МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ СЕМЕНА КУЗНЕЦЯ

МОДЕЛЮВАННЯ СИСТЕМ ТА МЕТОДИ ОПТИМІЗАЦІЙ

Методичні рекомендації до лабораторних робіт
для студентів галузі знань 12 "Інформаційні технології" першого (бакалаврського) рівня

Харків
ХНЕУ ім. С. Кузнеця
2019

УДК 004.94(07.034)
М74

Укладач В. М. Задачин

Затверджено на засіданні кафедри інформаційних систем.
Протокол № 5 від 20.12.2018 р.

Самостійне електронне текстове мережеве видання

Моделювання систем та методи оптимізацій [Електронний ресурс] : методичні рекомендації до лабораторних робіт для студентів галузі знань 12 "Інформаційні технології" першого (бакалаврського) рівня / уклад. В. М. Задачин. – Харків : ХНЕУ ім. С. Кузнеця, 2019. – 218 с.

М74

Запропоновано методичні рекомендації до виконання лабораторних робіт із метою набуття студентами навичок із розв'язання: різноманітних математичних задач чисельними методами; задач безумовної оптимізації; задач лінійного та нелінійного програмування; побудови математичних моделей із використанням пакета R і системи імітаційного моделювання GPSS World.

Рекомендовано для студентів галузі знань 12 "Інформаційні технології" першого (бакалаврського) рівня.

УДК 004.94(07.034)

Загальні положення

Сучасний розвиток науки й обчислювальної техніки характеризується стрімким зростанням рівня використання комп'ютерних моделей як для дослідження поведінки явищ і процесів, що оточують людину, так і для вирішення практичних завдань, пов'язаних з управлінням і прогнозуванням. Методи комп'ютерного моделювання широко застосовуються в усіх сферах людської діяльності – від конструювання моделей технічних, технологічних та організаційних систем до вирішення проблем розвитку людства та всесвіту.

Вивчення дисципліни "Моделювання систем та методи оптимізацій" передбачає набуття теоретичних знань та опанування практичних навичок стосовно основних підходів і принципів побудови моделей та розв'язання математичних задач, що виникають як у ході розроблення моделей, так і під час їх застосування з метою вирішення практичних завдань. Дисципліна спрямована на формування у студентів: основ застосування загальновідомих методологій та сучасних технологій моделювання складних систем; знань чисельних методів і методів оптимізації; практичних навичок роботи в середовищі універсальних і спеціалізованих пакетів моделювання.

Метою викладання навчальної дисципліни є формування знань і навичок стосовно основних підходів і принципів побудови моделей та надбуття навичок щодо їх застосування для розв'язання задач моделювання систем і методів їх оптимізації. Із цією метою велика увага приділяється практичній роботі студентів на персональних комп'ютерах із застосуванням математичних пакетів.

Об'єктом вивчення дисципліни є різні (технічні, фізичні тощо) системи (явища, процеси, об'єкти), з якими пов'язана людська діяльність. *Предметом* вивчення дисципліни є загальновідомі методології і сучасні технології моделювання складних систем і методи їх оптимізації.

Студенти повинні *вміти*:

визначати вид моделювання та вид моделі для даної системи;

знаходити вхідні та вихідні змінні системи та моделі;

складати рівняння та системи рівнянь для побудови моделі;

розв'язувати математичні задачі, застосовуючи математичні пакети, зокрема пакет R;

робити висновки щодо застосування моделі;

вносити корективи в модель у разі використання;

будувати й аналізувати імітаційні моделі систем масового обслуговування.

Під час розроблення складних комп'ютерних моделей нерідко доводиться розв'язувати математичні задачі різної складності, наприклад:

розв'язання рівнянь і систем алгебраїчних рівнянь, лінійних і нелінійних;

диференціювання та інтегрування функцій;

апроксимації (наближення) даних;

розв'язання диференціальних рівнянь (звичайних і із частинними похідними) та ін.

Основним інструментом розв'язання цих задач є чисельні методи.

У ході вибору оптимальних режимів функціонування складних систем на базі їх комп'ютерних моделей доводиться розв'язувати математичні задачі, наприклад:

задачі безумовної й умовної оптимізації;

задачі лінійного та нелінійного програмування та ін.

Основним інструментом розв'язання цих задач є методи оптимізації.

Тому вся дисципліна розбита на три модулі: чисельні методи, методи оптимізації та моделювання систем.

У світі розроблено ряд універсальних математичних пакетів, за допомогою яких можна достатньо швидко як розв'язати багато відомих математичних задач, так і провести попередні розрахунки з подальшою їх реалізацією вже в проектованій комп'ютерній системі. Треба зазначити, що останнім часом намітилася тенденція до зближення та інтеграції різних математичних пакетів. Тому немає значення, який математичний пакет застосовується в навчальному процесі та засвоюється студентами. Головне – принципово навчитися використовувати будь-який математичний пакет для розв'язання практичних задач.

Останнім часом математичний пакет R набуває все більшої популярності у світі, тому саме він використовується в цих методичних рекомендаціях щодо програмування чисельних методів, методів оптимізації, моделювання та наочного подання результатів розв'язання практичних і теоретичних задач.

Таблиця 1

Умовні позначення

Позначення	Значення
R^1	Простір дійсних чисел
$x \in G$	x належить множині G (x – елемент множини G)
$i = \overline{1,n}$	i приймає значення від 1 до n (еквівалентно $i = 1, \dots, n$)
$\forall x \in G$	Для всіх x, що належать множині G
$\exists x \in G$	Існує x, що належить множині G
R^n	n-вимірний дійсний простір векторів-стовпців, тобто векторів $x = (x_1, x_2, \dots, x_n)^T$, де $x_i \in R^1$, $i = \overline{1,n}$
$x \in R^n$	x – n-вимірний вектор-стовпець
x^T	Вектор, транспонований до x (вектор-рядок)
$x^{(k)} \in R^n$	$x^{(k)}$ – n-вимірний вектор-стовпець, k-й за номером у деякій нумерації
x_i	i-та компонента вектора x
$\{x^{(k)}\}$	Послідовність $x^{(k)}$, тобто $x^{(0)}$, $x^{(1)}$, $x^{(2)}$, ...
$\{x \in G: T\}$	Підмножина елементів множини G, що задовільнюють умові T
$x \geq 0, x \in R^n$	Усі компоненти n-вимірного вектора x невід'ємні
$A = (a_{ij})_{ij=1}^n$	Матриця розмірності n на n з елементами a_{ij}, де a_{ij} – елемент i-го рядка та j-го стовпця
$R^{n \times m}$	Множина дійсних матриць розмірності n на m
A^T	Матриця, транспонована до A, тобто матриця з елементами a_{ji}
$f: R^1 \to R^1$	Дійсна функція однієї дійсної змінної, тобто якщо $x \in R^1$ і $y = f(x)$, то $y \in R^1$
$f: R^n \to R^m$	Дійсна вектор-функція n дійсних змінних, тобто якщо $x \in R^n$ і $y = f(x)$, то $y \in R^m$
$x^T y$ або $\langle x, y \rangle$	Скалярний добуток векторів x, y, тобто якщо $x, y \in R^n$, то $\langle x, y \rangle = = x^T y = [\sum_{i=1}^n x_i y_i]$
$\|x\|$	Евклідова норма вектора x, тобто якщо $x \in R^n$, то $\|x\| = [\sum_{i=1}^n x_i^2]^{\frac{1}{2}} = = \sqrt{x^T x}$

Змістовий модуль 1
Чисельні методи

Лабораторна робота 1. Вступ у систему R

1.1. Мета роботи

Вивчення можливостей математичного пакета **R**; набуття навичок використання пакета R для розв'язання математичних задач і графічного подання результатів досліджень на комп'ютері.

1.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* загальні поняття лінійної алгебри та програмування; *вміти* застосувати засоби пакета R для розв'язання простих математичних задач [18 – 20].

1.2.1. Числові вектори

Вектори в системі R формуються функцією **c()**. Аргументи цієї функції є компонентами вектора. Наприклад, команда

```
> a = c(1, 2, 3, 4, 5)
> a
[1] 1 2 3 4 5
```

формує вектор-рядок **(1, 2, 3, 4, 5)** і привласнює його змінній **a**.

Аргументи функції **c()** можуть бути векторами. У цьому випадку як результат маємо конкатенацію цих векторів. Скалярні значення (тобто числа) сприймаються R як вектори довжини 1. Таким чином, аргументами функції **c()** можуть бути як вектори, так і скаляри.

Наприклад:

```
> c(c(1, 2, 3, 4, 5), 6, c(7, 8))
[1] 1 2 3 4 5 6 7 8
```

Вектор, що складається з послідовних чисел, можна отримати за допомогою команди <початкове_значення>:<кінцеве_значення>. Наприклад, 1:5.

На цю команду схожа функція **seq**, яка генерує відрізки арифметичної прогресії. Можна задати початкове значення, кінцеве значення і крок:

```
> seq(0, 1, by=0.1)
 [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
```

В іншому варіанті задається початкове значення, кінцеве значення і кількість точок. У результаті буде згенерований вектор, що складається із заданої кількості компонент, рівномірно розподілених на заданому відрізку:

```
> seq(0, 1, len=11)
 [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
```

Функція **rep(v,k)** створює вектор, що складається з **k** копій вектора **v**. Наприклад:

```
> rep(c(1, 2), 3)
[1] 1 2 1 2 1 2
```

Над векторами можна виконувати арифметичні операції й елементарні функції. Елементарні математичні функції застосовуються до кожної компоненти вектора. Наприклад:

```
> x = c(0, pi/2, pi)
> sin(x)
[1] 0.000000e+00 1.000000e+00 1.224606e-16
```

Доступ до елементів вектора здійснюється оператором **[i]**. Наприклад, **u[5]** – це п'ятий елемент вектора **u**. Нумерація елементів починається з 1. Вирази виду **u[i]** можуть зустрічатися і в лівій частині від знака привласнення. Водночас якщо вектор має довжину не менше **i**, то **u[i]** просто прийме нове значення. В інакшому випадку вектор **u** збільшить свою довжину до **i**, компонента **u[i]** прийме нове значення, а останні нові компоненти приймуть значення **NA**.

```
> u = 1
> u[5] = 5
> u
[1]  1 NA NA NA  5
```

Деякі корисні функції:

crossprod(v, w) – скалярний добуток векторів **v** та **w**;

sum(v) – сума елементів вектора **v**;

prod(v) – добуток елементів вектора **v**;

max(v) – максимальний елемент;

min(v) – мінімальний елемент;

length(v) – довжина вектора;

mean(v) – середнє значення елементів вектора **v** (оцінка математичного сподівання);

var(v) – варіація елементів вектора v (оцінка дисперсії);

sort(v) – повертає вектор тієї самої довжини, що і **v**, з елементами, відсортованими в порядку збільшення.

1.2.2. Рядкові назви елементів вектора

Елементам вектора (числового, логічного, символьного) можна задавати імена. Наприклад:

```
> fruit = c(5, 10, 1, 20)
> names(fruit) = c("orange", "banana","apple", "peach")
> fruit
orange banana  apple  peach
     5     10      1     20
```

Створений вектор довжиною 4, компоненти якого мають указані назви. Тепер звертатися до елементів вектора можна як за індексом, так і на ім'я:

```
> fruit[1] = 6
> fruit["peach"] = 60
> fruit
orange banana  apple  peach
     6     10      1     60
```

1.2.3. Матриці

Числову матрицю можна побудувати із числового вектора за допомогою функції **matrix**. Треба вказати кількість рядків **nrow=m** і/або кількість стовпців **ncol=m**. У результаті елементи з вектора будуть записані в матрицю вказаних розмірів. Наприклад:

```
> matrix(1:6, nrow=2, ncol=3)
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

Довжина вектора має бути кратною добутку потрібної кількості рядків і потрібної кількості стовпців:

```
> matrix(1, nrow=2, ncol =2)
     [,1] [,2]
[1,]    1    1
[2,]    1    1
> matrix(1:2, nrow=3, ncol =2)
     [,1] [,2]
[1,]    1    2
[2,]    2    1
[3,]    1    2
```

Елементи записуються в матрицю по стовпцях. Щоб записати їх по рядкам, треба задати значення додаткового параметра **byrow=TRUE**:

```
> matrix(1:6, nrow=2, byrow=TRUE)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

Функції **nrow(A)** і **ncol(A)** повертають, відповідно, кількість рядків і стовпців матриці **A**.

Доступ до елементів матриці відбувається за індексом. Так, **A[i,j]** посилається на елемент **i**-го рядка і **j**-го стовпця матриці **A**.

A[i,] еквівалентне **A[i,1:ncol(A)]**, а **A[,j]** еквівалентне **A[1:nrow(A),j]**. Можливий доступ до елементів матриці за допомогою одного індексу. Наприклад, **U[5]** означає п'ятий елемент матриці **U**, якщо вважати, що елементи перенумеровані по стовпцях. Якщо **i**-вектор, то **A[i]** – означає вибірку відповідних елементів і т. д.

Можна задати імена стовпцям і рядкам матриці:

```
> A = matrix(c(23, 31, 58, 16), nrow=2)
> rownames(A) = c("petal","sepal")
> colnames(A) = c("length","width")
> A
      length width
petal     23    58
sepal     31    16
```

Після цього доступ до рядків і стовпців може відбуватися за ім'ям.

Наприклад:

```
> A["petal","length"]
[1] 23
```

Функція **A = cbind(A, B)** створює матрицю, приписуючи до **A** справа **B** (для цього кількість рядків у **A** і **B** має співпадати).

Функція **A = rbind(A, B)** створює матрицю, приписуючи до **A** знизу **B** (для цього кількість стовпців у **A** і **B** має співпадати). Зазначимо, що в списках аргументів функцій **cbind** і **rbind** можна вказати більше двох матриць.

Арифметичні операції над матрицями здійснюються покомпонентно; тому, наприклад, щоб додати дві матриці, вони повинні мати однакові розміри.

Елементарні математичні функції також застосовуються до елементів матриці покомпонентно.

Транспонування матриці здійснює функція **t(A)**, а матричний добуток – операція **%*%**:

```
> A = matrix(1:6,nrow=2)
> A
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> B = t(A)
> B
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
> A%*%B
     [,1] [,2]
[1,]   35   44
[2,]   44   56
> B%*%A
     [,1] [,2] [,3]
[1,]    5   11   17
[2,]   11   25   39
[3,]   17   39   61
```

Для розв'язання системи лінійних рівнянь Ax = b з квадратною невиродженою матрицею A є функція **solve(A,b)**:

```
> A = matrix(1:4,ncol =2)
> A
     [,1] [,2]
[1,]    1    3
[2,]    2    4
> b = c(5,8)
> b
[1] 5 8
> solve(A,b)
[1] 2 1
```

Функція **det(A)** знаходить визначник, а **solve(A)** – обернену матрицю:

```
> det(A)
[1] -2
> solve(A)
     [,1] [,2]
[1,]   -2  1.5
[2,]    1 -0.5
```

1.2.4. Графіки функцій

Для зображення графіків функцій у R є функція **plot(x,y)**. Тут **x** – вектор значень абсцис і **y** – вектор значень ординат. Якщо вказаний тільки один аргумент – **plot(y)**, то передбачається, що **x=1:n**, де **n** – довжина вектора **y**. Наприклад, команди:

```
> x = seq(-pi, pi, len =30)
> y = sin(x)
> plot(x,y)
```

рисують 30 точок, що лежать на синусоїді **y=sin(x)** (рис. 1.1). Графіки з'являються в окремому вікні (або в кількох окремих вікнах).

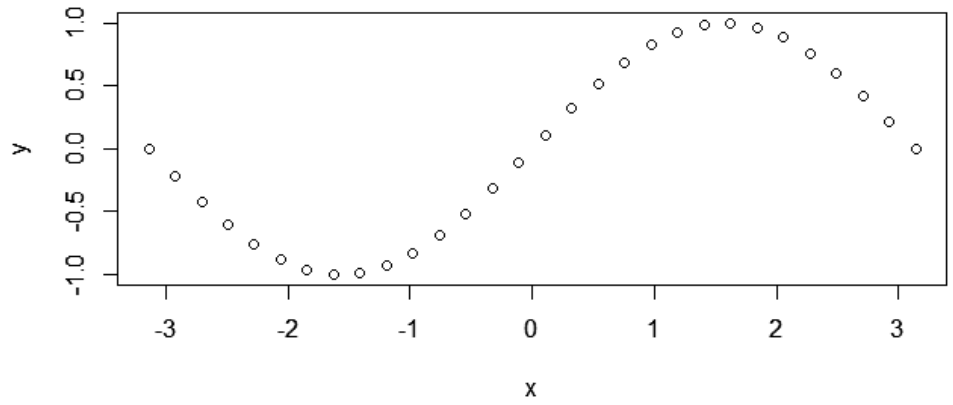

Рис. 1.1. **"Точковий" графік синуса**

Функція **plot** допускає багато додаткових параметрів. Деякі з них:

type=тип-графіка дозволяє вказати тип графіка, що виводиться; деякі значення параметра – "p" (точки, за замовчуванням), "l" (лінії), "b" (лінії і точки), "o" (лінії і точки перекриваються), "n" (нічого не рисується);

xlab=назва-осі-абсцис і **ylab=назва-осі-ординат** дозволяє вказати підписи до осі абсцис і ординат відповідно;

main=основний-надпис і **sub=додатковий-надпис** створюють надписи зверху та знизу графіка;

col=колір задає колір графіка; можливі значення "blue", "red", "green", "cyan", "magenta", "yellow", "black" та багато інших. Кольори можна задати rgb-вектор-функцією **rgb(r, g, b)**. Параметри цієї функції – значення від 0 до 1;

lty=стиль-лінії визначає стиль лінії; можливі значення "blank" (немає лінії), "solid", "dashed", "dotted", "dotdash", "longdash", "twodash".

Розглянемо такий приклад:

```
> plot(x, y, type="l", col="blue", main="Sine of x")
```

Результат виконання цієї команди відображений на рис. 1.2.

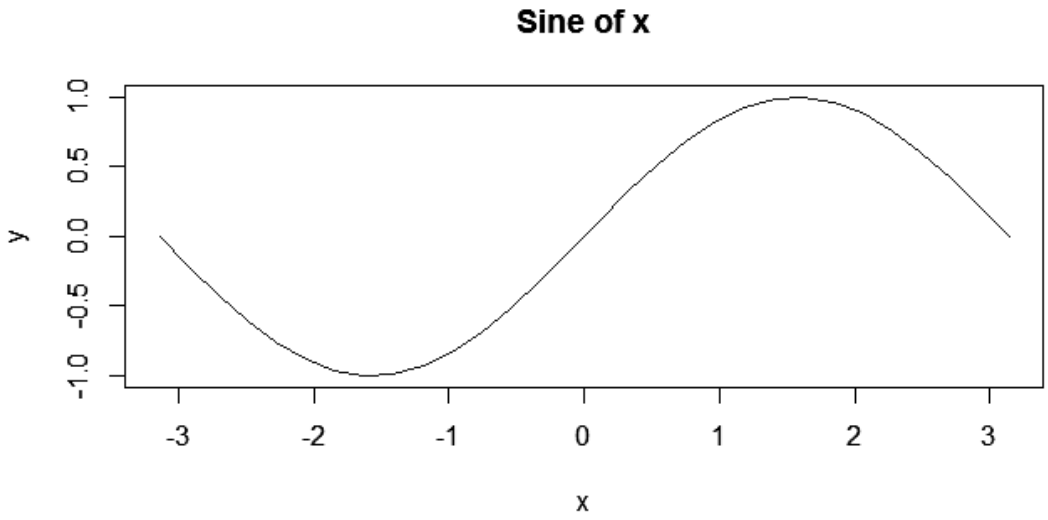

Рис. 1.2. **Графік синуса**

Наступні функції (що називають низькорівневими) ніколи не стирають зображення, яке вже є. Їм можуть бути передані також додаткові параметри, аналогічні відповідним параметрам функції **plot**:

points(x,y) рисує додаткові точки; **lines(x,y)** рисує ломану лінію, яка з'єднує вказані точки. Цим функціям можуть бути передані додаткові аргументи, наприклад, **type=тип-графика** (за замовчуванням він дорівнює "p" для **points** і "l" для **lines**):

text(x,y,текст) додає текст. За замовчуванням він центрується навколо точки **(x,y)**. Указані параметри можуть бути векторами. У цьому випадку буде розміщено кілька надписів. Можна вказати шрифт **font=шрифт** та ін.;

abline(k,b) рисує пряму y=kx+b. Можна вказати колір і стиль лінії тощо;

abline(h=y), **abline(v=x)** – для рисування горизонтальних і вертикальних прямих. Вказуються, відповідно, координати **y** і **x**;

polygon(x,y) рисує багатокутник з вершинами **(x,y)**. Можна вказати колір заповнення **col=колір** і колір межі **border=колір**;

legend(x,y,легенда) додає легенду в указану точку.

Розглянемо приклад:

```
> x = seq(-10, 10, by=0.1)
> y1 = x^2+2*x-5
> y2 = -x^2+60
> plot(x, y1, type="n",main="Графіки функцій")
> colors = c("blue", "red", "green")
> lines(x,y1, col=colors[1])
> lines(x,y2, col=colors[2], lty="dashed")
> legends = paste("y",1:2, sep="")
> legend(3, 110, legends, lty="solid",col=colors)
> abline(h=0) #Горизонтальна вісь
> abline(v=0) #Вертикальна вісь
```

Результат наведено на рис. 1.3.

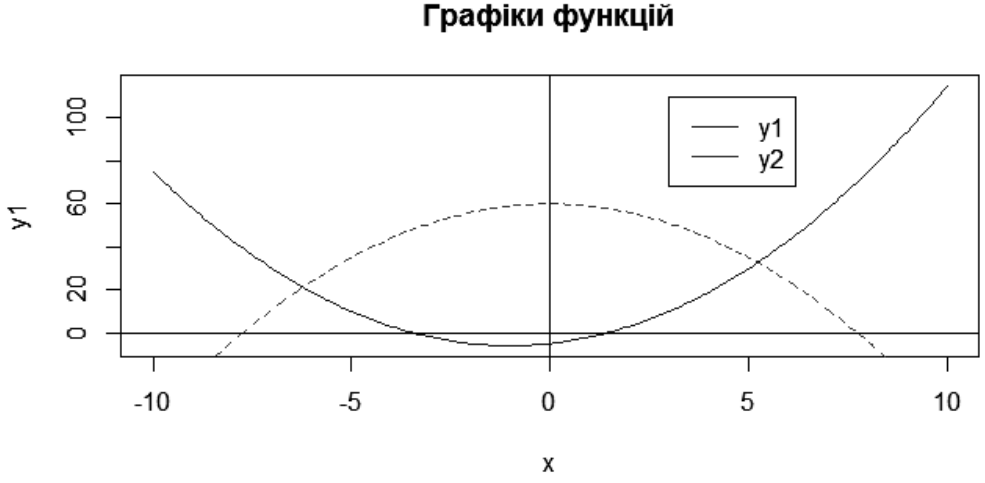

Рис. 1.3. **Два графіка в одному графічному вікні**

1.2.5. Управляючі конструкції

Кожна команда чи вираз у мові R повертає деякий результат. Навіть привласнення – це вираз, результат якого – привласнюване значення. Вираз може бути частиною іншого виразу. Зокрема, можливі множинні привласнювання, наприклад:

```
> x = y = z = 0
```

Перевірка умов здійснюється так, як і в мові С:

```
> if(умова) {вираз1} else {вираз2}
```

де *умова* – це логічний вираз, результатом якого є скалярне логічне значення.

Цикли з управляючою змінною реалізуються з допомогою конструкції:

```
> for (змінна in вираз1) {вираз2}
```

Результатом *вираз1* має бути вектор. Змінна на кожній ітерації циклу приймає значення чергового елемента цього вектора. Кількість ітерацій дорівнює кількості елементів у векторі.

```
> s=0
> for(i in 1:20)  s = s+i
> s
[1] 210
```

Зазначимо, що такий результат швидше можна отримати за допомогою команди. **sum(1:20)**.

Цикли з передумовою реалізуються за допомогою конструкції:

```
> while (умова) {вираз}
```

Передчасний вихід із циклу здійснюється командою **break**. Для переривання поточної ітерації і переходу до наступної слугує команда **next**. Ці команди можна також використовувати з конструкцією **for**.

Безкінечний цикл реалізує команда **repeat**:

```
> repeat {вираз}
```

Усередині цієї конструкції можна використовувати команди **break** і **next**.

1.2.6. Функції користувача

Формат визначення функції такий:

```
> ім'я = function(arg1,arg2, ...) {вираз}
```

Тут список формальних аргументів (формальних параметрів) **арг1, арг2,...** може мати довільну довжину (в тому числі бути порожнім). Як правило, вираз (тіло функції) є блоком. Значення, яке повертає функція, це значення цього виразу. Дострокове повернення з функції здійснює команда **return(вираз1)**. У цьому випадку функція повертає значення вказаного виразу.

Звернення до функції має вигляд **ім'я(вираз1, вираз2, ...)**. Тут значення виразів **вираз1, вираз2,...** є фактичними аргументами, які підставляються замість відповідних формальних аргументів.

Зі зміною значень формальних аргументів усередині функції не змінюються відповідні значення фактичних аргументів. Єдиний спосіб зробити це – використовувати в тілі функції <сильне> привласнювання: **arg1 <-- вираз**.

Якщо фактичні аргументи задані в <іменованому> вигляді **ім'я = вираз**, то вони можуть бути перераховані в довільному порядку. Більш того, список фактичних аргументів може починатися з аргументів, поданих у звичайній позиційній формі, після чого можуть іти аргументи в іменованій формі.

Наприклад, є функція **fun**, задана таким чином:

```
> fun = function(arg1,arg2,arg3,arg4)
+   {
+     #тіло функції опущене
+   }
```

Тоді **fun** може бути викликана численними еквівалентними способами, наприклад:

```
> ans = fun(d,f, 20, TRUE)
> ans = fun(d,f, arg4=TRUE, arg3=20)
> ans = fun(arg1=d, arg3=20, arg2=f, arg4=TRUE)
```

У багатьох випадках під час визначення функції деяким аргументам можуть бути привласнені значення за замовчуванням. Тоді з викликом функції ці аргументи можуть бути опущені. Припустимо, що функція **fun** визначена як:

```
> fun = function(arg1, arg2, arg3=20, arg4=TRUE)
+ {
+   #тіло функції опущене
+ }
```

Тоді звертання до цієї функції виду **fun(d,f)** еквівалентне трьом, наведеним. Наступне звертання:

```
> fun(d,f, arg4=FALSE)
```

змінює одне зі значень за замовчуванням.

Зауважимо, що в значеннях за замовчуванням можна використовувати будь-які вирази, у тому числі такі, що містять інші аргументи функції.

Усі символи, що зустрічаються в тілі функції, розподіляють на три групи: формальні аргументи, локальні змінні та вільні змінні.

Формальні аргументи (формальні параметри) – це аргументи, перераховані в заголовку функції (в круглих дужках після ключового слова **function**).

Локальні змінні – це змінні, які не є формальними аргументами, значення яких визначаються під час виконання функції.

Змінні, які не є формальними аргументами та локальними змінними, є *вільними змінними*.

Наприклад, у функції:

```
> f = function(x)
+ {
+   y = 2*x
+   print(x)
+   print(y)
+   print(z)
+ }
```

x – формальний аргумент, **y** – локальна змінна, **z** – вільна змінна.

Область видимості формальних аргументів і локальних змінних – лише сама ця функція. Це означає, що зміни цих змінних усередині функції ніяк не відображаються на змінних з такими ж іменами у зовнішній функції. Область видимості вільних змінних розповсюджується до зовнішньої функції, в якій вони були визначені. Зміна таких змінних у тілі функції впливає також на відповідні змінні в цій зовнішній функції.

1.3. Порядок виконання роботи та варіанти завдань

1.3.1. Зміст звіту

У практичній частині роботи необхідно:

виконати приклади, наведені в підрозділі 1.2;

виконати завдання наведене в підрозділі 1.3.2;

надати текст складеної програми та результати її виконання.

1.3.2. Завдання

Запрограмуйте функцію, яка приймає на вхід числовий вектор **x** і число розбиття відрізка **k** (за замовчуванням дорівнює кількості елементів вектора, поділеній на 10) і виконує таке: знаходить мінімальне (**xmin)** і максимальне (**xmax)** значення елементів вектора **x**, ділить отриманий відрізок **[xmin; xmax]** на **k** рівних інтервалів і підраховує кількість елементів вектора, що належать кожному інтервалу. Далі треба побудувати графік, де на осі абсцис – середини інтервалів, на осі ординат – кількість елементів вектора, що належать інтервалу, поділене на загальну кількість точок. Результатом виконання функції є побудова графіка.

Проведіть експеримент на даній функції, де **x** – вектор довжиною 5 000, згенерований із нормально розподіленої випадкової величини з математичним очікуванням a = 5.5 і середньоквадратичним відхиленням $\sigma = 1.5$ (функція **rnorm**).

1.4. Контрольні запитання

1. Які є засоби в системі R для створення та роботи з векторами та матрицями?
2. Які є засоби в системі R для побудови графіків?
3. Які є в системі R управляючі конструкції для програмування?

4. Як описати функцію користувача в системі R?

5. Чим відрізняються формальні аргументи, локальні змінні та вільні змінні, що зустрічаються в тілі функції користувача?

