». Якщо ж потрібно змінити папку за замовчуванням, то для цього слід звернутися до налаштувань програми, в групі «Загальні» встановити перемикач в положення «Використовувати типовий каталог» і задати шлях до нього. Відображення поточної папки для збереження доку- ментів здійснюється командою Файл ▸ Показати поточний каталог….

Документ системи.

Будь-які обчислення (часом досить складні) у системі можна виконати в режимі прямих обчислень, тобто без підготовки програми. Це перетворює Scilab у потужний калькулятор, здатний здійснювати не тільки звичайні для калькуляторів обчислення (наприклад, виконувати арифметичні операції й обчислювати елементарні функції), але й здійснювати операції з векторами й матрицями, комплексними числами, рядами й поліномами, розв’язувати системи лінійних рівнянь і т. ін.

Документ являє собою послідовність рядків введення у вигляді операторів, команд і функцій і результатів виведення. Нова інформація вводиться у рядку, на початку якого знаходиться символи «-->«, які є ознакою запрошення введення інформації.

В одному сеансі роботи з системою можна працювати тільки з одним документом (інші закриваються).

Для використання системи у режимі прямих обчислень потрібно знати, як саме у цьому разі вводиться інформація.

Правила введення інформації в документ:

1. Призначення функціональних клавіш при виконанні дій у рядку введення відповідає загальноприйнятим у текстових редакторах. Так, переміщення вліво або вправо досягається натисненням на клавіатурі стрілки вліво або вправо; на початок або кінець рядка – клавішами <Home> та <End>; <Esc> призначена для очищення рядка, <Ins> для ввімкнення або вимкнення режиму вставки і т. ін.

2. Введення нової інформації здійснюється відразу після символів «-->».

3. Будь-яке введення завершується натисканням <Enter>.

4. Для обчислення математичного виразу і відображення у документі результату за закінченням уведення виразу слід натиснути <Enter>.

2+3 <Enter> ans =

Результат обчислень виводиться в рядках виведення без символу «-->». При цьому результат обчислення автоматично присвоюється системній змінній ans (від англ. answer – відповідь):

5. Якщо не потрібно виводити результат, то введення інформації закінчується символом «;». В одному рядку можна ввести кілька операторів та

(або) команд, відокремлюючи їх символом «,»:

```
-->2+3, 3/5;
ans = 5.

ans = 0.6.
```

У цьому прикладі система послідовно виконує дві операції: додавання та ділення. Оскільки друга операція не містила присвоювання, результат збережено у змінній ans.

6. Якщо виведення результату обчислення не потрібне, то його можна відключити, завершуючи оператор символом «;»:

```
2+3; 3/5
```

7. Якщо вираз, що вводиться, є довгим і не поміщається у рядку, то наприкінці незавершеного рядка слід ввести дві крапки «..».

8. Нову інформацію можна додавати використовуючи стек раніше введеної інформації. Доступ до стеку здійснюється з рядку введення клавішами управління курсором «нагору» й «униз», які дозволяють гортати раніше введені команди знизу-вверх і зверху-вниз. Надалі їх можна повтор- но використовувати або створювати на їх основі інші.

9. Ще зручніше додавати інформацію використовуючи журнал команд. Для цього достатньо просто здійснити подвійне натискання на рядку з потрібним введенням з журналу.

10. Редагувати раніше введену інформацію не можна.

Сценарій.

Сценарій – це послідовність команд, операторів і функцій, що підключаються до документа як одне ціле (на зразок підпрограми у мовах програмування) і виконуються. Для створення сценаріїв система має спеціальний засіб – редактор сценаріїв. Доступ до редактора здійснюється за командою Програми ▸ SciNotes або натисканням на панелі інструментів кнопки «Запустити SciNotes»  . Після цього з’являється вікно редактора, в якому і створюється сценарій. Призначення редактора, так саме, як і взагалі для редакторів для створення програм алгоритмічними мовами, потрійне. По-перше, він дозволяє виконувати усі типові дії редагування послідовності введення, тобто сценарію. По- друге, він здійснює синтаксичну перевірку команд та функцій. І, нарешті, він дозволяє завантажити сценарій на виконання у вікно консолі.

Редактор сценаріїв є найбільш ефективним засобом створення нових математичних алгоритмів. Під час формування сценарію одночасно можна перевіряти його працездатність, тобто завантажувати його на виконання у середовищі Scilab, скориставшись командою редактора з групи Виконати. Наприклад, за командою Виконати ▸ файл з виведенням здійснюється завантаження сценарію до вікна консолі, де він виконується, і користувач одержує результати його виконання.

Файл сценарію призначений для виконання його надалі у середовищі Scilab.

Подальша робота з ним може відбуватися двома шляхами.

1. За командою Файл ▸ Виконати здійрахунки за сценарієм, при цьому виводяться тільки результати.
2. За командою Файл ▸ Відкрити файл відбувається завантаження сценарію в редактор сценарію SciNotes, звідки він надалі завантажується на виконання. За таким варіантом у документі відображається як результати, так і усе введення.

До редактора сценаріїв можна завантажити кілька файлів, кожний з яких буде відображатися в окремій вкладці.

Приклад.

1. Створюємо файл сценарію (рис.1.3):

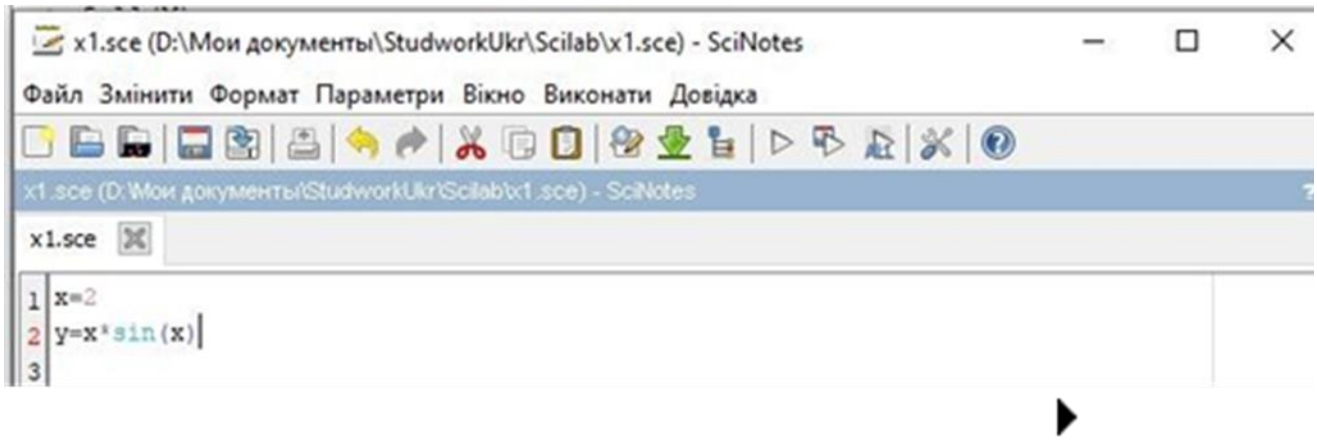


Рисунок 1.3 – Файл сценарію

2. Запам’ятовуємо його командою Файл Зберегти в потрібному місті.
3. У середовищі Scilab виконуємо Файл ▸ Відкрити і у вікні відкриття файлів виберемо збережений файл.

Результат виконання:

```
x  =

  2.
y  =

  1.8185949

Execution done.
```

Довідкова система

Звернення до довідкової системи здійснюється командою: Довідка ▸ Довідка Scilab або натисканням кнопки  на панелі стандартних інструментів. Отримати довідку можна також шляхом використання оператора help. Наприклад, для отримання довідки відносно оператора exec слід ввести:

```
help exec
```

Система має бібліотеку демонстраційних прикладів (необхідність їх встановлення на ПК користувача визначається під час інсталяції системи). Звернення до неї відбувається за командою Довідка ▸ Демонстрації Scilab. Демонстраційні приклади відкриваються в новому вікні, яке потрібно закривати по закінченню перегляду прикладу.

Вхідна мова системи

Алфавіт

Алфавіт визначає сукупність символів і слів, що використовуються для запису команд.

Алфавіт системи містить:

– малі і великі латинські літери;
– арабські цифри від 0 до 9;
– системні змінні;
– оператори;
– імена вбудованих функцій;
– спеціальні знаки.

Правила синтаксису мови системи.

1. Усі імена команд і функцій записуються літерами латинського алфавіту.
2. Аргументи операторів і функцій записуються у круглих дужках.
3. Велика і маленька літери розрізняються.
4. У числах ціла частина від дробової відокремлюється крапкою.
5. Знаки арифметичних операцій у виразах потрібно обов'язково вказувати.
6. Порядок дій у математичних виразах відповідає загальноприйнятому порядку дій у математиці.

Система здійснює синтаксичний контроль введення і за наявності помилки видає про це повідомлення.

Текстові коментарі.

Основним режимом роботи системи є, звичайно, робота з математичними об'єктами. Але система дозволяє вводити пояснення до документа (тобто коментарі), які роблять документ більш зрозумілим.

Для введення коментарів з будь-якої позиції рядка вводяться два символи «//», після чого вводиться безпосередньо сам коментар.

Об'єкти системи.

Обчислення здійснюються за допомогою математичних виразів, які є головним об'єктом будь-якої математичної системи. Вираз визначає те, що повинно бути обчислено в чисельному вигляді. Математичні вирази складаються з чисел, констант, змінних, операторів, функцій і спеціальних знаків.

Числа.

За найпростішим варіантом у режимі прямих обчислень відбувається робота з числами. Число – найпростіший об'єкт системи, що представляє кількісні дані. Вони можуть бути дійсними: цілими, дробовими, з фіксованою точкою. Ціла частина відокремлюється від дробової частин крапкою. Знак «+» для додатних чисел не використовується. Проміжки між символами в числах не допускаються. Наприклад: 0, -3, 2.301.

```
-->6+7
 ans  =

    13.

-->2.345*log(10)-1
 ans  =

    4.399562
```

Константи.

Константа – це попередньо визначене числове або символьне значення, представлене унікальним ім’ям. Числа (наприклад, 1, -2 і 1.23) є числовими константами без імені.

Оператори.

Оператори – це елементи мови, за допомогою яких створюються математичні вирази. До них, наприклад, належать знаки арифметичних операцій, обчислення сум, добутків, похідних, інтегралів і т. ін. Оператори використовуються разом з операндами. Наприклад, у виразі «5-1» знак «-» є оператором віднімання, а числа «5» і «1» – операндами. Після визначення операндів оператори стають блоками, що виконуються.

Арифметичні оператори.

Оператори – це елементи мови, за допомогою яких створюються математичні вирази. Вони призначені для виконання арифметичних дій над числовими величинами і конструювання математичних виразів. Оператори використовуються разом з операндами. Наприклад, у виразі «5-1» знак «-» є оператором віднімання, а числа «5» і «1» – операндами.

За замовчуванням результат обчислення містить вісім цифр:

```
-->3+7.123456789
ans =
10.123457.
```

Але цю кількість можна змінити. Для цього використовується функція printf, яка є емулятором однойменної функції мови С.

Синтаксис функції:

printf (формат, ім’я змінної 1, ім’я змінної 2, …)

Оскільки функція емулює функцію мови С, аргумент «формат» описується у відповідності з правилами цієї мови. Наприклад:

-->printf("%1.10f",ans); 10.1234567890

Змінні. Зазвичай у документі доцільно користуватися не конкретними числовими значеннями, а змінними. Змінна – це пойменований об’єкт, значення

якого в документі може змінюватися. Імена констант, змінних і інших об’єктів називають ідентифікаторами.

У свою чергу і значення обчислення можна запам’ятати у змінній.

Правила надання імен ідентифікаторам:

1. вони можуть складатися з будь-яких латинських і грецьких літер, цифр;
2. великі і малі літери розрізняються;
3. починаються тільки з літери;
4. не можуть містити проміжків;
5. не можуть збігатися з іменами вбудованих чи визначених користувачем функцій;
6. не можуть містити символів кирилиці.

```
-->c=2; b=3
 b  =

    3.

-->c+b
 ans  =

    5.
```

Для надання змінній значення застосовується символ «=«.

Змінні можуть бути локальними і глобальними. Локальні змінні використовуються і зберігаються тільки в поточній сесії. Для створення глобальної змінної її потрібно описати за допомогою оператора global.

За однім оператором global можна описати кілька змінних, при цьому вони повинні відокремлюватися проміжком.

-->global x z

Одержання списку змінних, що використовуються в поточній сесії, здійснюється функцією who. Застосування цієї функції з аргументом «global» виводить список глобальних змінних, а з аргументом «local» – список змінних, що використовуються в поточній сесії, а також системних змінних:

```
-->who("local")
ans =
!a                  !
!                   !
!editvar            !
…
!browsevar          !
!                   !
!printf             !
!                   !
!scicos_pal         !
!                   !
!%scicos_menu       !
…
```

Змінні розташовуються в робочій області, їх можна знищувати; а можна взагалі очистити всю цю область. Для цього використовується оператор ***clear***. При його використанні без аргументів знищуються всі змінні, а для знищення окремої або кількох змінної в якості операндів використовуються імена змінних, які відокремлюються одна від одної проміжком:

clear a [b c …]

Системні (зарезервовані) змінні.

Системні змінні – це невелика група особливих об'єктів, які не можна віднести ні до класу констант, ні до класу змінних. Прикладом такої змінної є *ans*, яка зберігає результат останньої операції. У системі також є невелика група скалярних системних змінних, відмінністю яких від інших є наявність перед їх іменем символу «%».

Перелік основних системних змінних містить наступна таблиці 1.1.

Таблиця 1.1 – Системні змінні

Уведення	Призначення
%i	Уявна одиниця($\sqrt{-1}$).
%pi	Число π.
%e	Число e (основа натурального логарифма, 2,7182818).
%eps	Похибка операцій над числами із плаваючою крапкою ($2{,}22^{-16}$).
%inf	Системна нескінченність.
%t	Логічне «істинне».
%f	Логічне «хибне».

Наприклад, введення системної змінної %pi за замовчуванням еквівалентно введенню числа «3.1415927». Системні змінні захищені і не можуть бути вилучені. Як правило їх не можна і перевизначити. Наприклад, спроба надати системної змінної %pi інше значення викличе помилку:

```
-->%pi=3.14
!--error 13
redefining permanent variable.
```

Функції.

Scilab має убудовані функції для розрахунку всіх математичних дій, а також багато спеціальних функцій.

Правила запису функцій:

1. Ім'я функції чутливе до регістру, їх назви записуються маленькими літерами;
2. Аргументи функції беруться до круглих дужок.
3. Функції можна вкладати одну до одної.

У наступній таблиці 1.2 наведено поширені математичні функції.

Таблиця 1.2 – Елементарні математичні функції

Функція	Назва
Математичні функції	
abs(x)	|x|
sqrt (x)	$\sqrt{x}$
Тригонометричні функції	
sin(x)	Синус
cos(x)	Косинус
tan(x)	Тангенс
ctg(x)	Котангенс
asin(x)	Арксинус
acos(x)	Арккосинус
Експоненціальні	
exp(x)	Експонента числа x
log(x)	Натуральний логарифм числа x

Функції користувача.

Користувач може самостійно створювати власні функції як для конкретного сеансу роботи, так і для постійного використання.

Функція користувача будується за певними правилами:

1. Вона починається з ключового слова «function», після якого через проміжок записується аналітичний вираз функції. Функція може мати вхідні параметри, наприклад, function y=f(x).
2. Математичний вираз функції, наприклад, x*cos(x).
3. Закінчується опис функції користувача ключовим словом «endfunction».
4. Складові функції користувача відокремлюються комою.

Отже, для наведеного прикладу функція користувача буде мати вигляд:

```
-->function y=f(x), y=x*cos(x), endfunction
```

Але такий варіант підходить у тому випадку, якщо «тіло» функції містить тільки один оператор. Якщо ж є потреба застосувати у функції кілька операторів, то вона формується за «лінійним» принципом, коли кожний рядок містить один оператор. Рядок введення можна закінчувати символом «;», але можна і взагалі нічого не ставити:

```
function difur=syst(t,y)
difur=zeros(2,1) difur(1)=cos(y(1)*y(2))
difur(2)=sin(y(1)+y(2)*t)
endfunction
```

При створенні користувацьких функцій у редакторі сценаріїв для їх відокремлення від інших введень передбачено автоматичне виділення їх іншим кольором.

Виклик функції:

```
--> y=[1 2];
```

```
--> difur=syst(1,y) difur =
-0.4161468
0.14112
```

Типи даних.

У системі визначається шість типів числових цілих даних: 32, 16 і 8-бітові дані зі знаком і без знака.

Визначити тип змінної можна за допомогою функції type (ім'я змінної), яка повертає ціле число, що визначає належність змінної до певного типу.

Наприклад, число «1» – означає дійсну або комплексну матрицю, 2 – поліноміальну матрицю, 4 – логічний тип і т. ін.

```
--> M=[0.25 1.26; 7 8]
M  =
0.25   1.26
7. 8.
--> type(M) ans  =
1.
--> D = [%t %f; %T %F] D  =
T F
T F
--> type(D)
ans  =
  4.
```

За допомогою спеціальних команд можна перетворювати дані одного типу до іншого.

Наприклад, перетворення даних дійсного типу до цілого здійснюється за допомогою функції iconvert (матриця дійсних елементів, тип перетворення).

Тип перетворення – код типу даних.

Наприклад, якщо тип = 1, то це функція повертає цілі числа в діапазоні [-128, 127].

```
--> m=[1.23, 1.25; 2.45, 3.45]
m  =
1.23   1.25
2.45   3.45
--> y=iconvert(m,1) y  =
1  1
2  3
```

Комплексні числа.

Система працює і з комплексними числами $a + bi$. Вони записуються в алгебраїчній формі з уявною одиницею, яка позначається %i у вигляді a+b*%i, де a і b – відповідно дійсна й уявна частини числа, наприклад: (3+5*%i) + 2*%i.

Логічний тип.

Система дозволяє використовувати змінні логічного типу. При цьому істина подається літералами %t або %T, а хибне – %f або %F.

З такими змінним можна проводити звичайні операції, що застосовуються над даними такого типу (кон'юнкція, диз'юнкція і т. ін.:

```
-> a=%t

a  =
  T
--> b=%f
b  =
  F

--> a&b
ans  =
  F
--> a|b
ans =
T
```

Завдання

1. Змініть шрифт інтерфейсу системи на «Courier».
2. Змініть папку для збереження інформації на Вашу папку.
3. Встановіть режим запису тексту сесії в текстовий файл з довільним ім'ям.
4. Введіть у документ коментар, наприклад, «Моя перша програма у Scilab».
5. Припиніть запис тексту сесії.
6. Відкрийте створений текстовий файл і переконай- теся, що він містить введену інформацію.
7. Відновіть режим запису тексту сесії в текстовий файл.
8. Обчисліть вирази:
   - π*100+179.
   - $c = a + b$ при $a = 2, b = 3$
   - R = 157,43 : 23,456.
9. При виведенні змінної R установіть кількість позицій дробової частини «12».
10. Виконайте такі дії з комплексними числами:
    - сформуйте комплексні числа x=6+2i і y=7-2i і виконайте з ними операції додавання, віднімання, ділення і множення;
    - добудьте корінь $\sqrt{x}$;
    - обчисліть тангенс *y*.
11. Виведіть список системних змінних і змінних, що використовуються в

поточній сесії.

12. Запам’ятайте змінні і їх значення з робочої області пам’яті поточної сесії.

13. Припиніть запис тексту сесії.

14. Закінчіть роботу з Scilab.

15. Завантажте збережені дані попередньої сесії.

16. Обчисліть вираз $l = c + a + b$, використовуючи значення змінних результатів збереженої сесії.

17. Знищить змінну a.

18. Спробуйте обчислити вираз $l = c + a + b$, *що* відбудеться?

19. Знищить усі змінні

20. Відкрийте вікно перегляду змінних і переконайтеся у відсутності створених Вами змінних.

21. Обчисліть вираз $c = a + b$ при $a = 2, b = 3$.

22. Переконайтеся у наявності створених Вами для обчислення виразу змінних.

23. Створіть сценарій, у якому передбачте виконання дій для обчислення виразів з пункту 8.

24. Збережіть сценарій.

25. Закінчіть роботу з Scilab.

26. Відкрийте Scilab і завантажте сценарій у новій сесії.

27. Виконайте сценарій (виведенням).

28. Виведіть значення змінних, що містить сценарій.

29. Створіть сценарій повного виконання лабораторної роботи. Переконайтеся у його працездатності.

Вказівки до виконання.

1. Для зміни локалізації (мови інтерфейсу) програми слід звернутися до настройок програми, у групі «Загальні» вибрати мову із списку «Типова мова», а потім натиснути кнопку «ОК». Систему після цього потрібно перезавантажити.

2. За замовчуванням система після її завантаження шукає документи системи в системній папці Documents and Setting / Ім’я користувача. Для зміни папки слід виконати команду Файл Змінити поточний каталог, знайти потрібну папку і натиснути «ОК». Визначити папку, де саме зберігаються документи, можна за командою Файл Показати поточний каталог.

3. Для збереження лабораторної роботи з метою подальшої демонстрації її роботи є збереження тільки введення в документ, тобто сценарію (файла з розширення SCE). Це можна зробити різними шляхами, наприклад скопіювати виконану роботу до текстового редактора або редактора SciNotes і вилучити усю зайву інформацію, в тому числі символи «-->«, залишивши тільки введення на зразок такого:

```
// Моя перша програма
%pi*100+179
 a=1; b=2;
c=a+b
```

4. Визначення змінних і функцій, які розташовані в робочій області пам’яті, можна записати на диск, виконавши команду Файл Зберегти середовище… з розширенням BIN. Завантаження з диска цих даних здійснюється за командою Файл Завантажити середовище…

Контрольні питання

1. Для яких розрахунків призначена система?
2. Аналогом якої системи є Scilab?
3. Як змінити папку для збереження інформації?
4. Для чого змінюється папка для збереження інформації?
5. Що таке «сесія»?
6. Що таке «сценарій»?
7. Як створити коментар?
8. Яке призначення редактора сценаріїв?
9. Як створити сценарій?
10. Як встановити, припинити режим запису тексту сесії?
11. Як запам’ятати визначення змінних і функцій, які розташовані в робочій області пам’яті?
12. Як завантажити визначення змінних і функцій, які розташовані в робочій області пам’яті?
13. Що слід зробити для того, щоб результат виконання математичного виразу не було відображено на екрані?
14. Скільки значущих цифр система виводить на екрані за замовчуванням?
15. Які принципи запису комплексних чисел.
16. Що таке «системна змінна»?
17. Які є основні системні змінні системи і як ввести їх в документ?
18. Як одержати список змінних, що використовуються в поточній сесії?
19. Як можна знищити окрему змінну, всі змінні?
20. Як у вікні консолі відкрити вікно перегляду змінних?
21. Як запам’ятати результати сесії?
22. Що містять результати сесії?
23. Як можна визначити значення змінної?

Практична робота №2

Створення векторів і матриць (масивів). Основні операції над масивами

Мета – навчитися створювати вектори і матриці, виконувати з ними операції.

Завдання: виконати базові математичні операції над матрицями,. Використати вбудовані функції для обчислення. Розв’язати завдання допомогою редактора сценаріїв

Література: [1-2, 4, 5, 7-8].

Теоретичні відомості

Створення масивів. Основним типом даних системи є масиви.

Створення векторів.

Для створення вектора-рядка і надання йому значень слід ввести його ім’я, знак присвоювання і в квадратних дужках через проміжок або кому елементи вектора.

Наприклад, якщо слід створити вектор-рядок V зі значеннями елементів 1, 2, 3 і 4, то використовується запис V=[l 2 3 4] або V=[1,2,3,4], які за змістом є ідентичними:

-->V=[1 2 3 4] V =1. 2. 3. 4.

При створенні вектора-стовпчика його елементи відокремлюються через «;». Створити вектор-стовпчик можна також шляхом транспонування вектора-рядка. Символом транспонування є апостроф:

-->V2=V' V2 =
1.
2.
3.
4.

Створити вектор можна і шляхом надання значення його довільному елементу.

Усі інші елементи за таким варіантом дорівнюватиме «0».

Ще один варіант створення вектора пов’язаний з використанням оператора «:». У цьому разі вектор створюється у вигляді числової послідовності, елементи якої змінюються з певним кроком від якогось початкового значення до якогось кінцевого. Синтаксис оператора має вигляд: ім’я масиву=початкове значення: крок зміни: кінцеве значення

Наприклад:

--> V3=1:2:5
V3 =
1. 3. 5.

Якщо опустити аргумент «крок зміни», він дорівнюватиме «1», тобто вектор сформується як послідовність цілих чисел.

Створити вектор можна також з матриці, про які мови піде нижче. Наведений приклад демонструє перетворення матриці у вектор-стовпчик:

--> M=[1 3; 6 8] M =
1. 3.
6. 8.
--> V=M(:) V =
1.
6.
3.
8.

Для звернення до n-го елемента вектору V використовується запис у вигляді ім’я вектору (номер елемента), наприклад:

k=V(1)+25

Створення матриць.

Створити матрицю можна так само як вектор, тобто ввести її ім’я, знак присвоювання, і в квадратних дужках – перелік її елементів. Для відокремлення елементів рядків використовується проміжок або кома, а для відокремлення одного рядка від іншого – крапка з комою. Наприклад, для створення матриці M розмірності 2×2 з елементами 1,2,3,4, у рядку введення слід ввести:

M = [1,2;3,4].

Створити вектор або матрицю можна також шляхом поєднання кількох векторів (дія конкатенації). Зрозуміло, що при створенні матриці обов’язковою є вимога однакової розмірності векторів.

Приклад.

```
-->A=[1 2];
-->B=[3 4];
-->C=[5 6];
-->V=[A B C] V =
1. 2. 3. 4. 5. 6.
-->M1=[A;B;C] M1 =
1. 2.
3. 4.
5. 6.
```

Для звернення до конкретного елемента матриці M використовується запис M(j,i), де M – ім’я матриці, j – номер рядка та i – номер стовпця:

```
-->M1(3,2)
ans =
6.
```

Як було зазначено раніше, масив може бути створений у вигляді числової послідовності, елементи якої змінюються з певним кроком від деякого початкового значення до деякого кінцевого за допомогою оператора «:». Взагалі цей оператор відіграє в системі важливу роль. Наприклад, він дає змогу одержати доступ до окремих блоків матриці – рядків, стовпчиків. Наступні приклади демонструють формування з матриці M векторів V1 і V2, перший з яких є другим рядком матриці M, а другий – першим стовпчиком матриці M.

```
-->M=[1 2 3; 4 5 6] M =
1. 2. 3.
```

4. 5. 6.
-->V1=M(2,:)
V1 =
4. 5. 6.
-->V2=M(:,1) V2 =
4.

Оператор «:» взагалі дає змогу виділити з матриці блок елементів, створивши таким чином нову матрицю. Наступний приклад демонструє створення нової матриці M1 з 1 і 2 рядків та 2 і 3 стовпчиків матриці M.

-->M1=M(1:2,2:3) M1 =
3. 5. 6.