Лабораторна робота 2.
Чисельні методи розв'язання систем лінійних алгебраїчних рівнянь

2.1. Мета роботи

Вивчення методу Гаусса із частковим вибором головного елемента та ітераційних методів для практичного розв'язання систем лінійних алгебраїчних рівнянь, набуття навичок використання цих методів для розв'язання систем лінійних алгебраїчних рівнянь із застосуванням комп'ютера.

2.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* загальне формулювання задачі розв'язання системи лінійних алгебраїчних рівнянь; *уміти* розв'язувати системи лінійних алгебраїчних рівнянь методом виключення Гаусса [1; 2; 6] і методом ітерацій; *вміти* застосовувати процедури пакета R для розв'язання систем лінійних алгебраїчних рівнянь.

Задача розв'язання системи алгебраїчних лінійних рівнянь у загальному вигляді формулюється в такий спосіб: необхідно знайти n невідомих $x_i \in R^1$, $i = \overline{1,n}$, що задовольняють системі рівнянь:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{cases}, \qquad (2.1)$$

де $a_{ij} \in R^1$, $i = \overline{1,n}$, $j = \overline{1,n}$ – задані коефіцієнти за невідомих; $b_i \in R^1$, $i = \overline{1,n}$ – задані праві частини (вільні члени).

Якщо ввести позначення $A = (a_{ij})_{ij=1}^{n}$ – матриця розмірності $n \times n$ з коефіцієнтами a_{ij}, $b = (b_i)_{i=1}^{n}$ – вектор-стовпець розмірності n з елементами b_i, $x = (x_i)_{i=1}^{n}$ – вектор-стовпець розмірності n з елементами x_i, то систему (2.1) можна записати у матричному вигляді:

$$A\,x = b. \qquad (2.2)$$

Для того щоб система (2.2) мала єдиний розв'язок, необхідно та достатньо, щоб $\det A \neq 0$.

Найбільш відомим з методів розв'язання системи (2.1) є **метод виключення Гаусса**, ідея якого полягає в послідовному виключенні невідомих з рівнянь.

Розрахункові формули метода Гаусса:

1. Позначимо $a_{ij}^{(0)} = a_{ij}$, $i = \overline{1,n}$, $j = \overline{1,n}$; $a_{i,n+1}^{(0)} = b_i$, $i = \overline{1,n}$.

2. Прямий хід, k -й крок $(k = \overline{1,n})$: $a_{kk}^{(k)} = 1$, $a_{kj}^{(k)} = \frac{a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}}$, $j = \overline{k+1,n+1}$;

$a_{ij}^{(k)} = a_{ij}^{(k-1)} - a_{ik}^{(k-1)} \times a_{kj}^{(k)}$, $i = \overline{k+1,n}$, $j = \overline{k+1,n+1}$.

3. Зворотний хід: $x_n = a_{n,n+1}^{(n)}$, $x_k = a_{k,n+1}^{(k)} - \sum_{j=k+1}^{n} a_{kj}^{(k)} x_j$, $k = \overline{n-1,1}$.

Для ручного розрахунку застосовують схему, що наведена в табл. 2.1 для випадку, коли $n = 4$.

Таблиця 2.1

Схема ручного розрахунку (метод Гаусса)

Коефіцієнти за невідомих				Вільні члени b
x_1	x_2	x_3	x_4	
a_{11}	a_{12}	a_{13}	a_{14}	$a_{15} = b_1$
a_{21}	a_{22}	a_{23}	a_{24}	$a_{25} = b_2$
a_{31}	a_{32}	a_{33}	a_{34}	$a_{35} = b_3$
a_{41}	a_{42}	a_{43}	a_{44}	$a_{45} = b_4$
1	$a_{12}^{(1)}$	$a_{13}^{(1)}$	$a_{14}^{(1)}$	$a_{15}^{(1)}$
0	$a_{22}^{(1)}$	$a_{23}^{(1)}$	$a_{24}^{(1)}$	$a_{25}^{(1)}$
0	$a_{32}^{(1)}$	$a_{33}^{(1)}$	$a_{34}^{(1)}$	$a_{35}^{(1)}$
0	$a_{42}^{(1)}$	$a_{43}^{(1)}$	$a_{44}^{(1)}$	$a_{45}^{(1)}$
	1	$a_{23}^{(2)}$	$a_{24}^{(2)}$	$a_{25}^{(2)}$
	0	$a_{33}^{(2)}$	$a_{34}^{(2)}$	$a_{35}^{(2)}$
	0	$a_{43}^{(2)}$	$a_{44}^{(2)}$	$a_{45}^{(2)}$
		1	$a_{34}^{(3)}$	$a_{35}^{(3)}$
		0	$a_{44}^{(3)}$	$a_{45}^{(3)}$
			1	$a_{45}^{(4)}$

Рядки, що містять одиницю, називають виділеними рядками. Діагональний елемент $a_{kk}^{(k-1)}$, на який проводиться ділення, є **головним** (ведучим) **елементом**. Якщо головний елемент близький до нуля за абсолютною величиною, то необхідно знайти у відповідному (k-ому) стовпці максимальний за модулем елемент і переставити рядки місцями так, щоб цей елемент став головним.

Процес отримання виділених рядків (приведення системи до трикутного вигляду) називають **прямим ходом**, а процес знаходження невідомих шляхом використання виділених рядків – **зворотним ходом** методу Гаусса.

Метод ітерацій є найбільш типовим прикладом ітераційного (наближеного) методу розв'язання системи лінійних рівнянь (2.2).

Для застосування методу ітерацій систему (2.2) необхідно подати у вигляді:

$$x = C \cdot x + d, \tag{2.3}$$

де $C \in R^{n \times n}$, $d \in R^n$.

Алгоритм методу ітерацій.

1. Задається $\varepsilon > 0$ – точність розв'язання задачі та початкове наближення розв'язку $x^{(0)} \in R^n$ (наприклад $x^{(0)} = d$).

2. На k-й ітерації методу ($k = 0,\ 1,\ 2,\ ...$) обчислюється наступне наближення $x^{(k+1)}$ за формулою:

$$x^{(k+1)} = C \cdot x^{(k)} + d.$$

3. Перевіряється критерій "останову" $\frac{q}{1-q}\left\|x^{k+1} - x^{k}\right\| \le \varepsilon$, де $0 < q < 1$ (визначається з умови збіжності).

Метод простої ітерації збігається тільки за виконанні умови:

$$\|C\| \le q, \tag{2.4}$$

де $0 < q < 1$ (умова збіжності) [1; 2; 6]. Тут, як норму матриці C можна розглядати величину:

$$\|C\| \equiv \max_j \sum_{i=1}^{n} |c_{ij}| \text{ або } \|C\| \equiv \max_i \sum_{j=1}^{n} |c_{ij}|. \tag{2.5}$$

Приведення системи (2.1) до виду (2.3) можна здійснити різними способами, важливо тільки, щоб виконувалася умова збіжності (2.4). Розглянемо один із них.

Якщо діагональні елементи матриці відмінні від нуля, тобто $a_{ii} \neq 0$, $i = \overline{1,n}$, то систему (2.1) можна записати у вигляді

$$\begin{cases} x_1 = \frac{1}{a_{11}}(-a_{12}x_2 - a_{13}x_3 - \ldots - a_{1n}x_n + b_1), \\ x_2 = \frac{1}{a_{22}}(-a_{21}x_1 - a_{23}x_3 - \ldots - a_{2n}x_n + b_2), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n = \frac{1}{a_{nn}}(-a_{n1}x_1 - a_{n3}x_3 - \ldots - a_{n,n-1}x_{n-1} + b_n). \end{cases} \quad . \qquad (2.6)$$

Застосування методу ітерацій до системи (2.6) називають **методом Якобі.** З (2.4) – (2.5) витікає, що для збіжності методу Якобі повинна виконуватись умова $|a_{ii}| > \sum_{j=1}^{n}|a_{ij}| \quad \forall\ i = \overline{1,\ n}$.

2.3. Контрольні приклади

Приклад 1. Розв'язати методом Гаусса систему рівнянь:

$$\begin{cases} 0{,}14x_1 + 0{,}24x_2 - 0{,}84x_3 = 1{,}11 \\ 1{,}07x_1 - 0{,}83x_2 + 0{,}56x_3 = 0{,}48 \\ 0{,}64x_1 + 0{,}43x_2 - 0{,}38x_3 = -0{,}83. \end{cases}$$

Ручний розрахунок. Результати ручного розрахунку наведені в табл. 2.2.

Таблиця 2.2

Результати ручного розрахунку

x_1	x_2	x_3	B
0,14	0,24	–0,84	1,11
1,07	–0,83	0,56	0,48
0,64	0,43	–0,38	–0,83
1	1,7143	–6	7,926
0	–2,6643	6,98	–8,0036
0	0,6672	–3,46	–5,9043
	1	–2,6198	3,004
	0	1,7121	–3,9
		1	–2,2779
	1	0	–2,9636
1			–0,6583

Розв'язок: $x_1 = -0{,}6583$, $x_2 = -2{,}9636$, $x_3 = -2{,}2779$.

Використання процедур пакета R. У пакеті R є процедура $solve$, призначена для розв'язання системи лінійних рівнянь у матричному вигляді (2.2). Для її виклику необхідно визначити та задати матрицю A та вектор b, тоді розв'язком системи є $x = \text{solve}\ (A, b)$.

Задається матриця коефіцієнтів *A* та вектор вільних членів *b*:

```
> A = matrix(nrow=3,ncol=3,c(0.14,1.07,0.64,0.24,-0.83,
+                            0.43,-0.84,0.56,-0.38))
> A
     [,1]  [,2]  [,3]
[1,] 0.14  0.24 -0.84
[2,] 1.07 -0.83  0.56
[3,] 0.64  0.43 -0.38
> b = matrix(c(1.11,0.48,-0.83))
> b
      [,1]
[1,]  1.11
[2,]  0.48
[3,] -0.83
```

Розв'язання з використанням процедури $solve$:

```
> x = solve(A,b)
> x
           [,1]
[1,] -0.658156
[2,] -2.963664
[3,] -2.277882
```

Перевірка розв'язку:

```
> A%*%x - b          # Перевіряємо рішення
            [,1]
[1,]  0.000000e+00
[2,] -1.110223e-16
[3,] -5.551115e-16
```

Приклад 2. Необхідно побудувати площину у тривимірному просторі, що проходить через три точки з координатами $(10,\ 5,\ 3)$, $(1,\ 7,\ 5)$, $(-2,\ 6,\ 1)$.

Розв'язання. У загальному вигляді рівняння площини у тривимірному просторі має вид: $z = a\ x + b\ y + c$, де $(x,\ y,\ z)$ – тримірні координати. Для побудови площини достатньо знайти параметри рівняння $a,\ b,\ c$. Оскільки площина має проходити через задані точки, то параметри рівняння $a,\ b,\ c$ повинні задовільнювати системі рівнянь:

$$\begin{cases} 10\ a\ + 5\ b + \ c = 3 \\ 1\ a\ + 7\ b + \ c = 5 \\ -2\ a\ + 6\ b + \ c = 1 \end{cases}.$$

Реалізація в пакеті R:

```
> A = matrix(nrow=3,ncol=3,c(10, 1, -2, 5, 7, 6, 1, 1, 1))
> A
     [,1] [,2] [,3]
[1,]   10    5    1
[2,]    1    7    1
[3,]   -2    6    1
> b = c(3,5,1)
> b
[1] 3 5 1
> x = solve(A,b)
> x
[1]   0.4   2.8 -15.0
```

Розв'язок: $a = 0.4$, $b = 2.8$, $c = -15$, тобто рівняння площини має вигляд: $z = 0.4\ x + 2.8\ y - 15$.

Приклад 3. Розв'язати методом ітерацій систему рівнянь:

$$\begin{cases} 5x_1 + x_2 + 2x_3 = 10, \\ 6x_1 + 18x_2 + 6x_3 = 54, \\ 10x_1 + 20x_2 + 40x_3 = 160. \end{cases}$$

Розв'язання. Перепишемо систему у вигляді

$$\begin{cases} x_1 = -\frac{1}{5}x_2 - \frac{2}{5}x_3 + 2, \\ x_2 = -\frac{1}{3}x_1 - \frac{1}{3}x_3 + 3, \\ x_3 = -\frac{1}{4}x_1 - \frac{1}{2}x_2 + 4. \end{cases}$$

Реалізація методу ітерацій у пакеті R:

```
MetIterSLE = function(C, d, n, eps){
    x0 = d
    repeat{
        x = C%*%x0 + d
        if( norm(x-x0) <= eps )
            break
        x0 = x
    }
    return(x)
}
```

Введення початкових даних і виклик запрограмованої процедури для отримання розв'язку:

```
> C = rbind(c(0,-1/5,-2/5), c(-1/3,0,-1/3), c(-1/4,-1/2,0))
> C
           [,1] [,2]       [,3]
[1,]  0.0000000 -0.2 -0.4000000
[2,] -0.3333333  0.0 -0.3333333
[3,] -0.2500000 -0.5  0.0000000
> d = c(2,3,4)
> d
[1] 2 3 4
> x = MetIterSLE(C,d,3, 0.0001)
> x
          [,1]
[1,] 0.4444534
[2,] 1.8666762
[3,] 2.9555659
```

Перевірка розв'язку:

```
> y = C%*%x + d
> y
          [,1]
[1,] 0.4444384
[2,] 1.8666602
[3,] 2.9555486
```

2.4. Порядок виконання роботи і варіанти завдань

2.4.1. Зміст звіту

У теоретичній частині роботи необхідно стисло описати:

постановку задачі розв'язання систем лінійних алгебраїчних рівнянь;

чисельні методи розв'язання систем лінійних алгебраїчних рівнянь.

У практичній частині роботи необхідно:

запрограмувати у вигляді окремого модуля метод Гаусса;

запрограмувати у вигляді окремого модуля метод Якобі;

надати тексти складених програм;

розв'язати задачу з індивідуального завдання (табл. 2.3) із застосуванням складених програм і засобів математичного пакета R.

2.4.2. Варіанти індивідуальних завдань

Таблиця 2.3

Індивідуальні завдання за варіантами

Варіант	Система
1	2
1	$\begin{cases} 5{,}23x_1 + 0{,}57x_2 + 0{,}61x_3 + 0{,}48x_4 = 6{,}72 \\ 0{,}66x_1 + 7{,}04x_2 + 0{,}77x_3 + 0{,}36x_4 = -6{,}96 \\ 0{,}34x_1 + 0{,}82x_2 + 6{,}81x_3 + 0{,}18x_4 = 80{,}33 \\ 0{,}11x_1 + 0{,}16x_2 + 0{,}90x_3 + 8{,}33x_4 = 26{,}05 \end{cases}$
2	$\begin{cases} 0{,}70x_1 + 5{,}09x_2 + 0{,}89x_3 + 0{,}17x_4 = -4{,}43 \\ 0{,}32x_1 + 0{,}15x_2 + 9{,}11x_3 + 0{,}87x_4 = 8{,}10 \\ 0{,}85x_1 + 0{,}26x_2 + 0{,}19x_3 + 7{,}33x_4 = 12{,}73 \\ 0{,}21x_1 + 0{,}26x_2 + 0{,}95x_3 + 5{,}13x_4 = 16{,}06 \end{cases}$
3	$\begin{cases} 4{,}97x_1 + 0{,}07x_2 + 0{,}36x_3 + 0{,}81x_4 = -15{,}43 \\ 0{,}66x_1 + 7{,}12x_2 + 0{,}32x_3 + 0{,}77x_4 = 8{,}28 \\ 0{,}59x_1 + 0{,}16x_2 + 9{,}43x_3 + 0{,}36x_4 = 17{,}00 \\ 0{,}87x_1 + 0{,}91x_2 + 0{,}42x_3 + 8{,}15x_4 = 8{,}22 \end{cases}$
4	$\begin{cases} 7{,}70x_1 + 0{,}38x_2 + 0{,}43x_3 + 0{,}83x_4 = -13{,}79 \\ 0{,}97x_1 + 6{,}49x_2 + 0{,}14x_3 + 0{,}06x_4 = 17{,}35 \\ 0{,}81x_1 + 0{,}53x_2 + 9{,}13x_3 + 0{,}05x_4 = 12{,}41 \\ 0{,}19x_1 + 0{,}27x_2 + 0{,}38x_3 + 8{,}47x_4 = 14{,}58 \end{cases}$
5	$\begin{cases} 5{,}19x_1 + 0{,}17x_2 + 0{,}32x_3 + 0{,}57x_4 = -10{,}23 \\ 0{,}86x_1 + 7{,}49x_2 + 0{,}75x_3 + 0{,}96x_4 = 11{,}25 \\ 0{,}63x_1 + 0{,}42x_2 + 8{,}13x_3 + 0{,}52x_4 = 11{,}33 \\ 0{,}27x_1 + 0{,}65x_2 + 0{,}28x_3 + 6{,}99x_4 = 9{,}56 \end{cases}$
6	$\begin{cases} 6{,}17x_1 + 0{,}63x_2 + 0{,}46x_3 + 0{,}24x_4 = 4{,}12 \\ 0{,}43x_1 + 7{,}08x_2 + 0{,}83x_3 + 0{,}66x_4 = 9{,}72 \\ 0{,}34x_1 + 0{,}52x_2 + 5{,}44x_3 + 0{,}17x_4 = -5{,}88 \\ 0{,}71x_1 + 0{,}93x_2 + 0{,}65x_3 + 8{,}83x_4 = 14{,}25 \end{cases}$

Закінчення табл. 2.3

1	2
7	$\begin{cases} 7{,}01x_1 + 0{,}93x_2 + 0{,}75x_3 + 0{,}53x_4 = -13 \\ 0{,}22x_1 + 3{,}67x_2 + 0{,}31x_3 + 0{,}57x_4 = 6 \\ 0{,}46x_1 + 0{,}67x_2 + 9{,}16x_3 + 0{,}69x_4 = 12 \\ 0{,}82x_1 + 0{,}37x_2 + 0{,}35x_3 + 7{,}24x_4 = 2 \end{cases}$
8	$\begin{cases} 3{,}47x_1 + 0{,}37x_2 + 0{,}56x_3 + 0{,}72x_4 = -6{,}57 \\ 0{,}52x_1 + 5{,}88x_2 + 0{,}23x_3 + 0{,}40x_4 = 7{,}27 \\ 0{,}72x_1 + 0{,}63x_2 + 9{,}65x_3 + 0{,}75x_4 = 9{,}77 \\ 0{,}17x_1 + 0{,}97x_2 + 0{,}47x_3 + 8{,}12x_4 = 11{,}72 \end{cases}$
9	$\begin{cases} 3{,}97x_1 + 0{,}17x_2 + 0{,}53x_3 + 0{,}68x_4 = -10{,}71 \\ 0{,}60x_1 + 8{,}71x_2 + 0{,}47x_3 + 0{,}93x_4 = 11{,}69 \\ 0{,}27x_1 + 0{,}59x_2 + 6{,}43x_3 + 0{,}63x_4 = 14{,}99 \\ 0{,}32x_1 + 0{,}51x_2 + 0{,}84x_3 + 8{,}77x_4 = 18{,}49 \end{cases}$
10	$\begin{cases} 4{,}47x_1 + 0{,}93x_2 + 0{,}76x_3 + 0{,}52x_4 = -14{,}66 \\ 0{,}38x_1 + 6{,}63x_2 + 0{,}43x_3 + 0{,}61x_4 = 21{,}65 \\ 0{,}53x_1 + 0{,}76x_2 + 8{,}11x_3 + 0{,}27x_4 = 14{,}25 \\ 0{,}44x_1 + 0{,}68x_2 + 0{,}83x_3 + 7{,}09x_4 = 16{,}86 \end{cases}$
11	$\begin{cases} 5{,}12x_1 + 0{,}82x_2 + 0{,}47x_3 + 0{,}68x_4 = -11{,}44 \\ 0{,}26x_1 + 6{,}49x_2 + 0{,}55x_3 + 0{,}37x_4 = 12{,}03 \\ 0{,}49x_1 + 0{,}42x_2 + 7{,}95x_3 + 0{,}95x_4 = 16{,}22 \\ 0{,}31x_1 + 0{,}59x_2 + 0{,}91x_3 + 8{,}55x_4 = 13{,}59 \end{cases}$
12	$\begin{cases} 6{,}01x_1 + 0{,}22x_2 + 0{,}33x_3 + 0{,}44x_4 = 7{,}62 \\ 0{,}11x_1 + 5{,}42x_2 + 0{,}55x_3 + 0{,}66x_4 = -6{,}84 \\ 0{,}63x_1 + 0{,}43x_2 + 6{,}88x_3 + 0{,}77x_4 = 10{,}43 \\ 0{,}99x_1 + 0{,}34x_2 + 0{,}57x_3 + 8{,}13x_4 = 18{,}44 \end{cases}$
13	$\begin{cases} 5{,}49x_1 + 0{,}35x_2 + 0{,}57x_3 + 0{,}79x_4 = -9{,}56 \\ 0{,}51x_1 + 7{,}82x_2 + 0{,}86x_3 + 0{,}64x_4 = 13{,}42 \\ 0{,}37x_1 + 0{,}38x_2 + 7{,}93x_3 + 0{,}42x_4 = 12{,}84 \\ 0{,}91x_1 + 0{,}53x_2 + 0{,}88x_3 + 0{,}02x_4 = 16{,}02 \end{cases}$
14	$\begin{cases} 4{,}75x_1 + 0{,}49x_2 + 0{,}07x_3 + 0{,}33x_4 = -14{,}30 \\ 0{,}24x_1 + 5{,}68x_2 + 0{,}63x_3 + 0{,}74x_4 = 10{,}07 \\ 0{,}46x_1 + 0{,}98x_2 + 7{,}08x_3 + 0{,}92x_4 = 8{,}53 \\ 0{,}53x_1 + 0{,}55x_2 + 0{,}76x_3 + 9{,}13x_4 = 9{,}99 \end{cases}$
15	$\begin{cases} 10{,}9x_1 + 1{,}2x_2 + 2{,}1x_3 + 0{,}9x_4 = -7 \\ 1{,}2x_1 + 11{,}2x_2 + 1{,}5x_3 + 2{,}5x_4 = 5{,}3 \\ 2{,}1x_1 + 1{,}5x_2 + 9{,}8x_3 + 1{,}3x_4 = 10{,}3 \\ 0{,}9x_1 + 2{,}5x_2 + 1{,}3x_3 + 12{,}1x_4 = 24{,}6 \end{cases}$

2.5. Контрольні запитання

1. Які є типи методів розв'язання систем лінійних алгебраїчних рівнянь?
2. У чому полягає сутність методу Гаусса з вибором головного елемента?
3. У чому полягає ідея методу ітерації?

4. Чим метод Якобі відрізняється від методу ітерацій?

5. За яких умов метод ітерації збігається? Якою тоді є оцінка швидкості збіжності?

6. Які проблеми виникають у розв'язанні систем лінійних алгебраїчних рівнянь великої розмірності та як вони вирішуються?

Лабораторна робота 3.
Чисельні методи розв'язання нелінійних рівнянь з одним невідомим і систем нелінійних рівнянь

3.1. Мета роботи

Вивчення чисельних методів розв'язання нелінійних рівнянь з одним невідомим і систем нелінійних рівнянь; набуття навичок з використання цих методів для пошуку розв'язків нелінійних рівнянь і систем нелінійних рівнянь із застосуванням комп'ютера.

3.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* загальне формулювання задачі пошуку коренів нелінійного рівняння та загальне формулювання задачі розв'язання системи нелінійних рівнянь; *уміти* розв'язувати ці задачі з використанням чисельних методів [1; 2; 6; 9].

3.2.1. Чисельні методи розв'язання нелінійних рівнянь з одним невідомим

Розглянемо рівняння виду:

$$f(x) = 0, \tag{3.1}$$

де $f(x)$ – нелінійна неперервна функція однієї змінної.

Розв'язати рівняння – означає знайти таке x^*, за якого рівняння перетворюється в тотожність (x^* ще називають коренем рівняння). У загальному випадку рівняння може мати багато коренів. Запропоновані чисельні методи розв'язання нелінійних рівнянь дозволяють знаходити один корінь на заданому відрізку $[a, b]$. Водночас на відрізку повинен існувати тільки один корінь.

Розглянемо кілька методів розв'язання нелінійного рівняння (3.1) на відрізку $[a, b]$.

Метод половинного ділення (метод дихотомії). Для розв'язання нелінійного рівняння методом половинного ділення задають відрізок $[a, b]$, на якому існує тільки один розв'язок, і бажана точність $\varepsilon > 0$. На 0-й ітерації методу $[a_0, b_0] = [a, b]$, на k-й ітерації методу маємо поточний відрізок $[a_k, b_k]$. Далі визначають середину відрізка $x_k = \frac{a_k + b_k}{2}$ і перевіряють умову $f(a_k) \times f(x_k) < 0$. Якщо зазначена умова виконується, то $b_{k+1} = x_k$, $a_{k+1} = a_k$. Якщо умова не виконується, то $a_{k+1} = x_k$, $b_{k+1} = b_k$. Ділення відрізка навпіл припиняється з досягненням заданої точності, тобто $|b_k - a_k| \leq \varepsilon$. Тут x_k є наближенням розв'язку на k-й ітерації методу. Очевидно, що

$$|x_{k+1} - x^*| \leq \frac{1}{2}|x_k - x^*| \leq \cdots \leq \left(\frac{1}{2}\right)^k |b - a|,$$

тобто метод збігається з лінійною швидкістю з коефіцієнтом $q = 1/2$.

Метод хорд. Для розв'язання нелінійного рівняння методом хорд задають відрізок $[a, b]$, на якому існує тільки один розв'язок, і точність ε. На 0-й ітерації методу $[a_0, b_0] = [a, b]$, на k-й ітерації методу маємо поточний відрізок $[a_k, b_k]$. Потім через дві точки з координатами $(a_k, f(a_k))$ і $(b_k, f(b_k))$ проводимо відрізок прямої лінії (хорду) та визначаємо точку перетину цієї лінії з віссю абсцис (точка x_k). Якщо $f(a_k) \times f(x_k) < 0$, то праву межу інтервалу переносимо в точку x_k (тобто $b_{k+1} = x_k$, $a_{k+1} = a_k$). Якщо зазначена умова не виконується, то в точку x_k переноситься ліва межа інтервалу ($a_{k+1} = x_k$, $b_{k+1} = b_k$). Пошук розв'язку припиняється з досягненням заданої точності, тобто $|f(x_k)| < \varepsilon$. Точку перетину хорди з віссю абсцис визначають за формулою:

$$x_{k+1} = a_k + \left|\frac{f(a_k)}{f(b_k) - f(a_k)}\right| \times (b_k - a_k). \qquad (3.2)$$

Метод дотичних (метод Ньютона). Для розв'язання нелінійного рівняння методом дотичних задають початкове наближення x_0 і точність ε. Потім у точці $(x_0, f(x_0))$ проводять дотичну до графіка $f(x)$ і визначають точку x_1 перетину дотичної з віссю абсцис. У точці $(x_1, f(x_1))$ знову будують дотичну, обчислюють наступне наближення шуканого розв'язку x_2 і т. д. Зазначена процедура повторюється доти, поки $|f(x_i)| > \varepsilon$. Точка перетину $(k+1)$-ї дотичної з віссю абсцис визначається за формулою:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \qquad (3.3)$$

Умова збіжності методу дотичних: $f(x_0) \times f''(x_0) > 0$. Швидкість збіжності буде квадратичною: $|x_{k+1} - x^*| \le M|x_k - x^*|^2$ для всіх $k > k_0$, $k_0 > 0$, $M > 0$.

Метод простої ітерації. Для розв'язання нелінійного рівняння (3.1) методом ітерацій його потрібно записати у вигляді:

$$x = \varphi(x). \tag{3.4}$$

Задають початкове наближення x_0 і точність ε. Перше наближення розв'язку x_1 знаходять із виразу $x_1 = \varphi(x_0)$, друге – $x_2 = \varphi(x_1)$ і т. д. У загальному випадку $k+1$ наближення обчислюється за формулою $x_{k+1} = \varphi(x_k)$. Зазначена процедура припиняється з досягненням заданої точності, тобто $|f(x_k)| = |x_k - \varphi(x_k)| \le \varepsilon$. Умова збіжності методу ітерацій: $|\varphi'(x)| \le q < 1$ для всіх $x \in [a, b]$. Швидкість збіжності буде лінійною: $|x_k - x^*| \le Mq^k$ для всіх $k > 0$, де $M > 0$.

3.2.2. Чисельні методи розв'язання систем нелінійних рівнянь

Розглянемо систему нелінійних рівнянь виду:

$$\begin{cases} f_1(x_1, x_2, \dots, x_n) = 0 \\ f_2(x_1, x_2, \dots, x_n) = 0 \\ \dots\dots\dots\dots\dots\dots\dots\dots \\ f_n(x_1, x_2, \dots, x_n) = 0 \end{cases}, \tag{3.5}$$

де $f_i(x_1, x_2, \dots, x_n)$, $i = \overline{1, n}$ – деякі нелінійні функції n змінних.

Якщо ввести позначення $x = (x_i)_{i=1}^n$ – вектор-стовпець розмірності n з елементами x_i, $F(x) = \begin{pmatrix} f_1(x) \\ \dots \\ f_n(x) \end{pmatrix}$ – векторна функція розмірності n, елементами якої є функції $f_i(x) = f_i(x_1, x_2, \dots, x_n)$, $i = \overline{1, n}$, то систему (3.5) можна записати у векторному вигляді:

$$F(x) = 0. \tag{3.6}$$

Розв'язати систему (3.6) – означає знайти таке $x^* \in R^n$, для якого $F(x^*) \equiv 0$, тобто $f_i(x_1^*, x_2^*, \dots, x_n^*) \equiv 0 \ \forall i = \overline{1, n}$.

Розглянемо три основні методи розв'язання систем нелінійних рівнянь виду (3.6).

Метод Ньютона. Метод будує ітераційну послідовність $\{x^{(k)}\}$, $k = 0,\ 1,\ 2, \ldots$ $(x^{(k)} \in R^n)$ наближень розв'язку x^* за такою ітераційною формулою (початкове наближення $x^{(0)}$ задається):

$$x^{(k+1)} = x^{(k)} - \left[F'(x^{(k)})\right]^{-1} F(x^{(k)}). \qquad (3.7)$$

Тут $F'(x) = \begin{pmatrix} \frac{\partial f_1(x)}{\partial x_1} & \cdots & \frac{\partial f_1(x)}{\partial x_n} \\ \cdots & \cdots & \cdots \\ \frac{\partial f_n(x)}{\partial x_1} & \cdots & \frac{\partial f_n(x)}{\partial x_n} \end{pmatrix}$, тобто $F'(x^{(k)})$ – матриця розмірності $n \times n$. Ітераційний процес (3.7) триває доти, поки не виконається умова $\left\|F(x^{(k)})\right\| \le \varepsilon$, де ε – задана точність розв'язання задачі (3.6).

Якщо послідовність $\{x^{(k)}\}$, $k = 0,\ 1,\ 2, \ldots$ збігається, то швидкість її збіжності квадратична, тобто $\left\|x^{(k+1)} - x^*\right\| \le M\left\|x^{(k)} - x^*\right\|^2$ починаючи з якогось k. Тут M – деяка додатна константа [1; 2; 6].

Основним недоліком методу Ньютона є те, що він збігається тільки для достатньо близьких до розв'язку початкових наближень $x^{(0)}$.

Метод ітерацій. Для розв'язання системи нелінійних рівнянь (3.6) методом ітерацій її потрібно записати у вигляді $x = \Phi(x)$. Тут $\Phi(x)$ – векторна функція розмірності n від $x \in R^n$. Задаються початкове наближення $x^{(0)}$ і точність ε. Метод будує ітераційну послідовність $\{x^{(k)}\}$, $k = 0,\ 1,\ 2, \ldots$ наближень розв'язків x^* за такою ітераційною формулою:

$$x^{(k+1)} = \Phi(x^{(k)}). \qquad (3.8)$$

Метод ітерацій збігається, якщо $\|\Phi'(x)\| \le q < 1$ для всіх x, що належних деякому околу $V(x^*)$ розв'язку x^* і $x^{(0)} \in V(x^*)$. Швидкість збіжності буде лінійною, тобто $\left\|x^{(k+1)} - x^*\right\| \le q\left\|x^{(k)} - x^*\right\|$ починаючи з якогось k. Тут $0 < q < 1$ – деяка константа [1; 2; 6].

Метод найменших квадратів. Для розв'язання системи нелінійних рівнянь (3.6) методом найменших квадратів треба ввести функцію виду

$$\Phi(x) \equiv \|F(x)\|^2 = \sum_{i=1}^{n} f_i^2(x). \qquad (3.9)$$

Тоді розв'язком задачі (3.6) буде точка, в якій функція $\Phi(x)$ досягає мінімального значення.

Для пошуку точки мінімума функції $\Phi(\mathrm{x})$ можна застосувати функцію **optim**(fn, par) пакету R, де **fn** – ім'я цільової функції, **par** – початкове наближення точки мінімума.

3.3. Контрольні приклади

Приклад 1. Розв'язати методом простої ітерації рівняння $\mathrm{x}^3 - 5\mathrm{x} = 0$ з точністю $\varepsilon = 10^{-4}$.

Розв'язання. Спочатку визначимо відрізок, що містить розв'язок. Зробимо це графічно.

Вводимо функцію лівої частини рівняння:

```
> f = function(x)
+ {
+   return (x^3-5*x+2)
+ }
```

Будуємо графік заданої функції (рис. 3.1):

```
> x = seq(-4, 4, len=100)
> y = f(x)
> plot(x, y, type="l")
> abline(h=0)
> abline(v=0)
```

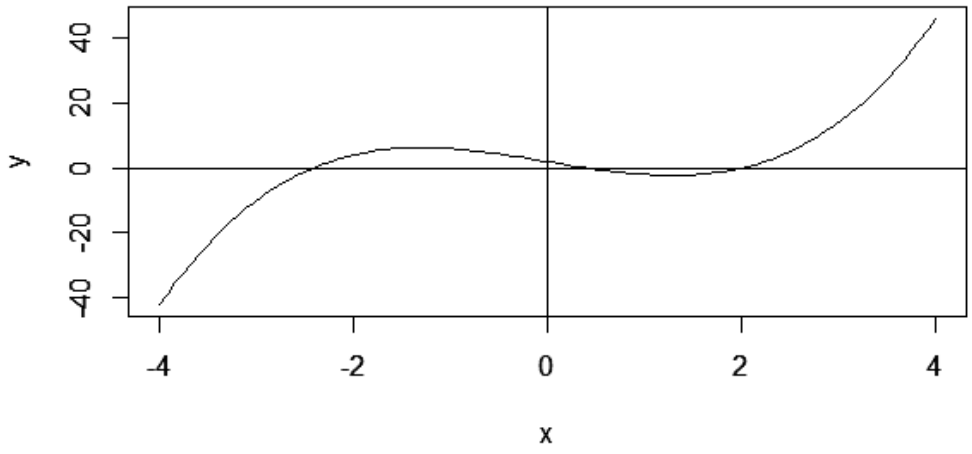

Рис. 3.1. **Графік функції лівої частини рівняння**

Із графіка видно, що коренів три. Знайдемо корінь на відрізку $[2,\ 4]$. Спробуємо застосувати метод простої ітерації. Для цього треба привести рівняння до виду (3.4):

а) розглянемо рівняння $x = \frac{x^3}{5}$, тобто $\varphi(x) = \frac{x^3}{5}$. Тоді $\varphi'(x) = \frac{3x^2}{5}$. Але $\left|\frac{3x^2}{5}\right| \geq \frac{12}{5} = 2.4$ для всіх $x \in [2, 4]$, тобто $|\varphi'(x)| \geq 2.4$. Отже, умова збіжності методу простої ітерації не виконується;

б) розглянемо рівняння $x = \sqrt[3]{5x}$, тобто $\varphi(x) = \sqrt[3]{5x}$. Тоді $\varphi'(x) = \frac{5\times(5x)^{-\frac{2}{3}}}{3} = \frac{5}{3(5x)^{\frac{2}{3}}}$, тому для всіх $x \in [2, 4]$ $|\varphi'(x)| \leq \frac{5}{3\times 10^{\frac{2}{3}}} < 0.36$. Тобто умова збіжності методу простої ітерації виконується $(q = 0.36)$.

Візьмемо $x_0 = 4$, далі $x_1 = \sqrt[3]{5x_0}$, $x_2 = \sqrt[3]{5x_1}$ і т. д. до досягнення точності $|x_k - x_{k-1}| < \varepsilon$.

Вводимо функцію правої частини рівняння (3.4):

```
> g = function(x)
+ {
+   return ((5*x)^(1/3))
+ }
```

Запрограмуємо в пакеті R метод ітерацій для розв'язання нелінійних рівнянь. Процедура має такий вигляд:

```
> MetIter = function(x0,eps)
+ {
+  repeat
+   {
+   x = g(x0)
+   if(abs(x-x0) <= eps ) break
+   x0 = x
+   }
+  return(x)
+ }
```

Виклик процедури для розв'язання нелінійного рівняння з початковим наближенням $x_0 = 4$ та точністю $\varepsilon = 10^{-4}$:

```
> x = MetIter(4, 0.0001)
> x
[1] 2.23609
```

Перевірка розв'язку:

```
> y = g(x)-x
> y
[1] -1.468216e-05
```

Приклад 2. Траєкторія польоту снаряда в безповітряному просторі під дією тільки однієї сили тяжіння описується рівнянням (параболою)

$$y = xtg\theta_0 - \frac{gx^2}{2V_0^2 \cos^2 \theta_0},$$

де V_0 – початкова швидкість у момент пострілу, θ_0 – кут відносно горизонту в момент пострілу (рад);

x, y – відповідно абсциса і ордината в площині польоту снаряда;

g – прискорення вільного падіння (9,81 м/с^2).

Необхідно знайти, під яким кутом був зроблений постріл, якщо снаряд пролетів 3 600 м, а початкова швидкість снаряда дорівнювала 800 км/год.

Розв'язання. Місце падіння снаряду має координати $(x_F,\ y_F) = (3600; 0)$. Тому для знаходження кута θ_0, під яким був зроблений постріл, треба розв'язати нелінійне рівняння $y_F = x_F tg\theta_0 - \frac{gx_F^2}{2V_0^2 \cos^2 \theta_0}$. Перепишемо його у вигляді (3.1), тобто $f(\theta_0) = x_F tg\theta_0 - \frac{gx_F^2}{2V_0^2 \cos^2 \theta_0} - y_F = 0$, і розв'яжемо за допомогою процедури **uniroot()** пакета R:

```
> V0 = 800000/3600; xF = 3600; yF = 0; g = 9.81
> fun = function(Teta)
+   {xF*tan(Teta)-g*xF*xF/(2*V0*V0*cos(Teta)*cos(Teta))-yF}
> fun(0)
[1] -1287.268
> fun(pi/3)
[1] 1086.31
> rc = uniroot(fun, lower=0, upper=pi/3, tol=1e-9)
> rc$root
[1] 0.3984186
> fun(rc$root)
[1] 3.751666e-11
```

Таким чином, постріл було зроблено під кутом 0.3984186 радіан або близько 23 градусів.

Приклад 3. Необхідно розв'язати методом Ньютона систему нелінійних рівнянь $\begin{cases} 2x_1^2 + x_2^2 - 1 = 0 \\ x_1^3 + 6x_1^2 x_2 - 1 = 0 \end{cases}$ з точністю $\varepsilon = 10^{-4}$ (початкове наближення $x_0 = (0.65,\ 0.35)^T$).

Розв'язання. Для розв'язання поставленої задачі реалізуємо метод Ньютона у пакеті R:

```
> Newton = function(Fx, FFx, x, eps)
+ {
+   y = Fx(x)
+   while ((sqrt(t(y)%*%y)) > eps)
+   {
+     yy = FFx(x)
+     x = x - solve(yy)%*%y
+     y = Fx(x)
+   }
+   return(x)
+ }
```

Введення початкових даних:

```
> # Вводимо вектор початкового наближення розв'язку
> x = c(0.65, 0.35)
> # Вводимо векторну функцію F
> Fx = function(x)
+ {
+   f = c(1:2)
+   f[1] = 2*(x[1])^2+(x[2])^2-1
+   f[2] = (x[1])^3+(6*(x[1])^2)*x[2]-1
+   return(f)
+ }
> # Вводимо матрицю частинних похідних
> FFx = function (x)
+ {
+   ff = matrix(nrow=2, ncol=2)
+   ff[1,1] = 4*x[1]
+   ff[1,2] = 2*x[2]
+   ff[2,1] = 3*(x[1]^2)+12*x[1]*x[2]
+   ff[2,2] = 6*(x[1]^2)
+   return(ff)
+ }
```

Виклик запрограмованої процедури для отримання розв'язку і перевірка правильності розв'язку:

```
> X = Newton(Fx, FFx, x, 0.00001)
> X
           [,1]
[1,] 0.6865944
[2,] 0.2391155
> Fx(X)
[1]  1.176896e-10 -2.472814e-10
```

Приклад 4. Пропозиція на деякий товар описується функцією:

$$S = \exp\left(\frac{P+5}{15}\right) - 2,$$

а попит на цей товар описується функцією:

$$D = -3\ln\left(\frac{P}{3}\right) + 15,$$

де S – величина пропозиції товару; D – величина попиту товару, P – ціна товару.

Треба знайти рівноважну ціну та кількість товару, за якої встановиться рівноважна ціна (рівноважна ціна – це ціна, за якої попит дорівнює пропозиції).

Розв'язання. Позначимо через x_1 ціну, а через x_2 – кількість товару. Тоді x_1 та x_2 повинні задовільняти системі:

$$\begin{cases} \exp\left(\frac{x_1+5}{15}\right) - 2 - x_2 = 0, \\ -3\ln\left(\frac{x_1}{3}\right) + 15 - x_2 = 0 \end{cases}.$$

Розв'яжемо цю систему.

Для вибору початкового наближення побудуємо графік (рис. 3.2):

```
> x = seq(10, 40, 0.1)
> y1 = exp((x+5)/15)-2
> y2 = -3*log(x/3)+15
> plot(x, y1, type="l")
> lines(x, y2, type="l")
```

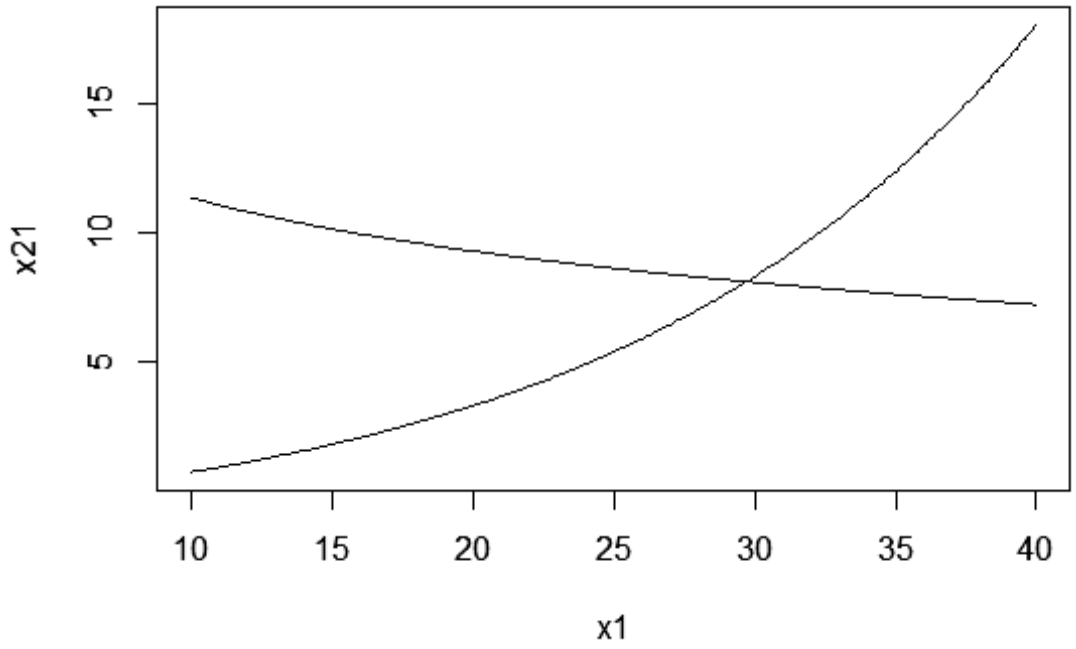

Рис. 3.2. **Графік пропозиції та попиту на товар**

Задамо початкове наближення таким: $x_0 = (28,\ 10)^T$.
Введемо початкові дані задачі:

```
> # Вводимо вектор початкового наближення розв'язку
> x = c(28, 10)
> # Вводимо векторну функцію F
> Fx = function(x)
+ {
+   f = c(1:2)
+   f[1] = exp((x[1]+5)/15)-2-x[2]
+   f[2] = -3*log(x[1]/3)+15-x[2]
+   return(f)
+ }
> # Вводимо матрицю частинних похідних
> FFx = function (x)
+ {
+   ff = matrix(nrow=2, ncol=2)
+   ff[1,1] = exp((x[1]+5)/15)*(1/15)
+   ff[1,2] = -1
+   ff[2,1] = -9/x[1]
+   ff[2,2] = -1
+   return(ff)
+ }
```

Скористаємося записаною в прикладі 1 процедурою і перевіримо отриманий результат:

```
> X = Newton(Fx, FFx, x, 0.00001)
> X
          [,1]
[1,] 29.718483
[2,]  8.120525
>
> Fx(X)
[1] 9.489298e-12 4.145522e-06
```

Таким чином, за ціною товару 8,12 у грошових одиницях попит і пропозиція будуть майже однакові та становитимуть 29 або 30 одиниць товару.

3.4. Порядок виконання роботи та варіанти завдань

3.4.1. Зміст звіту

У теоретичній частині роботи необхідно стисло описати:

постановку задачі розв'язання нелінійного рівняння з однією змінною;

чисельні методи розв'язання нелінійного рівняння з однією змінною;

постановку задачі розв'язання системи нелінійних рівнянь;

чисельні методи розв'язання системи нелінійних рівнянь.

У практичній частині роботи необхідно:

запрограмувати у вигляді окремих модулів методи половинного ділення, хорд, дотичних і простої ітерації для розв'язання нелінійних рівнянь з однією змінною;

запрограмувати у вигляді окремих модулів методи Ньютона, ітерацій та найменших квадратів для розв'язання систем нелінійних рівнянь;

навести тексти складених програм;

розв'язати задачі з індивідуального завдання (табл. 3.1 та табл. 3.2) із застосуванням складених програм та засобів математичного пакета R;

порівняти трудомісткість і швидкість збіжності методів.

3.4.2. Варіанти індивідуальних завдань

Таблиця 3.1

Індивідуальні завдання за варіантами

Варіант	Рівняння
1	$x^2 = \exp(-x^2) - 1$
2	$x = \cos(x)$
3	$x = x^2 - 1$
4	$x = 2\exp(-x)$
5	$x = 3\cos(x)$
6	$x = 2\exp(-x)$
7	$x = \mathrm{tg}(2x) - 1$
8	$x = \ln(x) + 2$
9	$x = \cos(2x)$
10	$x = \exp(-3x)$
11	$x = \exp(-3x) + 1$
12	$x = \exp(-x^2)$
13	$x = \exp(-3x^2)$
14	$x^2 = \exp(-x^2)$
15	$x = \mathrm{tg}(x) - 2$

Таблиця 3.2

Індивідуальні завдання за варіантами

Варіант	Система рівнянь	Початкове наближення
1	2	3
1	$\begin{cases} \ln\left(1 + \frac{x_1 + x_2}{5}\right) - \sin\left(\frac{x_2}{3}\right) - x_1 + 1.1 = 0, \\ \cos\left(\frac{x_1 x_2}{6}\right) - x_2 + 0.5 = 0 \end{cases}$	$x^{(0)} = (1;\ 1)$
2	$\begin{cases} \lg\left(\frac{x_2}{x_3}\right) - x_1 + 1 = 0, \\ 2x_1^2 + x_2 - x_3 - 0.4 = 0, \\ \frac{x_1 x_2}{20} - x_3 + 2 = 0 \end{cases}$	$x^{(0)} = (1;\ 2.2;\ 2)$
3	$\begin{cases} x_1 + x_1^2 - 2x_2 x_3 - 0.1 = 0, \\ x_2 - x_2^2 + 3x_1 x_3 + 0.2 = 0, \\ x_3 + x_3^2 + 2x_1 x_2 - 0.3 = 0 \end{cases}$	$x^{(0)} = (0;\ 0;\ 0)$

Закінчення табл. 3.2

1	2	3
4	$\begin{cases} x_1^3 + x_2^3 - 6x_1 + 3 = 0, \\ x_1^3 - x_2^3 - 6x_2 + 2 = 0 \end{cases}$	$x^{(0)} = (0.5;\ 0.5)$
5	$\begin{cases} x_1 + 3\lg x_1 - x_2^2 = 0, \\ 2x_1^2 - x_1x_2 - 5x_1 + 1 = 0 \end{cases}$	$x^{(0)} = (3.4;\ 2.2)$
6	$\begin{cases} 5x_1 - 6x_2 + 20\lg x_1 + 16 = 0, \\ 2x_1 + x_2 - 10\lg x_2 - 4 = 0 \end{cases}$	$x^{(0)} = (0;\ 0)$
7	$\begin{cases} 0.5\sin\left(\frac{x_2}{3}\right) - x_1 + 1 = 0, \\ 0.3\cos x_1 - x_2 = 0 \end{cases}$	$x^{(0)} = (1;\ 0)$
8	$\begin{cases} x_1^2 - x_2^2 - 1 = 0, \\ x_2(x_2 - 1) - 1 = 0 \end{cases}$	$x^{(0)} = (2;\ 10)$
9	$\begin{cases} 2x_1 - x_2 - 6\lg x_1 - 3 = 0, \\ 15x_1 - 10x_2 - 60\lg x_2 - 6 = 0 \end{cases}$	$x^{(0)} = (0;\ 0)$
10	$\begin{cases} x_1^2 - x_2^2 = 0, \\ x_1x_2 - 4 = 0 \end{cases}$	$x^{(0)} = (0;\ 0)$
11	$\begin{cases} x_1 - x_2 - 6\lg x_1 - 1 = 0, \\ x_1 - 3x_2 - 6\lg x_2 - 2 = 0 \end{cases}$	$x^{(0)} = (0.5;\ 0.2)$
12	$\begin{cases} 2x_1^3 - x_2^2 - 1 = 0, \\ x_1x_2^3 - x_2 - 4 = 0 \end{cases}$	$x^{(0)} = (0;\ 0)$
13	$\begin{cases} x_1 + x_2 + x_3 - 13 = 0, \\ x_1^2 + x_2^2 + x_3^2 - 91 = 0, \\ x_2^2 - x_1x_3 = 0 \end{cases}$	$x^{(0)} = (0;\ 0)$
14	$\begin{cases} \sin x_1 + 2x_2 - 2 = 0, \\ \cos x_1 + x_2 - 1.5 = 0 \end{cases}$	$x^{(0)} = (0;\ 0)$
15	$\begin{cases} \cos(x_1 + 0.5) - x_2 - 2 = 0, \\ \sin x_2 - 2x_1 - 1 = 0 \end{cases}$	$x^{(0)} = (0;\ 0)$

3.5. Контрольні запитання

1. Сформулюйте постановку задачі розв'язання рівнянь з одним невідомим.

2. У чому полягає сутність методу дихотомії? Яка у нього швидкість збіжності?

3. У чому полягає сутність методу хорд? Яка у нього швидкість збіжності?

4. У чому полягає сутність методу дотичних? Яка у нього швидкість збіжності?

5. У чому полягає сутність методу простої ітерації для задачі розв'язання рівняння з одним невідомим? Яка у нього швидкість збіжності?

6. Сформулюйте постановку задачі розв'язання системи нелінійних рівнянь.

7. У чому полягає сутність методу Ньютона? Яка у нього швидкість збіжності?

8. У чому полягає сутність методу ітерацій для задачі розв'язання системи нелінійних рівнянь? Яка у нього швидкість збіжності?

9. У чому полягає сутність методу найменших квадратів?

Лабораторна робота 4. Чисельні методи наближення функцій. Апроксимація та інтерполяція функцій

4.1. Мета роботи

Вивчення методу найменших квадратів (МНК) для практичного розв'язання задач апроксимації функцій. Вивчення методів інтерполяції для практичного розв'язання задачі наближення функцій. Набуття навичок використання зазначених методів для розв'язання задачі наближення функцій із застосуванням комп'ютера.

4.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* загальне формулювання задачі наближення (апроксимації) функцій і, зокрема, задачі інтерполяції; *вміти* розв'язувати задачі: апроксимації функцій методом найменших квадратів [1; 2; 6], наближення функцій за допомогою кусково-лінійної і кусково-квадратичної інтерполяції та полінома Лагранжа [1; 2; 6]*;* *уміти* застосовувати процедури пакета **R** для розв'язання задачі наближення функцій.

4.2.1. Апроксимація функцій

Задача апроксимації функцій у загальному вигляді формулюється таким чином.

Нехай є деяка функція $f(x)$, $f: R^1 \to R^1$, про яку відомо, що в n точках $x_1, x_2, \dots, x_n$ вона приймає, відповідно, значення $y_1, y_2, \dots, y_n$, тобто $y_i = f(x_i)$, $i = \overline{1, n}$. Потрібно відновити її значення за інших значень $x \in [x_1, x_n]$.

Функція $y = f(x)$ може бути невідомою, тобто її значення тільки вимірюються, або дуже складною для обчислень. У цих випадках функцію $f(x)$ намагаються замінити більш простою функцією $g(x, a)$, близькою до $f(x)$, тобто:

$$f(x) \approx g(x, a), \tag{4.1}$$

де $a \in R^k$, $a = (a_1, a_2, \dots, a_k)^T$ – деякі параметри функції g, вид якої відомий.

Процес наближення однієї функції іншою називають **апроксимацією**, а функцію g при цьому називають **апроксимуючою** для f.

Найбільш відомим і ефективним із методів розв'язання задачі апроксимації функцій є **метод найменших квадратів**, який застосовується саме для пошуку параметрів апроксимуючої функції.

Сутність методу найменших квадратів полягає в такому. Вводиться функція (від a) виду:

$$\Phi(a) \equiv \sum_{i=1}^{n} [(y_i - g(x_i, a)]^2, \tag{4.2}$$

яку можна розглядати як міру відхилення функції $g(x, a)$ (за аргументом x) від функції $f(x)$ за сукупністю точок $x_1, x_2, \dots, x_n$. Тоді параметри a функції $g(x, a)$ можна визначити з умови найменшого відхилення $g(x, a)$ від $f(x)$. Тобто параметри a визначаються як точка, у якій функція $\Phi(a)$ досягає за $a \in R^k$ мінімального значення (точка мінімуму). Нехай a^* – шукані параметри. Тоді величини $r_i \equiv y_i - g(x_i, a^*)$, $i = \overline{1, n}$ будуть **залишками** (**залишковими відхиленнями**), які використовуються для аналізу отриманого розв'язку задачі апроксимації.

У випадках, коли функція $g(x, a)$ є лінійною за параметрами a, точку мінімуму функції $\Phi(a)$ можна знайти аналітично з необхідної умови мінімуму 1-го порядку $\Phi'(a) = 0$, тобто:

$$\frac{\partial \Phi}{\partial a_j} = 0, \ \forall j = \overline{1, k}. \tag{4.3}$$

Розглянемо загальний випадок, коли функція $g(x, a)$ лінійна за параметрами a, тобто має вигляд:

$$g(x, a) = \sum_{m=1}^{k} a_m \psi_m(x), \tag{4.4}$$

де $\psi_m(x)$, $m = 1, \dots, k$ – деякі відомі функції.

Тоді з (4.2) – (4.4) отримуємо:

$$\forall j = \overline{1,k}:$$

$$\frac{\partial \Phi}{\partial a_j} = -2\sum_{i=1}^{n}[y_i - g(x_i,a)]\,\psi_j(x_i) = -2\sum_{i=1}^{n}\Big[y_i - \sum_{m=1}^{k} a_m \psi_m(x_i)\Big]\psi_j(x_i) = 0.$$

Звідки для $\forall\, j = \overline{1,k}$:

$$\sum_{m=1}^{k} a_m \left[\sum_{i=1}^{n} \psi_m(x_i)\psi_j(x_i)\right] = \sum_{i=1}^{n} y_i \psi_j(x_i). \qquad (4.5)$$

Таким чином, параметри a можна знайти як розв'язок лінійної системи алгебраїчних рівнянь (4.5), яку можна записати в матричному вигляді:

$$C \cdot a = b, \qquad (4.6)$$

де $c_{jm} = \sum_{i=1}^{n} \psi_m(x_i)\psi_j(x_i)$, , $j = \overline{1,k}$, $m = \overline{1,k}$, $b_j = \sum_{i=1}^{n} y_i \psi_j(x_i)$, $j = \overline{1,k}$.

Розглянемо ще випадок, коли апроксимуюча функція є поліномом заданої $(k-1)$-ї степені. Тоді вона, у загальному випадку, може бути записана у вигляді $g(x,a) = b_0 + b_1 x + b_2 x^2 + \cdots + b_{k-1}x^{k-1}$, де $a \in R^k$, $a_1 = b_0 a_2 = b_1, \ldots, a_k = b_{k-1}$, b_j – коефіцієнти полінома. Тут функція $g(x,a)$ є лінійною за параметрами a і має вигляд (4.4), де $\psi_1(x) = 1$, $\psi_2(x) = x, \ldots$, $\psi_k(x) = x^{k-1}$, тобто $\psi_m(x) = x^{m-1}$.

Тоді параметри a можна знайти, розв'язавши лінійну систему алгебраїчних рівнянь (4.6), з елементами:

$$c_{jm} = \sum_{i=1}^{n} {x_i}^{m-1} {x_i}^{j-1} = \sum_{i=1}^{n} {x_i}^{m+j-2},\ j = \overline{1,k},\ m = \overline{1,k}, \qquad (4.7)$$

$$b_j = \sum_{i=1}^{n} y_i {x_i}^{j-1},\ j = \overline{1,k}. \qquad (4.8)$$

Для пошуку точки мінімума функції декількох змінних можна застосувати функцію **optim**(fn, par) пакету R, де **fn** – ім'я цільової функції, **par** – початкове наближення точки мінімума.