Застосування комбінації з двох квадратних дужок «[]» дає змогу вилучати з масивів окремі елементи або їх блоки.
Наприклад:

-->M=[1 2 3; 4 5 6];
-->V1=M(2,:) V1 =
4. 5. 6.
-->M(2,:)=[] // Вилучення 2-го рядка матриці M M =
1. 2. 3.
-->V1(2)=[] // Вилучення 2-го елемента вектора V1 V1 =
4. 6.

Scilab має кілька функцій, за допомогою яких можна створити спеціальні матриці або перетворювати їх (таблиці 2.1).
Для усіх функцій аргументи *m,n* – це розмірність матриці.
Приклади перетворення матриці в інші за допомогою функції matrix:

-> M=[1 2; 0 3; 4 8] M =
1. 2.
0. 3.
4. 8.
--> matrix(M, 2, 3) ans =
1. 4. 3.
0. 2. 8.
--> V=matrix(M, 1, 6)
V = 1. 0. 4. 2. 3. 8.
--> V1=matrix(M, 6, 1) V1 =
1.
0.
4.

2.
3.
8.

Таблиця 2.1 – Функції для створення спеціальних матриць

Функція	Призначення
1	2
Matrix (M1 [,m,n])	Перетворення матриці M в матрицю з іншім розміром
cat(i,M1,M2)	Поєднання матриць M1 і M2. При цьому, якщо значення аргументу і дорівнює «1», то об'єднування здійснюється по рядках, а якщо «2», то – по стовпчиках. Дія функції тотожна дії оператора конкатенації M=[M1 M2].
eye(m,n)	Створення одиничної матриці (всі її елементи дорівнюють нулю, крім елементів головної діагоналі, значення яких дорівнює «1»).
ones(m,n)	Створення матриці, всі елементи якої дорівнюють «1». Для створення вектора одному з аргументів надати значення «1».
trial (M [,k])	Створення нижньої трикутної матриці, починаючи з k-ї діагоналі. За відсутності аргументу k або якщо k=0 матриця формується починаючи з головної діагоналі.
zeros(m,n)	Створення матриці, всі елементи якої дорівнюють «0».
sparse([i1 j1;i2 j2;…;in jn],[n1,n2,…,nn])	Створення розрідженої матриці. Аргументами функції є індекси не нульових елементів (i1 j1) і їх значення (n1).
full(M)	Відображення розрідженої матриці
rand(m,n, «тип розподілу»)	Створення матриці, всі елементи якої формуються генератором випадкових чисел.
grand(m,n,)	Створення матриці, всі елементи якої формуються генератором випадкових чисел з можливістю вибору розподілу.
diag(V)	Створення діагональної матриці, елементами головної діагоналі якої є значення вектора V; всі інші її елементи дорівнюють «0».
sort(M)	Впорядкування вектору. Якщо аргумент є матрицею, то сортування здійснюється по стовпцям.

Наведені приклади у тому числі демонструють як можна перетворити матрицю на вектор.

Створення розрідженої матриці

```
--> M=sparse([1 2; 2 3; 3 4], [3 6 9]) M =
(  3,  4) sparse matrix
```

```
(  1,  2) 3.
(  2,  3) 6.
(  3,  4) 9.
--> full(M)
 ans =
0.    3.   0. 0.
0.    0.   6. 0.
0.    0.   0. 9.
```

Приклади генерації випадкових чисел

Неперервний рівномірний випадковий розподіл на інтервалі [0,1]:

```
--> grand(4,4,"def")
ans  =
0.8147237   0.1269868   0.6323592    0.2784982
0.135477    0.9688678   0.3081671    0.188382
0.9057919   0.9133759   0.0975404    0.5468815
0.8350086   0.221034    0.5472206    0.9928813
```

Неперервний рівномірний випадковий розподіл на інтервалі [low,high) (high до інтервалу не включається)

```
--> grand(4,4,"uin",0,10)
 ans=
2.   6.   6.   1.
3.   5.   3.   6.
2.   7.   1.   6.
7.   8.   7.   0.
```

Нормальний розподіл. Параметри функції: (m, n, «nor», Av, Std), де Av – середнє значення, Std – середнє квадратичне відхилення)

```
--> grand(4,4,"nor",0,1)
ans  =
 0.4388861   -1.0682781   1.0433       -0.540479
 -1.0288046  0.0388806    -0.4565521   -0.827319
 0.1979328   0.1788246    -1.2085547   1.5340618
 0.3833912   -0.7169367   1.5310142    2.6169952
```

Операції над матрицями.

У системі для роботи з масивами виконуються різноманітні операції. Основні такі операції зведені у наступній таблиці 2.2.

Таблиця 2.2 – Операції з матрицями

	Операція	Примітка
	1	2
+	Додавання	Масиви мають бути однакової розмірності.
-	Віднімання	
*	Множення масивів	Масиви мають бути однакової розмірності. При цьому кількість рядків у першому масиві повинна дорівнювати кількості стовпчиків у другому, а кількість стовпчиків у першому – кількості рядків у другому. Отже, у разі виконання множення для векторів, один із них повинен бути вектором- стовпчиком, а другий – вектором-рядком.
*	Множення на скаляр	
^	Піднесення до ступеня	Операція еквівалента множенню матриці саму на себе. Якщо при цьому показник степені менше «1», то матриця множиться на матрицю, обернену до себе.
‘	Транспонування	
\	Ліве ділення	(A \ B) $\Rightarrow$ (A^{-1} · B) . Операція може бути використана для розв’язання рівняння $A \cdot X = B$, де X – вектор з невідомими значеннями.
.^	По елементне піднесення до ступеня	
.\	Поелементне ділення матриць справа наліво	Масиви мають бути однакової розмірності.
./	Поелементне ділення матриць зліва направо	Масиви мають бути однакової розмірності.
.^	Поелементне піднесення до степеня	Масиви мають бути однакової розмірності.

Приклади операцій:

```
-->V=[1; 2]; V1=[3 4];
-->V*V1
ans =
3. 4.
6. 8.
-->A=[1 2; 3 4]; B=[3 4; 5 6];
-->X=A\B // Розв’язування матричного рівняння AX=B
X =
- 1. – 2.
```

```
2. 3.
-->X=B/A // Розв’язування матричного рівняння XA=B
X =
0. 1.
- 1. 2.
-->X*A-B // Перевірка ans =
0. 0.
0. 0.
```

В усіх розглянутих прикладах елементами матриць є числа, але система дає змогу створювати і працювати з матрицями, елементами яких є рядки символів. Наприклад, з такими матрицями можна додавати, транспонувати.

Визначити, які елементи більше заданого числа:

```
--> a=[4,2,5,1]
a  =
4.  2.  5.  1.
--> k=[3]
k  = 3.
--> disp(a>3)
T F T F
--> disp(a>k)
T F T F
```

Списки

Список містить елементи, які є довільними об’єктами системи. Для створення списку використовується команда list:

```
--> l=list(12,["ab", "cd"])
l  =
        l(1)
        12.
        l(2)
        !ab  cd  !
```

Для роботи зі списками використовують спеціальні операції: вставка на певне місце, додавання на початок і в кінець списку, вилучення, конкатенація і т. ін.

Крім «звичайних» списків бувають також типізовані і орієнтовані на матрицю списки.

Завдання

Виконання практичної роботи здійснюється за допомогою редактора сценаріїв.

1. Створіть вектори-рядки з 5 елементів кожний:

– V1 – шляхом надання елементам вектора значень 1, 2, 3, 4, 5;

– V2 – з вектора V1 шляхом додавання до кожного його елемента числа «5*N, де N – номер по журналу»;

– V3 – з початковим значенням «індивідуальний номер» і кроком «2,5».

2. Введіть змінну x і надайте їй значення «індивідуальний номер».

3. Виконайте операцію множення вектора V1 на x.

4. Виконайте операцію додавання векторів V1 і V2.

5. Створіть вектор V4 шляхом транспонування вектора V1.

6. Виконайте операцію множення векторів V1 і V4.

7. Виконайте такі дії з елементами вектора V1:

– $V_1 \ - V_3$;

– $V_2 \ * V_4$;

– $V_3 \ / V_1$;

– $\sqrt{V_2}$;

– V_3^2.

8. Створіть дві матриці M1 і M2 з двох рядків і п’яти стовпчиків кожна. Значення матриць можуть бути будь- якими, окрім нульових.

9. Здійсніть з цими матрицями наступні дії:

– поелементне множення;

– поелементне ділення справа наліво;

– поелементне ділення зліва направо.

10. Створіть матрицю M3 шляхом транспонування матриці M1.

11. Створіть матрицю M4 з векторів V1 і V2.

12. Виконайте операцію множення вектора M1 на скаляр x.

13. Виконайте множення матриць M1 і M3.

14. Виконайте додавання матриць M1 і M2.

15. Виконайте дії з елементами матриць:

$M1_{1,1} + M2_{2,1}$; $M2_{1,1} + M3_{1,2}$; $\sqrt{\mathrm{M3}_{1,2}}$; $\mathrm{M1}_{2,1}$

16. З другого рядка матриці M1 створіть вектор-рядок V5.

17. Створіть матриці M5 і M51 шляхом поєднання матриць M1 і M2 по рядках і по стовпчиках.

18. За допомогою відповідної функції створіть одиничну матрицю M6 розмірністю 5*5.

19. За допомогою відповідної функції створіть матрицю M7 розмірністю 5*5, всі елементи якої дорівнюють «0».

20. За допомогою відповідної функції створіть матрицю M8 розмірністю 5*5, всі елементи якої дорівнюють «1».

21. Створіть вектор-рядок V5 з п’яти елементів, елементи якого сформуйте генератором випадкових чисел.

22. Створіть діагональну матрицю M8, елементами головної діагоналі якої є значення вектора V1.

23. Запам’ятайте змінні та їх значення з робочої області пам’яті поточної сесії.

Контрольні питання

1. Що таке «транспонування»?

2. Які є варіанти створення вектора?

3. Які є варіанти створення матриці?

4. Як створити з матриці вектор-рядок, вектор- стовпчик?

5. Як виконати з матрицями операції по елементного множення, по елементного ділення справа наліво та зліва направо.

6. Що являє собою операція по елементного множення?

7. Що являє собою операція по елементного ділення?

8. За допомогою якої функції створюються одиничні матриці?

9. За допомогою якої функції здійснюється поєднання матриць?

10. За допомогою якої функції створюється матриця, всі елементи якої дорівнюють «0», «1»?

11. Як створити вектор або матрицю, елементи якої повинні бути сформовані генератором випадкових чисел?

12. Як створити діагональну матрицю, елементами головної діагоналі якої є значення вектора V, а всі інші її елементи дорівнюють «0»?

13. Як створити вектор у вигляді послідовності цілих чисел з довільним кроком зміни?

Практична робота №3

Графічна візуалізація даних у системі Scilab

Мета – засвоєння методів побудови графіків функцій і візуалізації даних.

Завдання: побудувати графіки елементарних функцій. Виконати візуалізацію даних у 2D та 3D. Налаштувати осі, підписи, легенду, сітку. Дослідити можливості багатопанельних графіків.

Література: [2, 4, 5, 7].

Теоретичні відомості

Побудова графіків.

Побудова 2-D графіка.

Для побудови двовимірного графіка використовується функція plot2d, яка за

найпростішим варіантом має синтаксис:

plot2d([x,]y),

де x – вектор точок (координат), y є вектором значень функції від параметра x.

Зрозуміло, що вектори повинні бути однакової розмірності. За відсутності першого аргументу координати осі абсцис (x) визначаються автоматично.

Функцію можна визначити різними шляхами:

1. Описати її перед застосуванням plot2d.

x=[0:0.01:2*%pi]’; y=sin(exp(x)); plot2d(y)

2. Описати безпосередньо в plot2d.

x=[0:0.01:2*%pi]’; plot2d(x, sin(exp(x)))

Зверніть увагу, що x перетворюється у вектор- стовпчик.

На жаль, система спроможна побудувати не усі графіки. Наприклад, авторові так й не вдалося побудувати графік $y = x * \sin(1/x)$.

Разом із функцією plot2d для побудови двовимірних графіків можна використовувати функцію plot, яка є повним аналогом функції plot2d і призначена для сумісності із системою MATLAB.

Додаткові аргументи функції дозволяють визначити стиль графіка, встановити межі графіка і т. ін.

plot2d ([x],y,[, додаткові аргументи для графіків]).

Наприклад визначення кольору лінії здійснюється так:

```
x=[0:0.1:2*%pi]';
plot2d(x, sin(x), style=color("green"))
```

З прикладу зрозуміло, що значення кольору задається англійською назвою аргументу color (рис.3.1.).

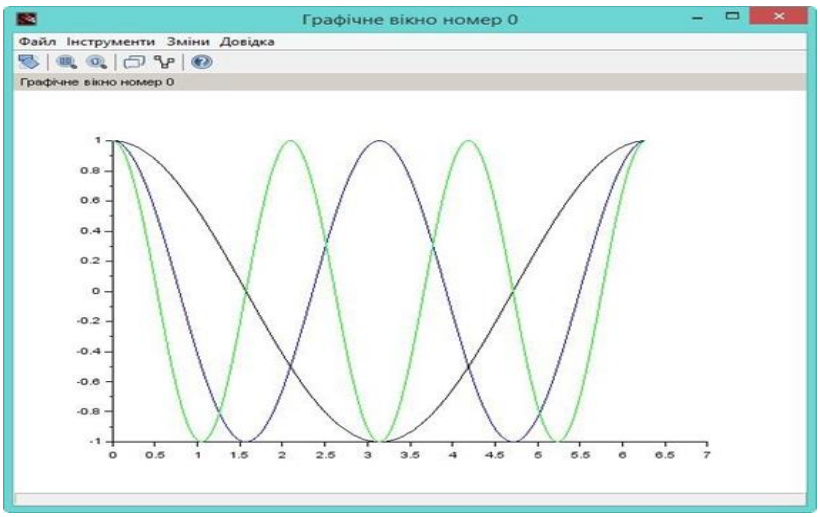


Рисунок 3.1 – Кольорові графіки

За допомогою функції в одному графічному вікні можна побудувати графіки кількох функцій. У цьому випадку перелік функцій подається у вигляді списку, тобто у квадратних дужках, а самі аналітичні вирази функцій відокремлюються проміжком:

plot2d(x,[cos(x) cos(2*x) cos(3*x)]).

Для побудови графіка у полярних координатах застосовується функція polarplot. Синтаксис функції:

polarplot (діапазон значень кута, функція, за якої будується полярній графік [, додаткові аргументи для графіків])

Зрозуміло, що і перший, і другий аргументи є обов'язковими.

Приклад. Побудувати у полярних координатах графік функції $4\cos(4\varphi)$, де φ змінюється на інтервалі $[0; 2\pi]$ з кроком «0,01» (рис.3.2).

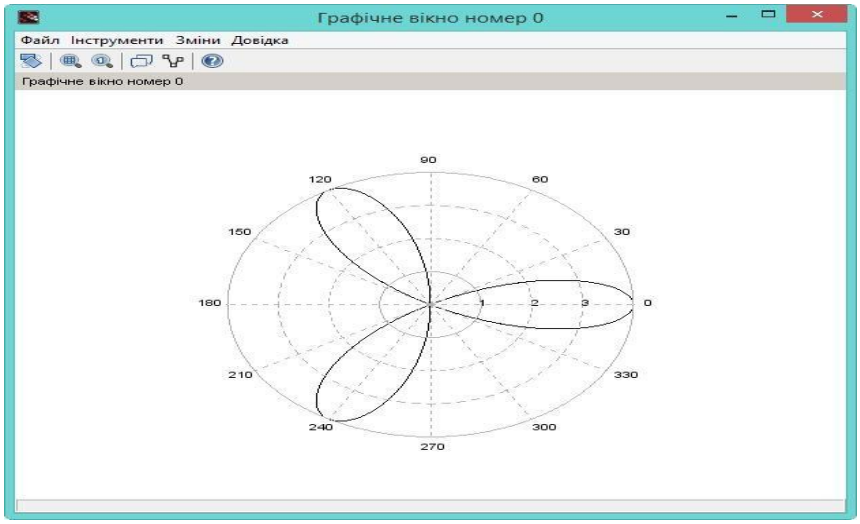


Рисунок 3.2 – Графік у полярних координатах

Розв'язок матиме вигляд:

```
--> fi=[0:0.01:2*%pi]
--> polarplot(fi, 4*cos(3*fi))
```

Або:

```
--> fi=[0:0.01:2*%pi]
--> f=4*cos(3*fi)
--> polarplot(fi, f)
```

Так саме як і для графіків у декартовій системі координат, при побудові графіка у полярній системі для нього можна задати колір лінії:

```
-->fi=[0:0.01:2*%pi];
-->polarplot(fi,f, style=color("red"))
```

Так саме як і для графіків у декартовій системі координат, в одному графічному вікні можна побудувати кілька графіків, але синтаксис при цьому інший: polarplot(функція_1, функція_2,…)

Як бачимо, при побудові графіків у полярній системі координат аргументами є перелік функцій, які відокремлюються комами.

Приклад:

```
polarplot(4*sin(fi), cos(8*fi))
```

Графічне вікно.

Графік відображається у спеціальному графічному вікні, ім’я якого складається з тексту «Графічне вікно номер» і порядкового номера вікна. Наприклад, для першого вікна ім’я буде «Графічне вікно номер 0». Це вікно стає активним і саме в ньому будуються усі інші графіки (рис.3.3).

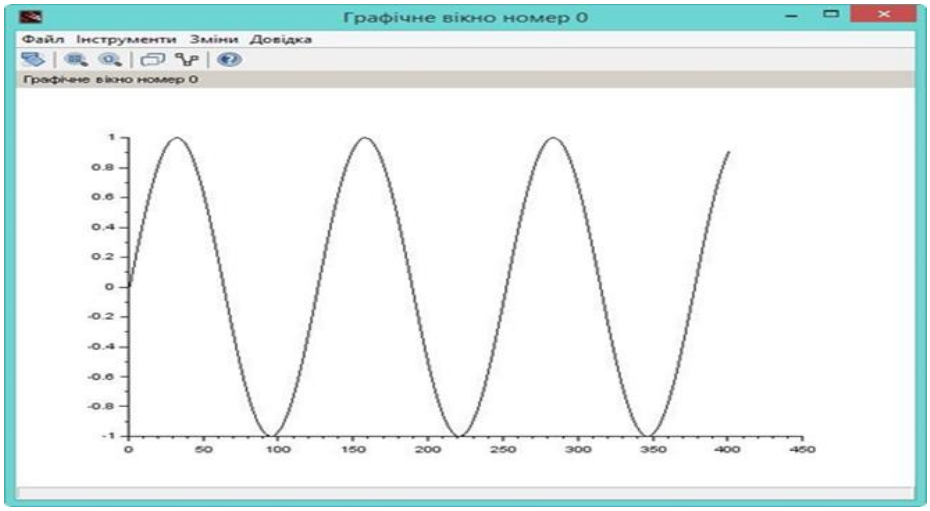


Рисунок 3.3 – Графік

Отже, якщо залишати це вікно відкритим, то це дозволяє будувати графіки кількох функцій в одному графічному вікні.

Для виведення графіка в іншому графічному вікні слід створити нове графічне вікно. Для цього в активному графічному вікні слід виконати команду Файл ▸ Створити рисунок. Після цього вже це вікно стає активним.

Але такий варіант є не дуже зручним і його можна використовувати під час роботи «в ручному режимі» з документом. Значно ефективнішим варіантом створення нового графічного вікна є застосування функції *scf*, яка має такий синтаксис:

```
scf([номер_графічного_вікна])[;]
```

Аргумент «номер_графічного_вікна» є не обов’язковим. За його відсутністю система надає йому значення (порядковий номер) першого вільного вікна. Функція повертає достатньо велику кількість параметрів вікна, тому за відсутності одержання якогось параметру доцільно закінчувати функцію символом «;».

Такий варіант створення графічного вікна має ще одне відчутне достоїнство: він дозволяє відкрити і не закривати в сесії кілька графічних вікон.

Scilab має також і функцію clf для повного очищення графічного вікна, що має синтаксис, аналогічний із функцією scf.

clf([номер_графічного_вікна]).

Головне меню вікна містить три пункти:

1. Файл. Серед команд цього пункту відмітимо такі:

1.1. Експортувати. Дозволяє зберегти графік у кількох графічних форматах, у тому числі таких поширених як GIF, BMP, а також PDF.

1.2. Копіювати до буфера. Дозволяє скопіювати графік до буферу обміну.

2. Інструменти.

2.1. Збільшити. Збільшення розміру ділянки графіка. Після вибору цього інструмента з'являється рамка, за допомогою якої обирається область для збільшення.

2.2. Обертання на площині/у просторі. Для цього потрібно виконати команду Інструменти ▸ Обертання на площині/у просторі або на панелі інструментів натиснути кнопку цього інструмента  , після чого встановити курсор на графік, натиснути праву кнопку миші і, не відпускаючи її, виконати обертання графіка.

3. Зміни. Містить кілька дій, основними з яких є редагування загальних властивостей графіка або властивостей осей графіка.

Під рядком команд розташована панель стандартних інструментів:

1. Обертання графіку.
2. Збільшення ділянки.
3. Повернення до початкових розмірів.
4. Підбір за вмістом.
5. Увімкнути або вимкнути режим підписів до даних.
6. Зміна режиму внесення змін до даних кривої.
7. Допомога.

Графічний редактор (GED).

Звернутися до графічного редактора можна за командою Зміни ▸ Властивості рисунка. Після цього з'являється вікно «Fugure Editor» (редактор графіка), у якому можна змінити параметри графіка, що знаходиться у графічному вікні (рис.3.4).

Вікно поділено на дві частини: ліва містить перелік об'єктів («Objects Browser»), а права («Object Properties») – відображає властивості об'єкта після виділення його в лівій частині.

Інформація у лівій частині вікна структурована таким чином:

1. Figure. Загальні властивості графіка. Наприклад, тут змінюються координати осей, колір фону.

2. Axes. Властивості осей графіка, при цьому для кожної осі відводиться своя

вкладка. Користувач може змінити місце розташування масштабної шкали, вивести допоміжну сітку, змінити її колір і т. ін.

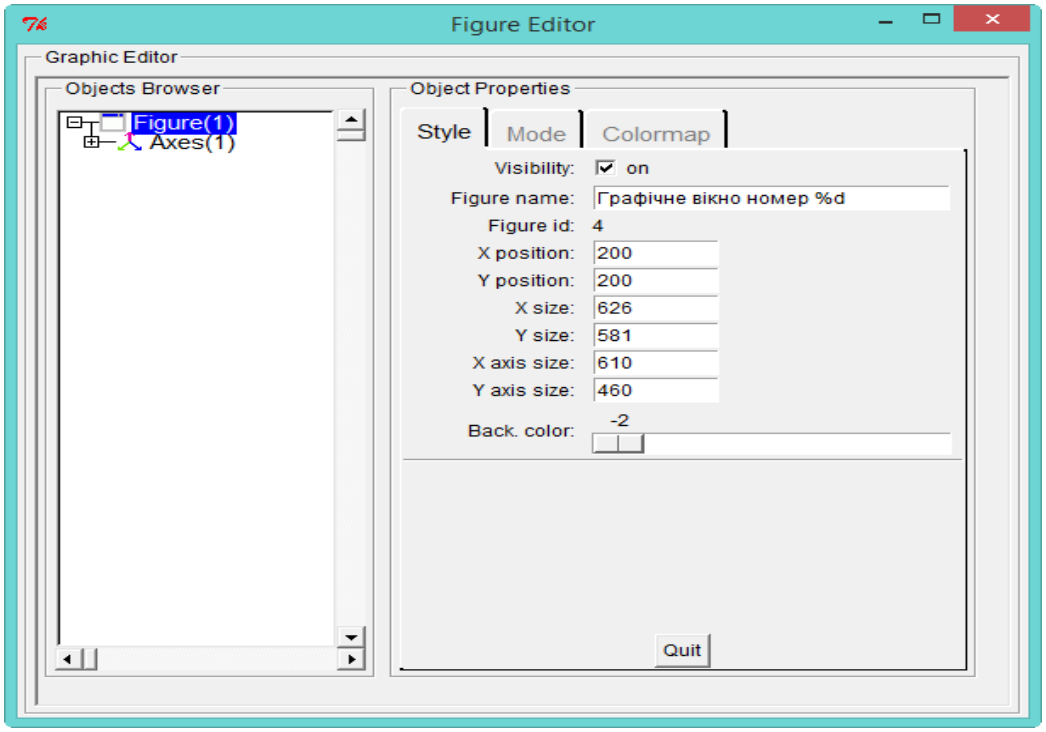


Рисунок 3.4 – Графічний редактор

3. Compound. Загальні властивості всіх графіків. Таких можливостей не багато: практично тут можна тільки тимчасово приховати самі графіки, знявши прапорець біля дії «Visibility: on».

4. Polyline. Властивості окремого графіка. Тут можна змінити колір лінії, її форму, товщину та багато іншого.

Інший варіант звернення до графічного редактора полягає у застосуванні команди Зміни ▸ Властивості осей. Знов-таки це призводить до відображення вікна «Axes Editor» (редактор осей), але при цьому відразу відкривається доступ до властивостей осей графіка.

Побудова 3-D графіка.

Для побудови тривимірного графіка використовується функція plot3d, яка за найпростішім варіантом має синтаксис:

plot3d(x',y,z),

де x і y – вектори однакової розмірності n, z – матриця розмірністю n*n, елементи якої розраховуються за значеннями x і y.

Така само, як і двовимірний графік, побудований 3-D графік можна обертати.

Зміна параметрів графіка із середовища Scilab.

За замовчуванням графік будується з конкретними параметрами, до яких належать шрифти, колір, розміри і т. ін. Для зміни цих параметрів після побудови графіка використовується функція xset. Синтаксис функції: xset(ім'я параметра[, арг_1, арг_2 арг_3, арг_4, арг_5])

де арг_1–арг_5 – додаткові аргументи, що конкретизують 1-й параметр; їх кількість і значення залежить від значення першого аргументу. Аргумент «ім'я параметра» може набувати досить багато значень, наприклад:

–default – відновлення параметрів за замовчуванням;

–«foreground», color – встановлення основного кольору;

–«background», color – встановлення кольору фона;

–«color», value – встановлення основного кольору;

–«font», fontid, fontsize – встановлення шрифту;

–«font size», fontsize – встановлення розміру шрифту;

–«line style», value – встановлення стиля шрифту і т. ін.

З повним переліком аргументів можна ознайомиться у довідковій системі. Конкретні значення того чи іншого аргументу можна також визначити з довідкової системи.

Наприклад, значення аргументу «color» можуть бути: 0 або 1 – чорний колір, 2 – синій, 3 – зелений, 4 – яскраво блакитний, 5 – червоний і т. ін. Наприклад, встановлення зеленого кольору для основного кольору відбувається за допомогою такого введення:

color=3; xset(«foreground», color)

Побудова графіку з вікна перегляду змінних.

Система взагалі дозволяє користувачеві дуже просто побудувати кілька поширених типів графіку для змінної (рис.3.5). Для цього у вікні перегляду змінних слід викликати контекстне меню на рядку з іменем змінної, після чого з'явиться меню з переліком видів графіків, з якого вибираємо потрібний.