У випадку, коли апроксимуюча функція лінійна за параметрами, систему (4.5) можна розв'язати за допомогою процедури $solve$, описаної в рекомендаціях до лабораторної роботи 1.

4.2.2. Інтерполяція функцій

Загальна задача апроксимації функцій була сформульована в розділі 4.2.1. Якщо параметри $a_1, a_2, \dots, a_k$ апроксимуючої функції $g(x, a)$ визначаються з умови збігу вихідної функції $f(x)$ і функції $g(x, a)$ у точках $x_1, x_2, \dots, x_n$ (тобто $g(x_i, a) = f(x_i),\ \forall\ i = \overline{1, n}$), то такий спосіб апроксимації (наближення) називають **інтерполяцією.**

Кусково-лінійна інтерполяція. Такий вид інтерполяції полягає в тому, що на кожному окремому інтервалі (x_i, x_{i+1}) функція $f(x)$ наближається лінійною функцією $g_i(x)$, що проходить через дві точки (x_i, y_i), (x_{i+1}, y_{i+1}). Неважко перевірити, що така лінійна функція може бути також записана у вигляді:

$$g_i(x) = \frac{(x - x_{i+1})}{(x_i - x_{i+1})} \times y_i + \frac{(x - x_i)}{(x_{i+1} - x_i)} \times y_{i+1}. \qquad (4.9)$$

Таким чином, у використанні кусково-лінійної інтерполяції для таблично заданої функції, з метою обчислення наближеного значення функції $f(x)$ у деякій точці $\hat{x} \in (x_1, x_n)$ дотримуються такого алгоритму. Спочатку знаходять інтервал (x_i, x_{i+1}), якому точка $\hat{x}$ належить, а потім обчислюють наближене значення функції $f(x)$ в точці $\hat{x}$ за формулою $f(\hat{x}) \approx g_i(\hat{x})$.

Якщо $f(x)$ двічі неперервно-диференційовна на відрізку $[x_1, x_n]$, то оцінка похибки у кусково-лінійній інтерполяції має вигляд [1; 2; 6]:

$$|f(x) - g(x)| \le \frac{M_2 h^2}{2},\ \forall x \in (x_1, x_n),$$

де $M_2 = \max\limits_{\xi \in (x_1, x_n)} |f''(\xi)|$, $h = \max\limits_{i=\overline{1,n}} |x_{i+1} - x_i|$.

Кусково-квадратична інтерполяція. Подібно кусково-лінійній інтерполяції можна використати для наближення $f(x)$ на кожному окремому інтервалі $(x_i,\ x_{i+1})$ поліноми більш високого порядку; наприклад, квадратичну функцію $g_i(x)$, яка проходить вже через три точки (x_i, y_i), (x_{i+1}, y_{i+1}), (x_{i+2}, y_{i+2}). Неважко перевірити, що така квадратична функція може бути записана у вигляді:

$$g_i(x) = \frac{(x - x_{i+1})(x - x_{i+2})}{(x_i - x_{i+1})(x_i - x_{i+2})} \times y_i + \frac{(x - x_i)(x - x_{i+2})}{(x_{i+1} - x_i)(x_{i+1} - x_{i+2})} \times y_{i+1} +$$

$$+ \frac{(x - x_i)(x - x_{i+1})}{(x_{i+2} - x_i)(x_{i+2} - x_{i+1})} \times y_{i+2}. \qquad (4.10)$$

Якщо $f(x)$ тричі неперервно-диференційовна на відрізку $[x_1, x_n]$, то оцінка похибки у кусково-квадратичній інтерполяції має вигляд [1; 2; 6]:

$$|f(x) - g(x)| \le \frac{M_3 h^3}{6}, \ \forall x \in (x_1, x_n),$$

де $M_3 = \max\limits_{\xi \in (x_1, x_n)} \left|f^{(3)}(\xi)\right|$, $h = \max\limits_{i=\overline{1,n}} |x_{i+1} - x_i|$.

Інтерполяційний многочлен Лагранжа. Якщо відомі значення функції $f(x)$ у n точках, то поліном мінімальної степені, що інтерполює $f(x)$, буде мати степінь $n - 1$. Він називається **інтерполяційним многочленом Лагранжа,** якщо має вигляд:

$$L_{n-1}(x) = \sum_{i=1}^{n} \left[\prod_{\substack{j=1 \\ j \ne i}}^{n} \left(\frac{x - x_j}{x_i - x_j} \right) \right] y_i. \qquad (4.11)$$

Неважко перевірити, що $L_{n-1}(x_i) = y_i$ для всіх $i = \overline{1, n}$.

Похибка наближення інтерпольованої функції многочленом Лагранжа сильно залежить від розкиду точок x_i. Тобто чим менший інтервал (x_1, x_n), тим точніше $L_{n-1}(x)$ наближає значення функції $f(x)$ у точках $x \in (x_1, x_n)$.

Якщо функція $f(x)$ n разів неперервно-диференційовна на відрізку (x_1, x_n). Тоді для всіх $x \in (x_1, x_n)$:

$$|f(x) - L_{n-1}(x)| \le \frac{M_n h^n}{n!},$$

де $M_n = \max\limits_{\xi \in (x_1, x_n)} \left|f^{(n)}(\xi)\right|$, $h = \max\limits_{i=\overline{1,n}} |x_{i+1} - x_i|$ [1; 2; 6].

4.3. Контрольні приклади

Приклад 1. Залежність (HQ-характеристика насоса) створюваного насосом напору (H) від витрати води (Q) описується функцією $H = H_F - S_F Q^2$, де H_F, S_F – параметри залежності. У результаті проведення натурних експериментів з насосом отримані дані, подані в табл.4.1.

Таблиця 4.1

Дані натурних експериментів

Q, м^3/год	5	20	30	35
H, м вод. ст.	209	194	165	142

Треба побудувати графік HQ-характеристики насоса й оцінити значення напору, який буде створювати насос за витрати води $Q = 32$ м^3/год.

Розв'язання 1. HQ-характеристика насоса задана табл. 4.1. Функція $H(Q) = H_F - S_F Q^2$ є апроксимуючою для неї. Використаємо метод найменших квадратів для оцінювання параметрів H_F, S_F апроксимуючої функції.

Використання процедури **optim** пакета R:

Вводимо початкові дані:

```
> n = 4
> x = c(5, 20, 30, 35)
> y = c(209, 194, 165, 142)
>
> # Задаємо вид апроксимуючої функції
> FunModel = function(x,a)
+ {
+  return( a[1]-a[2]*x^2 )
+ }
```

Визначаємо цільову функцію методу МНК:

```
> FunMNK = function(a)
+ {
+  S = 0
+  for(i in 1:n)
+   S = S + (y[i]-FunModel(x[i],a))^2
+  return( S )
+ }
```

Задаємо початкове значення для параметрів **a** апроксимуючої функції і оцінюємо параметри:

```
> a0 = c(210,1)
> res = optim(fn=FunMNK, par=a0)
> a1 = res$par
> a1
[1] 213.22313362   0.05605678
> FunMNK(a1)
[1] 29.68398
```

Для перевірки правильності отриманого розв'язку, а також наочної інтерпретації результатів, побудуємо графіки (рис. 4.1):

```
# Обчислюємо значення апроксимуючої функції в точках x
ym = FunModel(x,a1)
# Обчислюємо залишки
r = y - ym
a = 0; b = 40; n = 100
```

```
# Обчислюємо n точок на відрізку [a,b]
h = (b-a)/(n-1)
x1 = c(1:n)
for(i in 1:n){ x1[i] = a + h*(i-1) }
# Обчислюємо значення апроксимуючої функції в точках x
y1m = FunModel(x1,a1)
par(mfrow=c(1, 2))                    # задаємо дві графічні підобласті
plot(x, y, type="p", col="red")   # рисуємо перший графік у вигляді точок
lines(x1, y1m, col="blue")            # додаємо в 1-у графічну підобласть ще лінію
plot(x, r, type="p")          # рисуємо перший графік у вигляді точок
lines(x, r, col="blue")               # додаємо в 2-у графічну підобласть ще лінію
abline(h=0)
abline(v=0)
```

Порівняльний графік та графік залишків подано на рис. 4.1.

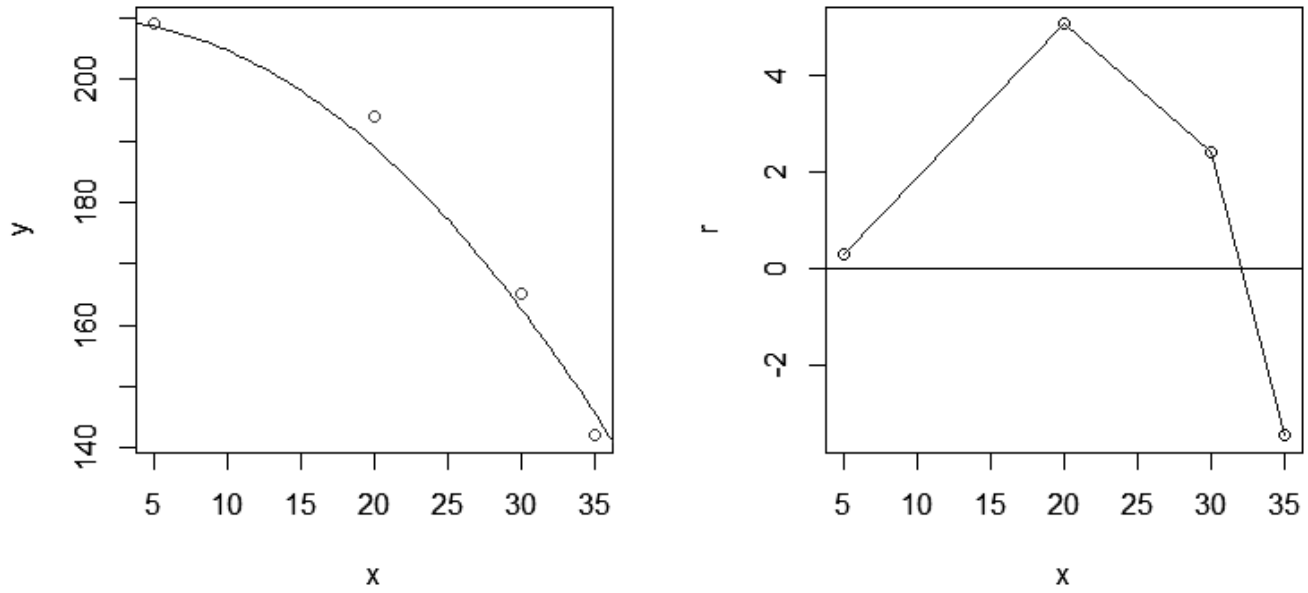

Рис. 4.1. **Порівняльний графік та графік залишків**

Розв'язання 2. Використання процедури solve пакета R

Апроксимуюча функція $H(Q) = H_F - S_F Q^2$ є поліномом другого степеня. Її можна записати у вигляді $g(x, a) = a_1 + a_2 x^2$, де $a \in R^2$ – коефіцієнти полінома ($a_1 = H_F, a_2 = S_F$). Тут функція $g(x, a)$ є лінійною за параметрами a і має вигляд (4.4), де $\psi_1(x) = 1$, $\psi_2(x) = -x^2$. Таким чином, у даному прикладі $n = 4$, $k = 2$. Тоді параметри a можна знайти, розв'язавши лінійну систему рівнянь (4.6).

Для оцінювання параметрів апроксимуючої функції виду (4.4) в загальному вигляді запишемо процедуру *MNKSolve*:

```
MNKSolve = function(x, y, n, Fi, k)
{
	# Формуємо матрицю значень векторної функції Fi в точках x[i] по рядкам
	FiX = matrix(nrow=n, ncol=k)
	for(i in 1:n)
	{
		FiX[i,] = Fi(x[i])
	}
	# Формуємо матрицю C
	C = matrix(nrow=k, ncol=k)
	for(j in 1:k)
	{
		for(m in 1:k)
		{
			S = 0
			for(i in 1:n)
			{
				S = S + FiX[i,j]*FiX[i,m]
			}
			C[j,m] = S
		}
	}
	# Формуємо вектор d
	d = c(1:k)
	for(j in 1:k)
	{
	 S = 0
	 for(i in 1:n)
	 {
		S = S + y[i]*FiX[i,j]
	 }
		d[j] = S
	}
	# Розв'язуємо систему
	a = solve(C, d)
	return(a)
}
```

Застосуємо цю процедуру для розв'язання поставленої задачі:

```
> # Вводимо початкові дані
> n = 4
> x = c(5, 20, 30, 35)
> y = c(209, 194, 165, 142)
> # Вводимо векторну функцію Fi
> Fi = function(x)
+ {
+   z = c(1:2)
+   z[1] = 1
+   z[2] = -x*x
+   return( z )
+ }
> # Знаходимо параметри апроксимуючої функції
> k=2
> a1 = MNKSolve(x, y, n, Fi, k)
> a1
[1] 213.19623060   0.05599409
```

Як бачимо, значення параметрів апроксимуючої функції такі самі, що з використанням процедури **optim**.

Приклад 2. Використовуючи кусково-лінійну, кусково-квадратичну інтерполяцію, поліном Лагранжа, обчисліть у точці $\hat{x} = 3$ наближене значення функції $f(x)$, заданої табл. 4.2.

Таблиця 4.2

Вихідні дані

x	0,5	2,5	4,5	6,5	8,5
y	4,0	205	1240	3840	8750

Розв'язання.

1. Кусково-лінійна інтерполяція. Оскільки точка $\hat{x} = 3$ належить інтервалу $(2.5;\ 4.5)$, то наближене значення $f(3)$ можна обчислити за формулою (4.9), тобто

$$f(3) \approx g_2(3) = \frac{(3-4.5)}{(2.5-4.5)} \times 205 + \frac{(3-2.5)}{(4.5-2.5)} \times 1240 = 463.75.$$

2. Кусково-квадратична інтерполяція. Оскільки точка $\hat{x} = 3$ належить інтервалу $(2.5;\ 4.5)$, то наближене значення $f(3)$ можна обчислити за формулою (4.10), тобто

$$\mathrm{f}(3) \approx \mathrm{g}_2(3) = \frac{(3-4.5)(3-6.5)}{(2.5-4.5)(2.5-6.5)} \times 205 + \frac{(3-2.5)(3-6.5)}{(4.5-2.5)(4.5-6.5)} \times 1240 + \\ + \frac{(3-2.5)(3-4.5)}{(6.5-2.5)(6.5-4.5)} \times 3840 = 497.$$

3. Інтерполяційний многочлен Лагранжа. Проводимо обчислення за формулою (4.11):

$$\mathrm{f}(3) \approx \mathrm{L}_4(3) = \frac{(3-2.5)(3-4.5)(3-6.5)(3-8.5)}{(0.5-2.5)(0.5-4.5)(0.5-6.5)(0.5-8.5)} \times 4 + \\ + \frac{(3-0.5)(3-4.5)(3-6.5)(3-8.5)}{(2.5-0.5)(2.5-4.5)(2.5-6.5)(2.5-8.5)} \times 205 + \\ + \frac{(3-0.5)(3-2.5)(3-6.5)(3-8.5)}{(4.5-0.5)(4.5-2.5)(4.5-6.5)(4.5-8.5)} \times 1240 + \\ + \frac{(3-0.5)(3-2.5)(3-4.5)(3-8.5)}{(6.5-0.5)(6.5-2.5)(6.5-4.5)(6.5-8.5)} \times 3840 + \\ + \frac{(3-0.5)(3-2.5)(3-4.5)(3-6.5)}{(8.5-0.5)(8.5-2.5)(8.5-4.5)(8.5-6.5)} \times 8750 = 357.247.$$

4.4. Порядок виконання роботи і варіанти завдань

4.4.1. Зміст звіту

У теоретичній частині роботи необхідно стисло описати:

постановку задачі наближення функцій;

метод найменших квадратів для апроксимації функцій;

методи інтерполяції функцій.

У практичній частині роботи необхідно:

запрограмувати у вигляді окремого модуля метод найменших квадратів для оцінювання параметрів будь-якої апроксимуючої функції;

запрограмувати у вигляді окремого модуля метод найменших квадратів для оцінювання параметрів апроксимуючого полінома деякої степені;

надати тексти складених програм;

підібрати із застосуванням складених програм найкращу апроксимацію для функції, заданої таблично, серед функцій виду:

1) $y = b_0 + b_1 x$;

2) $y = b_0 + b_1 x + b_2 x^2$;

3) $y = b_0 + b_1 x + b_2 x^2 + b_3 x^3$;

4) $y = b_0 + b_1 \sin(b_2 x) + b_3 \cos(b_2 x)$;

5) $y = b_0 \exp(b_1 x)$.

Побудувати графіки апроксимуючих функцій і відповідних залишків.

Запрограмувати у вигляді окремих модулів методи кусково-лінійної, кусково-квадратичної інтерполяції та інтерполяційний поліном Лагранжа;

надати тексти складених програм;

обчислити в точці $\hat{x} = \frac{(x_1 + x_2)}{2}$ наближене значення функції $f(x)$, заданої таблично в індивідуальному завданні;

побудувати графіки всіх інтерполюючих функцій.

4.4.2. Варіанти індивідуальних завдань

Варіант 1	x	–5	–4	–3	–2	–1	0	1
	y	54,9	37,6	23,7	14,4	8,3	6,6	8,9

Варіант 2	x	0,5	1,0	1,5	2,0	2,5	3,0	3,5	4,0
	y	0,4	0,3	1,0	1,7	2,1	3,4	4,1	5,8

Варіант 3	x	4,5	5,0	5,5	6,0	6,5	7,0	7,5	8,0
	y	7,7	9,4	11,4	13,6	15,6	18,6	21,2	24,1

Варіант 4	x	0,4	0,8	1,2	1,6	2,0	2,4	2,8	3,2	3,6
	y	0,43	0,94	1,91	3,01	4,0	4,56	6,45	8,59	11,15

Варіант 5	x	4,0	4,4	4,8	5,2	5,6	6,0	6,4	6,8
	y	13,88	16,93	20,47	24,15	28,29	32,61	37,41	42,39

Варіант 6	x	1	2	3	4	5	6
	y	2,11	2,45	2,61	2,73	2,75	2,81

Варіант 7	x	0,3	0,6	0,9	1,2	1,5	1,8
	y	4,39	4,75	4,98	5,11	5,12	5,18

Варіант 8	x	1	2	3	4	5	6
	y	0,1	0,21	0,43	0,51	0,62	0,81

Варіант 9	x	1,0	1,5	2,0	2,5	3,0	3,5	4,0
	y	4,11	4,16	4,23	4,29	4,36	4,42	4,53

Варіант 10	x	0,5	1,0	1,5	2,0	2,5	3,0	3,5	4,0
	y	2,47	2,86	3,01	2,91	2,55	2,11	2,61	1,25

Варіант 11	x	–1	0	1	2	3	4	5
	y	2,4	4,3	4,3	0,1	–6,1	–15,5	–26,5

Варіант 12	x	0,16	0,17	0,18	0,19	0,20	0,21
	y	6,61	6,5	6,1	5,80	4,3	3,81

Варіант 13	x	–5	–4	–3	–2	–1	0	1
	y	45,0	32,5	20,7	13,9	10,4	6,5	11,1

Варіант 14	x	–5	–4	–3	–2	–1	0	1
	y	37,0	29,6	18,5	16,2	9,2	9,5	9,1

Варіант 15	x	–1	0	1	2	3	4	5
	y	–0,6	3,4	3,4	–2,2	–12,6	–28,8	–48,9

4.5. Контрольні запитання

1. Сформулюйте постановку задачі наближення функцій.

2. У чому полягає метод найменших квадратів для апроксимації функцій?

3. Яким чином проводиться аналіз побудованої апроксимуючої функції?

4. У чому полягає кусково-лінійна інтерполяція функцій? Коли її слід застосовувати?

5. У чому полягає кусково-квадратична інтерполяція функцій? Коли її слід застосовувати?

6. Що називають інтерполяційним поліномом Лагранжа? Коли його слід застосовувати?

7. Що називають сплайном? Чим він відрізняється від інших методів інтерполяції?

Лабораторна робота 5. Чисельні методи розв'язання задачі Коші для звичайних диференціальних рівнянь

5.1. Мета роботи

Вивчення однокрокових і багатокрокових чисельних методів для практичного розв'язання задачі Коші для звичайних диференціальних рівнянь, набуття навичок використання цих методів для розв'язання задачі Коші із застосуванням комп'ютера.

5.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* загальне формулювання задачі Коші для звичайних диференціальних рівнянь; *уміти* розв'язувати цю задачу з використанням однокрокових і багатокрокових чисельних методів [1; 2; 6].

Задача Коші для звичайного диференціального рівняння першого порядку виду:

$$y' = f(x, y) \tag{5.1}$$

полягає в такому: знайти функцію $y = y(x)$ на заданому відрізку $[a, b]$, що задовольняє рівнянню (5.1) і початковій умові

$$y(a) = y_0, \tag{5.2}$$

де y_0 задано.

Чисельні методи розв'язання задачі (5.1), (5.2) знаходять розв'язок (тобто функцію $y(x)$ на відрізку $[a,b]$) у табличному вигляді, а саме у вигляді набору точок (x_i, y_i), $i = \overline{0,n}$, де $x_0 = a$, $x_i = x_0 + i \times h$, $i = \overline{1,n}$, $h = \frac{b-a}{n}$, n – задане число розбиття відрізка $[a,b]$, y_i, $i = \overline{1,n}$ – знайдені наближені значення функції $y(x)$ у вузлах сітки x_i.

Розглянемо кілька методів розв'язання задачі Коші для диференціального рівняння (5.1) на відрізку $[a,b]$.

5.2.1. Однокрокові методи

Метод Ейлера. Значення y_i обчислюють рекурентно за формулою:

$$y_{i+1} = y_i + h \times f(x_i, y_i),\ i = \overline{0, n-1}.$$

Похибка методу Ейлера, оцінювана для величини $|y_n - y(x_n)|$, має порядок $O(h)$ [1 – 3; 5; 6].

Перша модифікація методу Ейлера. Значення y_i обчислюють рекурентно за формулами:

$$y_{i+\frac{1}{2}} = y_i + \frac{h}{2} f(x_i, y_i),\ y_{i+1} = y_i + h \times f(x_i + \frac{h}{2}, y_{i+\frac{1}{2}}),\ i = \overline{0, n-1}.$$

Похибка першої модифікації методу Ейлера, оцінювана для величини $|y_n - y(x_n)|$, має порядок $O(h^3)$ [1 – 3; 5; 6].

Друга модифікація методу Ейлера. Значення y_i обчислюють рекурентно за формулами:

$$y^*_{i+1} = y_i + h \times f(x_i, y_i),\ y_{i+1} = y_i + \frac{h}{2}(f(x_i, y_i) + f(x_{i+1}, y^*_{i+1})),\ i = \overline{0, n-1}.$$

Похибка другої модифікації методу Ейлера, оцінювана для величини $|y_n - y(x_n)|$, має порядок $O(h^3)$ [1 – 3; 5; 6].

Метод Рунге – Кутта четвертого порядку. Значення y_i обчислюють рекурентно за формулами:

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4),\ i = \overline{0, n-1},$$

де $k_1 = h \times f(x_i, y_i)$, $k_2 = h \times f(x_i + \frac{h}{2}, y_i + \frac{k_1}{2})$, $k_3 = h \times f(x_i + \frac{h}{2}, y_i + \frac{k_2}{2})$, $k_4 = h \times f(x_i + h, y_i + k_3)$.

Похибка методу Рунге – Кутта четвертого порядку, оцінювана для величини $|y_n - y(x_n)|$, має порядок $O(h^4)$ [1 – 3; 5; 6].

5.2.2. Багатокрокові методи

У **методі прогнозу та корекції** четвертого порядку для розв'язання задачі Коші (5.1) значення y_i у вузлах сітки x_i, $i = \overline{0, n}$ обчислюють рекурентно за формулами:

$$y_{i+1}^{(pred)} = y_i + \frac{h}{24}(-9 \overleftarrow{\rightarrow} f_{i-3} + 37 f_{i-2} - 59 f_{i-1} + 55 \overleftarrow{\rightarrow} f_i),$$

$$f_{i+1}^{(pred)} = f(x_{i+1},\ y_{i+1}^{(pred)}),$$

$$y_{i+1} = y_i + \frac{h}{24}(\overleftarrow{\rightarrow} f_{i-2} - 5 f_{i-1} + 19 \overleftarrow{\rightarrow} f_i + 9 \overleftarrow{\rightarrow} f_{i+1}^{(pred)}),$$

де $f_i = f(x_i,\ y_i)$.

До початку розрахунків за методом прогнозу та корекції четвертого порядку необхідно три перші кроки зробити будь-яким однокроковим методом (наприклад, методом Рунге – Кутта).

Похибка методу прогнозу та корекції четвертого порядку, оцінювана для величини $|y_n - y(x_n)|$, має порядок $O(h^4)$ [1; 2; 6].

5.3. Контрольні приклади

Приклад 1. Розв'язати методом Ейлера задачу Коші: $y' = y - \frac{2x}{y}$, $y(0) = 1$ на відрізку [0,1] із числом розбиття відрізка $n = 80$.

Розв'язання. Складемо процедуру для метода Ейлера в пакеті R:

```
> eiler = function(a, b, y0, n)
+ {
+   h = (b-a)/n
+   x = c(1:n)
+   y = c(1:n)
+   x[1] = a
+   y[1] = y0
+   for (i in 1:n)
+   {
+     x[i+1] = x[i]+h
+     y[i+1] = y[i]+h*FunPr(x[i],y[i])
+   }
+   return(cbind(x,y))
+ }
```

Вводимо дані для розв'язання задачі:

```
> # задаємо рівняння (функцію правої частини)
> FunPr = function(x,y)
+ {
+   return (y - 2*x/y)
+ }
> a = 0
> b = 1
> y0 = 1
> n = 80
```

Викликаємо записану процедуру й отримуємо розв'язок диференціального рівняння у вигляді таблично поданої функції:

```
> Y = eiler(a, b, y0, n)
> Y
             x        y
  [1,] 0.0000 1.000000
  [2,] 0.0125 1.012500
  [3,] 0.0250 1.024848
  [4,] 0.0375 1.037048
  [5,] 0.0500 1.049107
  [6,] 0.0625 1.061030
  [7,] 0.0750 1.072820
  [8,] 0.0875 1.084483
  [9,] 0.1000 1.096022
 [10,] 0.1125 1.107441
 ..........................
 [80,] 0.9875 1.732003
 [81,] 1.0000 1.739400
```

Будуємо графік отриманого розв'язку:

```
> x = Y[,1]
> y = Y[,2]
> plot (x,y,type = "l")
```

Графік подано на рис. 5.1.

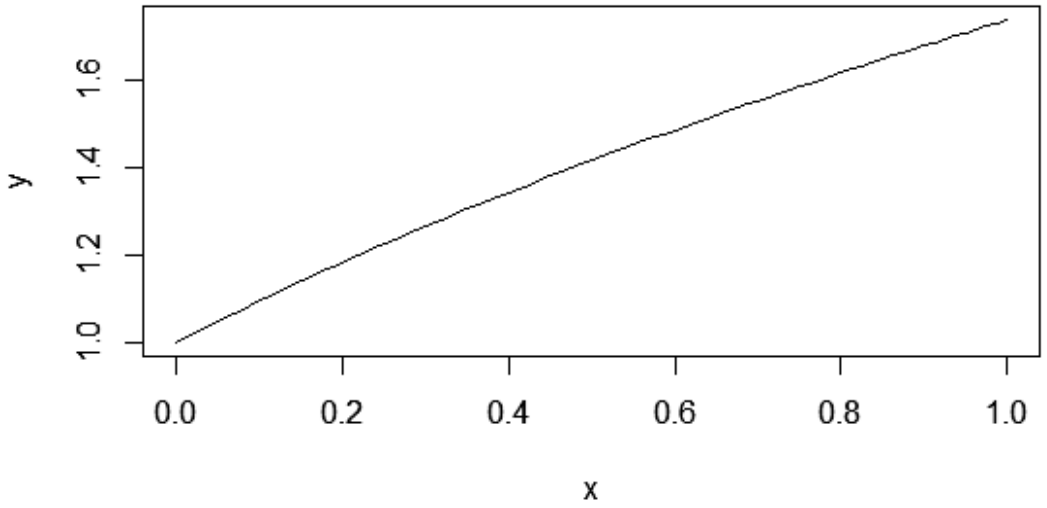

Рис. 5.1. **Графік розв'язку**

Приклад 2. Процес руху автомобіля на площині в найпростішому випадку можна описати системою диференціальних рівнянь [16]:

$$\begin{cases} \dfrac{\partial x}{\partial t} = V\cos\theta \\ \dfrac{\partial y}{\partial t} = V\sin\theta \\ \dfrac{\partial \theta}{\partial t} = -\dfrac{V}{WCtg\varphi + \dfrac{w}{2}} \end{cases}, \qquad (5.3)$$

де (x, y) – координати точки M на площині xOy;

θ – кут між повздовжньою віссю автомобіля і віссю Ox;

V – швидкість автомобіля;

φ – кут повороту передніх коліс відносно повздовжньої осі автомобіля;

W – відстань між передньою і задньою осями (колісна база);

w – відстань між колесами автомобіля на задній осі (колія задніх коліс) (рис. 5.2).