Вказівки до виконання.

1. Виконання практичної роботи здійснюється за допомогою редактора сценаріїв.

2. Пункти 1, 7, 8 виконуються за варіантами.

3. Кожний графік будується в окремому вікні. Нумерацію вікон починайте з «0».

4. Враховуючи велику розмірність векторів, виведення результатів не здійснюйте.

5. При побудові функцій для виконання дій над елементами матриць застосовуйте операції поелементного множення або ділення матриць зліва направо.

6. У вікні графічного редактора об'єкт «Figure» надає доступ до загальних властивостей графіка, «Axes» – властивостей осей графіка, «Compound» – загальних властивостей усіх графіків, «Polyline» – до властивостей окремого графіка.

7. Під час побудови графіка в полярній системі координат для завдання кольору осі використовуйте додатковий аргумент color, значення англійською мовою якого і визначає колір лінії, наприклад color(«red»).

8. За закінченням редагування у вікні графічного редактора натисніть кнопку «Quit».

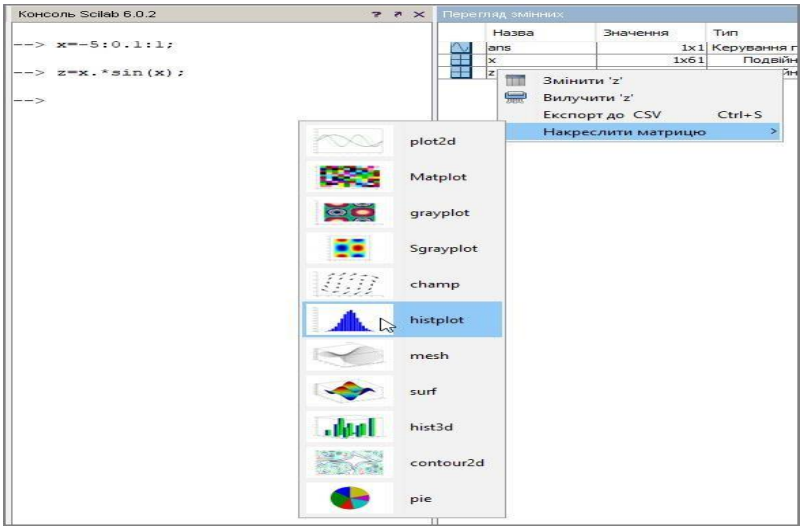


Рисунок 3.5 – Графік для змінної

Наприклад, вибираємо гістограму (histplot) (рис.3.6):

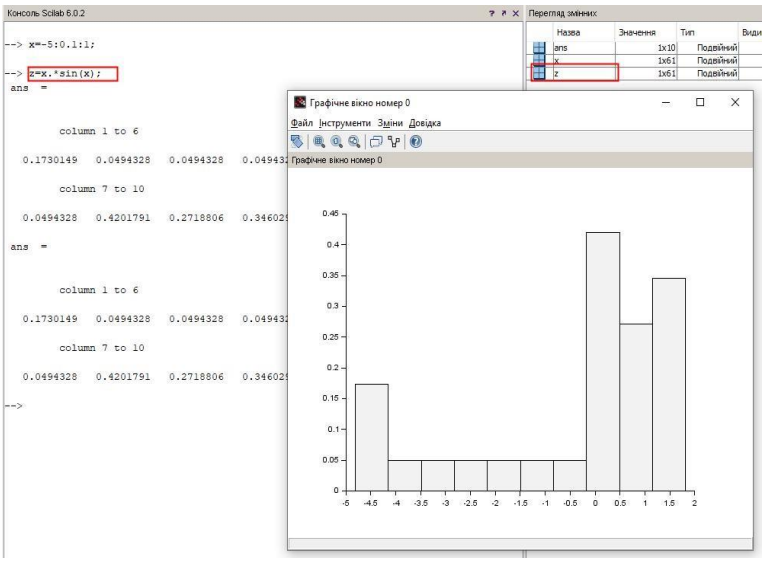


Рисунок 3.6 – Графік для змінної

Завдання

1. Побудуйте графіки наступних функцій. Для першого графіка визначити функцію графіка перед застосуванням plot2d, а для другого – опишіть її безпосередньо в plot2d. Кожний графік повинен мати унікальний колір і будуватися у власному вікні, тобто усі графічні вікна не закриваються.

$$y(x) = \frac{a^3}{(a^2 + x^2)}; \qquad f(x) = x \cdot (1 - \sin(x))$$

$$y(x) = \frac{x^3}{1000}; \qquad f(x) = x \cdot (2 - \cos(2x))$$

де a – константа, значення якої відповідає Вашому індивідуальному номеру, x змінюється від 0 до 20 з кроком 0,05.

2. Здійсніть повне очищення відкритого графічного вікна, а потім відновіть у ньому побудовані у попередньому пункті графіки.

3. Перегляньте панель інструментів графічного вікна та визначте, які інструменти вона містить.

4. Застосуйте відповідний інструмент для збільшення ділянки графіка та повернення до попереднього розміру.

5. Створіть вектор *z* як послідовність зі ста чисел за такими даними:

5.1 значення першого елемента дорівнює «індивідуальному номеру»;

5.2 крок приросту «0,1».

6. За допомогою тільки однієї функції plot2d побудуйте в одному графічному вікні графіки функцій sin(z) і cos(z).

7. Побудуйте в полярній системі координат в одному графічному вікні:

7.1 Три графіка функції $4\cos(3\varphi)$, де φ змінюється на інтервалі [0; 2π]: з кроком «0,5», колір лінії – чорний; з кроком «0,25», колір лінії – зелений; з кроком «0,01», колір лінії – червоний;

7.2 Три графіка функції $4\sin(3\varphi)$, де φ змінюється на інтервалі [0; 2π]: з кроком «0,5», колір лінії – чорний; з кроком «0,25», колір лінії – зелений; з кроком «0,01», колір лінії – червоний.

Поясніть відмінність між графіками.

8. Створіть вектор x2, що змінюється на інтервалі $[-4\cdot\pi; 4\cdot\pi]$ з кроком 0,01. Побудуйте графік параметричної функції:

$$f(x) = \frac{x^2}{\cos(x2)\pi};$$

$$f(x) = x2 \cdot \pi \cdot \sin(2\pi).$$

9. За допомогою графічного редактора виконайте для останнього графіка такі дії:

−змініть основний колір лінії графіка на червоний (властивість «foreground»);

−збільшить його розмір на всю ширину документа (властивість «X size»);

−встановіть лінії масштабної сітки для осі x (властивість «Grid color»);

−створіть загальний підпис до графіка (Axes, вкладка «Title», поле «Text»), Підпис вводиться у подвійних лапках;

−Збережіть графік у форматі JPEG.

10. Створіть вектор x1, що змінюється на інтервалі [0; 20] з кроком 1. Побудуйте матрицю ординат 3D-поверхні за формулою:

$$M = \left\{ \frac{(X1-10)}{5} \cdot \frac{(X1-10)}{5} \right\}.$$

11. За допомогою інструмента обертання надайте графіку наступний вигляд (рис.3.7):

12. Створіть для графіка підпис: «Приклад графіка функції в тривимірній декартовій системі координат».

13. Збережіть графік у форматах PDF, GIF і BMP.

14. Побудуйте для матриці всі види 3D-графіків.

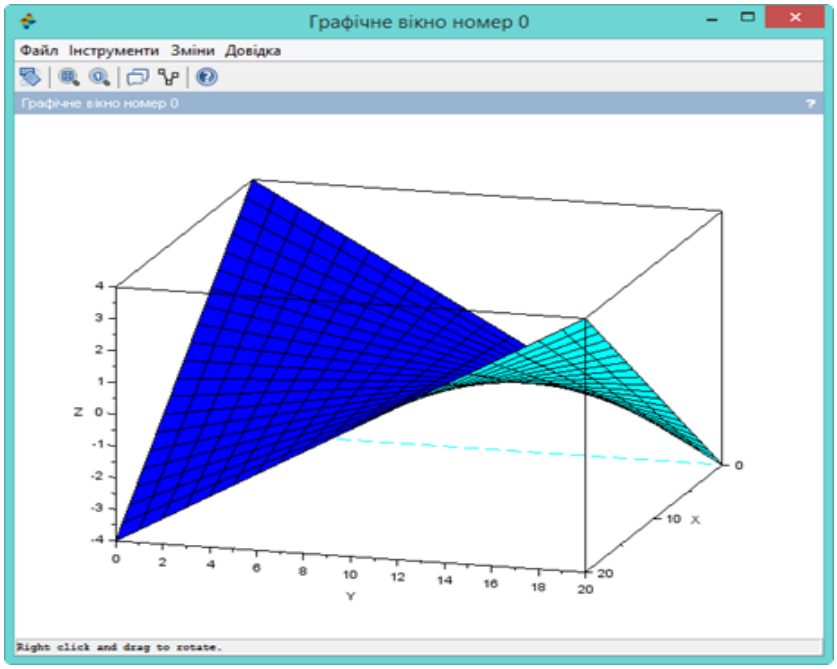


Рисунок 3.7 – Графік функції в тривимірній декартовій системі координат

Контрольні питання

1. Що таке «2-D графік»?
2. Що потрібно зробити для того, щоб результат виконання математичного виразу не був відображений на екрані?
3. Яка функція призначена для побудови 2-D графіків? Які аргументи має ця функція?
4. Яка функція призначена для створення нового графічного вікна?
5. Яка функція здійснює повне очищення відкритого графічного вікна?
6. Які є варіанти створення нового графічного вікна?
7. Яка структура графічного вікна?
8. Яка функція призначена для побудови 3-D графіків? Які аргументи вона має?
9. Яка функція призначена для побудови графіка у полярній системі координат? Які аргументи вона має?
10. Які інструменти містить панель інструментів графічного вікна?
11. Як змінити основний колір лінії графіка?
12. Як збільшити ділянку графіка та повернутися до попереднього розміру?
13. Як встановити лінії масштабної сітки для осі?
14. Як створити для графіка загальний підпис?
15. У яких популярних форматах можна зберегти графік?

Практична робота №4 – 5

Математичний аналіз. Розв'язок диференційних рівнянь в системі Scilab для динамічних систем.

Мета – навчитися застосовувати оператори математичного аналізу Scilab. Набуття умінь та навичок розв'язку рівнянь.

Завдання: Розв'язати систему рівнянь, яка описує динамічні системи з початковими умовами.

Література: [2, 4, 5].

Теоретичні відомості

Розв'язування завдань лінійної алгебри. Функції знаходження числових характеристик матриці.

Для визначення кількості рядків і стовпчиків матриці *M* використовують функцію size(M), аргументом якої є ім'я масиву:

```
-->M=[1 2; 3 4; 5 6]; size(M)
 ans = 2.
```

Якщо потрібно визначити тільки кількість рядків або тільки кількість стовпчиків матриці, то синтаксис функції модифікується: додається другий аргумент, який має значення «1» або «r», якщо слід визначити кількість рядків, «2» або «c» – якщо слід визначити стовпчиків. Наприклад:

```
-->M=[1 2; 3 4; 5 6]; size(M,1)
 ans =
      3.
```

Загальну кількість елементів матриці або довжину вектора обчислює функція length(M), аргументом якої є ім'я матриці або вектора:

```
-->M=[1 2; 3 4; 5 6]; length(M)
 ans =
      6.
-->V=[1 2 3 4]; length(V)
 ans =
       3.
```

Для визначення максимального за значенням елемента матриці або вектора використовується функція max(A), аргументом якої є ім'я матриці або вектора. Якщо потрібно визначити максимальне значення для кожного з рядків або стовпчиків матриці, то синтаксис функції модифікується: додається другий

аргумент, який має значення «r», якщо слід визначити максимальне значення для кожного рядка, або «c» – якщо для кожного стовпчика:

```
-->M=[1 2; 3 4; 5 6];
 -->max(M) ans =
     6.
 -->max(M,'c') ans =
     2.
     4.
     6.
```

Аналогічно застосовується функція min(A), яка призначена для визначення мінімального за значенням елемента матриці або вектора.

Для обчислення визначника (детермінанту) квадратної матриці використовується функція det(M), аргументом якої є ім’я матриці:

```
-->M=[1 2;3 4]
 M =
     1. 2.
     3. 4.
 -->det(M) ans =
       – 2.
```

Для обчислення рангу матриці використовується функція rank(M), аргументом якої є ім’я матриці:

```
-->M=[1 2; 3 4];
 -->rank(M) ans =
     2.
```

Функції, що реалізують чисельні алгоритми розв’язування задач лінійної алгебри.

Обчислення матриці, оберненої до M. Нагадаємо, що оберненою по відношенню до матриці M називається така матриця, яка при її множенні на матрицю M дає одиничну матрицю.

Для обчислення оберненої матриці використовується функція inv(M), аргументом якої є ім’я матриці, для якої знаходиться обернена матриця.

Приклад.

```
 -->M=[1 2; 3 4] M =
     1. 2.
     3. 4.
 -->A=inv(M) A =
     - 2. 1.
```

```
      1.5 – 0.5
-->M*A

  ans =
      1. 0.
      4.441D-16 1.
-->A*M
  ans =
      1. 0.
      1.110D-16 1.
```

Як бачимо, одержана обернена матриця А достатньо близька до одиничної, але все ж таки повністю не є одиничною, що є наслідком похибки чисельних обчислень.

Для створення полінома використовується функція poly.

Синтаксис функції:

[p]=poly(a, «x», [«flag»]),

де p – це ім'я полінома (його можна і не задавати). Аргументи функції: a – матриця або дійсне число, x –символьна змінна, «flag» – символьна змінна, яка визначає спосіб завдання полінома і може набувати значення «roots» (скорочення «r») або «coeff» (c).

За замовчуванням – «roots». Якщо «flag» має значення «r», то поліном створюється з параметрами a_i для відповідних змінних x_i . Якщо «flag» має значення «c», то значення параметрів a_i сприймаються як корені, для яких потрібно розрахувати коефіцієнти полінома.

Функція повертає вектор коефіцієнтів полінома.

Наступний приклад демонструє використання функції poly для створення поліномів p1, який має корінь що дорівнює «2» і p2 з коефіцієнтом «2»:

```
-->p1=poly(2,"x","r") p1 =
- 2 + x
-->p2=poly(2,"x","c") p2 =
2
```

У наступних прикладах створюються поліноми з відповідними коефіцієнтами для кубічного рівняння.

```
-->poly([1 2 3 4],"x","c")
ans =
1+ 2x + 3x^2 + 4x^3
-->poly([1 0 0 4],"x","c")
```

ans =

$1+ 4x^3$

З поліномами можна виконувати дії множення, ділення (створювати з них дроби), додавати та віднімати.

Розв'язування рівняння з одним невідомим.

Scilab може розв'язувати алгебраїчне рівняння з одним невідомим. Наприклад, потрібно знайти корені для рівняння $x^2 = 1$. Якщо – як у цьому прикладі – рівняння не представлено у вигляді поліному, то його попередньо слід перетворити в поліном: $x^2 - 1 = 0$. Після цього застосовується функція roots, єдиним аргументом якої є ім'я полінома. Функція повертає знайдені корені полінома.

Операції математичного аналізу.

Обчислення сум елементів матриці.

Для обчислення суми значень елементів матриці призначена функція sum, яка за найпростішим варіантом має синтаксис:

sum(ім'я масива), де ім'я масиву – вектор або матриця.

Якщо слід обчислити окремо суму значень кожного стовпчика матриці, то другий аргумент функції є числом «1», а якщо для рядка – «2».

Обчислення добутків елементів матриці.

Для обчислення добутків використовується функція prod, яка за найпростішим варіантом має синтаксис:

prod(ім'я масива), де ім'я масиву – вектор або матриця.

Якщо слід обчислити окремо добуток значень кожного стовпчика, то другий аргумент функції є числом «1», а якщо для рядка – «2».

Інтеграли.

Для знаходження визначених інтегралів в системі використовується функція intg. У найпростішому вигляді функція має такий синтаксис:

Intg (нижня межа інтегрування, верхня межа інтегрування, f),
де f – це підінтегральний вираз у вигляді текстового рядка або функції користувача.

Функція intg у відповідь повертає значення визначеного інтегралу:

-->intg(1,5,f) ans =

- 5.8927325

Результат функції можна записати у змінну:

-->k=intg(1,5,f)

Диференціали. Для знаходження диференціалів у певній точці використовується функція numderivative, яка за найпростішим варіантом має такий

синтаксис:

numderivative(f, coord)

де f – це є ім’я функції користувача, що диференціюється, а coord – координата точки, для якої потрібно обчислити похідну.

Координати точки можуть бути задані ідентифікатором (як скаляром, так і вектором) або безпосередньо числом. Наприклад:

```
-->function y=f(x), y=x^2+1/2, endfunction
-->numderivative(f,1) ans =
2.
```

Scilab чисельна система, але вона дозволяє виконати деякі символьні обчислення. Наприклад система має функцію devart, що обчислює в аналітичному вигляді похідну полінома.

```
--> p1=poly([3 -5 1], "y", "c")
p1 =
2
3 -5y +y
--> derivat(p1) ans =
-5 +2y
```

Розв’язування диференційних рівнянь. Для розв’язування диференційних рівнянь або систем диференційних рівнянь застосовується функція ode, яка за найпростішим варіантом має синтаксис:

ode(y_0,x_0,C,f)

де y_0 – початкова умова: дійсне число для одного диференційного рівняння для або вектор для системи диференційних рівнянь,

x_0 – початкове значення інтервалу інтегрування: дійсне число для одного диференційного рівняння для або вектор для системи диференційних рівнянь,

C – координати осі x: початкова_координата: крок: кінцева_координата (використовується зокрема під час побудови графіка функції),

f – функція користувача – права частина рівняння або системи рівнянь.

Результатом роботи функції є множина одержаних чисельних значень розв’язку.

Приклад. Знайти загальний розв’язок звичайного диференційного рівняння першого порядку $y' - 10x = 0$ на інтервалі [2,10] з початковою умовою $y_0 = -1$.

У першому рядку формується користувацька функція, на яку у функції ode здійснюється посилання. Оскільки вона має являти собою праву частину диференційного рівняння, то початкове рівняння перетворюється у вигляд $y' = 10x$.

У другому рядку задається початкова умова $y_0 = -1$, а у третьому – інтервал зміни незалежної змінної [2,10].

Розв’язування буде складатися з такої послідовності:

```
-->function yd=f(x,y), yd=10*x, endfunction;

-->y0=0; x0=-1;

-->x=2:1:10;

-->y=ode(y0,x0,x,f)
```

Одержимо графічний розв’язок диференційного рівняння (рис.4.1). Для цього застосуємо функцію plot2d (x,y).

Як бачимо, координатна сітка для осі x як раз і відбиває інтервал її зміни. Слід мати на увазі, що початкове значення x_0 має бути більшим за початкове значення координати осі x. У протилежному випадку система генерує помилку.

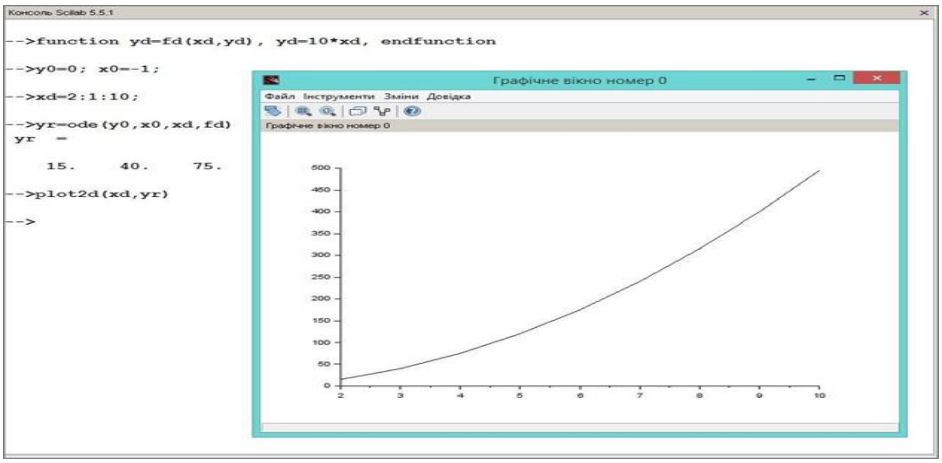


Рисунок 4.1 – Графік функції в тривимірній декартовій системі координат

Завдання

1. Створіть матрицю M1 розмірністю 3*3 шляхом надання її елементам довільних значень, відмінних від нульових.
2. За допомогою відповідних функцій знайдіть для матриці M1 такі її числові характеристики:
   - кількість рядків, стовпчиків;
   - кількість елементів;
   - максимальний, мінімальний за значенням елемент;
   - максимальний за значенням елемент 1-го рядка;
   - максимальний за значенням елемент 2-го стовпчика;
   - детермінант;
   - ранг.
3. Створіть вектор-рядок V1 з п’яти довільних елементів, значення яких відмінні від нуля.
4. За допомогою відповідних функцій знайдіть для нього такі числові характеристики:
   - довжину;
   - максимальний, мінімальний за значенням елемент.
5. Підрахуйте суму:

– елементів матриці M1;
– рядків матриці M1.
– елементів вектора V1.

6. Підрахуйте добуток:
– елементів матриці M1;
– стовпчиків матриці M1.
– елементів вектора V1.

7. Обчислить визначений інтеграл:

$$\int_0^4 (N * \mathrm{x}^N + 3\mathrm{x}^2 + 2)\mathrm{dx};$$

$$\int_0^\pi \sqrt{(N * 5 - \mathrm{c}^3)}\ \mathrm{dc},$$

де N – дорівнює індивідуальному номеру по журналу обліку.

8. Обчислить похідну для наступних функцій в точках 10, 1, 0 і 0,2:

1) $f(t) = (a0 + a_2 \cdot t^3) \cdot exp^{(a1 \cdot t)}$,
2) $f(t) = (a0 + a_1 \cdot t^2) \cdot exp^{(-a2 \cdot t)}$,

де константа *a0* – дорівнює індивідуальному номеру по журналу обліку: $a_1 = a0+5$, $a_2 = a0+3$.

9. Розв'яжіть систему лінійних алгебраїчних рівнянь:

1)
$$\begin{cases} 2 * N * \mathrm{x} - 3\mathrm{y} = 0 \\ 7 * N * \mathrm{x} - 5\mathrm{y} = 0 \end{cases};$$

2)
$$\begin{cases} 3 * N * \mathrm{x} - 3 * N * \mathrm{y} = 0 \\ 4 * N * \mathrm{x} - 4 * N * \mathrm{y} = 0 \end{cases};$$

3)
$$\begin{cases} 10 * N * \mathrm{x} - 2\mathrm{y} = 96 \\ -3 * N * \mathrm{x} + 5\mathrm{y} = 111 \end{cases};$$

4)
$$\begin{cases} 7 * N * \mathrm{x} - 7 * N * \mathrm{y} = -4 \\ 6 * N * \mathrm{x} - 6 * N * \mathrm{y} = -4 \end{cases}.$$

10. Розв'яжіть диференційне рівняння першого порядку на інтервалі [2,N] з початковою умовою $y_0 = -1$:

1) $y' + 3{,}5*\mathrm{N}*x = 0$
2) $y' - \sin(\mathrm{N}*x) = 0$

11. Розв'яжіть систему диференційних рівнянь на інтервалі $[0, \pi]$ з початковими умовами $x(0) = 3$ і $y(0) = 0$.

1)
$$\begin{cases} \mathrm{x}' = N * \mathrm{x} - 1 \\ \mathrm{y}' = N * \mathrm{x} + 2yt - 5 \end{cases};$$

2)

$$\begin{cases} x' = -N * 2x + 4y \\ y' = -N * x + 3yt \end{cases}$$

Вказівки до виконання:

1. Виконання лабораторної роботи здійснюється за допомогою редактора сценаріїв.
2. Пункти 7-11 виконуються за варіантами.
3. Під час виконання сценарію назви користувацьких функцій запам'ятовуються при іншому виконанні, зустрічаючи те саме ім'я система буде попереджувати про це повідомленням «Попередження: пере визначення функції…».

Для запобігання цієї ситуації додайте перед рядком з назвою користувацької функції (або на початку сценарію) рядок із системною функцією funcprot(0), яка за таким варіантом відключає виведення попереджень.

Контрольні питання

1. Як визначити кількість рядків, стовпчиків, елементів матриці?
2. Як визначити максимальний або мінімальний елемент вектора або матриці?
3. Як визначити максимальний або мінімальний елемент конкретного рядка або стовпчика матриці?
4. За допомогою якої функції обчислюється детермінант матриці?
5. За допомогою якої функції обчислюється ранг матриці?
6. Як обчислити довжину вектора? Як розв'язати систему лінійних алгебраїчних рівнянь виду
7. Як підрахувати суму всіх елементів матриці, окремого рядка або стовпчика?
8. Як підрахувати добуток всіх елементів матриці, окремого рядка або стовпчика?
9. Як обчислити визначений інтеграл?
10. Як обчислити похідну для певної точки?
11. За допомогою якої функції розв'язують диференційне рівняння першого порядку? Які аргументи вона має? Які є особливості використання цих аргументів?
12. Як розв'язати систему лінійних алгебраїчних рівнянь?
13. Як розв'язати систему диференційних рівнянь?

Практична робота №6

Імітаційне моделювання в середовищі Scilab/Xcos.

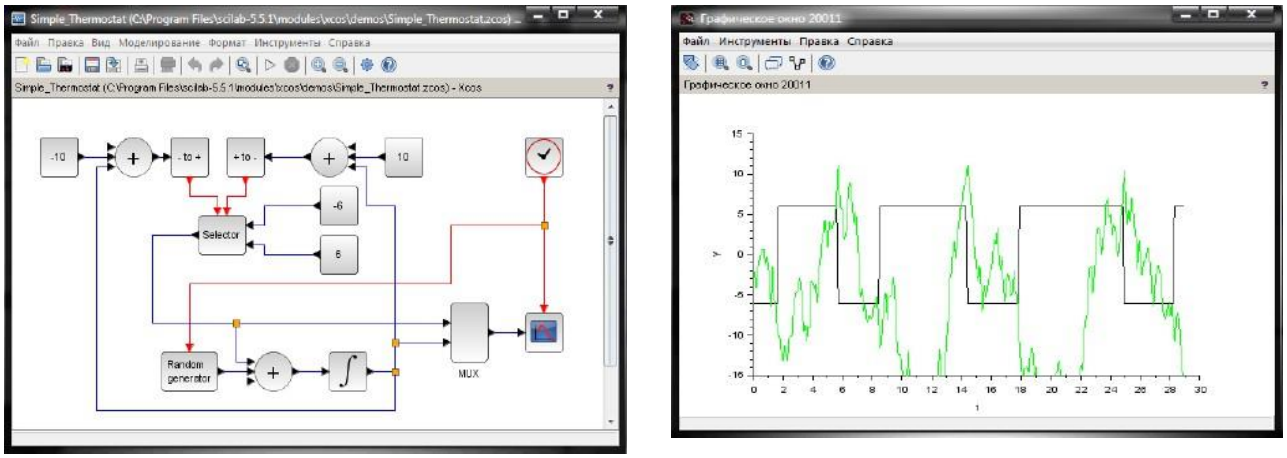
Мета – навчитися створювати імітаційні моделі систем у середовищі Scilab/Xcos.

Завдання: ознайомитися з бібліотеками Scilab/Xcos. Створити модель простої системи. Провести імітацію та отримати характеристики функціонування системи.

Література: [2, 4, 5, 11].

Теоретичні відомості

Scilab призначений для виконання інженерних і наукових обчислень та математичного моделювання систем. За своїми можливостями пакет Scilab можна порівняти з відомим математичним пакетом Mathcad, а за своїм інтерфейсом він схожий на пакет MATLAB. Однак при цьому пакет Scilab – програма вільного розповсюдження. Аналогічно тому, як у MATLAB є додаток для імітаційного моделювання систем Simulink, так у Scilab є додаток Xcos, який містить бібліотеку універсальних блоків для побудови моделі та здійснення імітаційного моделювання систем (рис.6.1).



а) приклад Xcos-моделі б) результати її роботи

Рисунок 6.1 – Робочі вікна Xcos

Імітаційне моделювання – це метод дослідження, який полягає у відтворені властивостей реальних об’єктів за допомогою віртуальних об’єктів. Всі розрахунки в комп’ютерній моделі виконуються в так званому системному часі, що відповідає реальному часу функціонування об’єкта дослідження або системи .

Рекомендації до виконання завдань з імітаційного моделювання.

Встановлення на комп’ютері системи SciLab/Xcos.

Вільно розповсюджену версію пакета разом з повною документацією англійською мовою можна отримати на сайті програми www.scilab.org.

Для того, щоб встановити Scilab на ПК, потрібно запустити однойменний виконуваний файл, після чого починає свою роботу Майстер встановлення. В першому вікні Майстра встановлення потрібно вибрати мову і натиснути кнопку OK для продовження встановлення. Надалі рекомендується погодитися з усіма пропозиціями Майстра (просто натискати Next).

Запуск Xcos.

Для запуску програми потрібно попередньо запустити пакет Scilab. Основне вікно пакета Scilab показано на рисунку 6.2. Там же показана підказка, що з’являється у вікні при наведенні покажчика миші на ярлик Xcos в панелі інструментів.

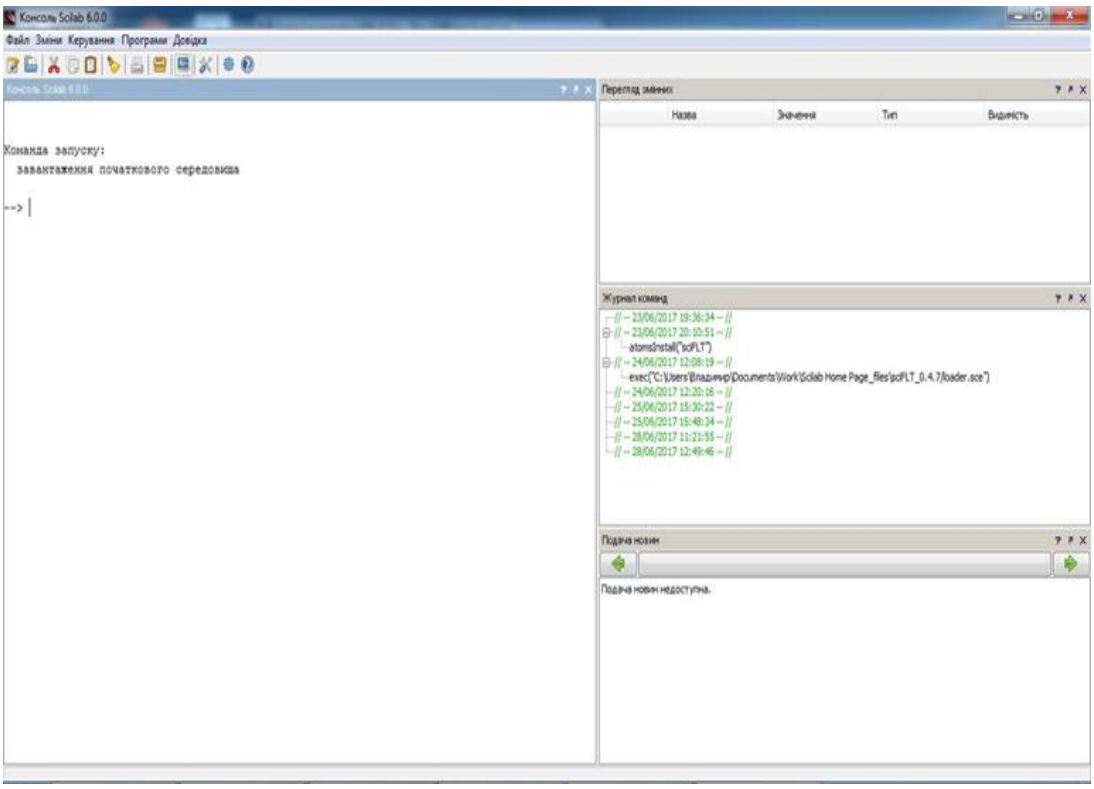
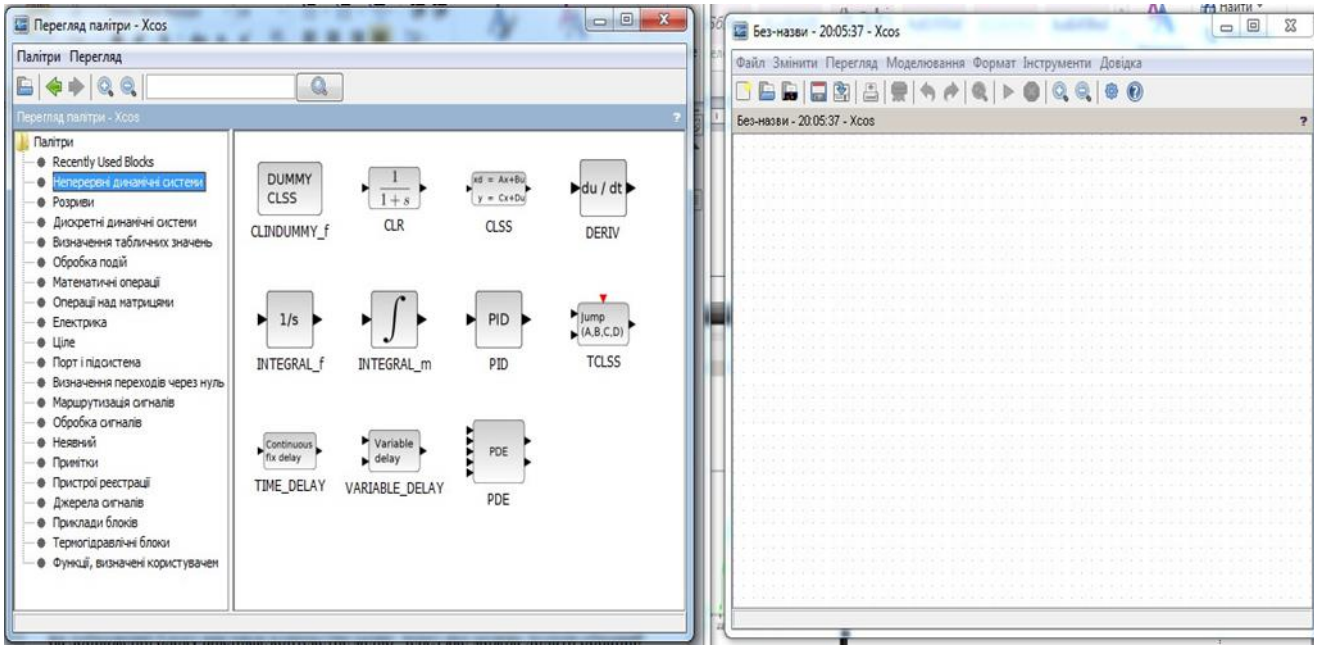


Рисунок 6.2 – Командне вікно Scilab

Для побудови блокової діаграми використовується графічний редактор Xcos.

Після запуску Xcos зазвичай відображаються два вікна (рис. 6.3): вікно Перегляд палітри блоків і вікно Графічного редактора.



а) Перегляд палітри блоків б) Вікно редактора

Рисунок 6.3 – Графічний редактор Xcos

Якщо вікно Перегляд палітри блоків відсутнє, його потрібно відобразити, вибравши Перегляд → Палітри блоків в головному меню вікна Графічного

редактора Xcos.

У вікні Перегляд палітри блоків перераховані групи блоків, з яких будується діаграма Xcos. Виділивши потрібну групу лівим кліком мишки (ЛКМ), ви побачите графічні зображення блоків, які входять до неї. Правий клік миші (ПКМ) на зображенні блока викликає контекстне меню, через яке можна додати обраний блок до діаграми або викликати довідку за цим блоком. Додати вибраний блок до діаграми можна, просто перетягнувши його мишею.

Найпростіша діаграма.

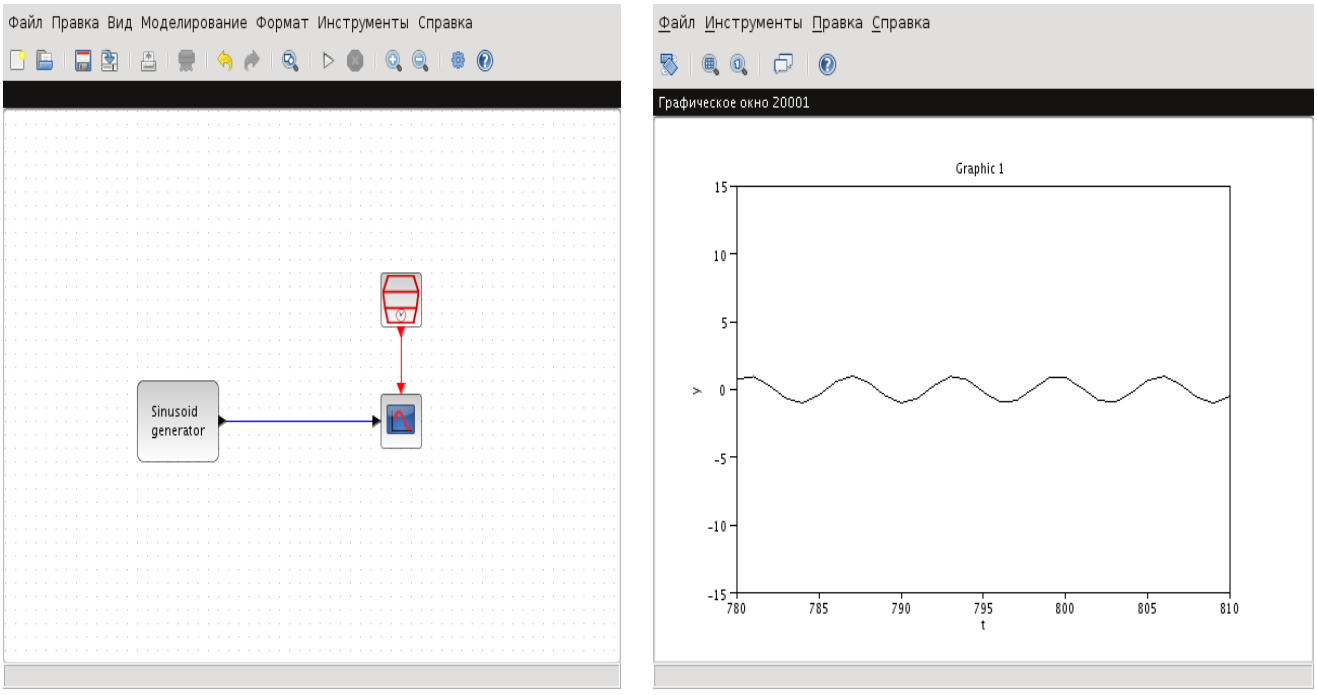
Виберіть палітру Джерела сигналів і впливів і перетягніть у вікно діаграми блоки 

(генератор синусоїди) і 

(лічильник часу). Потім перейдіть до палітри Пристрої реєстрації та додайте до діаграми блок 

(осцилограф).

З'єднайте вихід генератора з чорним входом осцилографа, а вихід лічильника – з червоним входом осцилографа (рис.6.4). Лічильник часу використовується для керування роботою осцилографа. Сполучні лінії проводяться від виходу до входу (або навпаки) при натиснутій ЛКМ. Дозволені з'єднання підсвічуються зеленим. Для видалення сполучної лінії виділіть її та натисніть Delete.



а) Діаграма б) Результат моделювання

Рисунок 6.4 – Найпростіша діаграма

Для створення відгалуження від лінії роблять на потрібному місці два клацання мишкою (КЛЦ) і тягнуть лінію в потрібне місце. На вже виділеному зв'язку можна зробити КЛЦ один раз. Для видалення ще не завершеної лінії зв'язку роблять один КЛЦ.

Для видалення блока потрібно вибрати блок (вказати курсором на його зображення і натиснути ліву клавішу миші), а потім натиснути клавішу Delete на клавіатурі.

Для зміни розмірів блока потрібно вибрати блок, встановити курсор в один з кутів блока і, натиснувши ліву клавішу миші, змінити розмір блока (курсор при цьому перетвориться в двосторонню стрілку).

Для встановлення часу моделювання вибрати пункт Моделювання → Налаштування в головному меню графічного редактора (рис. 6.5) і встановити параметр Загальний час інтегрування рівним потрібному значенню.

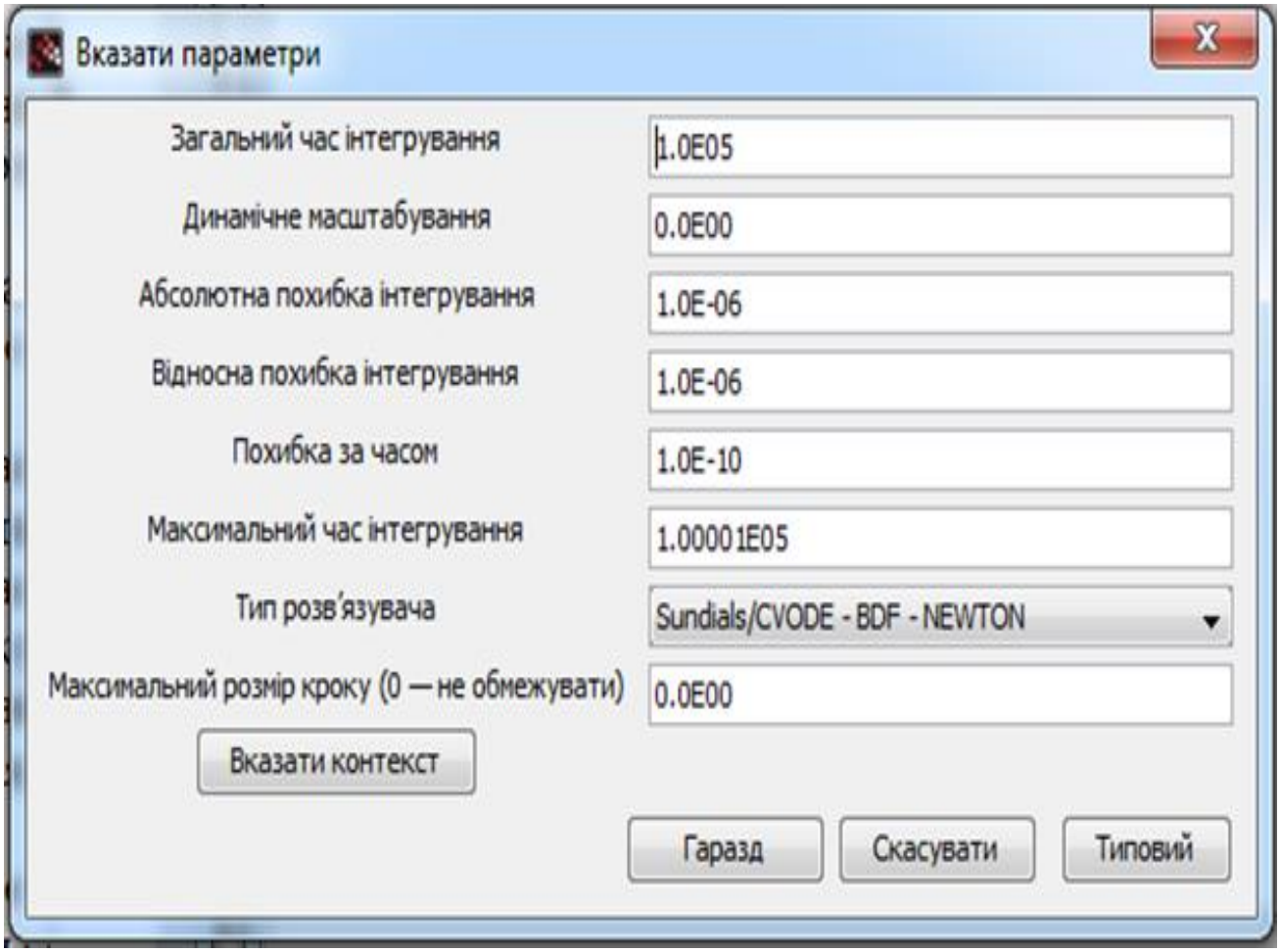


Рисунок 6.5 – Вікно встановлення загальних параметрів моделювання

Для запуску моделювання вибирають Моделювання → Виконати в головному меню редактора або просто натискають на відповідну кнопку в панелі інструментів. Для зупинки моделювання виберіть Моделювання → Завершити або ж скористайтеся відповідною кнопкою в панелі інструментів.

Викличте вікно «Введення значень» для осцилографа на діаграмі. Змініть значення змінних Ymin і Ymax, встановивши їх рівними -2 і 2 відповідно (рис. 6.6). Запустіть моделювання.

Збереження і завантаження. Зберігайте поточну діаграму частіше! Якщо у діаграмі робилися будь- які зміни, то вона може зберегтися неправильно. Тому відкрийте вікно створення нової діаграми: Файл → Створити діаграму, скопіюйте наявну діаграму у нове вікно (зафіксуйте ЛКМ над лівим верхнім кутом діаграми і потягніть її донизу направо, потім натисніть Ctrl-Insert, перейдіть у нове вікно і натисніть Shift-Insert). Збережіть діаграму, вибравши Файл → Зберегти як.

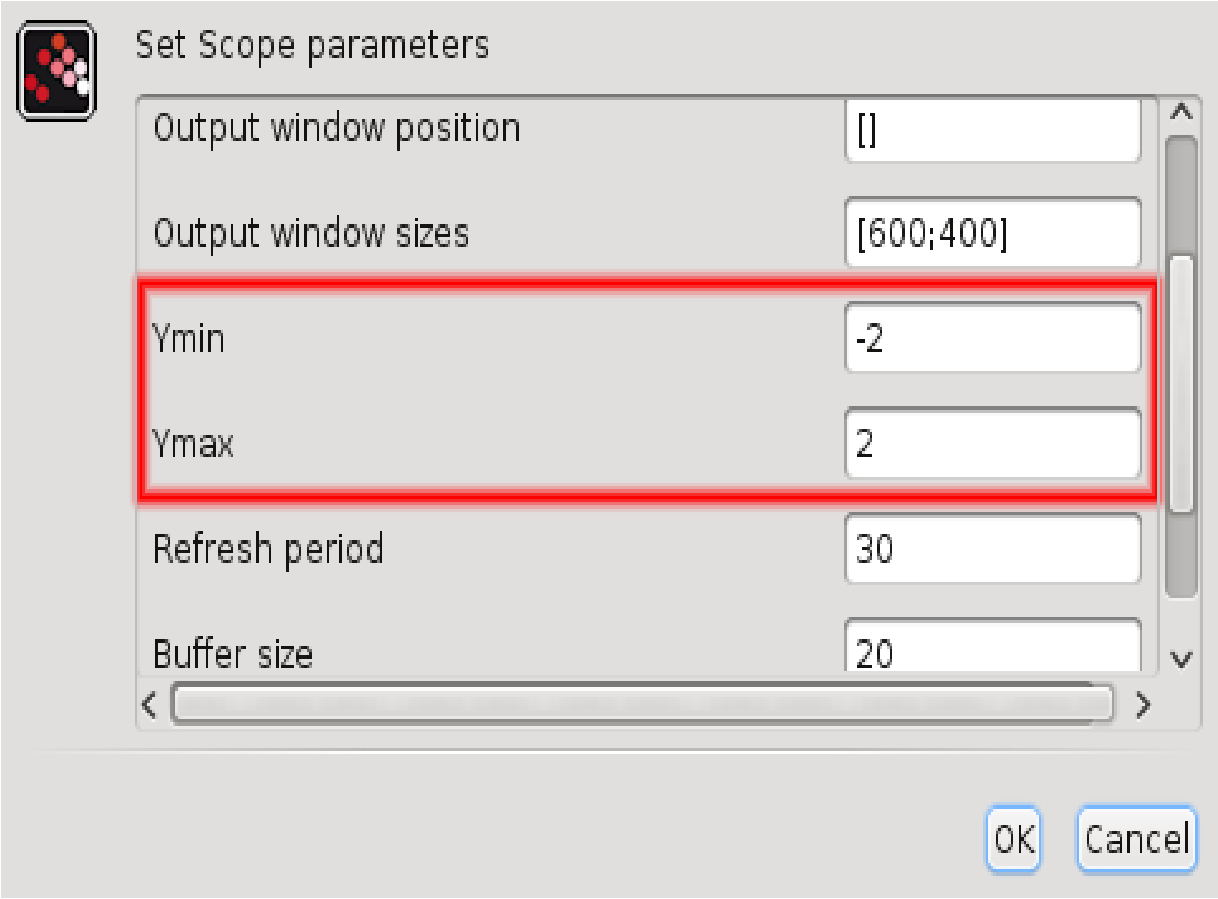


Рисунок 6.6 – Вікно зміни параметрів блока

Завжди зберігайте діаграми тільки в своїй папці! Завантажити збережену діаграму можна через Файл → Відкрити або

Файл → Недавні файли.

Завдання

1. Ознайомитися з можливостями пакета Scilab
2. Підготувати відповіді на контрольні питання.
3. Створити модель-діаграму калькулятора. Використати блоки (рис.6.7):

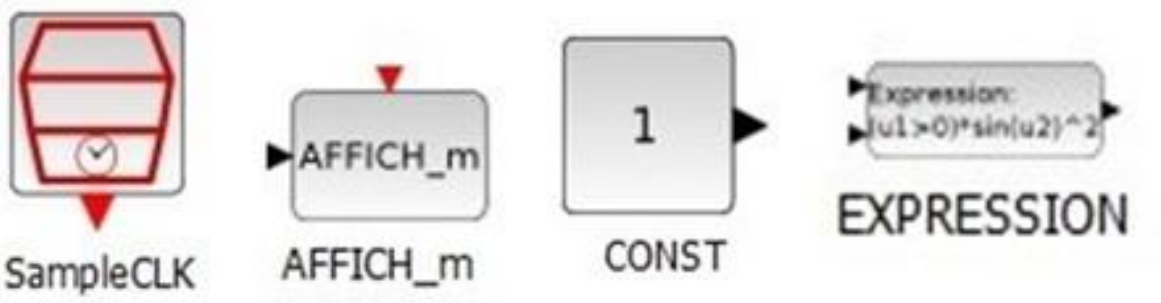


Рисунок 6.7 – Блоки для модель-діаграми калькулятора

4. Розрахувати значення за формулами наведеними в таблиці 6.1

Таблиця 6.1 – Вихідні значення

Варіант	Формула	x	y	z
1	$\frac{\sin xy}{\cos z}$	2,2	5,7	0,9
2	$x^2 - \sqrt{4yz}$	4,3	3,6	-5,1
3	$\frac{xyz}{x+y+z}$	4,2	2,3	6,0
4	$\sqrt{x^3+y^2+z^2}$	-5,1	2,3	4,1
5	$xe^{\frac{y-3}{z}}$	2,3	6,0	2,2
6	$\frac{xy^2}{x+y+z}$	2,2	5,7	0,9
7	$\sqrt{x^3+y^2+z^2}$	4,3	3,6	-5,1
8	$\frac{\sin xy}{\cos zx}$	4,2	2,3	6,0
9	$x^2 + \sqrt{4yz}$	2,2	5,9	0,9
10	$\frac{x(y-z)}{x+y+z}$	4,3	3,6	-5,1
11	$\sqrt{x^3+y^2-z^2}$	4,2	2,3	6,0
12	$xe^{\frac{xy-3}{z}}$	-5,1	4,3	4,1
13	$\frac{xy^2}{x+y+z}$	2,3	6,0	2,2
14	$\sqrt{x^3+y^2+z^2}$	2,2	5,7	0,9
15	$\sqrt{x^3+2y^2+3z^2}$	4,3	3,6	-5,1
16	$x^2 + \sqrt{4xyz}$	4,2	2,3	6,0
17	$\frac{xyz}{x+y+z}$	2,2	6,7	0,9
18	$\sqrt{22x^3+3y^2+z^2}$	4,3	3,6	-5,1
19	$5xe^{\frac{y-3}{z}}$	4,2	2,3	6,0
20	$\frac{x-y^2}{x+y+z}$	-2,1	2,3	4,1

5. Створити моделі-діаграми диференціаторів. Побудувати в одній системі координат графіки вхідної і вихідної функцій (таблиці 6.2).
6. Створити моделі-діаграми інтеграторів. Для кожної функції побудувати в одній системі координат 2 графіки: вхідної і вихідної функцій (таблиці 6.2).

Таблиця 6.2 – Вихідні значення

Вхідна функція	Варіанти параметрів									
	1	2	3	4	5	6	7	8	9	10
Константа. Значення	2.5	3.2	4.3	3.6	-5.1	2.3	6.0	2.2	6.0	2.2
Синусоїдальний сигнал: амплітуда, частота, фаза	2, 20, 0	3, 30, 0	4, 40, 0	5, 25, 0	6, 35, 0	7, 45, 0	8, 15, 0	9, 25, 0	8, 15, 0	9, 25, 0
Прямокутний сигнал: амплітуда, період	2, 2	3, 3	4, 4	5, 2	6, 3	7, 4	8, 1	9, 2	4, 4	5, 2
Лінійно змінний сигнал: нахил	2,5	3.2	4,3	3,6	-5,1	2,3	6,0	2,2	4,3	3,6
Ступінчастий сигнал: час перемикання, початкове значення, кінцеве значення	2, 2, 3	3, 3, 0	4, 4, -1	5, 0, 4	6, 3, -1	7, 4, -2	8, 1, 0	9, 0, 2	4, 4, 2	5, 0, 1

Продовження табл.6.2

Вхідна функція	Варіанти параметрів									
	11	12	13	14	15	16	17	18	19	20
Константа. Значення	2.3	6.0	2.2	6.0	2.2	2.5	3.2	4.3	3.6	-5.1
Синусоїдальний сигнал: амплітуда, частота, фаза	2, 20, 0	3, 30, 0	4, 40, 0	5, 25, 0	6, 35, 0	7, 45, 0	8, 15, 0	9, 25, 0	8, 15, 0	9, 25, 0
Прямокутний сигнал: амплітуда, період	7, 4	8, 1	9, 2	8, 1	9, 2	2, 2	3, 3	4, 4	5, 2	6, 3
Лінійно змінний сигнал: нахил	2,5	3.2	4,3	3,6	-5,1	2,3	6,0	2,2	6,0	2,2
Ступінчастий сигнал: час перемикання, початкове значення, кінцеве значення	7, 4, -2	6, 1, 0	9, 0, 2	7, 1, 0	9, 0, 2	2, 2, 3	3, 3, 0	4, 4, -1	5, 0, 4	6, 3, -1

7.Створити модель-діаграму для побудови поверхні $[t,\ f(t),\ f'(t)]$, (фазовий

портрет). Вихідні данні таблиці 6.3.