У моделі (5.3) усі кути вимірюються в радіанах.

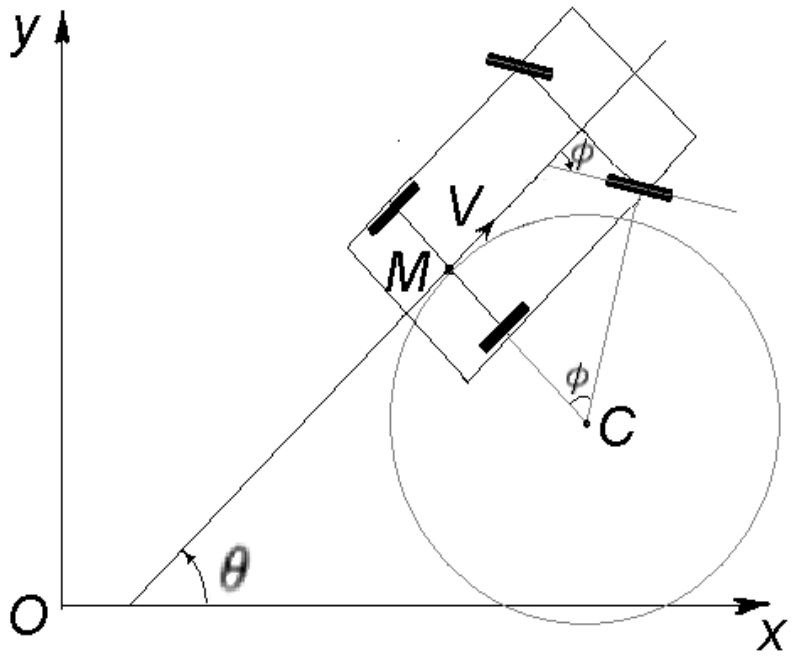

Рис. 5.2. **Модель руху автомобіля на площині**

Треба визначити траєкторію руху автомобіля (тобто точки M) впродовж 5 секунд, якщо його швидкість V була постійна і дорівнювала 5 км/год; кут φ також був постійним і дорівнював 30^0; $W = 2{,}47$ м; $w = 1{,}456$ м; на початку руху точка M мала координати $(5;\ 2)$, а кут $\theta = 0$.

Розв'язання. Оскільки в даному прикладі маємо три координати (x, y, θ), які змінюються у часі, то процедуру для метода Ейлера перепишемо в такому вигляді:

```
MetEiler = function(a, b, Y0, n)
{
  t = c(1:n)
  Y = matrix(0, nrow=n, ncol=3)
  h = (b-a)/(n-1)
  t[1] = a
  Y[1,] = Y0
  for (i in 1:(n-1))
  {
    t[i+1] = t[i] + h
    Y[i+1,] = Y[i,] + h*FunPr(t[i],Y[i,])
  }
  return(cbind(t,Y))
}
```

Вводимо дані для розв'язання задачі:

```
> W = 2.47; w = 1.456
> V = function(t){ return (5) }
> Fi = function(t){ return (pi/6) }
>
> # задаємо рівняння (функцію правої частини)
> FunPr = function(t,Y)
+ {
+   Yt = c(1:3)
+   Yt[1] = V(t)*cos(Y[3])
+   Yt[2] = V(t)*sin(Y[3])
+   Yt[3] = -V(t)/(W/tan(Fi(t))+w/2)
+   return (Yt)
+ }
>
> n = 20
> a = 0
> b = 5
> Y0 = c(5, 2, 0)
```

Викликаємо записану процедуру й отримуємо розв'язок. Будуємо графік отриманого розв'язку (рис. 5.3):

```
> tY = MetEiler(a, b, Y0, n)
> x = tY[,2]
> y = tY[,3]
> plot(x, y, type="l")
```

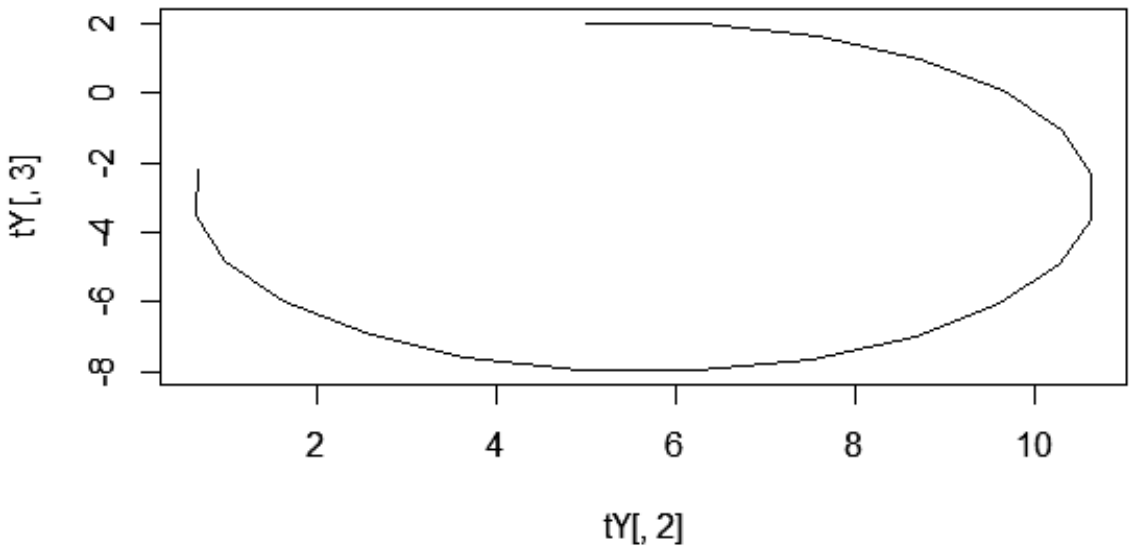

Рис. 5.3. **Траєкторія руху автомобіля**

Приклад 3. Припустимо, що рух гоночного автомобіля описується рівнянням $y' = \frac{1.3y}{t}$, де y – відстань, пройдена за час t з моменту початку руху.

Нехай через півгодини автомобіль віддалився на 75 км. Необхідно визначити на яку відстань він від'їде через 4 години після початку руху.

Розв'язання. Для розв'язання скористаємося записаною в прикладі 1 процедурою *eiler*:

```
> # задаємо рівняння (функцію правої частини)
> FunPr = function(x,y)
+ {
+   return (1.3*y/x)
+ }
> a = 0.5
> b = 4
> y0 = 75
> n = 100
> Y = eiler(a, b, y0, n)
> Y[n+1,2]
       y
1106.619
```

Таким чином, за 4 години гоночний автомобіль подолає відстань 1 106,619 км.

5.4. Порядок виконання роботи і варіанти завдань

5.4.1. Зміст звіту

У теоретичній частині роботи необхідно стисло описати:

постановку задачі Коші для звичайного диференціального рівняння першого порядку;

чисельні однокрокові методи розв'язання задачі Коші для звичайного диференціального рівняння першого порядку;

метод прогнозу та корекції розв'язання задачі Коші для звичайного диференціального рівняння першого порядку.

У практичній частині роботи необхідно:

запрограмувати у вигляді окремих модулів метод Ейлера, першу та другу модифікації методу Ейлера та метод Рунге – Кутта четвертого порядку з метою розв'язання задачі Коші для рівняння (5.1);

запрограмувати у вигляді окремого модуля алгоритм, за яким можна розв'язати задачу Коші із заданою точністю;

запрограмувати у вигляді окремого модуля метод прогнозу та корекції для розв'язання задачі Коші для рівняння (5.1);

надати тексти складених програм;

розв'язати задачу Коші для диференціального рівняння (5.1) чисельно методами Ейлера і його модифікаціями та методом Рунге – Кутта із числом розбиття відрізка $n = 10$ і $n = 20$;

розв'язати задачу Коші для диференціального рівняння (5.1) чисельно методом прогнозу та корекції із числом розбиття відрізка $n = 10$ і $n = 20$;

побудувати графіки отриманих розв'язків (в одному графічному вікні);

розв'язати задачу Коші для диференціального рівняння (5.1) чисельно методом Ейлера с точністю $\varepsilon = 10^{-7}$.

5.4.2.Варіанти індивідуальних завдань

Таблиця 5.1

Індивідуальні завдання за варіантами

Варіант	Рівняння	Інтервал	Умова
1	2	3	4
1	$y' = x^2 + y^2$	$[0,1]$	$y(0) = 0$
2	$y' = 1 + xy^2$	$[0,1]$	$y(0) = 0$
3	$y' = \frac{y}{x+1} - y^2$	$[0,1]$	$y(0) = 1$
4	$y' = \frac{y}{x}$	$[1,25]$	$y(1) = 1$
5	$y' = \frac{x^2 - y^2}{xy}$	$[2,25]$	$y(2) = 1$
6	$y' = \frac{1}{2}xy$	$[0,1]$	$y(0) = 1$
7	$y' = \frac{xy}{x^2 - y^2}$	$[2,25]$	$y(2) = 1$
8	$y' = \frac{x + y - 3}{x - y - 1}$	$[2,25]$	$y(2) = 1$
9	$y' = \frac{2xy - 1}{4x + 2y + 5}$	$[2,25]$	$y(2) = 1$
10	$y' = \frac{3x - y + 2}{2x - 3y + 1}$	$[3,30]$	$y(3) = 5$
11	$y' = \frac{xy^3}{x^2 + y^2}$	$[5,25]$	$y(5) = 7$

Закінчення табл. 5.1

1	2	3	4
12	$y' = \frac{y^4 + y}{x^2}$	$[2,34]$	$y(2) = 10$
13	$y' = \frac{2y + y^2}{x}$	$[2,22]$	$y(2) = 8$
14	$y' = \frac{3y^2 - y}{2 + x}$	$[3,33]$	$y(3) = 7$
15	$y' = \frac{5y + y^2}{22 - x}$	$[1,21]$	$y(1) = 18$

5.5. Контрольні запитання

1. Сформулюйте постановку задачі Коші для звичайного диференціального рівняння першого порядку.
2. Сформулюйте постановку задачі Коші для системи звичайних диференціальних рівнянь першого порядку.
3. У чому полягає ідея методу Ейлера та його модифікацій?
4. У чому полягає ідея методу Рунге – Кутта четвертого порядку?
5. Чим відрізняються багатокрокові методи розв'язання задачі Коші від однокрокових?
6. У чому полягає ідея методу прогнозу та корекції?
7. Від чого залежить точність розв'язку задачі Коші будь-яким методом?

Лабораторна робота 6.
Чисельні методи розв'язання лінійної крайової задачі для звичайних диференціальних рівнянь 2-го порядку

6.1. Мета роботи

Вивчення чисельних методів інтегрування диференціальних рівнянь для практичного розв'язання крайової задачі для звичайних диференціальних рівнянь, набуття навичок використання цих методів для розв'язання крайової задачі із застосуванням комп'ютера.

6.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: знати загальне формулювання крайової задачі для звичайних диференціальних рівнянь другого порядку; вміти розв'язувати цю задачу з використанням чисельного методу кінцевих різниць [1; 2; 6].

Крайова задача для звичайного диференціального рівняння другого порядку виду:

$$F(x,\ y,\ y',\ y'') = 0 \tag{6.1}$$

полягає в такому: знайти функцію $y = y(x)$ на заданому відрізку $[a, b]$, що задовольняє рівнянню (10.1) і крайовим умовам:

$$\begin{cases} \varphi_1(y(a), y'(a)) = 0 \\ \varphi_2(y(b), y'(b)) = 0 \end{cases}, \tag{6.2}$$

де F, φ_1, φ_2 – задані неперервні функції відповідного числа аргументів.

Крайова задача (6.1) – (6.2) є **лінійною**, якщо функції F, φ_1, φ_2 лінійні відносно y, y', y''. Таким чином, лінійна крайова задача для звичайного диференціального рівняння другого порядку полягає в такому: знайти функцію $y = y(x)$ на заданому відрізку $[a, b]$, що задовольняє лінійному рівнянню виду:

$$y'' + p(x)\, y' + q(x)\, y = f(x)) \tag{6.3}$$

і лінійним крайовим умовам:

$$\begin{cases} \alpha_0 y(a) + \alpha_1 y'(a) = A \\ \beta_0 y(b) + \beta_1 y'(b) = B \end{cases}, \tag{6.4}$$

де $p(x)$, $q(x)$, $f(x)$ – задані неперервні функції від x;

α_0, α_1, β_0, β_1, A, B – задані константи, причому $|\alpha_0| + |\alpha_1| \neq 0$, $|\beta_0| + |\beta_1| \neq 0$.

Чисельні методи розв'язання задачі (6.1) – (6.2), а зокрема і задачі (6.3) – (6.4), знаходять розв'язок (тобто функцію $y(x)$ на відрізку $[a, b]$)

у табличному вигляді, а саме у вигляді набору точок (x_i, y_i), $i = \overline{0,n}$, де $x_0 = a$, $x_i = x_0 + i \times h$, $i = \overline{1,n}$, $h = \frac{b-a}{n}$, n – задане число розбиття відрізка $[a,b]$, y_i, $i = \overline{0,n}$ – обчислені наближені значення функції $y(x)$ у вузлах сітки x_i.

Розглянемо **метод кінцевих різниць** розв'язання лінійної крайової задачі (6.3) – (6.4).

Замінимо приблизно в кожному внутрішньому вузлі x_i похідні $y'(x_i)$ та $y''(x_i)$ кінцево-різницевими формулами:

$$y'(x_i) \approx \frac{y_{i+1} - y_{i-1}}{2h}, \quad y''(x_i) \approx \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2}, \tag{6.5}$$

а на кінцях відрізка $[a,b]$ покладемо

$$y'(x_0) \approx \frac{y_1 - y_0}{h}, \quad y'(x_n) \approx \frac{y_n - y_{n-1}}{h}. \tag{6.6}$$

Також введемо позначення: $p_i = p(x_i)$, $q_i = q(x_i)$, $f_i = f(x_i)$, $i = \overline{1,n}$.

Використовуючи формули (6.5) – (6.6), приблизно замінимо рівняння (6.3) і крайові умови (6.4) системою рівнянь:

$$\begin{cases} \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} + p_i \frac{y_{i+1} - y_{i-1}}{2h} + q_i y_i = f_i \ (i = \overline{1, n-1}) ; \\ \alpha_0 y_0 + \alpha_1 \frac{y_i - y_0}{h} = A ; \\ \beta_0 y_n + \beta_1 \frac{y_n - y_{n-1}}{h} = B . \end{cases} \tag{6.7}$$

Отримана система (6.7) є лінійною алгебраїчною системою $(n+1)$ рівнянь відносно $(n+1)$ невідомих y_i, $i = \overline{0,n}$. Розв'язавши її, якщо це можливо, отримаємо таблицю наближених значень шуканої функції $y(x)$ на відрізку $[a,b]$.

Додамо, що систему (6.7) для її розв'язання краще записати у вигляді:

$$\begin{cases} (\alpha_0 h - \alpha_1) y_0 + \alpha_1 y_1 = Ah ; \\ (1 - \frac{p_i h}{2}) y_{i-1} + (q_i h^2 - 2) y_i + (1 + \frac{p_i h}{2}) y_{i+1} = f_i h^2 \ (i = \overline{1, n-1}) ; \\ -\beta_1 y_{n-1} + (\beta_0 h + \beta_1) y_n += Bh . \end{cases} \tag{6.8}$$

Похибка методу кінцевих різниць, оцінювана для величин $|y_i - y(x_i)|$, $i = \overline{0,n}$ має порядок $O(h^2)$ [1; 2; 6].

6.3. Контрольні приклади

Приклад 1. Розв'язати методом кінцевих різниць крайову задачу $x^2y'' + xy' = 1,\ y(1) = 0,\ \ y(1.4) = 0{,}05661$ із числом розбиття відрізка $n = 4$.

Розв'язання. Оскільки всі функції в цій задачі лінійні відносно y, y', y'', то це лінійна крайова задача. Тоді, насамперед, необхідно привести дану крайову задачу до виду (6.3) – (6.4), записавши диференціальне рівняння так: $y'' + \frac{1}{x}y' = \frac{1}{x^2}$, тобто $p(x) = \frac{1}{x}$, $q(x) = 0$, $f(x) = \frac{1}{x^2}$.

Тому вхідні дані для розв'язання задачі запишемо так:

```
p1 = function(x) { return(1/x) }
q1 = function(x) { return(0) }
f1 = function(x) { return(1/(x^2)) }
a1 = 1
b1 = 1.4
alpha1 = c(1,0)
beta1 = c(1,0)
A1 = 0
B1 = 0.05661
```

Систему (6.8) будемо розв'язувати за допомогою процедури *solve* пакета R. Процедура формування системи лінійних рівнянь (6.8) і її розв'язання має такий вигляд:

```
LinDifUrKZ = function(p, q, f, a, b, alpha, A, beta, B, n)
{
    h = (b-a)/n
    x = c(1:(n+1))
    for (i in 1:(n+1))
    {
        x[i] = a + (i-1)*h
    }
```

```
    C = matrix(0, nrow=(n+1), ncol=(n+1))
    d = c(1:(n+1))

    C[1,1] = (alpha[1])*h - alpha[2]
    C[1,2] = alpha[2]
    d[1] = A*h
    for (i in 2:n)
    {
       C[i,(i-1)] = 1 - h*p(x[i])/2
       C[i,i] = (h^2)*q(x[i]) - 2
       C[i,(i+1)] = 1 + h*p(x[i])/2
       d[i] = f(x[i])*h^2
    }
    C[(n+1),n] = -beta[2]
    C[(n+1),(n+1)] = (beta[1])*h + beta[2]
    d[n+1] = B*h

    y = solve(C, d)
    return(cbind(x,y))
}
```

Для отримання розв'язку викликається записана процедура:

```
n = 4
XY = LinDifUrKZ(p1, q1, f1, a1, b1, alpha1, A1, beta1, B1, n)
```

Побудуємо графік отриманого чисельного розв'язку та порівняємо його з аналітичним розв'язком (рис. 6.1).

```
# Будуємо графік розв'язку
plot(XY[,1], XY[,2], type="l")
# Порівнюємо отриманий розв'язок з точним (аналітичним) розв'язком
kT = 100
x = seq(a1,  b1, len=kT)
y = c(1:kT)
for(i in 1:kT){ y[i] = 0.5*(log(x[i]))^2 }
lines(x, y, type="l", col="red")
```

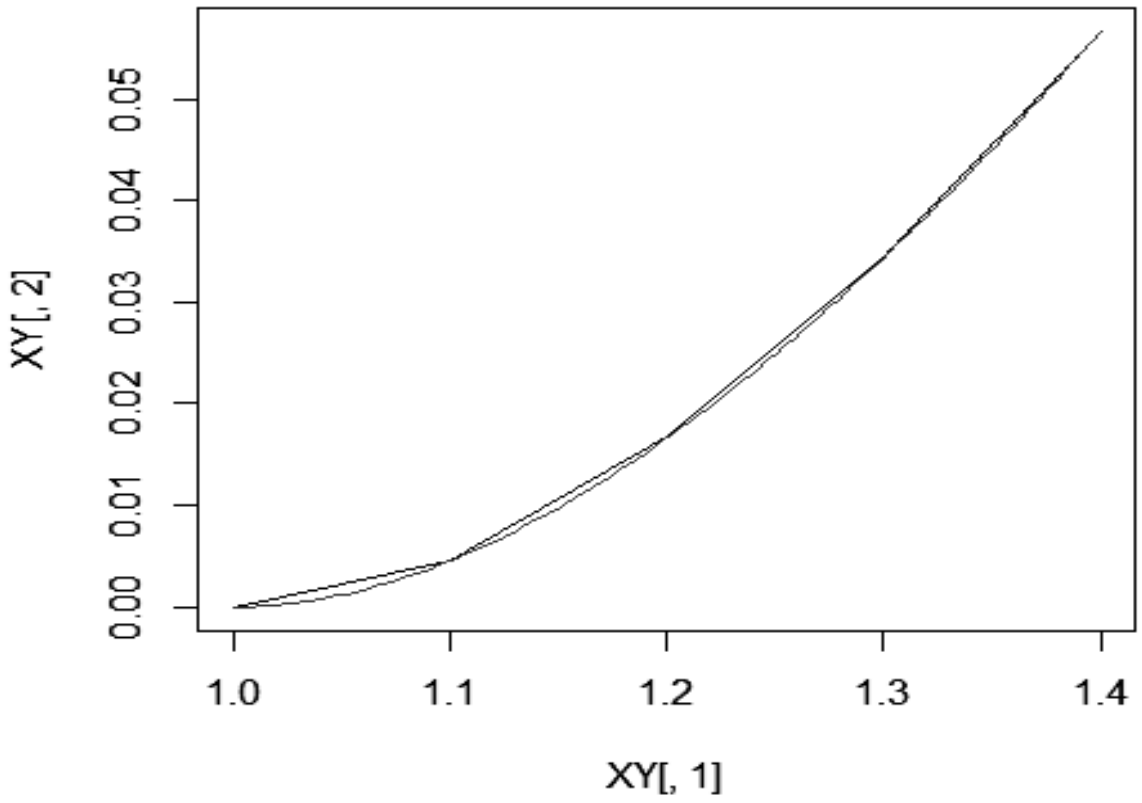

Рис. 6.1. **Графік розв'язку**

6.4. Порядок виконання роботи та варіанти завдань

6.4.1. Зміст звіту

У теоретичній частині роботи необхідно стисло описати:

постановку крайової задачі для звичайного диференціального рівняння другого порядку;

метод кінцевих різниць розв'язання лінійної крайової задачі для звичайного диференціального рівняння другого порядку.

У практичній частині роботи необхідно:

запрограмувати у вигляді окремого модуля метод кінцевих різниць розв'язання лінійної крайової задачі для звичайного диференціального рівняння другого порядку;

надати текст складеної програми;

розв'язати методом кінцевих різниць крайову задачу відповідно варіанту (табл. 6.1) із числом розбиття відрізка $n = 5$ і $n = 10$;

побудувати графік отриманого розв'язку.

6.4.2. Варіанти індивідуальних завдань

У табл. 6.1 подано індивідуальні завдання за варіантами.

Таблиця 6.1

Індивідуальні завдання за варіантами

Варіант	Рівняння	Інтервал	Умова
1	$y'' - 2xy' - 2y = -4x$	$[0,1]$	$y(0) - y'(0) = 0,$ $y(1) = 3.718$
2	$y'' + \frac{x}{2}y' + (1 + 2\pi^2 x^2)y = 4x$	$[0,1]$	$y(0) = 1, y(1) = 1.367$
3	$y'' + x^2y' + (1 - x)y = \frac{x}{x^2 + 3}$	$[0,1]$	$y(0) = 0, y(1) = 0$
4	$y'' + (x - 1)y' + 3.125y = 4x$	$[0,1]$	$y(0) = 1, y(1) = 1.367$
5	$y'' + x^2y' + (1.4 - x)y = \frac{x}{x^2 + 2.5}$	$[0,1]$	$y(0) = 0, y(1) = 0$
6	$y'' + 2xy' + 2y = \frac{2 \cdot (5 - 2x)}{(2 - x)^3}$	$[0,1]$	$y(0) = 1, y(1) = 1.367$
7	$y'' + (1 + x^3)y' + (1 - x^2)y = e^{1-2.5x^2}$	$[0,1]$	$y(0) = 0, y(1) = 0$
8	$y'' + (1.4 + x^3)y' + (1 - x^2)y = e^{1-3x^2}$	$[0,1]$	$y(0) = 0, y(1) = 0$
9	$y'' + x^2y' + (1.8 - x)y = \frac{x}{x^2 + 3.5}$	$[0,1]$	$y(0) = 0, y(1) = 0$
10	$y'' + (1.8 + x^3)y' + (1 - x^2)y = e^{1-3.5x^2}$	$[0,1]$	$y(0) = 0, y(1) = 0$
11	$y'' + x^2y' + (2.2 - x)y = \frac{x}{x^2 + 4}$	$[0,1]$	$y(0) = 0, y(1) = 0$
12	$y'' + (2.2 + x^3)y' + (1 - x^2)y = e^{1-4x^2}$	$[0,1]$	$y(0) = 0, y(1) = 0$
13	$y'' + y' \sin(x) + y = \frac{1}{2.5 + \sin^2(x)}$	$[0,1]$	$y(0) = 0, y(1) = 0$
14	$y'' + x^2y' + (3 - x)y = \frac{x}{x^2 + 5}$	$[0,1]$	$y(0) = 0, y(1) = 0$
15	$y'' + \frac{y'}{\sqrt{x^2 + 2.5}} + y = x$	$[0,1]$	$y(0) = 0, y(1) = 0$

6.5. Контрольні запитання

1. Сформулюйте постановку крайової задачі для звичайного диференціального рівняння другого порядку.

2. Які є чисельні методи для розв'язання крайової задачі для звичайного диференціального рівняння другого порядку?

3. Від чого залежить точність розв'язання крайової задачі чисельним методом?

Змістовий модуль 2
Методи оптимізації

Лабораторна робота 7.
Чисельні методи розв'язання задач одномірної оптимізації та задач безумовної оптимізації

7.1. Мета роботи

Вивчення чисельних методів одномірної оптимізації та методів безумовної оптимізації для практичного розв'язання задач пошуку точки оптимуму функції однієї змінної, пошуку точки оптимуму функції декількох змінних; набуття навичок використання цих методів для розв'язання задач одномірної оптимізації та безумовної оптимізації із застосуванням комп'ютера.

7.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* загальне формулювання задачі одномірної оптимізації та задачі безумовної оптимізації; *уміти* розв'язувати ці задачі з використанням чисельних методів одновимірної чи безумовної оптимізації [4; 9].

7.2.1. Чисельні методи розв'язання задач одномірної оптимізації

Розглянемо нелінійну неперервну функцію $f(x)$ однієї змінної $x \in R^1$. Задача одновимірної оптимізації полягає в знаходженні точки $x^* \in R^1$, у якій функція $f(x)$ приймає мінімальне значення. У загальному випадку функція $f(x)$ може мати кілька точок мінімуму. Запропоновані чисельні методи розв'язання задачі одновимірної оптимізації дозволяють знаходити одну точку мінімуму на заданому відрізку $[a, b]$. Водночас на відрізку повинна існувати тільки одна точка мінімуму.

Розглянемо кілька чисельних методів пошуку точки мінімуму функції $f(x)$ на відрізку $[a, b]$.

Метод половинного ділення (дихотомії). Для пошуку точки мінімуму функції $f(x)$ методом половинного ділення задають відрізок $[a, b]$, на якому існує тільки одна точка мінімуму, і бажана точність $\varepsilon > 0$. На 0-му кроці методу $[a_0, b_0] = [a, b]$, на k-му кроці методу маємо поточний відрізок $[a_k, b_k]$. Далі визначають дві точки $x_{k1} = \frac{a_k+b_k}{2} - \delta$ і $x_{k2} = \frac{a_k+b_k}{2} + \delta$ (відступають в обидва боки від середини відрізка $x_k = \frac{a_k+b_k}{2}$ на величину $\delta \le \frac{\varepsilon}{4}$) і перевіряють умову $f(x_{k1}) < f(x_{k2})$. Якщо зазначена умова виконується, то $b_{k+1} = x_{k2}$, $a_{k+1} = a_k$. Якщо умова не виконується, то $a_{k+1} = x_{k1}$, $b_{k+1} = b_k$. Ділення відрізка навпіл триває доти, поки $|b_k - a_k| > \varepsilon$. Тут справедлива така оцінка швидкості збіжності:

$$|b_k - a_k| \le \frac{1}{2^{\frac{k}{2}}} + \left(1 - \frac{b-a}{2^{\frac{k}{2}}}\right)\delta,$$

тобто метод збігається приблизно з геометричною швидкістю з коефіцієнтом $q = \frac{1}{\sqrt{2}}$.

Недоліком методу половинного ділення є те, що на кожній ітерації методу значення функції $f(x)$ потрібно обчислювати у двох точках x_{k1}, x_{k2}. Це може виявитися істотним, коли кожне обчислення значення функції $f(x)$ вимагає великих ресурсних витрат. Цей недолік усувається в методі золотого перетину.

Метод золотого перетину. Для пошуку точки мінімуму функції $f(x)$ методом золотого перетину задають відрізок $[a, b]$, на якому існує тільки одна точка мінімуму, і бажана точність $\varepsilon > 0$. На 0-му кроці методу $[a_0, b_0] = [a, b]$, на k-му кроці методу маємо поточний відрізок $[a_k, b_k]$. Далі на першій ітерації визначаються дві точки $x_{k1} = a_k + (1 - \lambda)(b_k - a_k)$ і $x_{k1} = a_k + \lambda(b_k - a_k)$, де $\lambda = \frac{-1+\sqrt{5}}{2}$, і перевіряється умова $f(x_{k1}) < f(x_{k2})$. Якщо зазначена умова виконується, то $b_{k+1} = x_{k2}$, $a_{k+1} = a_k$, $x_{k2} = x_{k1}$, $x_{k1} = a_{k+1} + (1 - \lambda)(b_{k+1} - a_{k+1})$. Якщо умова не виконується, то $a_{k+1} = x_{k1}$, $b_{k+1} = b_k$, $x_{k1} = x_{k2}$, $x_{k2} = a_{k+1} + \lambda(b_{k+1} - a_{k+1})$. Таким чином, на всіх наступних ітераціях (починаючи із другої) значення функції $f(x)$ додатково обчислюється тільки в одній точці, друга ж точка (а тобто і значення функції в ній) просто перевизначається. Золотий перетин відрізка триває

доти, поки $|b_k - a_k| > \varepsilon$. Тут справедлива наступна оцінка швидкості збіжності:

$$|b_k - a_k| \le \frac{1}{(1+\lambda)^{k-1}}(b-a),$$

тобто метод збігається з геометричною швидкістю з коефіцієнтом $q = \frac{1}{1+\lambda} = \lambda$.