Таблиця 6.3 – Вихідні значення

Варіант	f(t)	t_{min}	t_{max}
1	$\sqrt{\frac{t}{t+1}}$	2,2	5,7
2	$\sin t^2$	3,6	4,3
3	$sin\, t\, /t$	2,3	4,2
4	$\sqrt{t^2\ +1}$	-5,1	2,3
5	$\sin t^3$	2,3	6,0
6	$\sin t^2$	2,2	5,7
7	$sin\, t\, /t$	3,6	4,3
8	$\sqrt{t^2\ +1}$	2,4	5,0
9	$\sqrt{\frac{t}{t+1}}$	3,5	4,0
10	$\sin t^2$	3,6	4,3
11	$\frac{\sin t}{t}$	2,3	4,2
12	$\sqrt{t^2\ +1}$	-5,1	2,3
13	$\sqrt{\frac{t}{t+1}}$	2,3	6,0
14	$\sin t^2$	2,2	5,7
15	$sin\, t\, /t$	4,6	4,3
16	$\sqrt{t^2\ +1}$	2,3	4,2
17	$\sqrt{\frac{t}{t+1}}$	-4,8	2,0
18	$\sin t^2$	2,5	5,8
19	$sin\, t\, /t$	2,4	5,2
20	$\sqrt{t^2\ +1}$	4,4	4,1

Контрольні питання

1. У чому полягає сутність імітаційного моделювання та які його основні завдання?
2. У чому полягає агрегатний принцип імітаційного моделювання та як він використовується при побудові моделей складних систем?
3. Які основні можливості та призначення системи Scilab/Xcos?
4. Які основні палітри блоків Xcos існують та для чого вони призначені?

Практична робота №7– 8

Статистичний аналіз випадкових процесів за допомогою Scilab/Xcos.

Мета – навчитися проводити статистичний аналіз випадкових процесів у середовищі Scilab/Xcos.

Завдання: ознайомитися з бібліотеками Scilab/Xcos. Створити модель генератора випадкових процесів. Провести імітацію та отримати характеристики функціонування системи.

Література: [2, 4, 5, 9-10].

Теоретичні відомості

Стохастичною називають невизначеність, яка зумовлена дією випадкових факторів впливу.

Випадковою називається така величина, яка в результаті випробувань може прийняти те чи інше значення.

Випадкові величини поділяються на дискретні та неперервні. Крім дискретної і неперервної випадкових величин зустрічаються випадкові величини змішаного типу, для яких, разом з ділянками неперервних значень, є окремі, ізольовані значення.

Для того, щоб задати випадкову величину, потрібно задати множину значень, які вона може приймати, та ймовірності, з якими вона приймає ці значення. Відповідь на це питання дає вичерпна характеристика випадкової величини – закон розподілу.

Закон розподілу дозволяє визначити ймовірність появи випадкової величини в будь-якому інтервалі (і, зокрема, ймовірності будь-яких значень дискретної випадкової величин).

Для характеристики неперервної випадкової величини визначають ймовірність появи значення випадкової величини X меншого за x, де x –значення поточної змінної, тобто визначають ймовірність події $X < x$.

Ймовірність цієї події залежить від x ,тобто є функцією x. Ця функція $F(x)= P\ (X<x)$ називається функцією розподілу випадкової величини X .

При вирішенні багатьох практичних задач часто досить вказати окремі числові характеристики, що визначають особливості того чи іншого розподілу випадкової величини. Ці характеристики називають моментами розподілу.

Для оцінювання ступеня розкиду, розсіювання значень випадкової величини відносно середнього, використовують перший початковий і другий центральний моменти:

– математичне сподівання;

– дисперсію

та пов’язані з ними показники:

– середнє квадратичне відхилення;

– коефіцієнт варіації.

Зручним способом опису процесів є їх спектральне подання. Але випадкові процеси не відповідають умовам перетворення Фур’є, яке використовується для

знаходження спектра. Тому для моделювання динаміки стаціонарних систем в умовах невизначеності використовують спектральну щільність потужності – зображення за Фур'є не самого процесу, а його кореляційної функції:

Найрозповсюдженішим способом побудови імітаційної моделі є статистичне моделювання.

Статистичне моделювання полягає у проведенні чисельного експерименту з функціональною моделлю. Методика статистичного моделювання передбачає ряд послідовних етапів:

– моделювання на ЕОМ випадкових сигналів у вигляді числових послідовностей з заданою кореляцією та законом розподілу ймовірностей, які імітують вхідні сигнали і збурювальні впливи;

– моделювання перетворення сигналів і впливів;

– статистична обробка результатів моделювання.

Задачу генерування випадкових чисел із заданим законом розподілу розв'язують в декілька етапів. Спочатку отримують послідовність рівномірно розподілених на інтервалі [0,1] випадкових чисел, а з неї – послідовність випадкових чисел із заданим законом розподілу.

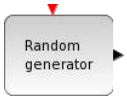
Окремим випадком є генерування числової послідовності з нормальним розподілом ймовірностей і експоненціальною автокореляційною функцією. Цей вид випадкової послідовності є найпоширенішим і найлегшим для генерування. Генерування здійснюється у два етапи:

– генерування некартельованої рівномірно розподіленої послідовності x_i за допомогою будь-якої програми-генератора, які наразі є в усіх мовах програмування;

– перетворення на задану послідовність x'_j методом підрахунку ковзного середнього.

Згенеровані таким чином дані піддаються перетворенню за формула ми, які моделюють роботу блоків системи, у послідовності, яка відповідає розташуванню блоків системи.

Імітаційне моделювання стохастичних процесів з використанням Scilab/Xcos здійснюється за допомогою генератора випадкових чисел, який знаходиться у палітрі «Джерела сигналів».

Блок RAND_m 

використовується для отримання випадкових чисел, розподілених за нормальним або рівномірним законом. Блок має один керувальний вхід і один інформаційний вихід. Параметри блока:

– Datatype (тип вихідних даних): 1 – дійсні числа, 2 – комплексні;

– Flag: прапорець, який визначає вид закону розподілу: 0 – рівномірний, 1 – нормальний (гаусівський);

– A і B: для рівномірного розподілу величина A визначає мінімальне значення, а величина A + B – максимальне. Для нормального розподілу A визначає математичне сподівання, а B – середнє квадратичне відхилення (СКВ).

– SEED: числа, використовувані для ініціалізації машинного генератора псевдовипадкових чисел. Перше значення належить до дійсної, а дру- ге – до уявної

частини вихідного сигналу. Два генератора з однаковим параметром SEED видаватимуть два ідентичних псевдовипадкових сигнали.

Для статистичної обробки даних можна застосувати блоки, показані на рисунку 7.1 і рисунку 7.2.

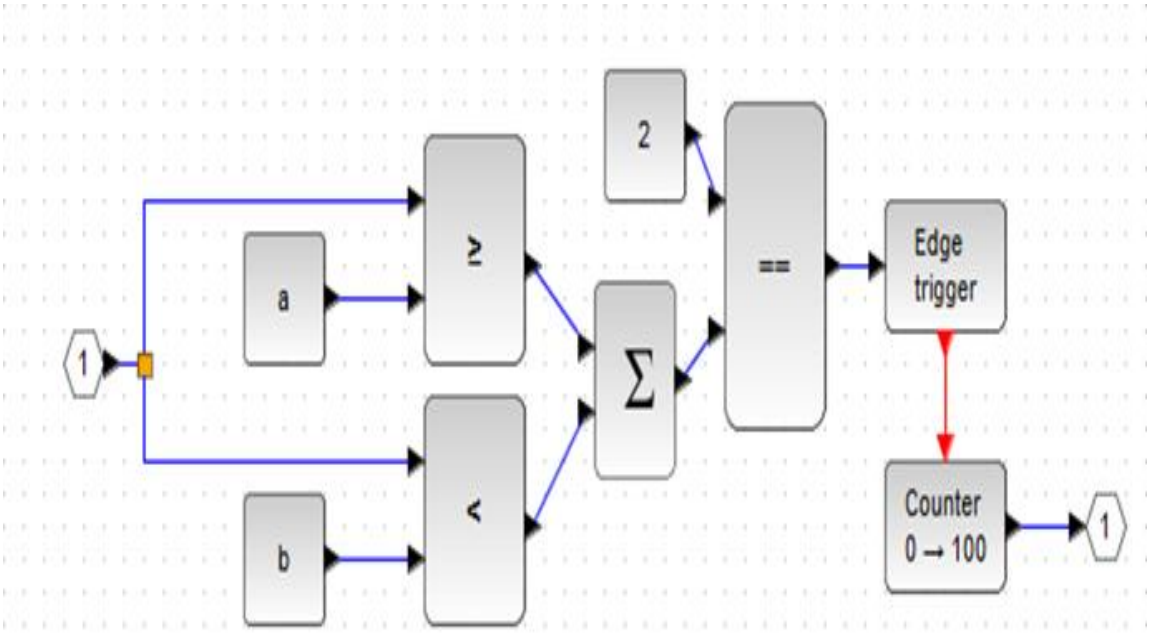


Рисунок 7.1 – Суперблок для підрахунку потраплянь значення випадкового процесу у проміжок $[a,b)$

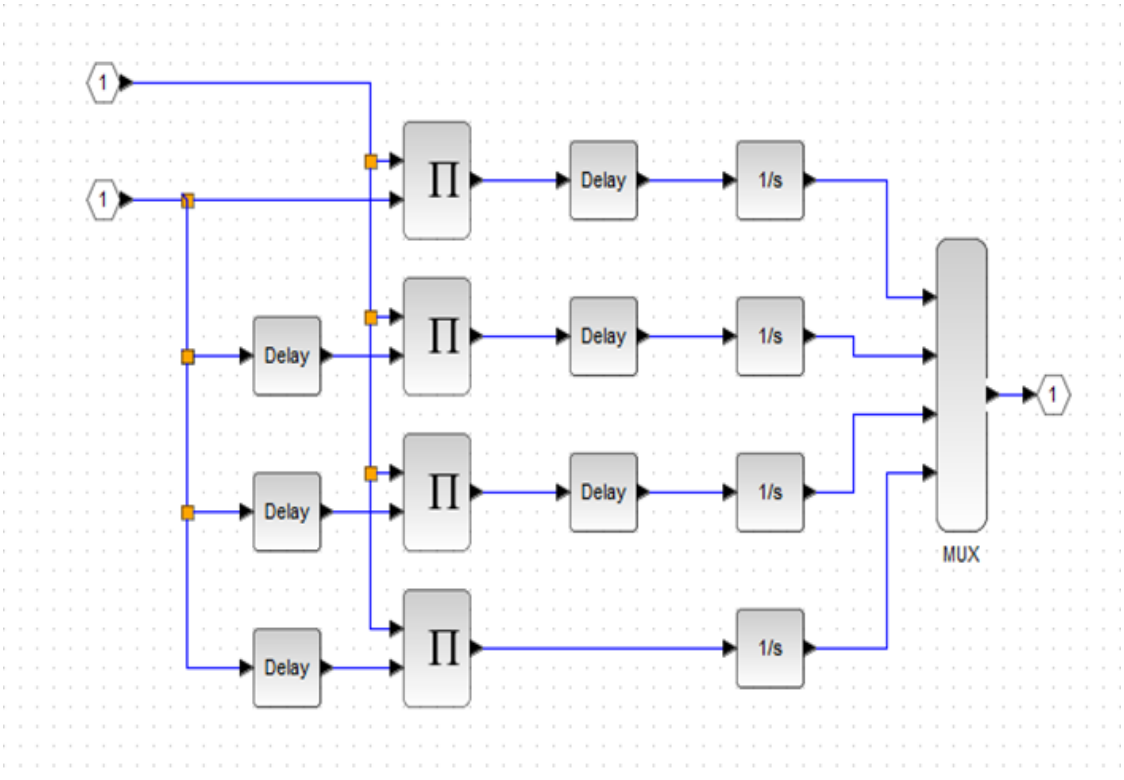


Рисунок 7.2 – Суперблок для вимірювання кореляційної функції (на інтервалі вимірювання отримується 4 значення, починаючи з $\tau = 0$)

Завдання

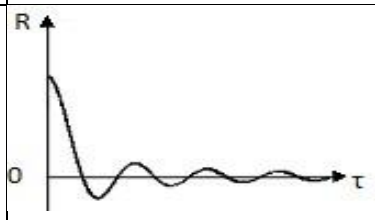
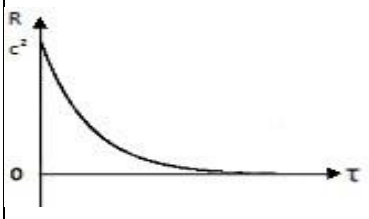
1. Розробити і дослідити генератор сигналу з заданим розподілом ймовірностей відповідно до структурної схеми (рис.7.3).

Рисунок 7.3 – Генератор сигналу з заданим розподілом ймовірностей

2. Розробити і дослідити генератор сигналу з заданою кореляційною функцією відповідно до структурної схеми (рис.7.4). Вихідні данні в таблиці 7.1.

Рисунок 7.4 – Генератор сигналу з заданою кореляційною функцією

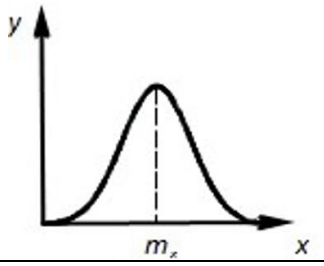
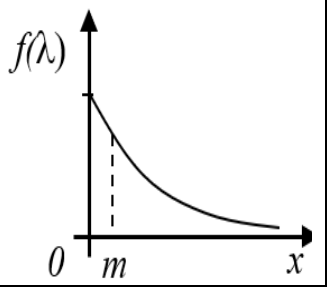
Таблиця 7.1 – Вихідні данні журналу обліку (де N–номер варіанту по журналу обліку)

Варіант	Вид процесу	Кореляційна функція	
		Вираз	Графік
парний	Результат проходження білого шуму через ідеальний фільтр низьких частот	$\frac{c^2}{\pi t} sin\, N\omega t$	
непарний	Результат проходження білого шуму через реальний RC-фільтр низьких частот 1-го порядку	$c^2 e^{-\alpha N\lvert\ \ t\rvert}$, де $\alpha=\frac{1}{RC}$	

3. Проаналізувати вплив нелінійного перетворення на кореляційну функцію

і динамічного перетворення – на закон розподілу ймовірностей. Вихідні данні в таблиці 7.2.

Таблиця 7.2 – Вихідні данні журналу обліку (де N – номер варіанту по журналу обліку)

Варіант	Закон розподілу	Щільність розподілу ймовірності (ймовірність – для дискретних величин)	Графік
Парний	Експонен-ціальний	$f(x) = \frac{1}{\sqrt{\pi a}} \cdot e^{-\frac{(x-b)^2}{Na}}$	
Непарний	Нормальний	$f(x) = \begin{cases} \lambda e^{-N\lambda x}, \text{при } x \geq 0, \lambda > 0 \\ 0, \quad \text{при } x < 0, \lambda > 0 \end{cases}$	

Контрольні питання

1. Чим відрізняються випадкова величина і випадковий процес?
2. Що може стати джерелом стохастичної невизначеності в системі керування?
3. Що спільного і чим відрізняються спектр і спектральна щільність потужності сигналу?
4. Як отримати кореляційну функцію, якщо відома спектральна щільність потужності?
5. Чим відрізняється математичне сподівання від середнього значення?

Практична робота №9 –10.

Імітаційне моделювання динамічних систем в середовищі Scilab/Xcos.

Мета – навчитися створювати імітаційні моделі динамічних систем у середовищі Scilab/Xcos.

Завдання: Створити імітаційну модель динамічних систем. Провести імітацію та отримати характеристики функціонування системи.

Література: [2, 4, 5].

Теоретичні відомості

Імітаційне моделювання динаміки систем широко використовується як при їх проектуванні, так і в самих системах для розрахунку оптимального управління. Відповідно, для задач проектування переважно використовуються пакети імітаційного моделювання, зокрема Scilab/Xcos, а для задач оптимального управління та адаптації використовуються спеціалізовані програми, записані певною алгоритмічною мовою, переважно – мовою Сі.

Програмні імітаційні моделі є дискретними.

Для імітаційного моделювання в Scilab/Xcos використовуються блоки палітри «Неперервні динамічні системи» (рис. 9.1, табл. 9.1) і «Дискретні динамічні системи» (рис. 9.2, табл. 9.2)

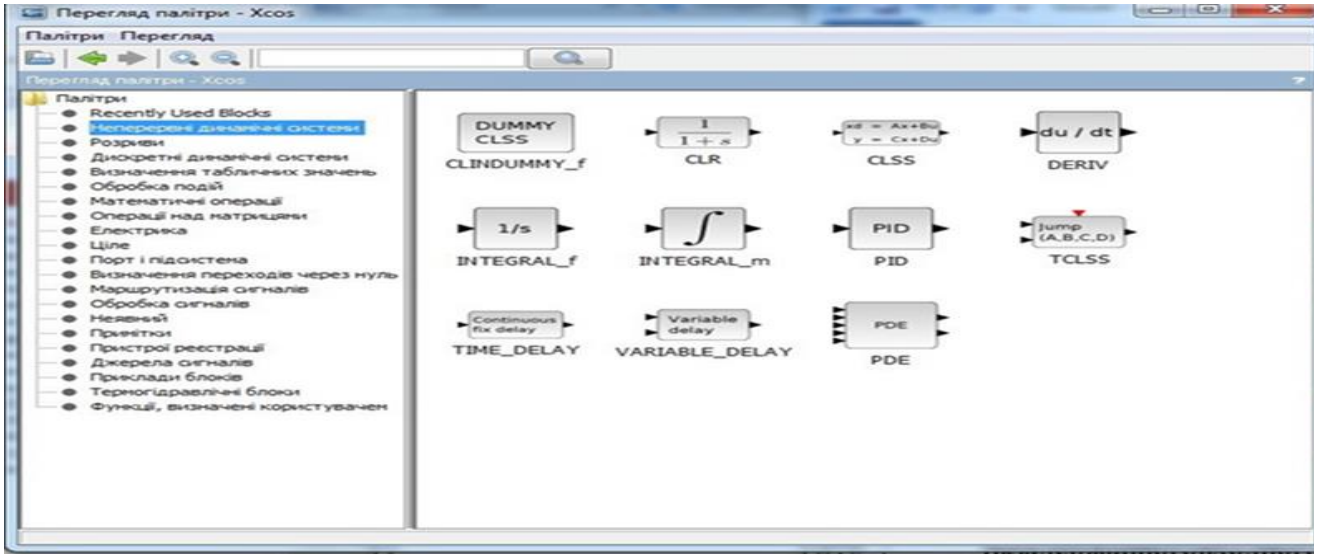


Рисунок 9.1 – Блоки палітри «Неперервні динамічні системи»

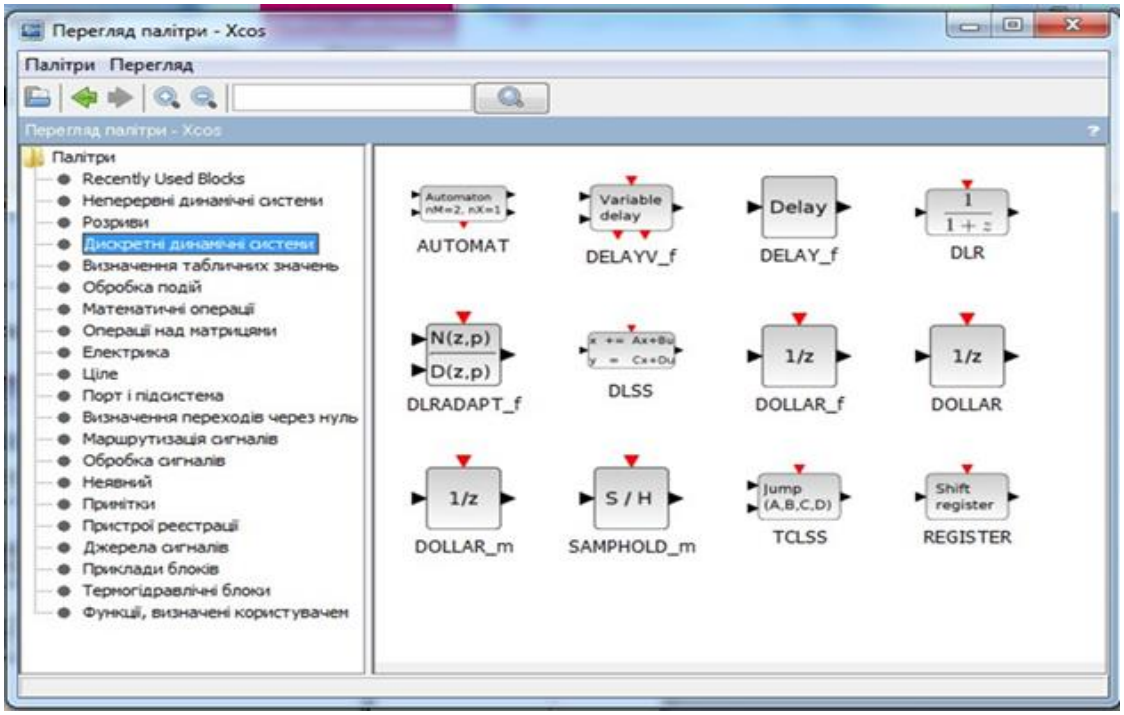
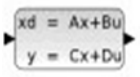
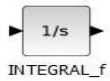
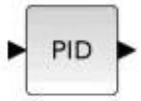
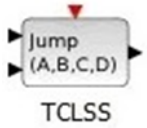
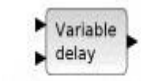
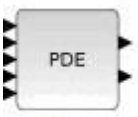
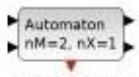
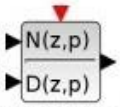
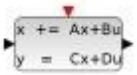


Рисунок 9. 2 – Блоки палітри «Дискретні динамічні системи»

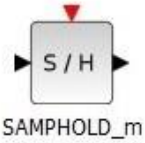
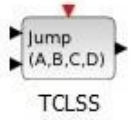
Таблиця 9.1 – Опис блоків «Неперервні динамічні системи»

Блок	Опис
	CLSS – неперервна система в просторі станів. Мають бути задані матриці A, B, C, D і початковий стан x0. x – це вектор змінних стану, u – вектор вхідних функцій, y – вектор вихідних змінних.
	DERIV – блок похідної. Вона обчислюється за вхідним сигналом $\Delta u/\Delta t$. Початковий вихід для блока – 0.
	Цей блок є інтегратором, визначеним за допомогою передатної функції. Вихід є інтегралом входу.
	INTEGRAL_ m – інтегратор: y0 – початкова умова, а t0 – відповідний час.
	ПІД-регулятор Закон ПІД-регулятора (алгоритм) містить три параметри: коефіцієнти пропорційної складової k_p, інтегральної k_i та диференціальної k_d
	TCLSS – неперервна лінійна система зі стрибком. Цей блок реалізує неперервну лінійну систему з можливістю стрибків у стані. Кількість входів до цього блока – два. Перший вхід є звичайним входом лінійної системи, другий – це нове значення стану, яке копіюється в стан, коли «подія» надходить на порт «події» цього блока. Це означає, що стан системи переходить до значення, при- сутнього на другому вході. Система визначається матрицями (A, B, C, D) та початковим станом x_0. Розміри мають бути сумісними. Розмір входів та виходів регулюється автоматично.
	TIME_DELAY – постійна затримка за часом. На початку моделювання блок виводить параметр *Initial input*, доки час моделювання перевищить параметр *Time delay*, тоді блок починає створювати затримку входу. Параметр «Час затримки» має бути невід'ємним.
	Блок Variable Transport Delay може бути використаний для імітації змінної затримки часу між дією та її ефектом. Величина затримки задається значенням на другому вході.
	Цей блок є реалізацією кількох чисельних методів (кінцеві елементи, різниці і об'єми 1-го та 2-го порядків) розв'язання одновимірних диференціальних рівнянь в частинних похідних (ЧДУ) в Xcos. Математичні рамки обмежують ЧДУ лінійним скаляром, максимальним по- рядком два (час і простір). Система вибирає найбільш ефективний чисельний спосіб залежно від типу рівняння і управляє рішенням

Таблиця 9.2 – Опис блоків «Дискретні динамічні системи»

Блок	Опис
 AUTOMAT	Цей блок дає можливість будувати гібридні автомати, тобто гібридні системи, чия дискретна частина визначена способами і переходами між способами, а неперервна частина визначена через диференціальні рівняння алгебри. Блок забезпечує автоматичне перемикання між підсистемами. Підсистеми побудовані таким способом, що вони мають стаціонарний вхідний вектор і обчислюють плавну і стрибкоподібну функції (перетин нуля) і передають їх назад до блока автомата. Стаціонарні змінні визначені в блоці автомата, підсистеми – статичні функції.
 DELAYV_f	Блок Variable Delay може бути використаний для імітації змінної часу затримки між дією та її ефектом. (Аналогічний блок є у палітрі «Розриви»)
 DELAY_f	Цей скомпільований суперблок реалізує дискретизовану затримку. Вона побудована з регістром зсуву та годинником. Значення затримки визначається часом дискретизації, помножене на значення стану регістра.
 DLR	Дискретна передавальна функція. Блок реалізує лінійну дискретну систему, подану раціональною функцією
 DLRADAPT_f	Ця система моделей та блоків, яка подана нулями та полюсами дискретної передавальної функції.
 DLSS	Дискретна система в просторі станів. Мають бути задані матриці A, B, C, D і початковий стан x_0. Розміри мають бути сумісними. Після надходження вхідної «події» на порт «події», стан входу оновлюється.
 DOLLAR_f	Цей блок еквівалентний оператору дискретного часу 1/z. Блок приймає один вхід і генерує один вихід, який може бути або скалярним, або вектором. Якщо вхід є вектором, всі елементи вектора затримуються однаково.
 DOLLAR_m	Оператор затримки (1/z). Вхідна величина подається на вихід за сигналом активації, а після цього на вході запам’ятовуються нові вхідні величини.

Продовження табл.9.2

	Відтворення на виході вхідного значення і зчитування нового вхідного значення за сигналом активації. Кожного разу, коли на блок надходить «подія», блок копіює свій вхід на виході і тримає до наступної «події». Для періодичної вибірки та утримання, вхід «події» має бути згенерований годинником.
	Цей блок реалізує неперервну лінійну систему з можливістю стрибків у стані. Кількість входів до цього блока становить два. (Див. аналогічний блок у палітрі «Неперервні динамічні системи»).
	Зсувний регістр Цей блок реалізує регістр зсуву. На кожній вхідній «події» регістр зміщується на один крок.

Приклад.

Нехай задана система керування (рис.9.3). Розробимо алгоритм її дискретного моделювання.

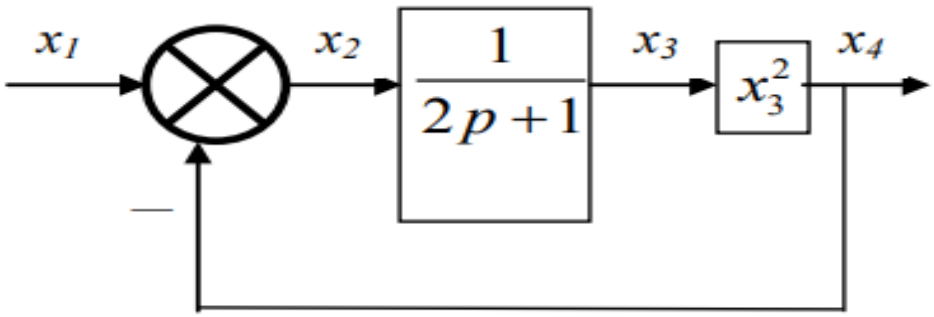


Рисунок 9.3 – Схема системи керування

Динамічний блок буде моделюватися перетворенням

$$x_3 = \frac{\frac{2}{\Delta t}x_3 + x_2}{1 + \frac{2}{\Delta t}}$$

Вважатимемо Δt=1с. Далі проводять аналітичний розрахунок і строюється імітаційна модель.

Імітаційна модель системи в Scilab/Xcos для умов цього прикладу наведена на рисунку 9.4.

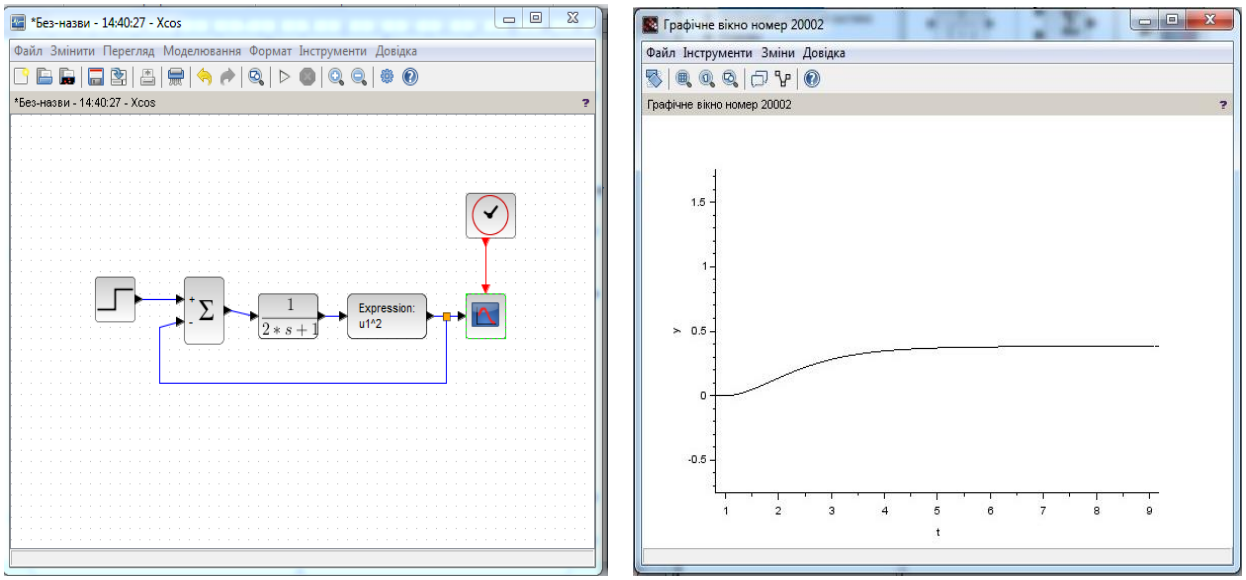


Рисунок 9.4 – Приклад імітаційної моделі

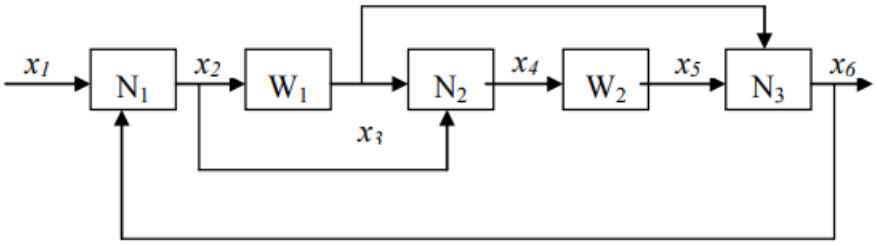
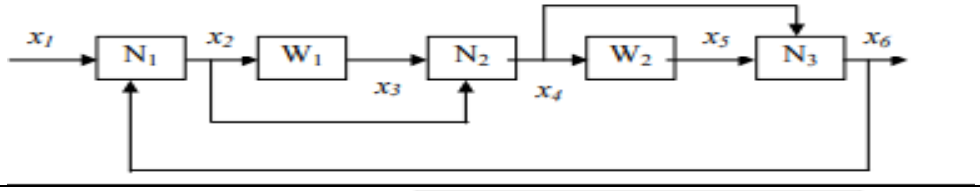
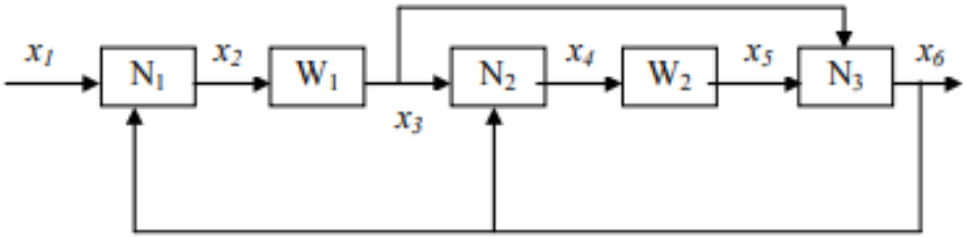
Завдання

1. В пакеті Scilab/Xcos створити імітаційну модель тої самої системи.
2. Виконати імітаційне моделювання в умовах надходження на вхід системи синусоїдального впливу $x_1 = \sin(2t)$.
3. Виконати імітаційне моделювання в умовах надходження на вхід системи одиничного ступінчастого впливу.
4. Виконати імітаційне моделювання в умовах надходження н на вхід системи нормальних випадкових вхідних впливів з експоненціальною кореляційною функцією і одиничною дисперсією.
5. Виконати статистичну обробку результатів: визначити дисперсію вихідного процесу.
6. Розрахувати коефіцієнт фільтрації (збільшення/зменшення дисперсії).
7. Проаналізувати результати, зробити висновки.
8. Вихідні данні наведено в таблиці 9.3. та таблиці 9.4.

Таблиця 9.3 – Вихідні данні журналу обліку (де А – номер варіанту по журналу обліку)

Варіант	Схе-ма	N_1	N_2	N_3	W_1	W_2
Парний	1	$y = A^* x_1 - x_6$	$y = A^* x_3 + x_2$	$y = A * x_5^2 + x_3^2$	p	$1/(p+1)$
Не парний	2	$y = A^* x_1 + x_6$	$y = A^* x_3 - x_2$	$y = A * x_5^2 + x_3^2$	1/p	3p-1
Для всіх	3	$y = A^* x_1 - x_6$	$y = A^* x_3 + x_2$	$y = A * (x_5^2 + x_3^2)^2$	4p+1	2p

Таблиця 9.4 – Вихідні данні журналу обліку

Варіант	Структурна схема системи
1	
2	
3	

Контрольні питання

1. В чому полягає концепція імітаційного моделювання?
2. Як здійснюється генерування даних з заданим законом розподілу ймовірностей?
3. Як здійснюється генерування даних з заданою кореляційною функцією?
4. Як здійснюється моделювання лінійних динамічних підсистем в пакеті Mathcad?
5. Як здійснюється моделювання лінійних динамічних підсистем в пакеті Scilab/Xcos?
6. Як здійснюється моделювання нелінійних підсистем?

Практична робота №11

Імітаційне моделювання нечітких систем в середовищі Scilab/Xcos.

Мета – провести імітаційне моделювання нечітких систем у середовищі Scilab/Xcos.

Завдання: здійснити моделювання нечіткого обчислення залежності відповідно до варіанта.

Література: [5, 9-10].

Теоретичні відомості

У багатьох сучасних системах (експертних, управління, інформаційних, інтелектуальних тощо), де є необхідність прийняття рішень в умовах непевної інформації, використовується нечітка логіка. В системах управління для цього

використовують нечіткі контролери, проте систему нечіткого висновку можна реалізувати і на звичайному комп'ютері. В зв'язку з поширенням нечітких систем для Scilab/Xcos розроблено спеціальний пакет – Fuzzy Logic Toolbox (FLT).

Висновки в нечітких системах здійснюються на основі бази знань. База знань складається з набору відомих фактів (бази даних) і набору відомих залежностей між ними (правил «якщо…, то…»). Алгоритм прийняття рішень на основі фактів і правил практично не залежить від призначення системи управління. Конкретні особливості системи впливають лише на структуру бази даних і вид правил.

Структура системи нечіткого висновку наведена на рисунку 11.1.

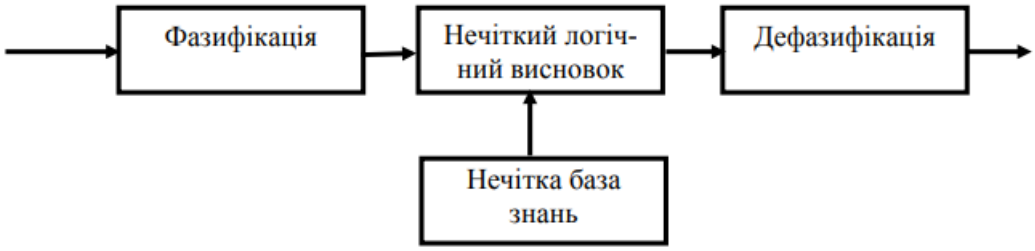


Рисунок 11.1 – Схема нечіткого логічного висновку

Фазифікація – це процес перетворення чіткого або лінгвістичного значення x на нечітке, яке характеризується функцією належності $\mu(x)$.

Для здійснення нечіткого висновку з бази нечітких даних вибираються рядки з однаковими значеннями функції і з них утворюється диз'юнктивна нормальна форма. Дефазифікація полягає у перетворенні вхідної величини в число. Існує багато методів дефазифікації, але найбільш популярним є метод центра ваги

Для отримання нечіткого висновку існують алгоритми Мамдані, Цукамото, Сугено, Ларсена та інші, але найпопулярнішими є алгоритми Мамдані та Такагі-Сугено. Саме ці алгоритми і використовуються в Scilab FLT.

Нечіткі кон'юнкції і диз'юнкції можуть здійснюватися на основі так званих t-норм і s-норм.

Моделювання буде проводитися за допомогою пакета Scilab/Xcos- FLT, в якому можна також моделювати системи з використанням нечіткого контролера.

FLT (Fuzzy Logic Toolbox) – це додаток до Scilab, який складається з інструментів для моделювання та використання нечітких множин.

Fuzzy Logic Toolbox не є вбудованим додатком у Scilab, його потрібно скачати відповідно до версії інструментального засобу та операційної сис- теми вашого персонального комп'ютера.

Послідовні дії встановлення Fuzzy Logic Toolbox:

– завантажити на комп'ютер «sciFLT» з сайту (atoms.scilab.org);
– завантажену папку перенести в «scilab\contrib»;
– відкрити інструментальний засіб Scilab;
– в меню обрати «Інструменти – Управління модулями Atoms», (наведено на рисунку 11.2).
– у вікні «Основні категорії-ATOMS», вибираємо категорію Xcos та завантажуємо Fuzzy Logic Toolbox (рис. 11.3).

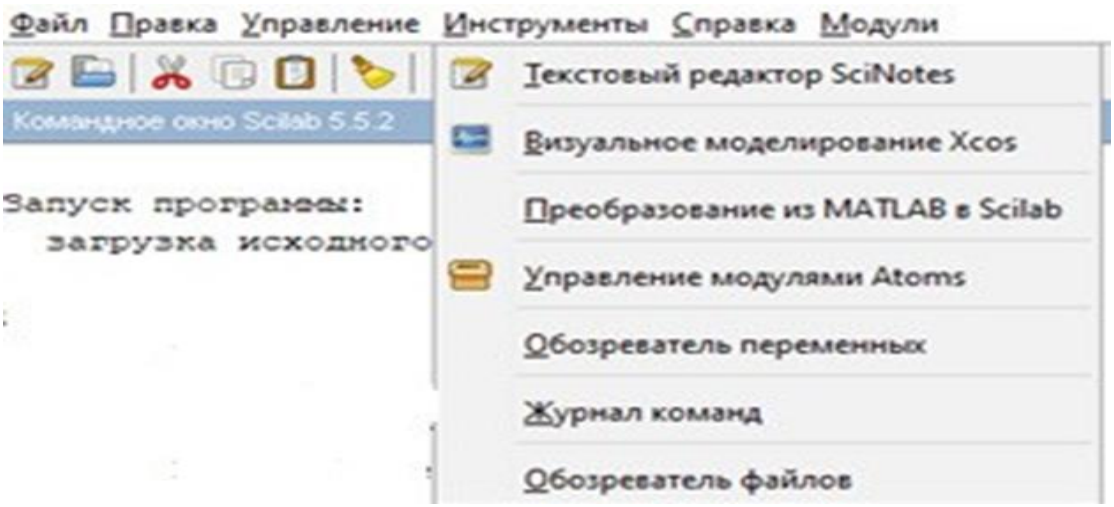


Рисунок 11.2 – Меню «інструменти»

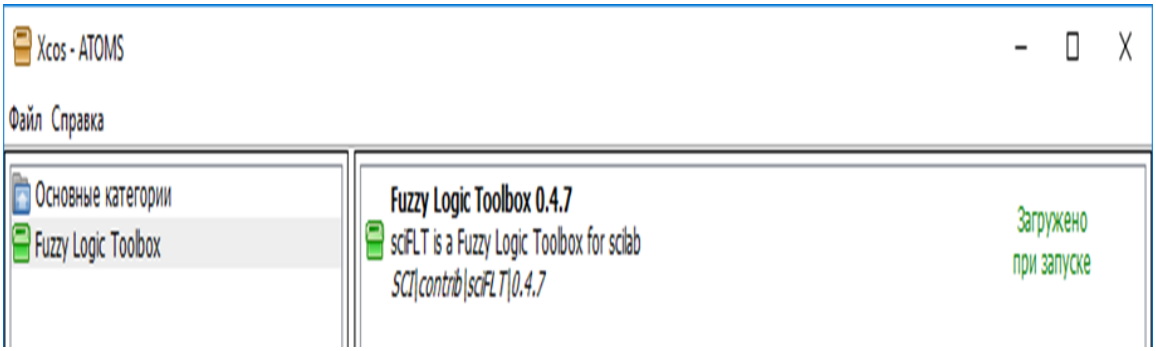


Рисунок 11.3 – Fuzzy Logic Toolbox

– перезавантаження інструментального засобу Scilab та запуск з до- датком нечіткої логіки, який запускається автоматично. Запуск Fuzzy Logic Toolbox, виводить повідомлення в командному вікні Scilab, наведені на рисунку 11.4.

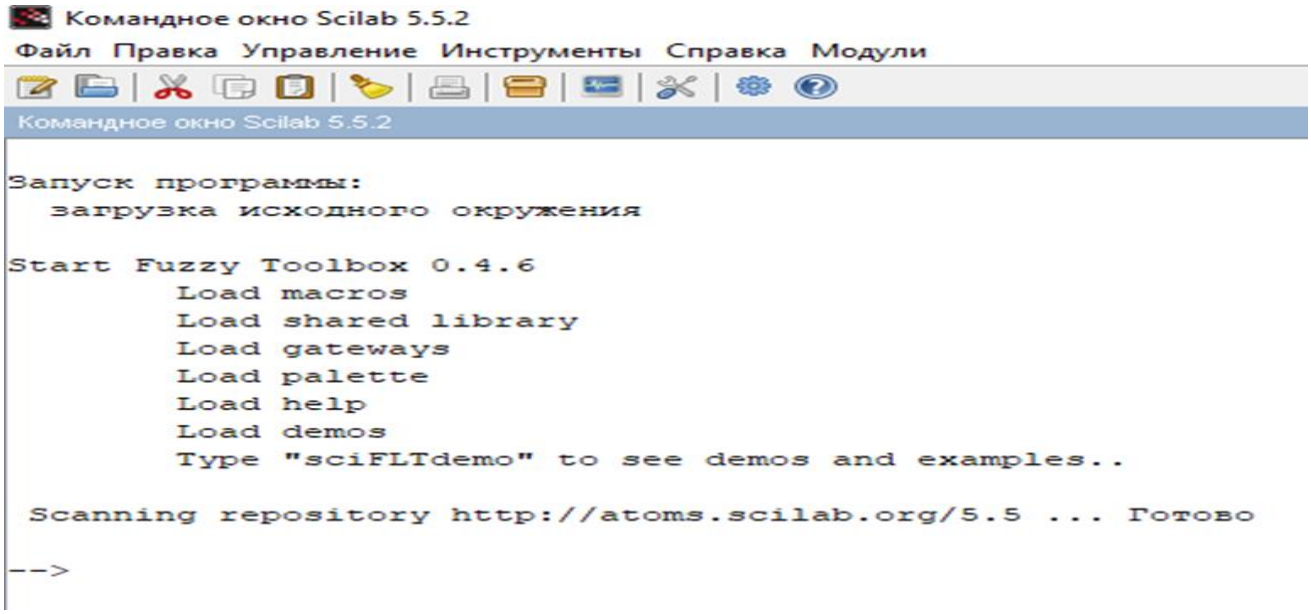


Рисунок 11.4 – Інформація про запуск додатка Fuzzy Logic Toolbox

Додаток складається з блоків нечіткої логіки для моделювання систем. Щоб їх використовувати, потрібно у вікні палітри блоків обрати групу блоків «sciFLT»(рис. 11.4., табл. 11.1)

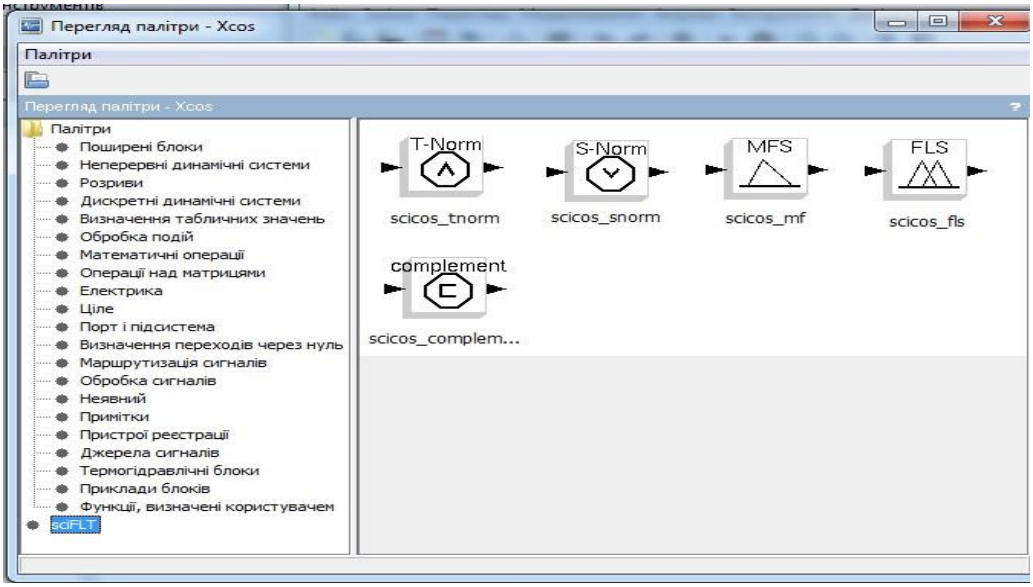
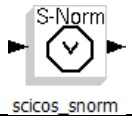


Рисунок 11.4 – Палітра нечіткого моделювання

Таблиця 11.1 – Опис блоків нечіткого моделювання

Блок	Опис
	scicos_tnorm – обчислення нечіткого класу T-Norm за алгоритмами: Dubois, Yager, Drastic Product, Einstein Product, Algebraic Product, Minimum. Користувач також може вибрати для обчислення клас S-Norm для цього встановити параметри
	scicos_snorm – обчислення, використовуючи клас S-Norm, за алгоритмами: Dubois, Yager, Drastic Sum, Einstein Sum, Algebraic Sum, Minimum.
	scicos_mf – обчислює функції належності Triangular, Trapezoidal, Gaussian, Extended Gaussian, Generalized Bell, Sigmoidal, Product of two Sigmoidal, Absolute difference of tho Sigmoidal, S-Shaped, Z-Shaped, Pi-Shaped.
	scicos_fls – алгоритм обирається в редакторі
	scicos_complement – обчислює нечіткий клас доповнення: One, Yager, Sugeno

Методика імітаційного моделювання систем з нечіткими контролерами в середовищі SCILAB/XCOS/FLT

Найпростіше здійснювати моделювання за допомогою блока scicos_fls, який налаштовується редактором editf.ls. За допомогою редактора можна налаштувати Scilab на моделювання системи двох типів:

– імітаційне моделювання систем з нечітким контролером за алгоритмом Мамдані;

– імітаційне моделювання систем з нечітким контролером за алгоритмом Сугено.

Щоб розпочати моделювання з використанням нечіткого контролера в середовищі Scilab, потрібно в командному вікні програми ввести команду «editfls», яка відкриє графічний редактор системи нечіткого виведення (рис. 11.5).

Рисунок 11.5 – Графічний редактор системи нечіткого виведення

У меню редактора FLS для вибору методу моделювання потрібно натиснути кнопку «Файл – Новый FLS» і обрати метод нечіткого висновку для моделювання Мамдані чи Такагі-Сугено.

Розглянемо моделювання системи з нечітким висновком Мамдані за допомогою редактора FLS (рис. 11.6).

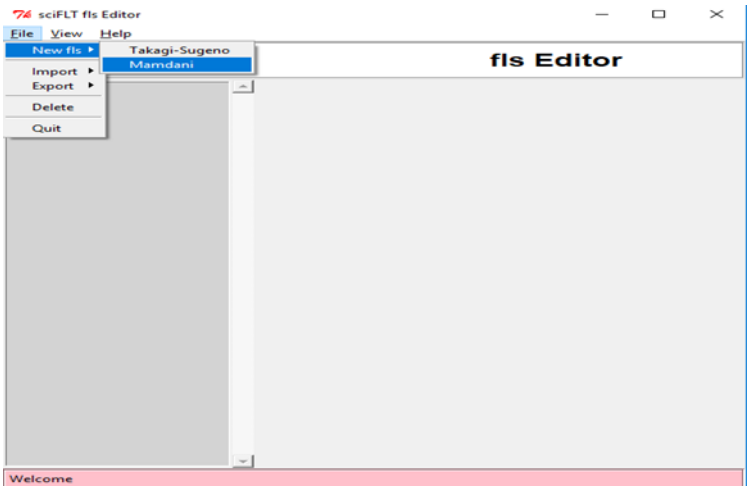


Рисунок 11.6 – Вибір нечіткого моделювання за допомогою нечіткого висновку Мамдані

У вікні редактора натискаємо «Description» та заповнюємо поля (рис. 11.7):

– Name – задаємо ім’я файлу налаштування;

– Comment – опис файлу (коментар щодо конкретної задачі, наприклад: «Лабораторна робота № 11»);

– Type – тип нечіткого виведення буде автоматично зазначено (в даному випадку Мамдані);

– s-norm class – спосіб виконання операції нечіткої диз’юнкції: Dubois-Prade, Yager, Drastic sum, Einstein sum, Algebraic sum, Maximum;

– t-norm class – спосіб виконання операції нечіткої кон’юнкції: Dubois-Prade, Yager, Drastic sum, Einstein sum, Algebraic sum, Minimum;

– Complement – додаткові можливості оброблення параметрів: One, Yager, Sugeno. Звичайно обирають One та задають кількість параметрів;

– implication method – метод висновку: Minimum, Product (стандартно в редакторі обрано), Einstein product;

– aggregation method: Maximum (стандартно в редакторі обрано), Sum, Prob. OR, Einstein sum;

–defuzzification method: Centroide (стандартно в редакторі обрано),Bisector.

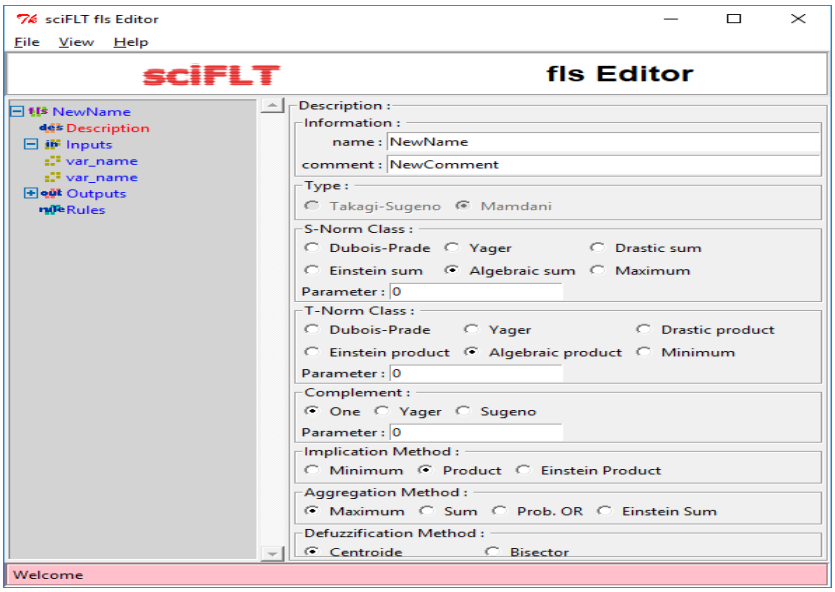


Рисунок 11.7 – Вікно налаштування контролера

Визначаємо вхідні змінні в редакторі, обравши пункт меню «Inputs» (рис. 11.8, а), та вихідні змінні – пункт меню «Outputs» (рис. 11.8 б) та заповнюємо поля:

– number of input variables – кількість доданих вхідних/вихідних змінних за допомого клавіши «Add»;

– variables – додані вхідні/вихідні змінні, які можна редагувати, натиснувши клавішу «Edit» та видаляти – «Delete».