7.2.2. Чисельні методи розв'язання задач безумовної оптимізації

Розглянемо нелінійну неперервну функцію $f(x)$ кількох змінних $x \in R^n$. Задача безумовної оптимізації полягає в знаходженні точки $x^* \in R^n$, у якій функція $f(x)$ приймає мінімальне значення. Короткий запис задачі безумовної оптимізації:

$$f(x) \to \min,\ x \in R^n. \qquad (7.1)$$

У загальному випадку функція $f(x)$ може мати кілька локальних точок мінімуму. Запропоновані чисельні методи розв'язання задачі багатовимірної оптимізації дозволяють знаходити одну з локальних точок мінімуму.

Розглянемо кілька чисельних методів пошуку точки мінімуму функції $f(x)$, які в принципі мають одну загальну схему.

Узагальнена схема чисельних методів розв'язання задачі безумовної оптимізації. Для пошуку точки мінімуму функції $f(x)$ чисельним методом задають початкове наближення розв'язку $x^{(0)} \in R^n$ й бажану точність $\varepsilon > 0$ розв'язку по градієнту. На k-му кроці методу маємо поточне наближення розв'язку $x^{(k)} \in R^n$. Наступне наближення розв'язку $x^{(k+1)} \in R^n$ визначається за формулою:

$$x^{(k+1)} = x^{(k)} + \alpha_k h^{(k)}, \qquad (7.2)$$

де напрямок пошуку $h^{(k)} \in R^n$ визначається **конкретним** чисельним методом, а кроковий множник $\alpha_k > 0$ зазвичай визначається одним із способів:

а) як точка мінімуму функції однієї змінної $\varphi_k(\alpha) = f(x^{(k)} + \alpha h^{(k)})$, $\alpha > 0$ (повний пошук за напрямком);

б) за алгоритмом дроблення кроку.

Обчислення тривають доти, поки не виконається умова $\|f'(x^k)\| \le \varepsilon$ (хоча можливі й інші критерії "зупинення").

Алгоритм дроблення кроку для визначення крокового множника в чисельних методах розв'язання задачі безумовної оптимізації. В алгоритмі дроблення кроку для визначення крокового множника α_k у чисельних методах розв'язання задачі безумовної оптимізації, що підпадають під схему (7.2), задають параметри:

ρ – максимальне значення кроку, $\rho > 0$;

γ – коефіцієнт рівня спадання функції, $\gamma \in (0, \frac{1}{2})$;

λ – коефіцієнт дроблення кроку, $\lambda \in (0,\ 1)$.

Вектор $h^{(k)}$ повинен бути напрямком спадання функції $f(x)$, тобто задовольняти умові $\langle f'(x^{(k)}), h^{(k)}\rangle < 0$.

На початку вибирається $\alpha = \rho$ і перевіряється нерівність:

$$f(x^{(k)} + \alpha h^{(k)}) \le f(x^{(k)}) + \alpha\gamma\langle f'(x^{(k)}), h^{(k)}\rangle. \qquad (7.3)$$

Якщо нерівність (7.3) не виконується, то α зменшується шляхом множення на λ (тобто $\alpha = \lambda\alpha$) і знову перевіряється (7.3). Якщо нерівність (7.3) виконується, то кроковий множник α_k приймається рівним поточному значенню α.

Якщо вектор $h^{(k)}$ є напрямком спадання функції $f(x)$, то описаний процес дроблення кроку буде скінченим, тобто нерівність (7.3) виконається через скінчене число кроків.

Метод найшвидшого спуску. Метод будує ітераційну послідовність $\{x^{(k)}\}$, $k = 0,\ 1,\ 2,\ldots$, наближень розв'язку за схемою (7.2), де напрямок пошуку $h^{(k)}$ визначається за формулою:

$$h^{(k)} = -f'(x^{(k)}), \qquad (7.4)$$

кроковий множник $\alpha_k > 0$ визначається як точка мінімуму функції однієї змінної $\varphi_k(\alpha) = f(x^{(k)} + \alpha h^{(k)})$. Обчислення триває доти, поки не виконається умова $\|f'(x^k)\| \le \varepsilon$.

Недоліком методу найшвидшого спуску є його досить низька швидкість збіжності для "яружних" (погано обумовлених) функцій.

Метод сполучених градієнтів. Метод будує ітераційну послідовність $\{x^{(k)}\}$, $k = 0,\ 1,\ 2,\ldots$, наближень розв'язку за схемою (7.2), де напрямок пошуку $h^{(k)}$ визначається за формулою:

$$\begin{cases} h^{(k)} = -f'(x^{(k)}), & k \in \{0, n,\ 2n,\ 3n, \ldots\} \\ h^{(k)} = -f'(x^{(k)}) + \beta_{k-1} h^{(k-1)}, & k \notin \{0, n,\ 2n,\ 3n, \ldots\} \end{cases} \tag{7.5}$$

коефіцієнт β_{k-1} визначається за однією з формул:

$$\text{а)} \quad \beta_{k-1} = \frac{\langle f'(x^{(k)}),\ f'(x^{(k)}) - f'(x^{(k-1)})\rangle}{\|f'(x^{k-1})\|^2}; \tag{7.6}$$

$$\text{б)} \quad \beta_{k-1} = \frac{\|f'(x^{(k)})\|^2}{\|f'(x^{(k-1)})\|^2}.$$

Кроковий множник $\alpha_k > 0$ в (7.2) може бути визначений одним із способів:

а) як точка мінімуму функції однієї змінної $\varphi_k(\alpha) = f(x^{(k)} + \alpha h^{(k)})$;

б) за алгоритмом дроблення кроку (7.3).

Обчислення тривають доти, поки не виконається умова $\|f'(x^k)\| \le \varepsilon$.

Перевагою методу сполучених градієнтів є його висока швидкість збіжності для нормальних (не "яружних") функцій.

Недоліком методу сполучених градієнтів є його низька швидкість збіжності для "яружних" (погано обумовлених) функцій.

Метод Ньютона з регулюванням кроку. Метод будує ітераційну послідовність $\{x^{(k)}\}$, $k = 0,\ 1,\ 2,\ldots$, наближень розв'язку за схемою (7.2), де напрямок пошуку $h^{(k)}$ визначається за формулою:

$$h^{(k)} = -\left[f''(x^{(k)})\right]^{-1} f'(x^{(k)}), \tag{7.7}$$

кроковий множник $\alpha_k > 0$ у (7.2) зазвичай визначається за алгоритмом дроблення кроку (7.3), у якому $\rho = 1$. У класичному методі Ньютона кроковий множник α_k завжди приймається рівним 1.

Перевагою методу Ньютона з регулюванням кроку є його висока (квадратична) швидкість збіжності навіть для "яружних" функцій.

Недоліком методу Ньютона є необхідність на кожній ітерації обчислення й обернення матриці других похідних цільової функції (метод другого порядку).

7.3. Контрольні приклади

Приклад 1. Знайдіть точку мінімуму функції $f(x) = x^2 + 10\cos(x)$ методом: половинного ділення з точністю 10^4. Відрізок $[a, b]$, що містить точку мінімуму треба знайти самостійно.

Розв'язання. Спочатку визначимо відрізок, що містить розв'язок. Зробимо це графічно (рис. 7.1).

```
Fun = function(x)
{
	return(x^2 + 10*cos(x))
}

n = 100
a = -10
b = 10
h = (b-a)/(n-1)
x = c(1:n)
for(i in 1:n)
	x[i] = a + h*(i-1)
y = Fun(x)
plot(x, y, type="l")
```

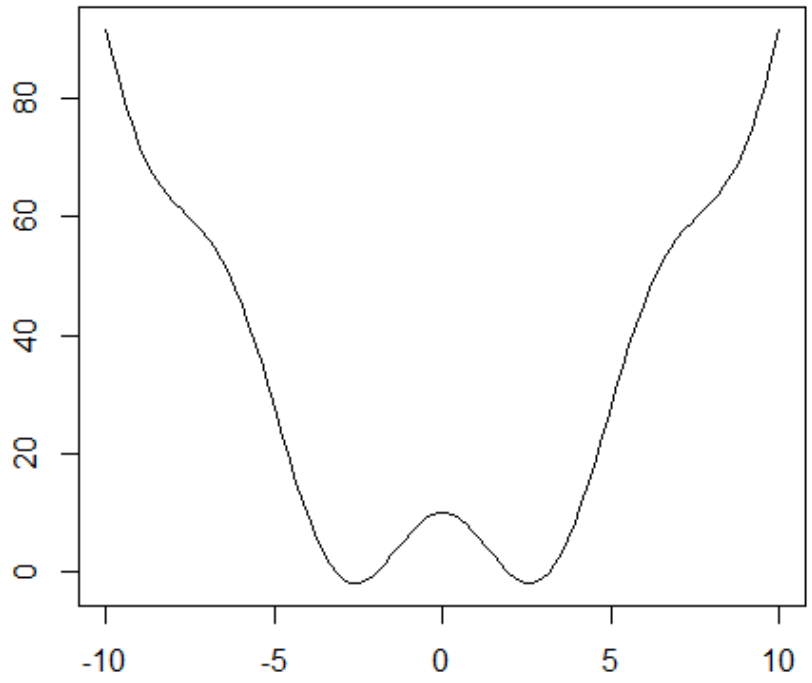

Рис. 7.1. **Графік цільової функції**

Судячи із графіка та виду функції, вона має дві точки мінімуму, причому вони симетричні відносно точки початку координат. Знайдемо праву точку мінімуму, відрізок $[a, b]$ візьмемо рівним $[0,\ 5]$.

Реалізуємо метод половинного ділення у вигляді процедури засобами пакета R, параметр методу δ візьмемо рівним $\frac{\varepsilon}{4}$:

```
BiSection = function(f, a, b, eps)
{
        del = eps/4
        while ((b-a) > eps)
        {
                x = (a+b)/2
                x1 = x - del
                x2 = x + del
                if (f(x1) < f(x2))
                        b = x2
                else
                        a = x1
        }
        return((a+b)/2)
}
```

Знайдемо першу (праву) точку мінімуму.

```
a = 0
b = 5
eps = 0.0001
X1 = BiSection(Fun, a, b, eps)
X1
```

Друга (ліва) точка мінімуму може бути знайдена за допомогою тієї самої процедури.

```
a = -5
b = 0
eps = 0.0001
X2 = BiSection(Fun, a, b, eps)
X2
```

Приклад 2. Знайдіть точку мінімуму функції двох змінних $f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$ (функція Розенброка) методом Ньютона з регулюванням кроку з точністю 10^{-4}. Початкове наближення $x^{(0)} = (3,\ 4)^T$.

Розв'язання. Спочатку визначимо всі необхідні функції для реалізації методу Ньютона:

Fun – функція обчислення значень цільової функції;

FunX – функція обчислення градієнта цільової функції;

FunXX – функція обчислення матриці Гессе цільової функції;

```
Fun = function(x)
{
      return(100*(x[2]-x[1]^2)^2 + (1 - x[1])^2)
}

FunX = function(x)
{
      z = c(1:2)
      z[1] = 200*(x[2]-x[1]^2)*(-2*x[1]) - 2*(1 - x[1])
      z[2] = 200*(x[2]-x[1]^2)
      return(z)
}

FunXX = function(x)
{
      Z = matrix(0, nrow=2, ncol=2)
      Z[1,1] = 1200*x[1]^2 - 400*x[2] + 2
      Z[2,1] = -400*x[1]
      Z[1,2] = -400*x[1]
      Z[2,2] = 200
      return(Z)
}
```

Реалізуємо алгоритм дроблення кроку (7.3) у вигляді процедури засобами пакета **R:**

```
AlDrStep = function(Fun, xk, hk, fk, scphp, ro, kvF)
{
        Lambda = 0.8
        Gamma = 0.4
        alfa = ro
        x1 = xk + alfa*hk
```

```
        f1 = Fun(x1)
        kvF = kvF + 1
        while (f1 > (fk + alfa*Gamma*scphp))
        {
                alfa = alfa*Gamma
                x1 = xk + alfa*hk
                f1 = Fun(x1)
                kvF = kvF + 1
        }
        return(list(xk=x1, fk=f1, kvF = kvF))
}
```

Запрограмуємо функцію обчислення норми вектора для обчислення норми градієнта:

```
NormaV = function(y)
{
      return(sqrt(crossprod(y, y)))
}
```

Реалізуємо метод Ньютона з регулюванням кроку у вигляді процедури засобами пакета **R:**

```
MetodNewton = function(Fun, FunX, FunXX, x0, eps, kmax)
{
    xk = x0
    fk = Fun(xk)
    gk = FunX(xk)
    kvF = 1

    k = 0
    while ((NormaV(gk) > eps) && (k < kmax))
    {
        Hk = FunXX(xk)
        hk = -solve(Hk)%*%gk
        scphp = crossprod(gk, hk)
        res = AlDrStep(Fun, xk, hk, fk, scphp, 1, kvF)
        xk = res$xk
        fk = res$fk
        print(res$kvF-kvF)
        kvF = res$kvF
        gk = FunX(xk)
```

```
        k = k + 1
      }
      return(list(x=xk, k=k, kvF = kvF))
    }
```

Застосуємо цю процедуру для розв'язання задачі:

```
x0 = c(3, 4)
eps = 0.0001
kmax =50
ans = MetodNewton(Fun, FunX, FunXX, x0, eps, kmax)
ans
x1 = ans$x
FunX(x1)
Fun(x1)

res = optim(fn=Fun, par=x0)
res
```

Відповідь, її аналіз і порівняння з результатом, отриманим за допомогою процедури ***optim*** пакета **R**:

```
> ans
$`x`
      [,1]
[1,] 1.000000
[2,] 1.000001

$k
[1] 20

$kvF
[1] 27

> x1 = ans$x
> FunX(x1)
[1]  2.436106e-06 -7.762449e-07
> Fun(x1)
[1] 1.967006e-13
>
>
```

```
> res = optim(fn=Fun, par=x0)
> res
$`par`
[1] 1.001459 1.003154

$value
[1] 7.619629e-06

$counts
function gradient
    139       NA
```

Для візуальної інтерпретації розв'язку задачі побудуємо графік функції в околі точки мінімуму із застосуванням бібліотеки **rgl** (рис. 7.2):

```
# завдання точок сітки для побудови графіка
X1left = -2.0
X1rigth = 2.5
X2left = -5
X2rigth = 8

N = 10
x1 = seq(X1left, X1rigth, len= N)
x2 = seq(X2left, X2rigth, len= N)
Y = matrix(0, nrow=N, ncol=N)
# обчислення значень функції на сітці
xv = c(1:2)
for(i in 1:N){
        xv[1] = x1[i]
        for(j in 1:N){
                xv[2] = x2[j]
                Y[i,j] = Fun(xv)
        }
}
x1 = x1*1000
x2 = x2*1000
```

```
library(rgl)
ylim = range(Y)
ylen = ylim[2] - ylim[1] + 1
colorlut = terrain.colors(ylen) # height color lookup table
col = colorlut[ Y-ylim[1]+1 ]   # assign colors to heights for each point
rgl.surface(x1, x2, Y, coords=1:3, color=col, back="fill")
axes3d()                # виведення осей на графік
title3d('f(x)','','x1','Y','x2') # виведення назви осей
```

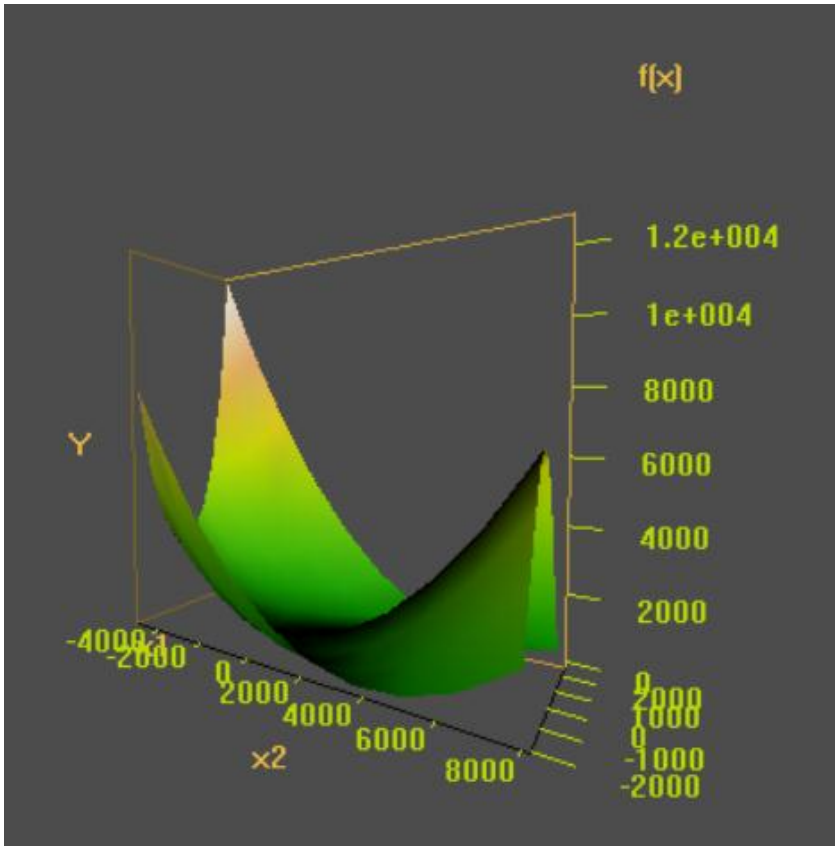

Рис. 7.2. **Графік цільової функції**

7.4. Порядок виконання роботи та варіанти завдань

7.4.1. Зміст звіту

У теоретичній частині роботи необхідно стисло описати:
постановку задачі одномірної оптимізації;
методи розв'язання задачі одномірної оптимізації;

постановку задачі безумовної оптимізації;

методи розв'язання задачі безумовної оптимізації.

У практичній частині роботи необхідно:

запрограмувати у вигляді окремих модулів методи половинного ділення і золотого перетину;

запрограмувати у вигляді окремих модулів методи найшвидшого спуску, сполучених градієнтів і Ньютона;

надати тексти складених програм;

знайти точку мінімуму функції однієї змінної (табл. 7.1) за двома методами: половинного ділення (дихотомії) і золотого перетину. Точність розв'язку – 10^{-4}, відрізок $[a, b]$, що містить точку мінімуму, знайдіть самостійно.

знайти точку мінімуму функції декількох змінних (табл. 7.2) за трьома методами: найшвидшого спуску, сполучених градієнтів і Ньютона (точність розв'язку 10^{-4});

порівняти трудомісткість і швидкість збіжності методів.

7.4.2. Варіанти індивідуальних завдань

Таблиця 7.1

Задачі одномірної оптимізації

Варіант	Функція	Варіант	Функція
1	$f(x) = (x - \exp(-x))^2$	9	$f(x) = (x - \cos(2x))^2$
2	$f(x) = (x - \cos(x))^2$	10	$f(x) = (x - tg(2x) - 1)^2$
3	$f(x) = (x - x^2 - 1)^2$	11	$f(x) = (x - \exp(-3x) + 1)^2$
4	$f(x) = (x - 2\exp(-x))^2$	12	$f(x) = (x - \exp(-x^2))^2$
5	$f(x) = (x - \exp(-3x))^2$	13	$f(x) = (x - \ln(x) + 2)^2$
6	$f(x) = (x - 3\cos(x))^2$	14	$f(x) = (x - \exp(-3x) - 2)^2$
7	$f(x) = (x - \exp(-3x^2))^2$	15	$f(x) = (x^2 - \exp(-x^2))^2$
8	$f(x) = (x - tg(x))^2$		

Таблиця 7.2

Задачі безумовної оптимізації

Варіант	Функція $f(x)$	Початкове наближення	Точка мінімуму $x*$	Значення $f(x*)$
1	$f(x) = 6x_1 + 2x_1^2 - 2x_1x_2 + 2x_2^2$	(-1,-1)	(-2,-1)	-6
2	$f(x) = x_1 + x_2^2 + \left(\frac{x_1 + x_2 - 10}{3}\right)^2$	(-1,-1)	(5,0.5)	7.5
3	$f(x) = (x_1 - 1)^2 + 100(x_1 - x_2)^2$	(3,4)	(1,1)	0
4	$f(x) = 5(x_1 - 3)^2 + (x_2 - 5)^2$	(0,0)	(3,5)	0
5	$f(x) = x_1^2 - x_1x_2 + x_2^2$	(1,2)	(0,0)	0
6	$f(x) = 9x_1^2 + 16x_2^2 - 90x_1 - 128x_2$	(0,3)	(5,4)	-481
7	$f(x) = 2x_1^2 + 2x_2^2 + 2x_1x_2 - 4x_1 - 6x_2$	(1,1)	(1/3,4/3)	-14/3
8	$f(x) = x_1^2 - x_1x_2 + x_2^2 - 2x_1 + x_2$	(3,5)	(1,0)	-1
9	$f(x) = 5x_1^2 + 4x_1x_2 + x_2^2 - 16x_1 - 12x_2$	(1,1)	(-4,14)	-152
10	$f(x) = 2x_1^2 + 2x_2^2 + x_1x_2 - 11x_1 - 8x_2$	(-3,-5)	(2,3)	-23
11	$f(x) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$	(1,1)	(-1,1.5)	-1.25
12	$f(x) = x_1^2 + x_2^2 + x_1x_2$	(1,1)	(0,0)	0
13	$f(x) = x_1^2 + 16x_2^2$	(2,2)	(0,0)	0
14	$f(x) = (1 - x_1)^2 + (x_1 - x_2)^2$	(-5,-8)	(1,1)	0
15	$f(x) = x_1^2 + 4x_2^2 + 1$	(3,5)	(0,0)	1

7.5. Контрольні запитання

1. Сформулюйте постановку задачі одномірної оптимізації.
2. Які чисельні методи застосовують для розв'язання задачі одномірної оптимізації? Чим вони відрізняються?
3. Сформулюйте постановку задачі безумовної оптимізації.
4. Які чисельні методи застосовують для розв'язання задачі безумовної оптимізації? Чим вони відрізняються?

Лабораторна робота 8.
Чисельні методи розв'язання задач нелінійного програмування

8.1. Мета роботи

Вивчення чисельних методів умовної оптимізації для практичного розв'язання задачі пошуку точки оптимуму функції кількох змінних з обмеженнями на змінні; набуття навичок використання цих методів для розв'язання задачі умовної оптимізації із застосуванням комп'ютера.

8.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* загальне формулювання задачі умовної оптимізації та задачі нелінійного програмування; *уміти* розв'язувати задачі з використанням чисельних методів умовної оптимізації [4; 7].

Розглянемо нелінійну неперервну функцію $\mathrm{f(x)}$ кількох змінних $\mathrm{x} \in \mathrm{R^n}$. Задача умовної оптимізації полягає в знаходженні точки $\mathrm{x^*} \in \mathrm{X}$, в якій функція $\mathrm{f(x)}$ приймає мінімальне значення на деякій множині $\mathrm{X} \subset \mathrm{R^n}$. Короткий запис задачі умовної оптимізації:

$$\mathrm{f(x)} \to \min, \quad \mathrm{x} \in \mathrm{X}. \tag{8.1}$$

Якщо множину X можна записати у вигляді:

$$\mathrm{X} = \{\mathrm{x}: \mathrm{g_i(x)} = 0,\ \mathrm{i} = \overline{1, \mathrm{m1}},\ \mathrm{h_i(x)} \le 0,\ \mathrm{i} = \overline{\mathrm{m1}+1, \mathrm{m}}\}, \tag{8.2}$$

де $g_i(x)$, $h_i(x)$ – задані неперервні функції від x, то задача (8.1) – (8.2) є ***задачею нелінійного програмування***.

Розглянемо кілька чисельних методів розв'язання задачі (8.1) – (8.2).

Метод штрафних функцій (квадратичний штраф). У методі вводиться штрафна функція:

$$\Phi(\mathrm{x}, \mathrm{c}) = \mathrm{f(x)} + \frac{\mathrm{c}}{2}\sum_{\mathrm{i}=1}^{\mathrm{m1}} \mathrm{g_i^2(x)} + \frac{\mathrm{c}}{2}\sum_{\mathrm{i}=\mathrm{m1}+1}^{\mathrm{m}} [\max\{0, \mathrm{h_i(x)}\}]^2,$$

де $\mathrm{c} > 0$ – штрафний множник.

Для пошуку розв'язку задачі (8.1) – (8.2) методом штрафних функцій задають початкове наближення розв'язку $x^{(0)} \in R^n$, послідовність $\{c_k\}$, $k = 0,\ 1, \ldots, c_k \to \infty$ з $k \to \infty$ і бажану точність $\varepsilon > 0$ розв'язку задачі. На k-му кроці методу маємо поточне наближення розв'язку $x^{(k)} \in R^n$ і $c_k > 0$. Наступне наближення розв'язку $x^{(k+1)} \in R^n$ визначається як точка безумовного мінімуму функції $\Phi(x, c_k)$ за x.

Обчислення триває доти, поки не виконається умова $\left|\Phi(x^{(k)}, c_k) - f(x^{(k)})\right| \le \varepsilon$.

Перевагою методу штрафних функцій є його висока швидкість збіжності для великого класу задач (8.1) – (8.2).

Недоліком методу штрафних функцій є необхідність постійного збільшення на кожній ітерації значення штрафного множника c_k. Це призводить до того, що функція $\Phi(x, c_k)$ стає погано обумовленою ("яружною"), і тому пошук її точки мінімуму сильно ускладнюється.

Метод модифікованої функції Лагранжа. У методі вводиться модифікована функція Лагранжа:

$$M(x, y, c) = f(x) + \sum_{i=1}^{m1} y_i g_i(x) + \frac{c}{2}\sum_{i=1}^{m1} g_i^2(x) + \frac{1}{2c}\left[\sum_{i=m1+1}^{m}(\max\{0, y_i + c h_i(x)\})^2 - \sum_{i=m1+1}^{m} y_i^2\right],$$

де $y \in R^m$ – множники Лагранжа;

$c > 0$ – штрафний множник.

Для пошуку розв'язку задачі (8.1) – (8.2) методом модифікованої функції Лагранжа задають початкове наближення $y^{(0)} \in R^m$, початкове значення $c_0 > 0$, і бажану точність $\varepsilon > 0$ розв'язку задачі. На k-му кроці методу маємо поточне наближення $y^{(k)} \in R^m$ й $c_k > 0$, а також поточне наближення розв'язку $x^{(k)} \in R^n$. Наступне наближення розв'язку $x^{(k+1)} \in R^n$ визначається як точка безумовного мінімуму функції $M(x, y^{(k)}, c_k)$ за x, чергове наближення $y^{(k+1)} \in R^m$ обчислюють за формулою:

$$y_i^{(k+1)} = y_i^{(k)} + c_k g_i(x^{(k+1)}),\ i = \overline{1, m1};$$

$$y_i^{(k+1)} = \max(0,\ y_i^{(k)} + c_k g_i(x^{(k+1)})),\ i = \overline{m1+1, m}.$$

Значення крокового множника c_{k+1} визначається залежно від ходу процесу: якщо пошук точки мінімуму функції $\mathrm{M}(\mathrm{x}, \mathrm{y}^{(\mathrm{k})}, \mathrm{c}_{\mathrm{k}})$ за x виявився утрудненим (тобто функція $\mathrm{M}(\mathrm{x}, \mathrm{y}^{(\mathrm{k})}, \mathrm{c}_{\mathrm{k}})$ стала "яружною" по x), то значення c_{k+1} береться меншим відносно c_k (наприклад, $\mathrm{c}_{\mathrm{k+1}} = \frac{2}{3}\mathrm{c}_{\mathrm{k}}$); якщо ж проблем не було, то значення c_{k+1} збільшується відносно c_k (наприклад, $c_{k+1} = 2c_k$).

Обчислення триває доти, поки не виконається умова $\left|\mathrm{M}(\mathrm{x}^{(\mathrm{k})}, \mathrm{y}^{(\mathrm{k})}, \mathrm{c}_{\mathrm{k}}) - \mathrm{f}(\mathrm{x}^{(\mathrm{k})})\right| \leq \varepsilon$.

Перевагою методу модифікованої функції Лагранжа є його висока швидкість збіжності для великого класу задач (8.1) – (8.2).

8.3. Контрольні приклади

Приклад 1. Розв'язати задачу нелінійного програмування

$$\mathrm{f}(\mathrm{x}) = \mathrm{x}_1^2 + \mathrm{x}_2^2 - 16\mathrm{x}_1 - 10\mathrm{x}_2 \to \min_{\mathrm{x}}$$

з обмеженнями:

$$\begin{aligned} &\mathrm{x}_1^2 - 6\mathrm{x}_1 + 4\mathrm{x}_2 - 11 \leq 0 \\ &3\mathrm{x}_2 - \mathrm{x}_1\mathrm{x}_2 + \mathrm{e}^{\mathrm{x}_1 - 3} - 1 \leq 0 \\ &0 \leq \mathrm{x}_1 \leq 6, \\ &0 \leq \mathrm{x}_2 \leq 5 \end{aligned}$$

методом штрафних функцій з квадратичним штрафом із точністю 10^{-4}. Початкове наближення $\mathrm{x}^{(0)} = (10{,}10)^{\mathrm{T}}$.