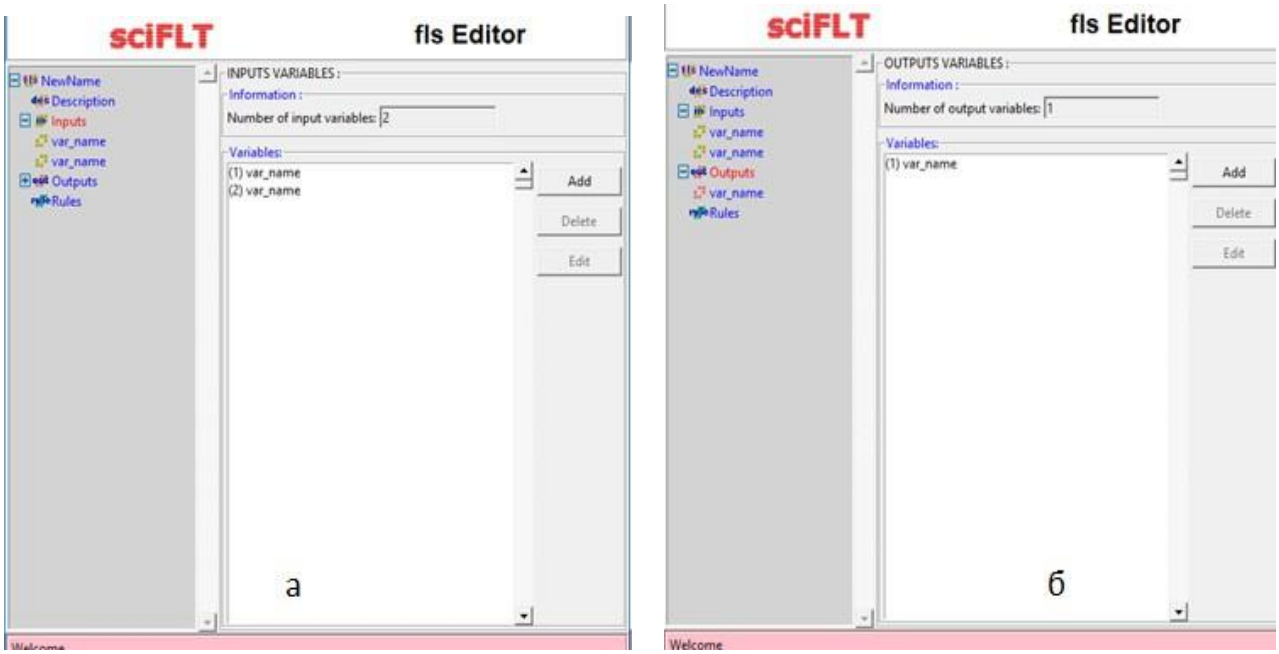
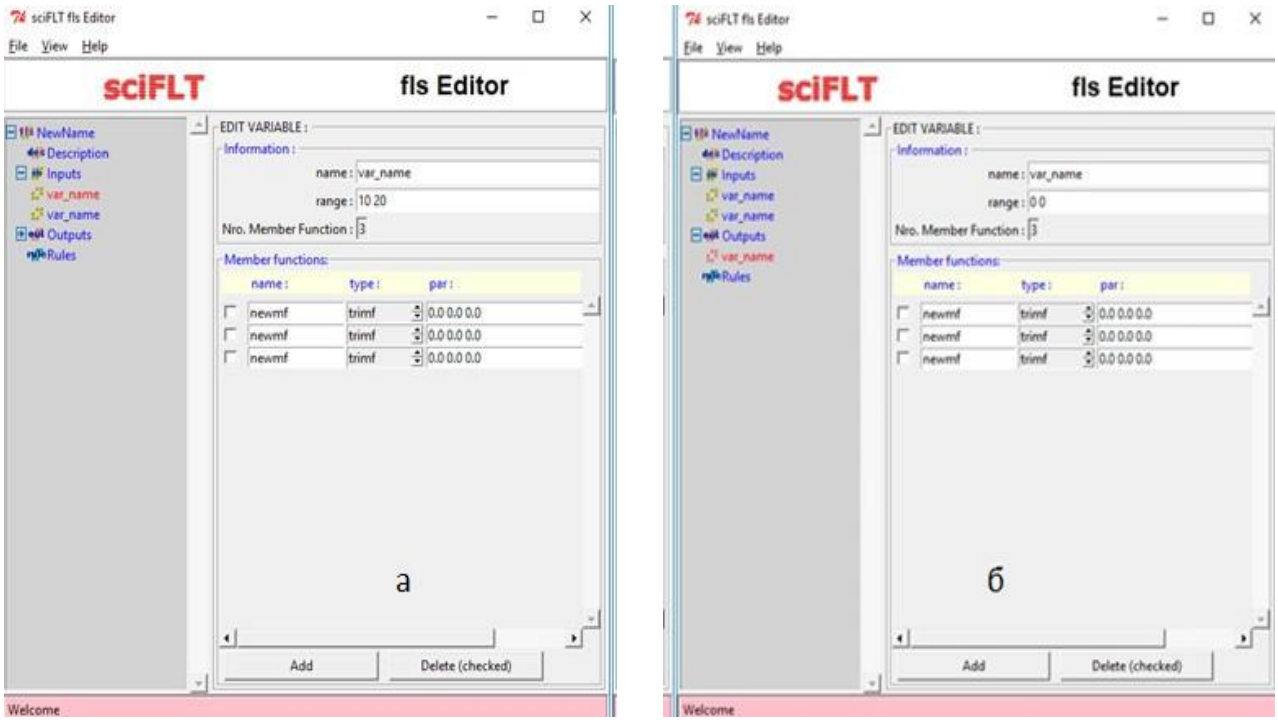


Рисунок 11.8 – Змінні: а) вхідні; б) вихідні

Натиснувши клавішу «Edit», відкриється редактор вхідних/вихідних змінних (рис. 11.9 а, б), в якому є поля:

– name – ім’я вхідної/вихідної змінної;
– range – діапазон вхідної/вихідної змінної, nro;



а) б)

Рисунок 11.9 – Редагування а) вхідних змінних; б) вихідних змінних

– Member function – параметри функції належності вхідної/вихідної змінної;

– name – ім’я параметра вхідної/вихідної змінної;

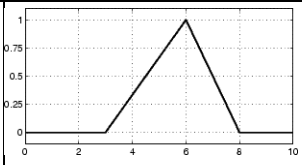
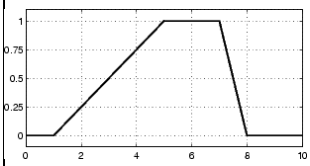
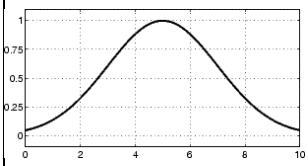
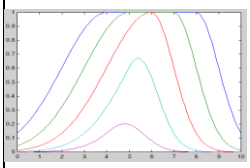
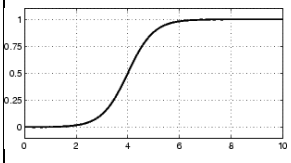
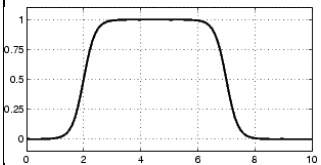
– type – обирається тип параметра;

– par – значення параметра можуть бути задані за допомогою числових або лінгвістичних значень.

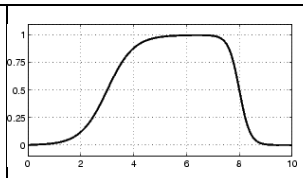
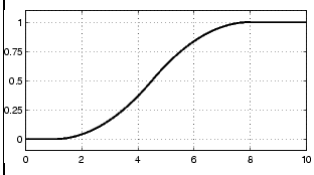
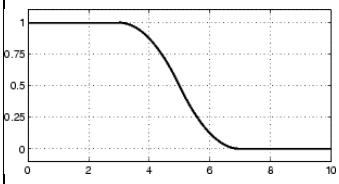
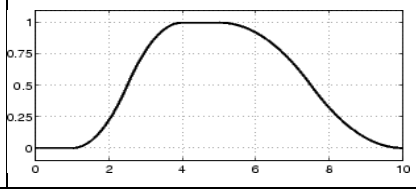
Для того, щоб додати параметри для вхідної/вихідної змінної, натискаємо клавішу «Edit», яка створиться у меню «Member function», а також можна видаляти – «Delete».

У пакеті передбачені такі типи функцій належності термів вхідних і вихідних змінних (таблиці 11.2).

Таблиця 11.2 – Типи функцій належності термів вхідних і вихідних змінних

Назва	Графік	Опис
Trimf		$f(x;a,b,c)=\begin{cases}0, & x\le a\\ \frac{x-a}{b-a}, & a\le x\le b\\ \frac{c-x}{c-b}, & b\le x\le c\\ 0, & c\le x\end{cases}$
Trapmf		$f(x;a,b,c,d)=\begin{cases}0, & x\le a\\ \frac{x-a}{b-a}, & a\le x\le b\\ 1, & b\le x\le c\\ \frac{c-x}{c-b}, & c\le x\le d\\ 0, & d\le x\end{cases}$
gaussmf		$f(x;\sigma,c)=e^{e^{-\frac{(x-c)^2}{2\sigma^2}}}$
gauss2mf		комбінація двох гаусових функцій
Sigmf		$(x;a;c)=\frac{1}{1+e^{-a(x-c)}}$
dsigmf		різниця двох сигмоїдних кривих

Продовж. табл.11.2.

psigmf		добуток двох сигмоїдних кривих
Smf		крива сплайна s-форми
Zmf		крива сплайна z-форми
Pimf		крива сплайна П-форми

Після задання вхідних та вихідних змінних переходимо до пункту меню «Rules», в якому створюється нечітка база правил за алгоритмом Мамдані, відповідне вікно показано на рисунку 11.10. В редакторі можна додавати правила, натиснувши клавішу «Add rule», видаляти правила – «Delete rule» та змінювати правила – «Change rule».

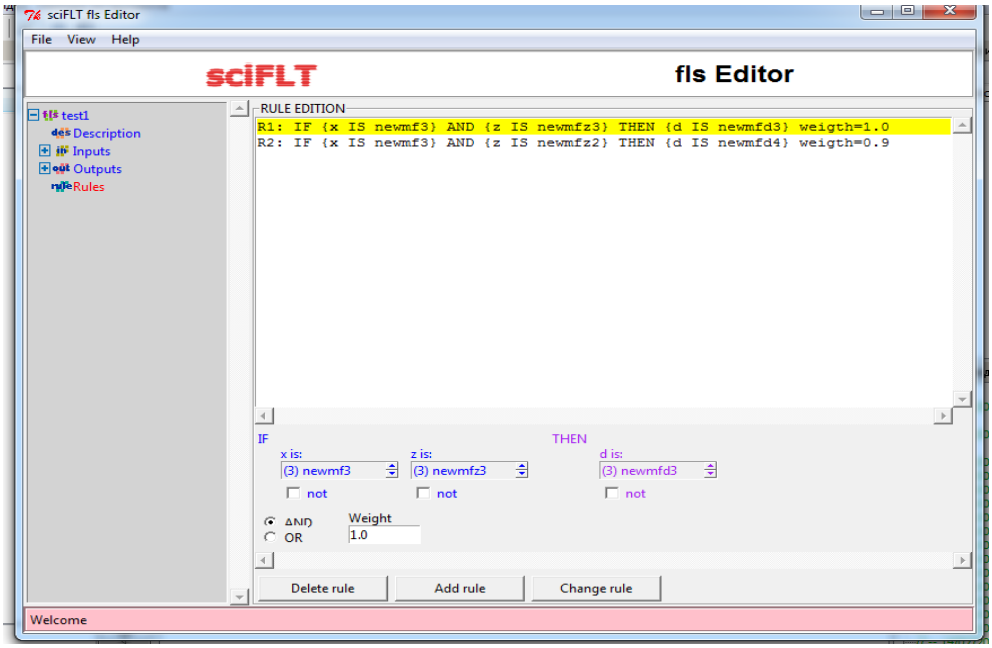


Рисунок 11.10 – Редактор правил

Нечіткі правила за алгоримом Мамдані мають нижченаведений вигляд. ПРАВИЛО_1: IF «умова», THEN «ім'я функції належності результата» weigth=»вага правила.

Вагові коефіцієнти відповідних правил можуть приймати значення з інтервалу [0, 1]. У разі, якщо ці вагові коефіцієнти відсутні, зручно прийняти, що їх значення 1.

Для створення правил за допомого редактора потрібно вибрати відповідну комбінацію термів і залежностей, вибрати тип зв'язку: or або and, вагу правила Weight, значення вихідної змінної та натиснути кнопку Add rule.

Метод імітаційного моделювання систем з нечітким контролером за алгоритмом Сугено за допомогою редактора FLS виконується аналогічно алгоритму Мамдані, відмінність полягає у тому, що визначається одразу чітке значення результату.

Створені нечіткі бази правил за обраним методом на диску персонального комп'ютера (Файл – Експорт – в файл fls).

Нечіткі бази правил, які створені за допомогою FLS-редактора, використовуються при візуальному моделюванні різних систем в редакторі Xcos. Для цього потрібно:

- запустити вікно Xcos;
- у вікні «Палітри», обрати палітру блоків «sciFLT»;
- з даної палітри блоків обрати нечіткий блок, який Вам потрібний для візуального моделювання певної системи, та перетягнути його у вікно моделі Xcos;
- натискаємо двічі лівою клавішою миші на нечіткому блоці. Відкривається вікно «sciFLT Scicos», за допомогою якого ми імпортуємо створений файл з нечіткою базою правил (import from file – відкриваємо файл – Ok);
- запускаємо створену імітаційну модель системи з нечітким контролером.

Завдання

1. Ознайомитися з теоретичними основами моделей статистики [2], документацією пакета Scilab/Xcos−TLT та іншими джерелами.
2. Підготувати відповіді на контрольні питання.
3. Здійснити моделювання нечіткого обчислення залежності відповідно до варіанта в таблиці 11.3:

$$y = ax^2_1 + bx^2_2 + c\,x^2_3 .$$

База правил має містити 10 правил для $x_1, x_2, x_3 \in \{1,2,3\}$.

Розрахувати у для вхідних даних відповідно до варіанта трьома способами:

А) вручну прямим підставлянням значень в початкову формулу;

Б) вручну за алгоритмом Мамдані;

В) налаштувати контролер scicos_fls і подати на входи задані значення X. Порівняти результати.

4. Розробити імітаційну модель системи управління з одиничним зворотним зв'язком і аперіодичним об'єктом з передатною функцією:

$$W(s)=\frac{2}{3s+1}$$

і пропорційно-інтегрально-диференціальним законом управління:

$$u=a(x-y)+b\int_0^t(x-y)dt+c\frac{d(x-y)}{dt}$$

Отримати графік перехідної характеристики.

Таблиця 11.3 – Вихідні данні по журналу обліку

Варіант	Параметри залежності			Вхідні дані		
	a	b	C	x_{10}	x_{20}	x_{30}
1	-1	2	3	3.5	-1	2
2	2	-1	3	3	2.5	-1
3	3	2	-1	-1	3	2.5
4	-2	3	1	1	-2.3	3
5	2	-2	1	1.3	2	-2
6	3	1	-3	-3	3.7	1
7	1	-3	2	2	1	-3.7
8	-2	2	3	3.5	-1	3
9	3	-1	3	3	2.5	-2
10	4	2	-1	-1	3	2.8
11	-3	3	1	1	-2.3	4
12	3	-2	1	1.3	2	-2
13	4	1	-3	-3	3.7	2
14	2	-3	2	2	1	-3.0
15	-2	2	3	3.5	-1	3
16	3	-1	3	3	2.5	-2
17	4	2	-1	-1	3	3.5
18	-3	3	1	1	-2.3	4
19	3	-2	1	1.3	2	-3
20	4	1	-3	-3	3.7	2

Контрольні питання

1. Що таке база знань та які основні особливості її структури порівняно з традиційними базами даних?

2. У чому полягає сутність правил нечіткого логічного висновку та як вони застосовуються в нечітких експертних системах?

3. Які існують види t-норм і s-норм у нечіткій логіці та для чого вони використовуються?

4. Які види функцій належності застосовуються в нечітких системах та які їх основні характеристики?

5. У чому полягає алгоритм нечіткого логічного висновку Мамдані та які основні етапи його реалізації?

Практична робота №12–13

Імітаційне моделювання систем масового обслуговування (СМО) в середовищі Scilab/Xcos.

Мета – створити СМО у середовищі Scilab/Xcos.

Завдання: побудувати модель одноканальної СМО. Провести імітацію та отримати характеристики функціонування системи.

Література: [1, 5, 9 -10].

Теоретичні відомості

Розглянемо просту систему масового обслуговування на рисунку 12.1 з вхідним потоком Пуассона та експоненціальним часом обслуговування. Це буде моделювати чергу біля каси в магазині. Клієнти підходять до каси згідно з законом Пуассона, і час, який витрачає кожен клієнт на розрахунок, описується експоненціальним законом.

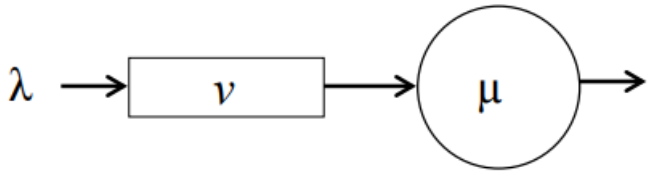


Рисунок 12.1 – Система масового обслуговування

Для генерування потоку запитів використовуємо автогенератор подій (показано на рисунку 12.2). Генерування експоненціально розподілених значень здійснюється на основі перетворення рівномірного розподілу на експоненціальний із застосуванням оберненого нелінійного перетворення (форм.12.1):

$$f(y)=\frac{d\left[N^{-1}(y)\right]}{dy}\cdot f_x\left[N^{-1}(y)\right], \qquad (12.1)$$

Блок генератора випадкових чисел Random generator видає рівномірно розподілене випадкове число від 0 до 1. Взяття логарифму (з палітри математичних операцій) і множення на (-1/ λ) (використовуємо блок Gain)

перетворює закон розподілу на експоненціальний з параметром λ. Подаючи вихід блока затримки події (з палітри обробки подій) до його порту активації введення, генеруємо послідовність подій, причому час між двома подіями є незалежною випадковою величиною з експоненціальним законом. Результатом є процес Пуассона з параметром λ. Блок початкової активації Event at time 0 починає процес. Блок «плюс (+)» являє собою блок об'єднання потоків подій. На рисунку 12.2 також наведено вікно задання змінного параметра – інтенсивності потоку запитів – і графік роботи.

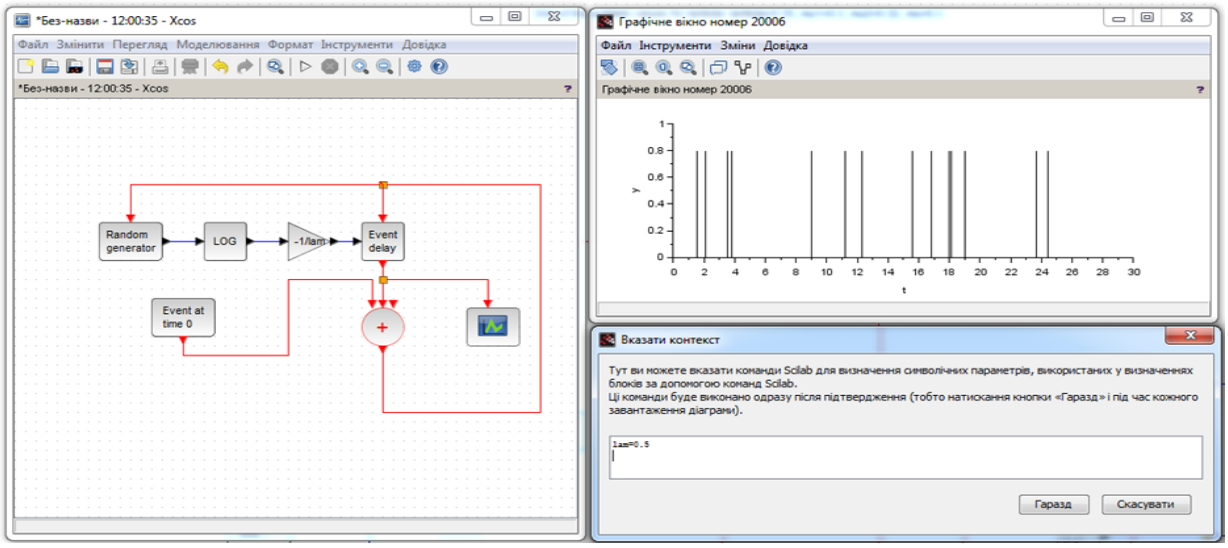


Рисунок 12.2 – Генератор потоку подій

Для використання в складних моделях СМО генератор потоку запитів зручно перетворити на суперблок, як на рисунку 12.3.

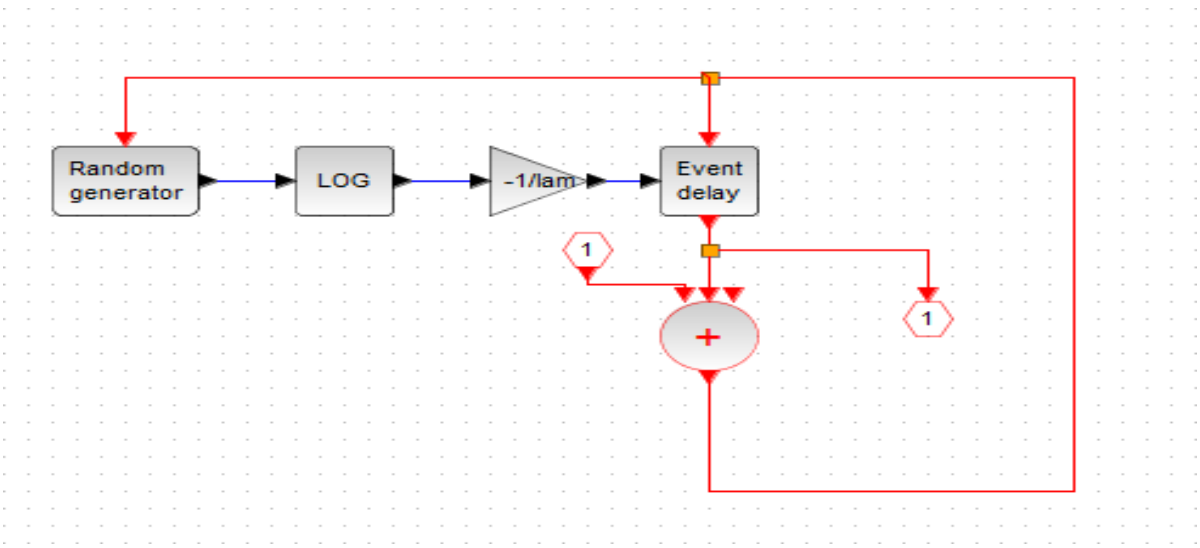


Рисунок 12.3 – Суперблок генерування потоку запитів

Модель черги є складнішою. Черга працює в двох режимах: збільшення черги при надходженні запиту і зменшення при обслуговуванні запиту, проте, якщо черга

пуста, то зменшення розміру черги не відбувається (черга не може бути від'ємною). Модель черги показана на рисунку 12.4. У моделі вхідний порт «1» – порт надходження запиту, порт «2» – порт сигналу переходу запиту на обслуговування, вихідний порт «1» – довжина черги. Стан черги зберігається в блоці 1/z. Його значення збільшується або зменшується на одиницю, залежно від події.

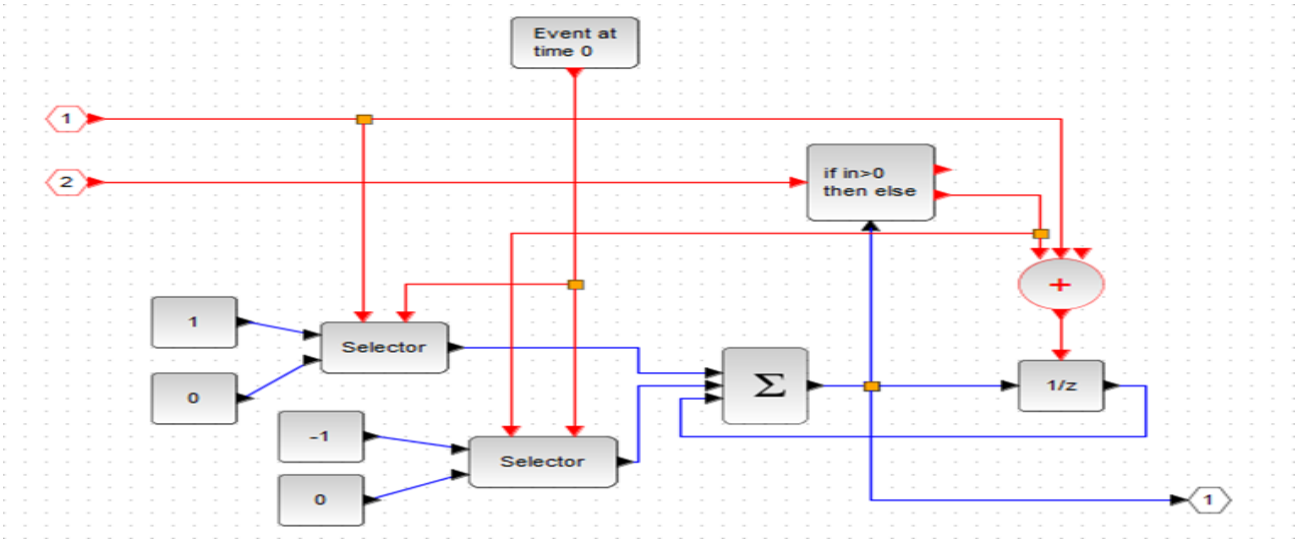


Рисунок 12.4 – Модель черги

Обслуговування запитів залежить від стану черги: пуста чи непуста. Поки черга непуста, час між двома подіями є незалежною випадковою змінною. Але коли черга порожня, не слід створювати події для виходу. Цей процес має бути неактивним. Для реалізації цього процесу ми маємо дозволити перезапуск процесу зовнішньою подією (прихід клієнта в порожню чергу). Тому модель подій відправлення (рис. 12.5) має два входи активації. Один призначений для ініціалізації, а інший – для перезапуску.

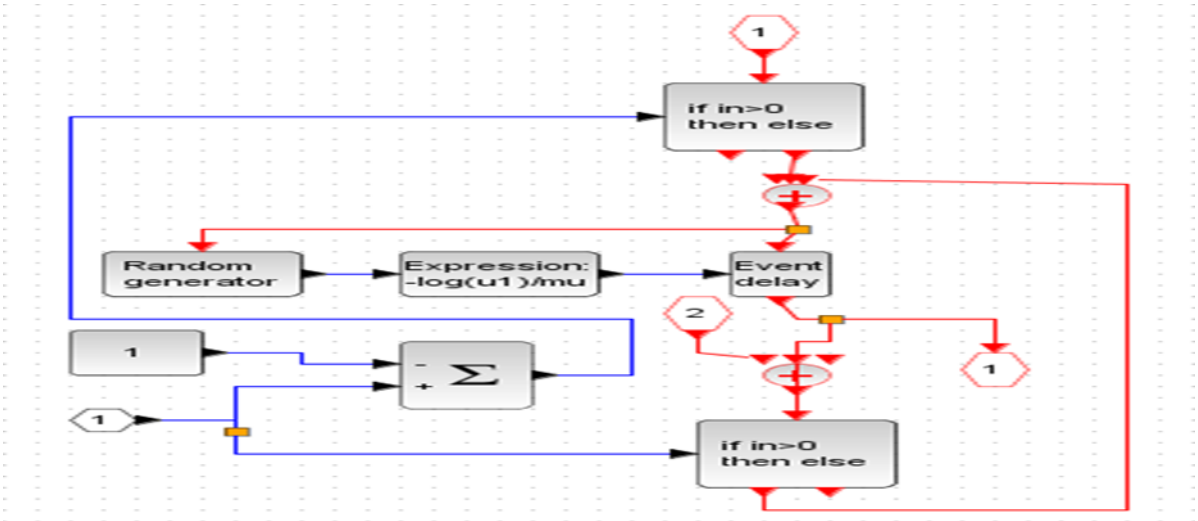


Рисунок 12.5 – Модель процесу обслуговування

Для моделювання експоненціального закону розподілу часу обслуговування виконується нелінійне перетворення (-log (u1)/ μ), що приводить до

експоненціального закону з параметром μ. Блок If-Then-Else знаходиться з палітри обробки подій.