Розв'язання. Задача може бути записана у вигляді (8.1) – (8.2). Тут $m1 = 0$, $m = 6$, оскільки подвійні нерівності мають бути записані парою одинарних. Визначимо цільову функцію та векторну функцію обмежень, що відповідають цій задачі нелінійного програмування:

```
FunC = function(x)
{
        return(x[1]^2 + x[2]^2 - 16*x[1] - 10*x[2])
}

Gcon = function(x)
{
```

```
        z = c(1:6)
        z[1] = x[1]^2 - 6*x[1] + 4*x[2] - 11
        z[2] = 3*x[2] - x[1]*x[2] + exp(x[1] - 3) - 1
        z[3] = x[1] - 6
        z[4] = -x[1]
        z[5] = x[2] - 5
        z[6] = -x[2]
        return(z)
}

m1 = 0
m = 6
```

Далі визначимо необхідну штрафну функцію для реалізації метода штрафних функцій.

```
FunSfine = function(x, parFun)
{
        c = parFun$c
        g = Gcon(x)
        S = 0
        if (m1 > 0)
        {
                for (i in 1:m1)
                {
                        t = g[i]
                        S = S + t*t
                }
        }
        if ((m-m1) > 0)
        {
                for (i in m1+1:m)
                {
                        t = max(0, g[i])
                        S = S + t*t
                }
        }
        return(FunC(x) + c*0.5*S)
}
```

Тоді метод штрафних функцій можна реалізувати у вигляді:

```
MetodFineFun = function(x0, eps, kmax)
{
        xk = x0
        ck = 10
        fck = FunC(xk)
        parC = list(c=ck)
        Fik = FunSfine(xk, parC)
        k = 0
        while ((abs(Fik - fck) > eps) && (k < kmax))
        {
                parC = list(c=ck)    #передаємо штрафний множник C через список
                res = optim(fn=FunSfine, par=xk, parFun=parC)
                xk = res$par
                ck = ck*2
                fck = FunC(xk)
                parC = list(c=ck)
                Fik = FunSfine(xk, parC)
                k = k + 1
        }
        return(list(x=xk, k=k))
}
```

Розв'яжемо задачу умовної оптимізації за допомогою такої процедури:

```
x0 = c(10, 10)
FunC(x0)
Gcon(x0)

eps = 0.0001
kmax = 50
ans = MetodFineFun(x0, eps, kmax)
ans
x1 = ans$x
FunC(x1)
Gcon(x1)
```

Для аналізу отриманого розв'язку обчислимо значення цільової функції та функцій обмежень:

```
> ans
$`x`
[1] 5.239367 3.746320

$k
[1] 13

> x1 = ans$x
> FunC(x1)
[1] -79.80719
> Gcon(x1)
[1]  4.485072e-05 -1.999219e-03 -7.606331e-01 -5.239367e+00 -1.253680e+00
[6] -3.746320e+00
```

Як видно, отриманий розв'язок лежить на межі допустимої множини. Активними є перше та друге обмеження (значення функцій $h_1(x)$ і $h_2(x)$ у точці мінімуму близькі до нуля). Останні обмеження є пасивними, тобто суттєвого впливу не мають.

8.4. Порядок виконання роботи і варіанти завдань

8.4.1. Зміст звіту

У теоретичній частині роботи необхідно стисло описати:

постановку задачі нелінійного програмування;

методи розв'язання задачі нелінійного програмування.

У практичній частині роботи необхідно:

запрограмувати у вигляді окремих модулів метод штрафних функцій та метод модифікованої функції Лагранжа;

надати тексти складених програм;

розв'язати задачу нелінійного програмування (табл. 8.1) методом штрафних функцій та методом модифікованої функції Лагранжа (точність розв'язку 10^{-4});

порівняти трудомісткість і швидкість збіжності методів.

8.4.2. Варіанти індивідуальних завдань

Таблиця 8.1

Задачі умовної оптимізації

Варіант	Цільова функція, $f(x)$	Обмеження	Початковий вектор, $x^{(0)}$	$f(x^{(0)})$	Точка мінімуму, x^*	Значення $f(x^*)$
1	2	3	4	5	6	7
1	$-x_1^2 - x_2^2$	$(x_1-1)^2 + x_2^2 \le 1,$ $x_1, x_2 \ge 0$	$[1;1]$	-4	-2	-4
2	$x_1^2 + x_2^2 + x_3^2$	$2x_1 + x_2 - 5 \le 0,$ $x_1 + x_3 - 2 \le 0,$ $-x_1 + 1 \le 0,$ $-x_2 + 2 \le 0,$ $-x_3 \le 0$	$[1.5; 2; 0.5]$	6.5	$[1; 2; 0]$	5
3	$2x_1^2 + 2x_2^2 + 2x_1x_2 - 4x_1 - -6x_2$	$-x_1 + 2x_2 \le 2,$ $x_1, x_2 \ge 0$	$[0.5; 0.5]$	-3.5	$\left[\frac{1}{3}; \frac{5}{6}\right]$	4.16
4	$2x_1^2 + 2x_2^2 + 3x_3^2 + 2x_1x_2 + + x_2x_3 + x_1 - 3x_2 - 5x_3$	$x_1 + x_2 + x_3 \ge 1,$ $3x_1 + 2x_2 + x_3 \le 6,$ $x_1, x_2, x_3 \ge 0$	$[1; 1; 1]$	-2.35	$[0; 0.4; 0.7]$	-2.35
5	$x_1^2 + x_2^2 + x_3^2$		$[0.1; 0.9; 2]$	4.82	$[1; 1; 1]$	3
6	$2x_1^2 + 2x_1 + 4x_2 - 3x_3$	$8x_1 - 3x_2 + 3x_3 \le 40,$ $2x_1 + x_2 - x_3 = -3,$ $x_2 \ge 0$	$[1; 1; 1]$	2	$[-2; 0; 7]$	-17
7	$\exp(x_1 - x_2) - x_1 - x_2$	$x_1 + x_2 \le 1,$ $x_1, x_2 \ge 0$	$[0; 0]$	1	$[0; 1]$	-0.6
8	$3x_2^2 - 11x_1 - 3x_2 - x_3$	$x_1 - 7x_2 + 3x_3 + +7 \le 0,$ $5x_1 + 2x_2 - x_3 \le 2,$ $x_3 \ge 0$	$[0; 2; 0]$	6	$[0; 1; 0]$	0
9	$x_1^2 + x_2^2 - 4x_1 - 2x_2$	$2x_1 + x_2 \le 4,$ $x_1 + 2x_2 \le 6,$ $x_1, x_2 \ge 0$	$[1; 1]$	-4	$[1.6; 0.8]$	4.8
10	$0.5x_2^2 + 0.5x_1^2 - x_1 - 2x_2 + 5$	$2x_1 + 3x_2 \le 6,$ $x_1 + 4x_2 \le 5,$ $x_1, x_2 \ge 0$	$[0; 0]$	5	$\left[\frac{13}{17}; \frac{18}{17}\right]$	$\frac{101}{34}$
11	$-2x_1 + 0.2x_1^2 - 3x_2 + 0.2x_2^2$	$2x_1 + 3x_2 \le 13,$ $2x_1 + x_2 \le 10,$ $x_1, x_2 \ge 0$	$[1; 1]$	-4.6	$[2; 3]$	-10.4

Закінчення табл. 8.1

1	2	3	4	5	6	7
12	$0.5x_1^2 + 0.5x_2^2 - x_1 - 2x_2 + 8$	$2x_1 + 3x_2 \le 6,$ $x_1 + 4x_2 \le 5,$ $x_1, x_2 \ge 0$	$[1.5; 0.5]$	3.75	$[0.7647; 1.0588]$	2.9705
13	$\ln x_1 - x_2$	$x_1 - 1 \ge 0,$ $x_1^2 + x_2^2 - 4 = 0$	$[2; 0]$	0.69	$[1; \sqrt{3}]$	$-\sqrt{3}$
14	$x_1^2 + x_2^2 - 2.4x_1 - 5.6x_2$	$-2x_1 - 3x_2 - 3 \le 0,$ $x_1 + x_2 - 3 \le 0,$ $2x_1 - x_2 - 4 \le 0,$ $x_1, x_2 \ge 0$	$[0; 1]$	-4.6	$[1.2; 1.8]$	-7.88
15	$(x_1 - 2)^2 + (x_2 - 1)^2$	$x_1 - 2x_2 + 1 = 0,$ $-0.25x_1^2 - x_2^2 + 1 \ge 0$	$[2; 2]$	1	$[0.823; 0.911]$	1.393

8.5. Контрольні запитання

1. Сформулюйте постановку задачі умовної оптимізації.
2. Сформулюйте постановку задачі нелінійного програмування.
3. Які чисельні методи застосовують для розв'язання задачі нелінійного програмування? Чим вони відрізняються?
4. У чому полягає ідея методу штрафних функцій для розв'язання задачі нелінійного програмування?
5. Які види штрафних функцій застосовують у методі штрафних функцій для розв'язання задачі нелінійного програмування?
6. У чому полягає ідея методу модифікованої функції Лагранжа для розв'язання задачі нелінійного програмування?

Лабораторна робота 9.
Розв'язання задач лінійного програмування

9.1. Мета роботи

Вивчення чисельних методів лінійного програмування для розв'язання практичних задач; набуття навичок використання цих методів для розв'язання задач лінійного програмування із застосуванням математичних пакетів.

9.2. Методичні вказівки по організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* загальне формулювання задачі лінійного програмування; *уміти* розв'язувати цю задачу з використанням математичних пакетів [4].

Задача лінійного програмування в **загальній формі** записується у вигляді:

$$f(x) = \sum_{j=1}^{n} c_j x_j \to \min_{x \in X} \tag{9.1}$$

з обмеженнями

$$X = \{x: \sum_{j=1}^{n} a_{ij} x_j = b_i,\ i = \overline{1, m_1},\ \sum_{j=1}^{n} a_{ij} x_j \le b_i,\ i = \overline{m_1 + 1, m},\ x_j \ge 0,\ j = \overline{1, n}\}, \tag{9.2}$$

де $c_j \in R^1, j = \overline{1, n}$, $a_{ij} \in R^1, i = \overline{1, m}, j = \overline{1, n}$; $b_i \in R^1$, $i = \overline{1, m}$ – задані.

Основним чисельним методом розв'язання задач лінійного програмування є так званий **симплекс-метод**. Його теоретичні й обчислювальні аспекти добре розроблені. Є безліч стандартних програм, що реалізовують симплекс-метод.

У математичному пакеті **R** є функція *solveLP*, що розв'язує задачу лінійного програмування. Вона належить до бібліотеки ***linprog***. Наведемо її опис.

Опис:

Ця функція оптимізує лінійну функцію $c^T x$ з урахуванням обмежень $Ax \le b$ і $x \ge 0$. Можлива максимізація або мінімізація (за замовчуванням – мінімізація).

Використання:

solveLP(cvec, bvec, Amat, maximum = FALSE)

Аргументи:

cvec	вектор довжини n, який задає коефіцієнти цільової функції $c^T x$;
bvec	вектор довжини m, який задає праву частину обмеження $Ax \le b$;
Amat	матриця коефіцієнтів розмірності $m \times n$ обмеження $Ax \le b$;
maximum	логічний параметр, який вказує мінімізацію, якщо FALSE (за замовчуванням) і максимізацію – в іншому випадку.

9.3. Контрольні приклади

Приклад 1. Розв'язати задачу лінійного програмування

$$f(x) = 200x_1 + 6000x_2 + 3000x_3 - 200x_4 \to \max_x$$

з обмеженнями:

$$800x_1 + 6000x_2 + 1000x_3 + 400x_4 \leq 13800;$$
$$50x_1 + 3x_2 + 150x_3 + 100x_4 \geq 600;$$
$$10x_1 + 10x_2 + 75x_3 + 100x_4 \geq 300;$$
$$150x_1 + 35x_2 + 75x_3 + 5x_4 \geq 550;$$
$$x \geq 0.$$

Розв'язання. Задаємо початкові дані задачі та використовуємо функцію *solveLP*:

```
cf = c(200, 6000, 3000, -200)
A1 = c(800, 6000, 1000, 400)
b1 = 13800
vitx = c(50, 3, 150, 100)
vity = c(10, 10, 75, 100)
vitz = c(150, 35, 75, 5)
A2 = rbind(vitx, vity, vitz)
b2 = c(600, 300, 550)

library(linprog)
A = rbind(A1, -A2)
b = c(b1, -b2)
A
b
res = solveLP(cf, b, A, maximum=TRUE)
res
```

Функція *solveLP* надає такий результат:

```
Results of Linear Programming / Linear Optimization
```

```
Objective function (Maximum): 41400

Iterations in phase 1: 3
Iterations in phase 2: 4
Solution
  opt
1  0.0
2  0.0
3 13.8
4  0.0

Basic Variables
      opt
3    13.8
S 2 1470.0
S 3  735.0
S 4  485.0

Constraints
  actual dir  bvec free dual dual.reg
1  13800  <= 13800    0    3  6466.67
2  -2070  <=  -600 1470    0  1470.00
3  -1035  <=  -300  735    0   735.00
4  -1035  <=  -550  485    0   485.00
```

Отримуємо розв'язок: $x_1 = 0;\quad x_2 = 0;\quad x_3 = 13.8;\quad x_4 = 0$. Як видно, отриманий розв'язок лежить на межі допустимої множини; активним є перше обмеження. Останні обмеження є пасивними, тобто суттєвого впливу не мають.

Приклад 2. Треба спланувати на добу оптимальний раціон "бідного" студента з точки зору мінімуму матеріальних затрат, але з дотриманням добової норми споживання білків, жирів, вуглеводів і калорій. Тобто треба визначити кількість пакетиків у 100 грам різних круп (пшеничної, гречаної, манної та рисової), що задовільнять виконання норм споживання. Необхідні дані наведені в табл. 9.1.

Таблиця 9.1

Вихідні дані

Раціон студента	Пакетики, 100 г				Добова норма, г	
Добова норма споживача та ціна	Крупа пшенична	Крупа гречана	Крупа манна	Крупа рисова		Max
Білки, г	11,5	12,6	10,30	70,00	90	150
Жири, г	3,3	3,3	1,0	1,00	103	158
Вуглеводи, г	67,00	63,00	68,00	72,00	400	500
Калорійність, Ккал	348,00	335,00	328,00	330,00	2 600	3 000
Ціна за 100 г, грн	0,63	1,76	0,80	1,72	–	–

Розв'язання. Позначимо через x_1, x_2, x_3, x_4 кількість пакетиків пшеничної, гречаної, манної та рисової груп послідовно. Тоді ціна всіх пакетиків буде дорівнювати $0.63x_1 + 1.76x_2 + 0.8x_3 + 1.72x_4$, добова норма споживання білка $11.5x_1 + 12.6x_2 + 10.3x_3 + 70x_4$, добова норма споживання жирів $3.3x_1 + 3.3x_2 + 1.0x_3 + 1.0x_4$, добова норма споживання вуглеводів $67x_1 + 63x_2 + 68x_3 + 72x_4$, добова норма споживання калорій $348x_1 + 335x_2 + 328x_3 + 330x_4$. Таким чином, задача мінімізації затрат на покупку їжі, за умов виконання мінімальних норм споживання білків, жирів, вуглеводів та калорій, зводиться до розв'язання задачі лінійного програмування:

$$f(x) = 0.63x_1 + 1.76x_2 + 0.8x_3 + 1.72x_4 \to \min_x$$

з обмеженнями:

$$11.5x_1 + 12.6x_2 + 10.3x_3 + 70x_4 \geq 90;$$
$$3.3x_1 + 3.3x_2 + 1.0x_3 + 1.0x_4 \geq 103;$$
$$67x_1 + 63x_2 + 68x_3 + 72x_4 \geq 400;$$
$$348x_1 + 335x_2 + 328x_3 + 330x_4 \geq 2600;$$
$$x \geq 0.$$

Розв'яжемо цю задачу з використанням функції ***solveLP:***

```
#Суточна норма
DNmin = c(90., 103., 400., 2600)
```

```
#Ціна
cf = c(0.63, 1.76, 0.8, 1.72)

#Вміст білка і т.д. в крупах
vit1 = c(11.5, 12.6, 10.3, 70)
vit2 = c(3.3, 3.3, 1.0, 1.0)
vit3 = c(67.0, 63.0, 68.0, 72.0)
vit4 = c(348., 335., 328., 330.)
A = rbind(vit1, vit2, vit3, vit4)

#Формування обмежень типу Ax <= b
A = -A
b = -DNmin

cf
A
b

library(linprog)
res = solveLP(cf, b, A, maximum=FALSE)
res
```

Функція *solveLP* дає такий результат:

```
Results of Linear Programming / Linear Optimization

Objective function (Minimum): 19.6636

Iterations in phase 1: 4
Iterations in phase 2: 1
Solution
      opt
1 31.2121
2  0.0000
3  0.0000
4  0.0000

Basic Variables
         opt
1     31.2121
```

```
S 1  268.9394
S 3 1691.2121
S 4 8261.8182

Constraints
     actual dir  bvec     free     dual dual.reg
1   -358.939  <=   -90  268.939 0.000000  268.939
2   -103.000  <=  -103    0.000 0.190909      Inf
3  -2091.212  <=  -400 1691.212 0.000000 1691.212
4 -10861.818  <= -2600 8261.818 0.000000 8261.818
```

Отримуємо розв'язок: $x_1 = 31.2121; x_2 = 0; \quad x_3 = 0; x_4 = 0$. Тобто, оптимальним буде план: їсти тільки крупу пшеничну в кількості 31 пакетик на день. Для чого буде достатньо 19.66 гривень на добу.

9.4. Порядок виконання роботи та варіанти завдань

9.4.1. Зміст звіту

У теоретичній частині роботи необхідно стисло описати:

постановку задачі лінійного програмування.

У практичній частині роботи необхідно:

розв'язати задачу лінійного програмування за допомогою функції *solveLP з* бібліотеки ***linprog***.

9.4.2. Варіанти індивідуальних завдань

Варіанти 1 – 5. (Транспортна задача). З трьох холодильників $A_i, i = \overline{1,3}$, що вміщають морожену рибу в кількості a_i тон, необхідно останню доставити в п'ять магазинів $B_j, j = \overline{1,5}$ рибу в кількості b_j тон. Вартість перевезення 1 тони риби з холодильника A_i в магазин B_j задана у вигляді матриці $C=(c_{ij})$ розмірності 3 х 5.

Напишіть математичну модель задачі та сплануйте перевезення так, щоб їх загальна вартість була мінімальною. Під плануванням перевезення слід розуміти визначення x_{ij} – кількість тон риби, що перевозиться з холодильника (постачальника) A_i в магазин (споживач) B_j.

Вихідні дані за варіантами подані в табл. 9.2.

Таблиця 9.2

Вихідні дані

Варіант	a_i	b_j	C
1	(320; 280; 250)	(150; 140; 110; 230; 220)	20 23 20 15 24 29 15 16 19 29 6 11 10 9 8
2	(250; 380; 220)	(120; 140; 140; 210; 240)	21 22 17 16 25 26 15 17 19 25 5 10 10 11 7
3	(420; 210; 220)	(130; 130; 160; 190; 240)	22 23 21 16 24 24 15 17 19 28 7 11 11 9 8
4	(500; 300; 100)	(150; 350; 200; 100; 100)	3 3 5 3 1 4 3 2 4 5 3 7 5 4 2
5	(320; 260; 270)	(110; 140; 170; 180; 250)	26 23 21 16 24 22 15 17 19 28 7 11 11 79 8

Варіант 6. (Задача оптимального розподілу ресурсів). Транспортне господарство має можливість придбати не більше дев'ятнадцяти тритонних автомашин і не більше сімнадцяти п'ятитонних. Відпускна ціна тритонної вантажівки – 400 000 грн, п'ятитонної – 500 000 грн. Господарство може виділити для придбання автомашин 14 100 тис. грн. Скільки потрібно придбати автомашин, щоб їх сумарна вантажопідйомність була максимальною?

Варіант 7. Серед ненегативних чисел x і y, що задовільнюють умовам: $x + y \leq 1$; $x - 4y \geq -2$, знайдіть такі, для яких різниця $(x - y)$ приймає найменше значення. Складіть математичну модель задачі.

Варіант 8. Серед ненегативних чисел x і y, що задовільнюють умовам: $-2x - 6y \leq 6$; $-x + 2y \leq 6$; $x \leq 3$, знайдіть такі, для яких сума $(x + y)$ приймає найбільше значення. Складіть математичну модель задачі.

Варіант 9 (задача оптимального розподілу ресурсів). Для виготовлення трьох видів виробів A, B і C використовується токарне, фрезерне, зварювальне та шліфувальне обладнання. Витрати часу на обробку одного виробу для кожного з типів обладнання, загальний фонд робочого часу кожного з типів використовуваного обладнання, а також прибуток від реалізації одного виробу кожного виду надані в табл. 9.3.

Таблиця 9.3

Вихідні дані

Типи обладнання	Витрати часу (верстато-години) на обробку одного виробу кожного виду			Загальний фонд робочого часу обладнання (години)
	A	*B*	*C*	
Фрезерне	2	4	5	120
Токарне	1	8	6	280
Сварочне	7	4	5	240
Шлифовальне	4	6	7	360'
Прибуток (грн)	100	140	120	

Потрібно визначити, скільки виробів і якого виду слід виготовити підприємству, щоб прибуток від їх реалізації був максимальним. Складіть математичну модель задачі.

Варіант 10 (задача оптимального планування). Продукцією міського молочного заводу є молоко, кефір і сметана, розфасовані в пляшки. На виробництво 1 т молока, кефіру та сметани потрібно, відповідно, 1 010, 1 010 і 9 450 кг молока. Водночас витрати робочого часу для розливу 1 т молока та кефіру складають 0,18 і 0,19 машино-годин. На розфасовці 1 т сметани задіяні спеціальні автомати протягом 3,25 годин. Усього для виробництва продукції з незбираного молока завод може використовувати 136 000 кг сировини. Основне обладнання може працювати протягом 21,4 машино-годин, а автомати з розфасовки сметани – 16,25 годин. Прибуток від реалізації 1 т молока, кефіру та сметани, відповідно, дорівнює 30, 22 і 136 грн. Завод повинен щодня виробляти не менше 100 т молока, розфасованого в пляшки. На виробництво іншої продукції немає жодних обмежень.

Потрібно визначити, яку продукцію і в якій кількості слід щодня виготовляти заводу, щоб прибуток від її реалізації був максимальним. Складіть математичну модель задачі.

Варіанти 11 – 15 (задача оптимального розкрою). На швейній фабриці тканина може бути розкроєна кількома способами для виготовлення потрібних деталей швейних виробів. Нехай у j-му варіанті розкрою $(j = \overline{1, n})$

зі 100 м^2 тканини виробляється b_{ij} деталей i-го виду $(i = \overline{1,m})$, а величина відходів дорівнює c_j м^2. Знаючи, що деталей і-го виду треба виробляти B_i штук, потрібно розкроїти тканину так, щоб було отримано необхідну кількість деталей кожного виду з мінімальними загальними відходами. Складіть математичну модель задачі. Під оптимальним розкроєм слід розуміти визначення x_j сотень м^2 тканини для j-го варіанта розкрою, що забезпечують мінімальні загальні відходи.

Початкові дані за варіантами надані в табл. 9.4.

Таблиця 9.4

Вихідні дані

Варіант	c_j	B_i	b_{ij}
11	(1.4; 0.1; 2.1;0.1)	(240; 200; 120; 140)	$\begin{matrix}1&4&0&1\\1&0&4&0\\1&0&0&3\\1&1&0&3\end{matrix}$
12	(0.2; 0; 0.9;0.7;0.2;0.5;0)	(600; 720; 900)	$\begin{matrix}3&2&2&1&1&0&0\\0&1&0&1&0&2&1\\0&0&1&1&3&1&3\end{matrix}$
13	(0.4; 0.3; 0.1;0;0.1;0.2)	(155; 190; 300)	$\begin{matrix}0&2&1&3&1&&0\\1&1&0&0&2&&0\\1&0&1&0&0&&2\end{matrix}$
14	(1.3; 0.2; 2.1;0.1)	(230; 210; 125; 134)	$\begin{matrix}1&4&0&1\\1&0&4&0\\1&0&0&3\\1&1&0&3\end{matrix}$
15	(0.4; 0.3; 0.1;0;0.1;0.2)	(150; 200; 300)	$\begin{matrix}0&2&2&4&1&&0\\1&1&0&0&2&&0\\1&0&1&0&0&&2\end{matrix}$

9.5. Контрольні запитання

1. Сформулюйте загальну постановку задачі лінійного програмування.
2. Які чисельні методи застосовуються для розв'язання задачі лінійного програмування?
3. У чому полягає ідея симплекс-методу для розв'язання задачі лінійного програмування?

Змістовий модуль 3
Моделювання систем

Лабораторна робота 10.
Вивчення можливостей математичного пакета R

10.1. Мета роботи

Ознайомлення із засобами математичного пакета R, що можуть бути застосовані в комп'ютерному моделюванні.

10.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* основні закони розподілу випадкових величин та основні поняття лінійної алгебри та теорії диференціальних рівнянь; *уміти* застосовувати процедури статистичного пакета R для розв'язання задач лінійної алгебри, задачі Коші для систем звичайних диференціальних рівнянь, генерації значень випадкових величин за різними розподілами [17 – 19].

10.2.1. Функції для обробки випадкових величин

У статистичному пакеті R є багато функцій, призначених для обробки випадкових величин. Наведемо кілька з них, розбивши на чотири групи.

Функції для генерування випадкових чисел *(префікс "r" на початку назви функції):*

runif(n, min = 0, max = 1) – повертає вектор *n* випадкових чисел, що мають рівномірний розподіл на інтервалі [min, max];

rnorm(n, mean = 0, sd = 1) – повертає вектор *n* випадкових чисел, що мають нормальний розподіл (*mean* – середнє значення, *sd* > 0 – середнє квадратичне відхилення);

rexp(n, rate = 1) – повертає вектор *n* випадкових чисел, що мають експонентний розподіл (*rate* > 0 – параметр розподілу);

rpois(n, lambda) – повертає вектор *n* випадкових чисел, що мають розподіл Пуассона (*lambda* > 0 – середнє значення).