Повна імітаційна модель СМО, зображеної на рисунку 12.1, наведена на рисунку 12.6. Параметри потоків визначаються в контексті.

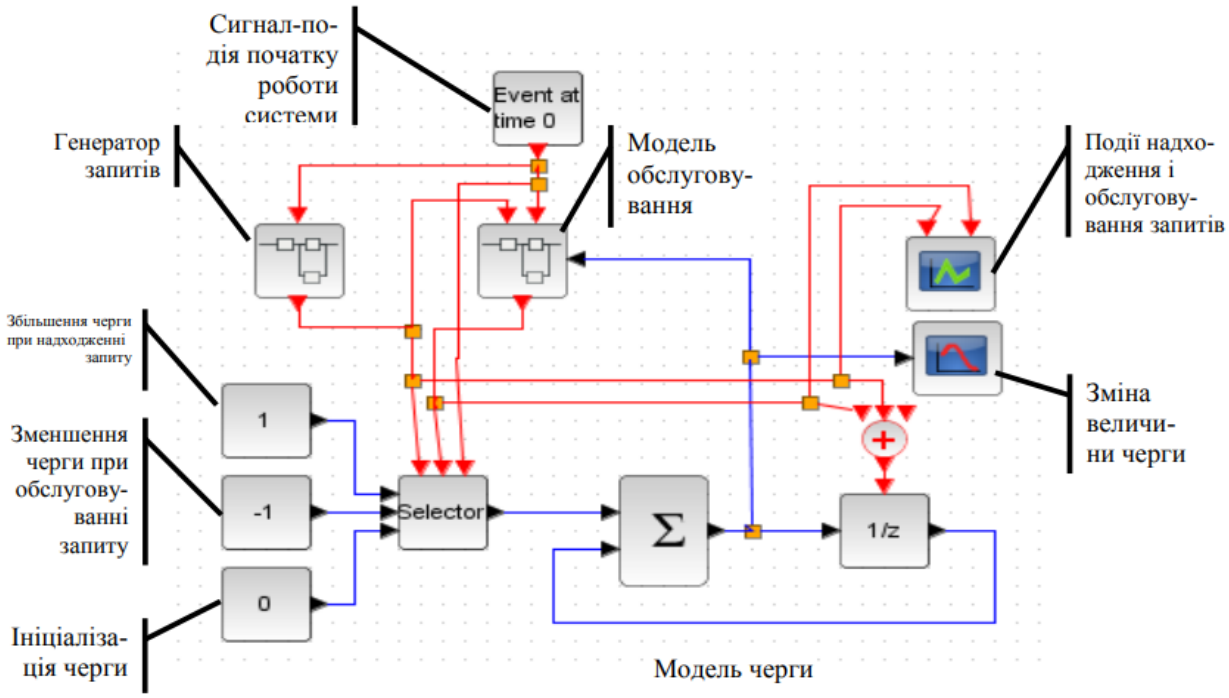


Рисунок 12.6 – Загальна модель одноканальної однофазної СМО з необмеженою чергою

Результат моделювання (еволюцію черги) наведено на рисунку 12.7, на рисунку 12.8 – події (темні лінії відповідають подіям надходження запитів).

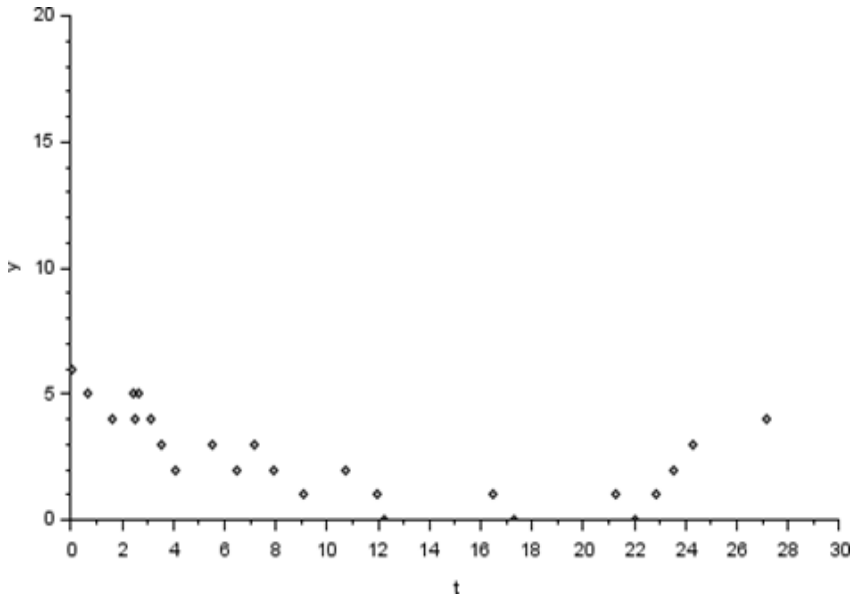


Рисунок 12.7 – Еволюція черги

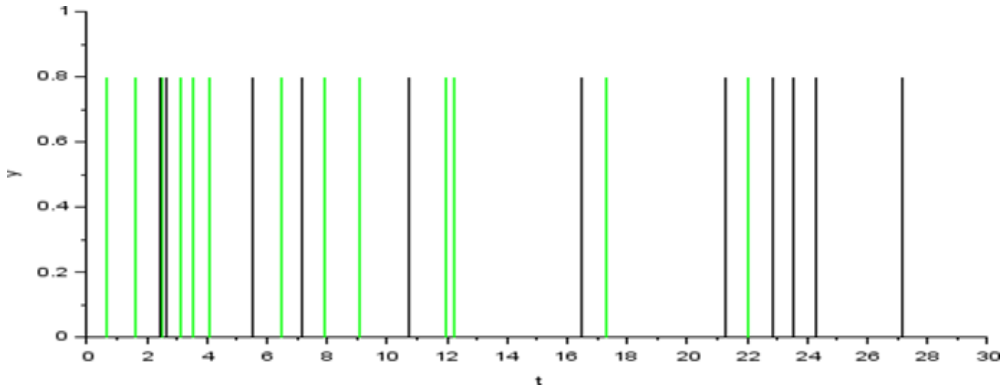


Рисунок 12.8 – Події запитів і обслуговування

Складніші системи утворюються комбінацією простих. На рисунку1 2 .9 показано приклад моделі двофазної СМО з двома каналами у першій фазі.

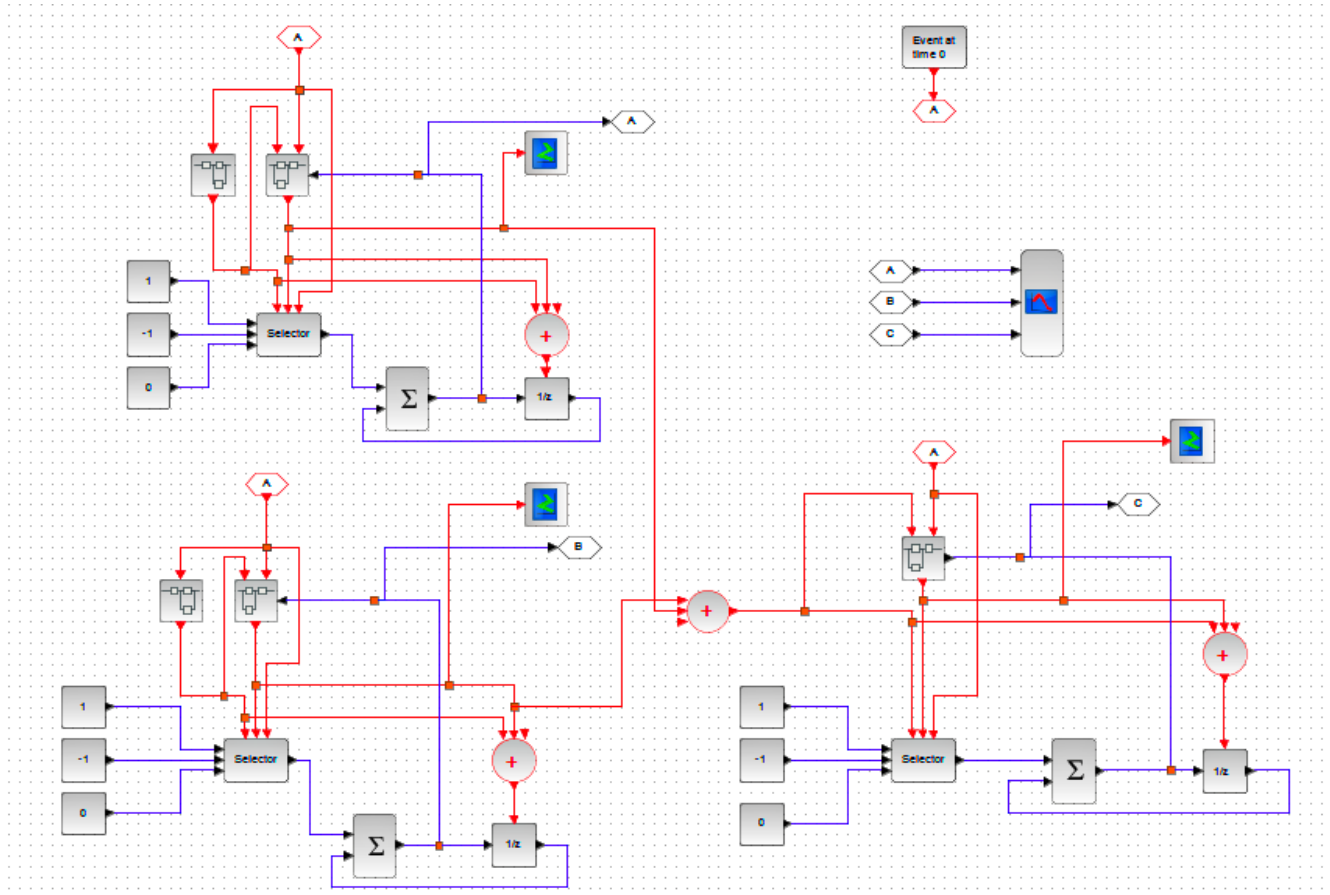
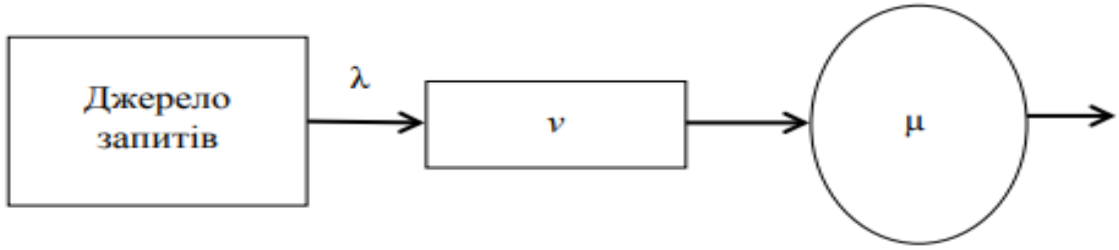


Рисунок 12.9 – Модель двофазної СМО з двома каналами у першій фазі

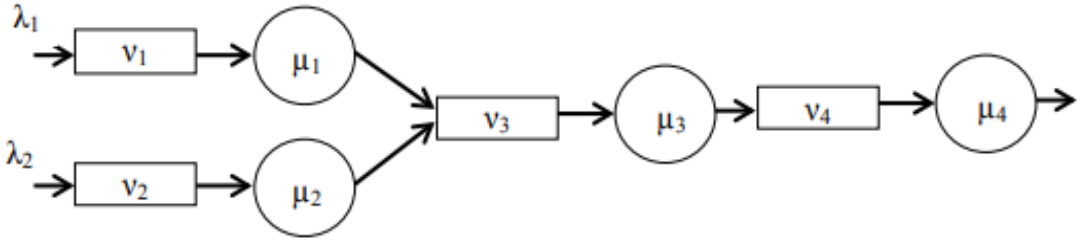
Завдання

1. Створити у Scilab/Xcos моделі, які наведені у теоретичних відомостях, і дослідити їх.

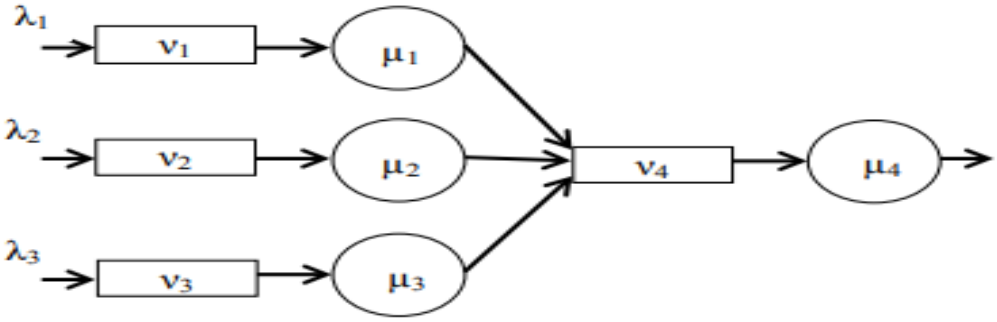
2. З декількох моделей найпростішої СМО з чергою, яка наведена у теоретичній частині, скласти модель складнішої системи (відповідно до варіанта) і дослідити її при різних інтенсивностях потоків запитів і обслуговування.



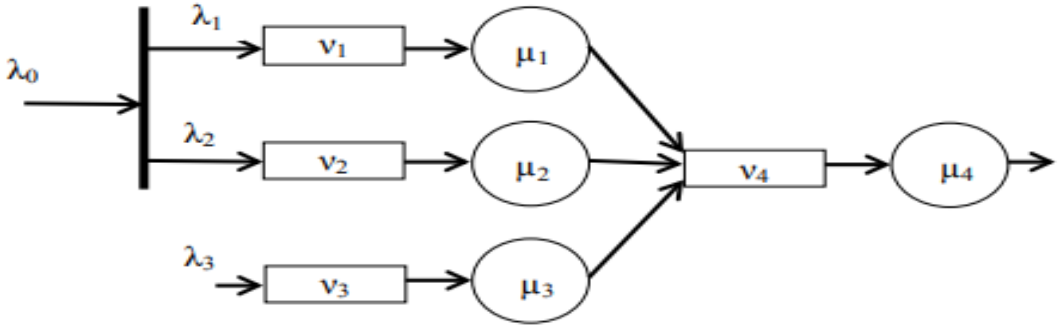
Завдання по варіантам з журналу обліку.
Номер варіанту: 1.6.11.16.



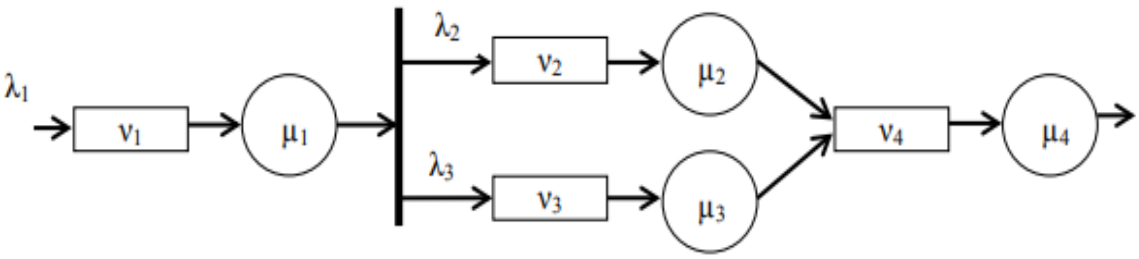
Номер варіанту: 2.7.12.17.



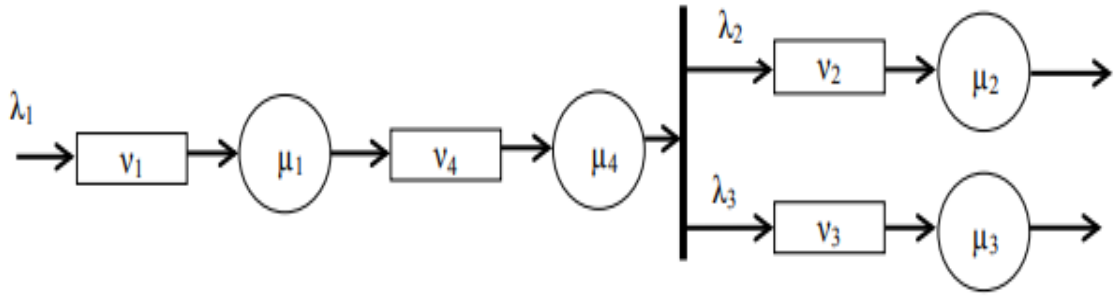
Номер варіанту: 3.8.13.18.)



Номер варіанту: 4.9.14.19.



Номер варіанту: 5.10.15.20.



3. Провести імітаційне моделювання та розрахувати характеристики системи (таблиця 12.1)

Таблиця 12.1 –Вихідні дані по варіантам (одиниці: заявки/хв або хвилини)

Варіант	λ_1(заяв/хв)	λ_2	μ_1	μ_2	μ_3	μ_4
1	0.5	0.8	0.60	0.70	0.80	0.90
2	0.6	0.9	0.65	0.75	0.85	0.95
3	0.55	0.85	0.58	0.68	0.78	0.88
4	0.52	0.82	0.57	0.67	0.77	0.87
5	0.58	0.88	0.59	0.69	0.79	0.89
6	0.6	0.9	0.63	0.73	0.83	0.93
7	0.53	0.83	0.55	0.65	0.75	0.85
8	0.57	0.87	0.60	0.70	0.80	0.90
9	0.51	0.81	0.56	0.66	0.76	0.86
10	0.59	0.89	0.61	0.71	0.81	0.91
11	0.5	0.8	0.57	0.67	0.77	0.87
12	0.6	0.9	0.62	0.72	0.82	0.92
13	0.55	0.85	0.59	0.69	0.79	0.89
14	0.52	0.82	0.58	0.68	0.78	0.88
15	0.58	0.88	0.60	0.70	0.80	0.90
16	0.6	0.9	0.63	0.73	0.83	0.93
17	0.53	0.83	0.55	0.65	0.75	0.85
18	0.57	0.87	0.60	0.70	0.80	0.90
19	0.51	0.81	0.56	0.66	0.76	0.86
20	0.59	0.89	0.61	0.71	0.81	0.91

Для всіх варіантів: початкова довжина черги = 0, початковий стан каналів = всі вільні, запис подій увімкнений.

Умовні позначення: λ – інтенсивність вхідного потоку (заявок/хв), μ – інтенсивність обслуговування у фазі N для кожного каналу (заявок/хв), V_1 – кількість каналів у фазі 1, μ_2 – інтенсивність у фазі 2, s_2 – кількість каналів у фазі 2.

Рекомендовані варіанти експериментів (варіювати по черзі)

1. Змінити λ: 0.15; 0.20; 0.25; 0.30.
2. Змінити μ: 0.20; 0.25; 0.30.
3. Збільшити s1s від 1 до 3 (замість 2) і порівняти.

4. Короткий прогін: Tsim=120 хв. VS довгий: Tsim=1440 хв (перевірити стабільність оцінок)

Порядок виконання роботи (по кроках).

1. Відкрити Scilab/Xcos, створити нову модель.

2. Реалізувати генератор пуассонівського потоку: блок Random generator → трансформація.

3. Побудувати модель черги: блок збереження стану 1/z, логіка збільшення/зменшення при подіях.

4. Реалізувати процес обслуговування для кожного каналу: трансформація , механізм перезапуску при порожній черзі.

5. Встановити контекстні змінні.

6. Запустити симуляцію для базових параметрів, записати журнали подій і збудувати графіки.

7. Отримати симуляційні результати; проаналізувати розбіжності.

8. Виконати рекомендовані варіації експериментів, оформити таблиці й графіки.

9. Оформити звіт за стандартною структурою: ціль → модель → вихідні дані → результати → висновки.

Контрольні питання

1. Які види систем масового обслуговування (СМО) існують та за якими ознаками вони класифікуються?
2. Наведіть приклади систем масового обслуговування з реального життя та поясніть принцип їх роботи.
3. Як побудувати модель системи масового обслуговування з обмеженою чергою у вигляді зваженого графа станів системи?

Практична робота №14 –15

Аналітичне моделювання системи із застосуванням СМО.

Мета – формування навичок розрахунку характеристик СМО аналітичними методами.

Завдання: Виконати аналітичне дослідження заданої СМО. Визначити середній час очікування, довжину черги та інші параметри. Порівняти аналітичні результати з імітаційними.

Література: [3-8].

Теоретичні відомості

Системи масового обслуговування (СМО) – це такі системи, в які у випадкові моменти часу надходять заявки (запити, вимоги) на обслуговування, при цьому заявки, що надійшли, обслуговуються за допомогою наявних у розпорядженні системи каналів обслуговування.

Теорія масового обслуговування розглядає і дозволяє розрахувати

основні характеристики СМО:

- інтенсивність вхідного потоку λ;
- інтенсивність обслуговування μ;
- середня кількість запитів у системі $\bar{n}$;
- середня кількість запитів у черзі $\bar{\upsilon}$;
- середня кількість запитів, що обслуговуються у системі $\bar{j}$;
- кількість каналів обслуговування s ;
- середня кількість вільних каналів $\bar{r}$;
- середній час очікування у черзі $\bar{\tau}$;
- середня тривалість перебування запиту у системі $\bar{T}$.

Для пуасонівського вхідного потоку запитів і експоненціального закону розподілу часу обслуговування для одно канальної однофазної системи з необмеженою чергою:

- завантаження системи

$$\psi = \frac{\lambda}{\mu}.$$

- Середня кількість запитів у системи

$$\bar{n} = \frac{\psi}{1-\psi}.$$

- Середня кількість запитів у черзі

$$\bar{v} = \frac{\psi^2}{1-\psi}.$$

- Середній час перебування запиту в системі

$$\bar{T} = \frac{1}{\mu-\lambda}.$$

(перевірка через $T = \frac{n}{\lambda}$ – розбіжність через округлення)

- Середній час очікування в черзі

$$\bar{\tau} = \frac{\psi}{\mu(1-\psi)}$$

або простіше:

$$\bar{\tau} = T - \frac{1}{\mu}.$$

Багатоканальна система з однаковими каналами (інтенсивності обслуговування $\forall \mu_i = \mu$) та з різними каналами (для $S = 2$, причому $\mu_1 \neq \mu_2$) описується іншими фізичними законами.

Для інших типів вхідних потоків і законів обслуговування виконати розрахунки важко, тому зручно використовувати імітаційне моделювання в SciLab/xcos з застосуванням механізму подій.

Використати у Scilab/Xcos моделі, які наведені у теоретичних відомостях попередньої роботи №12-13

Завдання

1. Використати для розрахунку систему, створену в попередній роботі (практичне заняття 12-13).
2. Розрахувати характеристики системи аналітично при різних інтенсивностях потоків запитів і обслуговування.
3. Провести порівняння результатів імітаційного моделювання (з попередньої роботи) та аналітичні значення розрахунків при різних інтенсивностях потоків запитів і обслуговування.

Контрольні питання

1. Як формується пуассонівський потік заявок та які його основні властивості?
2. Яким чином утворюються потоки Ерланга та в яких випадках вони застосовуються?
3. Що називають марковським процесом та які його основні характеристики?
4. Яку умову повинно задовольняти співвідношення інтенсивності вхідного потоку заявок λ та інтенсивності обслуговування μ у реальній системі масового обслуговування?

СПИСОК РЕКОМЕНДОВАНИХ ДЖЕРЕЛ

1. Цибко Г. Ю., Горошко Ю. В. Основи системного аналізу : конспект лекцій : метод. рек. з дисципліни «Системний аналіз». Чернігів : НУЧК, 2025. 117 с.
2. Горбань О. М., Бахрушин В. Є. Основи теорії систем і системного аналізу. Запоріжжя : ГУ «ЗІДМУ», 2020. 204 с.
3. Клен К. С. Методи моделювання інформаційних систем : конспект лекцій : навч. посіб. Київ : КПІ ім. І. Сікорського, 2023. 193 с.
4. Олевський В. І., Олевська Ю. Б., Соколова Н. О. Моделювання інформаційних систем : конспект лекцій для здобувачів ступеня бакалавра спец. 126 «Інформаційні системи та технології» / М-во освіти і науки України ; Нац. техн. ун-т «Дніпровська політехніка». Електрон. дані. Дніпро : НТУ «ДП», 2024. 499 с.
5. Чорней Н. Б., Чорней Р. К. Теорія систем і системний аналіз : навч. посіб. Київ : МАУП, 2021. 256 с.
6. Катренко А. В. Системний аналіз : підручник. Львів : Новий світ–2000, 2021. 396 с.
7. Шарапов О. Д., Дербенцев В. Д., Семьонов Д. Є. Системний аналіз : навч.-метод. посіб. для самост. вивч. дисципліни. Київ : КНЕУ, 2020. 154 с.
8. Кисіль Т. М. Моделювання систем : навч. посіб. Хмельницький : ПП Мельник А. А., 2021. 256 с.
9. Muller G. System Modeling and Analysis: A Practical Approach. University of South-Eastern Norway, NISE, 2021. 128 p. URL: www.gaudisite.nl/SystemModelingAndAnalysisBook.pdf. (дата звернення: 20.12.2025).
10. Дубовой В. М., Никитенко О. Д., Юхимчук М. С., Галущак А. В. Моделювання об’єктів і систем : лаб. практикум. Вінниця : ВНТУ, 2021. 157 с.

Системний аналіз та технології моделювання інформаційних систем: методичні вказівки до практичних занять для здобувачів першого (бакалаврського) рівня вищої освіти освітньої програми «Інформаційні системи та технології охорони і безпеки» галузі знань F/12 Інформаційні технології спеціальності F6/126 Інформаційні системи та технології денної та заочної форм навчання / уклад. О.М. Любименко. Луцьк: ЛНТУ, 2026. 88 с.

Комп’ютерний набір та верстка: О.М. Любименко

Редактор: О.М. Любименко

Підп. до друку «___» _____ 2026р.
Формат 60х84/16. Папір офс. Гарнітура Таймс.
Ум. друк. арк. ____. Тираж 10 прим. Зам. ____

Відділ іміджу та промоцій
Луцького національного технічного університету
43018, м. Луцьк, вул. Львівська, 75