Приклад 1. Застосування функції **runif**. Генеруються точки в колі радіусу 1 (рис. 10.1).

```
Circle = function(n)
{
  r = runif(n, 0, 1)
  teta = runif(n, 0, 2*pi)
  x = c(1:n)
  y = c(1:n)
  for(i in 1:n)
  {
          x[i] = r[i]*cos(teta[i])
          y[i] = r[i]*sin(teta[i])
  }
  plot(x, y, type="p", col="red")
}

n = 500
Circle(n)
```

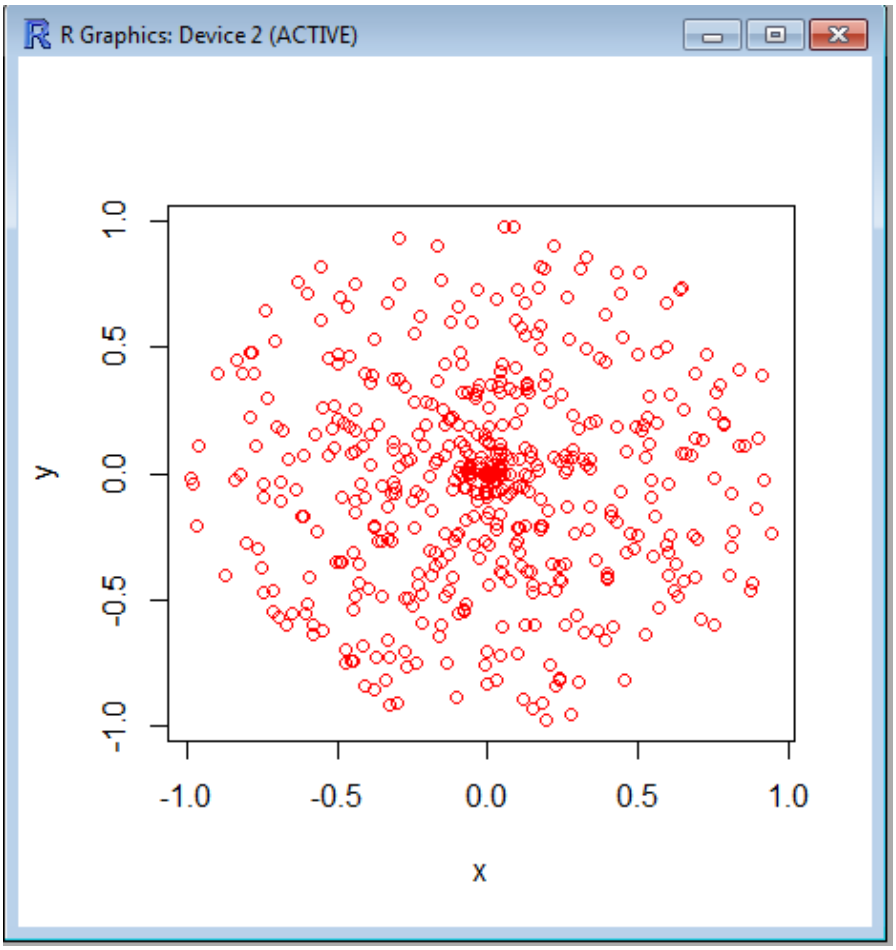

Рис. 10.1. **Точки в колі радіусу 1**

Приклад 2. Застосування функції **rexp**. Будується графік значень випадкової величини, що підкоряється експонентному розподілу з параметром 5 (рис. 10.2).

```
n = 100
y = rexp(n, 5)
plot(y, type="l", col="red")
```

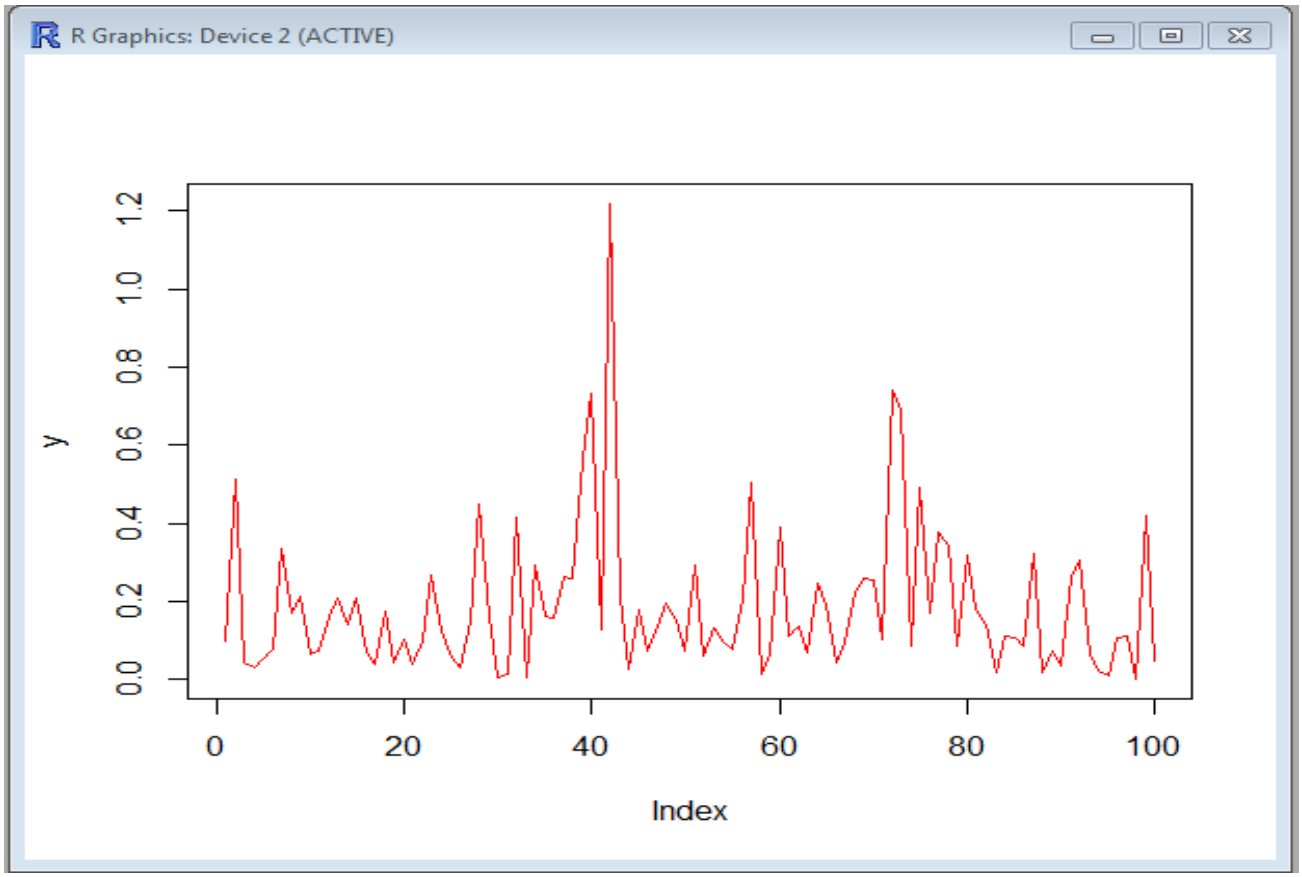

Рис. 10.2. **Графік значень випадкової величини, що підкоряється експонентному розподілу**

Щільності розподілу ймовірності *(префікс "d" на початку назви функції):*

dunif(x, min = 0, max = 1) – функція від x щільності розподілу ймовірності для рівномірного закону на інтервалі [min, max];

dnorm(x, mean = 0, sd = 1) – функція від x щільності розподілу ймовірності для нормального закону з параметрами *mean, sd*;

dexp(x, rate = 1) – функція від x щільності розподілу ймовірності для експонентного закону з параметром *rate*;

dpois(x, lambda) – функція від x щільності розподілу ймовірності для закону Пуассона з параметром *lambda*.

Функції розподілу *(префікс "p" на початку назви функції):*

punif(x, min = 0, max = 1) – функція від x розподілу для рівномірного закону на інтервалі [min, max];

pnorm(x, mean = 0, sd = 1) – функція від x розподілу для нормального закону з параметрами *mean, sd*;

pexp(x, rate = 1) – функція від x розподілу для експонентного закону з параметром *rate*;

ppois(x, lambda) – функція від x розподілу для закону Пуассона з параметром *lambda*.

Приклад 3. Приклад застосування функцій **pnorm, dnorm**. Будуються графіки функції розподілу та функції щільності розподілу ймовірності для нормального закону розподілу із середнім значенням 3 і середнім квадратичним відхиленням 2 (рис. 10.3).

```
mean = 3
sd = 2
n = 21
x = c(1:n)
yp = c(1:n)
yd = c(1:n)
for(i in 1:n)
{
        x[i] = -10 + i
        yp[i] = pnorm(x[i], mean, sd)
        yd[i] = dnorm(x[i], mean, sd)
}
plot(x, yp, type="l", col="red")
lines(x, yd, type="l", col="blue")
```

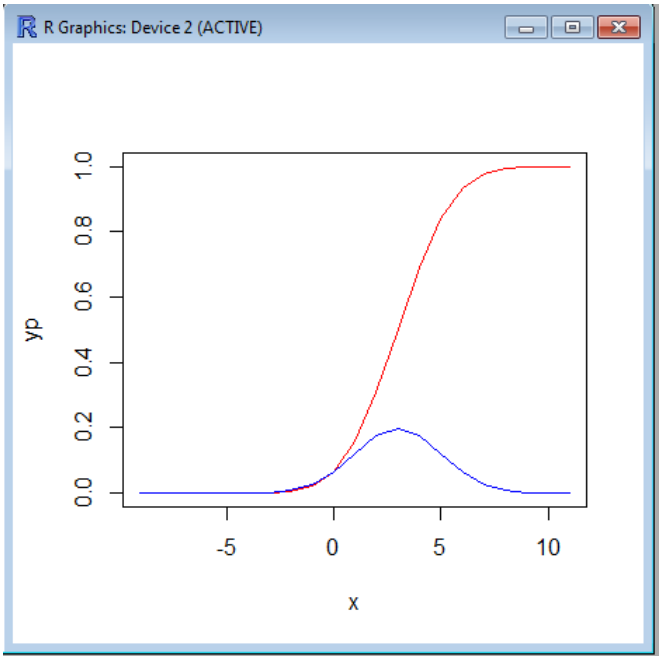

Рис. 10.3. **Графік функції розподілу та функції щільності розподілу ймовірності для нормального закону розподілу**

Функції, обернені до функції розподілу *(префікс "q" на початку назви функції)**:***

qt(p, df) – функція від *p*, обернена до функції розподілу для *t*-розподілу Стьюдента, де *df* – число степенів свободи;

qchisq(p, df) – функція від *p*, обернена до функції розподілу для *xi*-квадрат розподілу, де *df* – число степенів свободи.

10.2.2. Функції для розв'язання задач лінійної алгебри

Функції, призначені для розв'язання задач лінійної алгебри, можна розподілити на три групи.

Функції визначення матриць та операції з блоками матриць:

matrix(вираз, nrow=n, ncol=m) – створює матрицю розмірності n x m і заповнює її значеннями виразу;

diag(вираз, nrow=n) – створює діагональну матрицю розмірності *n* x *n* і заповнює її значеннями виразу;

t(A) – траспонування матриці;

solve(A) – знаходження оберненої матриці;

A[i,] – *i*-й рядок матриці A;

A[,j] – *j*-й стовпець матриці A;

cbind(A, B) – створює матрицю, приписуючи до **A** справа **B** (для цього кількість рядків у матриць **A** та **B** має співпадати);

rbind(A, B) – створює матрицю, приписуючи до **A** знизу **B** (для цього кількість стовпців у матриць **A** та **B** має співпадати).

Функції пошуку різноманітних числових характеристик матриць:

det(A) – визначник (детермінант) матриці A;

nrow(A) – визначення числа рядків у матриці A;

ncol(A) – визначення числа стовпців у матриці A;

max(A) – визначення найбільшого елемента в матриці A;

min(A) – визначення найменшого елемента в матриці A;

norm(A) – обчислення норми квадратної матриці A.

Функції, що реалізують чисельні методи розв'язання задач лінійної алгебри:

eigen(A) – обчислення власних значень та власних векторів квадратної матриці A;

solve(A, b) – розв'язання системи лінійних алгебраїчних рівнянь виду **Ax=b**.

Приклад 4. Приклад застосування функцій **eigen, solve.**

```
> A = diag(1:5, nrow=5)
> eigen(A)
eigen() decomposition
$`values`
[1] 5 4 3 2 1

$vectors
     [,1] [,2] [,3] [,4] [,5]
[1,]    0    0    0    0    1
[2,]    0    0    0    1    0
[3,]    0    0    1    0    0
[4,]    0    1    0    0    0
[5,]    1    0    0    0    0

>
```

```
> b = c(1:5)
> x = solve(A,b)
> A%*%x - b
     [,1]
[1,]    0
[2,]    0
[3,]    0
[4,]    0
[5,]    0
```

10.2.3. Функції для розв'язання систем звичайних диференціальних рівнянь, записаних у нормальній формі

Оскільки математичний пакет **R** сьогодні не має функцій, призначених для розв'язання **систем** звичайних диференційних рівнянь, то для цих цілей застосовуватимемо пакет ***deSolve***, який треба підключити до пакету **R**.

Функції пакету ***deSolve***, що призначені для розв'язання задач Коші, розв'язують їх для **нормальних систем** звичайних диференціальних рівнянь. Задачі для диференціальних рівнянь вищих порядків зводяться до відповідних задач для нормальних систем.

Розглянемо задачу Коші:

$$\begin{aligned} y_1' &= f_1(x, y_1, y_2, \dots, y_k), \\ y_2' &= f_2(x, y_1, y_2, \dots, y_k), \\ &\dots\dots\dots\dots\dots\dots\dots\dots \\ y_k' &= f_k(x, y_1, y_2, \dots, y_k) \end{aligned} \qquad (10.1)$$

за початкових умов:

$$\begin{cases} y_1(x_0) = y_{0,1} \\ y_2(x_0) = y_{0,2} \\ \dots \\ y_k(x_0) = y_{0,k} \end{cases}.$$

Чисельне розв'язання задачі (10.1) полягає в побудові таблиці наближених значень $y_{i,1}, y_{i,2}, \dots, y_{i,k}$ $(i = 1,2, \dots, N)$. Розв'язок $y_1(x)$, $y_2(x), \dots,$ $y_k(x)$

на відрізку $[x_0, x_N]$ у точках $x_1, x_2, \dots, x_N$, які називають вузлами сітки. Позначимо:

$$Y = \begin{pmatrix} y_1(x) \\ y_2(x) \\ \dots \\ y_k(x) \end{pmatrix}, \qquad Y_0 = \begin{pmatrix} y_{0,1} \\ y_{0,2} \\ \dots \\ y_{0,k} \end{pmatrix}, \qquad Y' = \begin{pmatrix} y_1'(x) \\ y_2'(x) \\ \dots \\ y_k'(x) \end{pmatrix},$$

$$F(x, Y) = \begin{pmatrix} f_1(x, y_1, y_2, \dots y_k) \\ f_2(x, y_1, y_2, \dots y_k) \\ \dots \\ f_k(x, y_1, y_2, \dots y_k) \end{pmatrix},$$

де Y – шуканий розв'язок;

Y_0 – вектор початкових умов;

$F(x, Y)$ – вектор правих частин системи.

Запишемо задачу Коші для системи диференціальних рівнянь (10.1) у векторній формі:

$$Y' = F(x, Y),\ Y(x_0) = Y_0. \qquad (10.2)$$

Із застосуванням пакета ***deSolve*** розв'язати задачу Коші для системи (10.2) можна, наприклад, за допомогою функції rk4:

rk4(y0, time, Dfunc, parms) – розв'язання задачі на відрізку методом Рунге – Кутта.

Тут y0 – вектор початкових умов Y_0;

time – вектор вузлів сітки $x_1, x_2, \dots, x_N$;

Dfunc – векторна функція $F(x, Y)$ правих частин системи (10.2);

parms – вектор або список параметрів векторної функції $F(x, Y)$, якщо вони є.

Приклад 7. Застосування функції rk4 з метою розв'язання задачі Коші для диференціального рівняння другого порядку.

Знайдемо на відрізку [0, 3] наближений розв'язок рівняння

$$y'' = \exp(-xy),$$

що задовільнює початковим умовам

$$y(0) = 1,\ y'(0) = 1,$$

і побудуємо графік знайденого розв'язку.

Зведемо розв'язання задачі для рівняння другого порядку до задачі для еквівалентної нормальної системи першого порядку. Позначимо:

$$y_1 = y(x), \quad y_2 = y'(x).$$

Оскільки $y''(x) = (y'(x))' = y_2'(x)$, то маємо:

$$\begin{cases} y_1'(x) = y_2(x) \\ y_2'(x) = \exp(-xy_1) \end{cases}, \quad \begin{cases} y_1(0) = 1 \\ y_2(0) = 1 \end{cases}.$$

Розв'яжемо цю задачу чисельно, застосовуючи функцію **rk4** з фіксованим кроком на сітці з двадцяти рівновіддалених вузлів:

```
# Залаємо функцію правої частини диференційного рівняння
FunRight = function(t, y, parms) {
  dy = c(2)
  dy[1] = y[1]
  dy[2] = exp(-x*y[1])
  return(list(dy))
}
# Залаємо відправні дані
t0 = 0   # початкова точка  відрізка
t1 = 3   # кінцеве точка відрізка
y0 = c(1, 1) # початкове значення змінної y

library(deSolve)
n = 100                # кількість поділень відрізка
h = (t1 - t0)/n          # шаг поділення відрізка
time = seq(t0, t1, h) # сітка на відрізку
parms = c(1:1)         # невикористовувані параметри рівняння

# rk4 - метод Рунге-Кутта
out = rk4(y0, time, FunRight, parms)
out
T = out[, 1]
Y = out[, 2]
plot(T, Y, type = "l", col = "red")
```

Побудуємо графік (рис. 10.4):

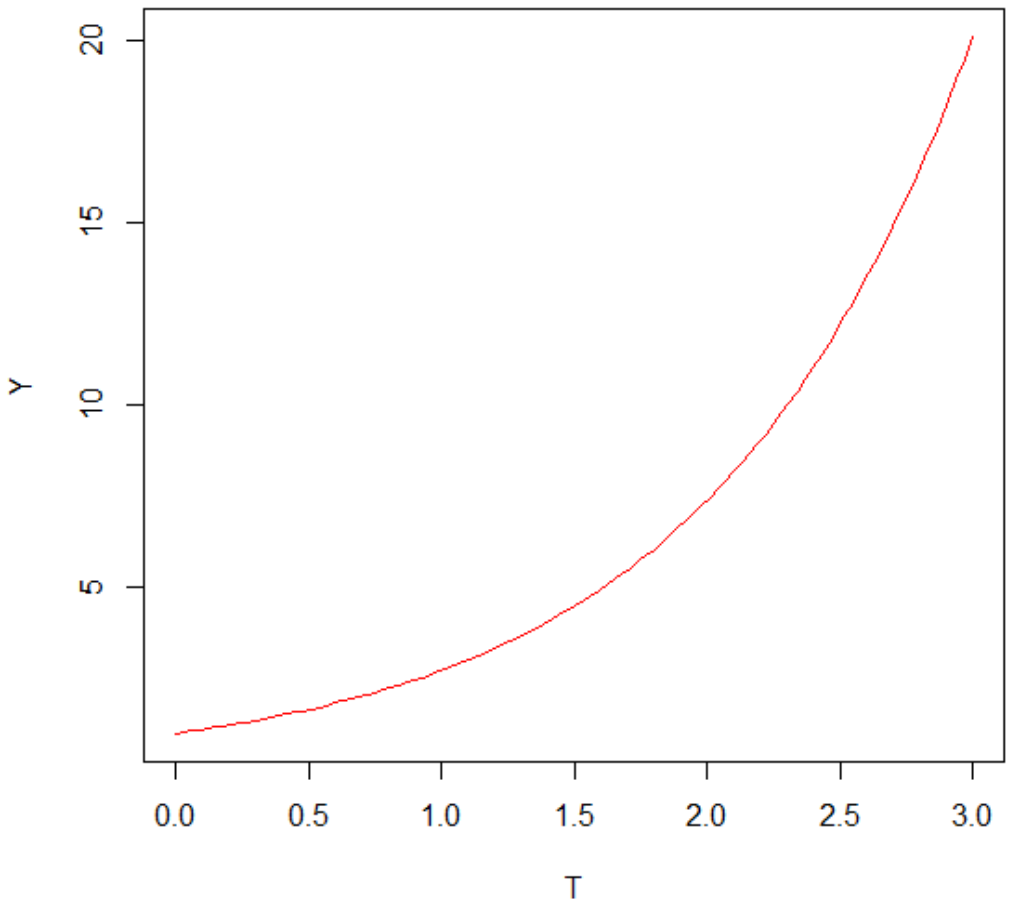

Рис. 10.4. **Графік розв'язку**

10.3. Порядок виконання роботи і варіанти завдань

10.3.1. Зміст звіту

У практичній частині роботи необхідно:
виконати всі приклади наведені в розділі 10.2;
виконати завдання, запропоновані у 10.3.2;
надати тексти складених програм і результати їх виконання.

10.3.2. Варіанти індивідуальних завдань

Виконати всі завдання:

1. Побудуйте графік значень випадкової величини, розподіленої за нормальним законом з параметрами: середнє значення – 0, середнє

квадратичне відхилення – 5, відносно номера експерименту з нею (порядкового номера).

2. Побудуйте графік значень випадкової величини, розподіленої за рівномірним законом на відрізку [-5, 5], відносно номера експерименту з нею (порядкового номера).

3. Побудуйте графіки функції розподілу та функції щільності розподілу ймовірності для випадкової величини, розподіленої за рівномірним законом на відрізку [5, 15].

4. Побудуйте матрицю розмірності 5 x 4, (i, j) – елемент якої дорівнює $i^2 - j^2$.

5. Розв'яжіть задачу Коші для системи диференціальних рівнянь першого порядку на відрізку [0, 2]:

$$\begin{cases} y_1' = -11y_1 + 9y_2 \\ y_2' = 9y_1 - 11y_2 \end{cases},$$

за початкових умов

$$y_1(0) = 1, \quad y_2(0) = 0$$

і побудуйте графік знайденого розв'язку.

6. Розв'яжіть задачу Коші для диференціального рівняння другого порядку на відрізку [0, 4π]:

$$y'' + xy = \frac{x}{4\pi},$$

за початкових умов

$$y(0) = 0, \quad y'(0) = 1$$

і побудуйте графік знайденого розв'язку.

10.4. Контрольні запитання

1. Які в системі R є засоби для генерації значень випадкових величин?

2. Які в системі R є засоби для створення та роботи з векторами та матрицями?

3. Які в системі R є засоби для чисельного розв'язання задач лінійної алгебри?

4. За допомогою яких засобів у системі R можна розв'язати задачу Коші для звичайних диференціальних рівнянь?

Лабораторна робота 11.
Обробка й аналіз даних експерименту. Підбір параметрів розподілу

11.1. Мета роботи

Засвоєння прийомів підбору законів розподілів випадкових величин за експериментальними даними з використанням середовища статистичного пакета **R**.

11.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* основні види законів розподілу випадкових величин; *уміти* розв'язувати задачу визначення параметрів розподілів шляхом перевірки статистичних гіпотез; *уміти* застосовувати процедури статистичного пакета **R** для визначення закону розподілу випадкових величин за вибірками [3; 11; 18; 19].

У деяких випадках імітаційна модель складної системи може бути реалізована у вигляді набору окремих моделей її підсистем. У ході проведення експериментів із такою моделлю з метою скорочення витрат часу буває необхідно замінити моделювання роботи однієї з підсистем деяким числовим параметром (принцип параметризації) або випадковою величиною, розподіленою за заданим законом. Щоб така заміна була виконана коректно, дослідник повинен мати у своєму розпорядженні опис залежності даного числового параметра від часу й інших факторів, що фігурують у моделі.

В імітаційному моделюванні підбір законів розподілів виконується на основі статистичних даних, отриманих у ході експерименту. В основу процедури відшукання закону розподілу деякої випадкової величини за експериментальними даними закладена перевірка статистичних гіпотез.

Перевірка гіпотези полягає в такій послідовності дій. На підставі вибірки значень випадкової величини (даних експерименту) обчислюється z – часткове значення деякого критерію Z. Якщо $z > z_{\text{кр}}$, то від гіпотези H_0 відмовляються. Якщо $z \le z_{\text{кр}}$, то говорять, що отримані спостереження не суперечать прийнятій гіпотезі. Тут $z_{\text{кр}}$ – граничне (критичне) значення критерію Z, що відповідає обраному рівню значущості α, обумовлене

стандартним чином (за таблицею або через обернену функцію відповідного розподілу).

Зрозуміло, перш ніж висувати гіпотезу щодо значень параметрів розподілу, необхідно визначити вид самого закону розподілу. Найпоширеніший на практиці та досить ефективний метод підбору виду розподілу заснований на використанні графічного подання експериментальних даних. Вони відображаються у вигляді так званої гістограми відносних частот, яка може бути побудована як вручну, так і за допомогою відповідних інструментальних засобів, що входять до складу більшості пакетів моделювання.

Для ефективного використання графічних засобів пакета **R** корисно знати **методику** покрокової **побудови гістограми** відносних частот.

Крок 1. Обчислюється величина інтервалу гістограми з такого співвідношення: $h = (ymin_{max}$, де $(ymax_{min}$ – діапазон зміни спостережуваної змінної y; m – число інтервалів, обраних дослідником.

Крок 2. За результатами (або в процесі) моделювання визначається число попадань значень у в i-й інтервал.

Крок 3. Обчислюється відносна частота попадань спостережуваної змінної в кожен інтервал:

$$g_i = \frac{n_i}{N},$$

де n_i – число попадань в i-й інтервал;

N – загальне число вимірів (обсяг вибірки).

Крок 4. На кожному i-му інтервалі будується прямокутник зі сторонами $h \times g_i$.Сума площ прямокутників гістограми дорівнює одиниці.

Для найчастіше використовуваних статистичних гіпотез розроблені критерії, що дозволяють проводити їхню перевірку з найбільшою вірогідністю. Розглянемо основні з них.

t-критерій слугує для перевірки гіпотези про рівність середніх значень двох нормально розподілених випадкових величин (x і y) у припущенні, що їх дисперсії рівні (хоча й невідомі). Порівнювані вибірки можуть мати різний обсяг (n_1 і n_2).

Як критерій використовують величину:

$$T = \frac{\bar{x} - \bar{y}}{\sqrt{(n_1 - 1)Dx + (n_2 - 1)Dy}} \sqrt{\frac{n_1 n_2 (n_1 + n_2 - 2)}{n_1 + n_2}}. \qquad (11.1)$$

Величина T підкоряється *t-розподілу* Стьюдента.

Критичне значення $t_{кр}$ для *t-критерію* визначається за таблицею для обраного значення α і числа степенів свободи $k = n_1 + n_2 - 2$.

Якщо обчислене за (11.1) значення задовольняє нерівності $T > t_{кр}$, то гіпотезу H_0 відкидають.

Стосовно припущення про "нормальну розподіленість" величин x і y *t-критерій* не дуже чутливий. Його можна застосовувати, якщо розподіли випадкової величини мають декілька вершин і не занадто асиметричні.

F-критерій призначений для перевірки гіпотез про рівність дисперсій Dx і Dy за умови, що випадкові величини x і y розподілені нормально.

Гіпотези такого роду мають велике значення в техніці, тому що дисперсія є мірою таких характеристик, як погрішності вимірювальних приладів, точність біологічних процесів, точність наведення для стріляння тощо.

Як контрольна величина використовується відношення дисперсій $F = Dx/Dy$ (або Dy/Dx – більша дисперсія повинна бути в чисельнику).

Величина F підкорюється *F-розподілу* (Фішера) з (m_1, m_2) степенями свободи $m_1 = n_1 - 1;\ m_2 = n_2 - 1$. Перевірка гіпотези полягає в такому. Для величини $a = \alpha/2$ і величин m_1, m_2 за таблицею *F-розподілу* вибирають значення $F_{a,m1,m2}$. Якщо обчислене за вибіркою F більше цього критичного значення, гіпотеза відхиляється з імовірністю помилки α.

Критерії згоди – це критерії, за допомогою яких перевіряють, чи задовільнює розглянута випадкова величина даному закону розподілу.

Критерій згоди Пірсона (χ^2) слугує для перевірки гіпотези H_0 про те, що $F_y(y) = F_0(y)$, де $F_y(y)$ – істинний розподіл випадкової величини y; $F_0(y)$ – гіпотетичний розподіл. Перевірка проводиться за такою послідовністю дій.

1. Область значень випадкової величини *y* розбивається (довільно) на m непересічних множин ("класів").

2. У результаті N експериментів формується вибірка $(y_1, \ldots, y_N)$.

3. Обчислюється контрольна величина χ^2:

$$\chi^2 = \sum_{i=1}^{m} \frac{(n_i - Np_i)^2}{Np_i}, \qquad (11.2)$$

де n_i – число значень y, що потрапили в i-й клас;

p_i – теоретична ймовірність (для $F_0(y)$) попадання значення y в i-й клас.

4. За таблицею χ^2-розподілів (або через обернену функцію χ^2-розподілу) знаходять критичне значення χ^2_α для рівня значущості α і $k = m - 1$ степенів свободи. Якщо $\chi^2 > \chi^2_\alpha$, то гіпотеза відкидається.

Зрозуміло, що проведення вручну розрахунків, необхідних для перевірки статистичних гіпотез, вимагає значних витрат часу й сил. Тому багато сучасних математичних пакетів, серед яких R, мають у своєму складі засоби, що дозволяють звести до мінімуму кількість операцій, виконуваних користувачем уручну.

11.3. Контрольні приклади

Приклад 1. Згенерувати вибірку для випадкової величини, що має нормальний закон розподілу з параметрами: середнє значення дорівнює 3, середнє квадратичне відхилення дорівнює 0.5.

Розв'язання. Виконаємо завдання з використанням пакета *R.*

Згенеруємо вибірку V із 100 значень випадкової величини, розподіленої за нормальним законом з параметрами: середнє значення дорівнює 3, середнє квадратичне відхилення дорівнює 0.5:

```
mean = 3
sd = 0.5
N=100
V = rnorm(N, mean, sd)
V
```

```
> mean = 3
> sd = 0.5
> N=100
> V = rnorm(N, mean, sd)
> V
 [1] 3.649932 3.180631 3.354651 2.492110 2.753465 3.028869 1.921820 2.549060
 [9] 2.990830 3.727476 3.050849 3.309048 2.942106 3.304776 2.836391 3.514170
[17] 3.826973 2.592287 3.068678 3.631452 3.053338 2.966088 2.410615 3.289271
[25] 2.295583 3.482386 3.136363 3.862807 2.934053 3.251396 3.097008 3.473262
[33] 3.376660 2.103254 2.848340 2.092730 2.279497 2.186818 3.597512 3.154971
[41] 3.004268 3.012171 3.353703 2.491733 2.765573 2.468789 3.196046 2.963648
```

```
[49] 2.965182 2.928811 3.157296 2.542312 3.121009 3.316966 3.533716 3.541426
[57] 3.209555 3.079594 3.417606 3.309514 3.425295 2.905591 3.160393 3.075602
[65] 2.451891 2.920443 2.780081 3.261926 3.269127 2.273307 3.858314 2.542832
[73] 2.548376 3.215115 2.264697 3.685690 2.664152 2.046300 3.094422 3.052247
[81] 2.582047 2.937942 2.574445 3.530718 3.647533 3.284510 3.235885 2.973506
[89] 2.879184 2.992299 2.687381 3.452414 2.593236 3.667018 2.562273 4.029166
[97] 2.942984 1.975018 3.401109 3.164180
```

Приклад 2. Використовуючи згенеровану вибірку з прикладу 1 як вихідні дані експерименту, оцінити вид і параметри закону розподілу відповідної випадкової величини.

Розв'язання. Побудуємо гістограму для вибірки V (рис. 11.1):

```
hist(V, col="red")
```

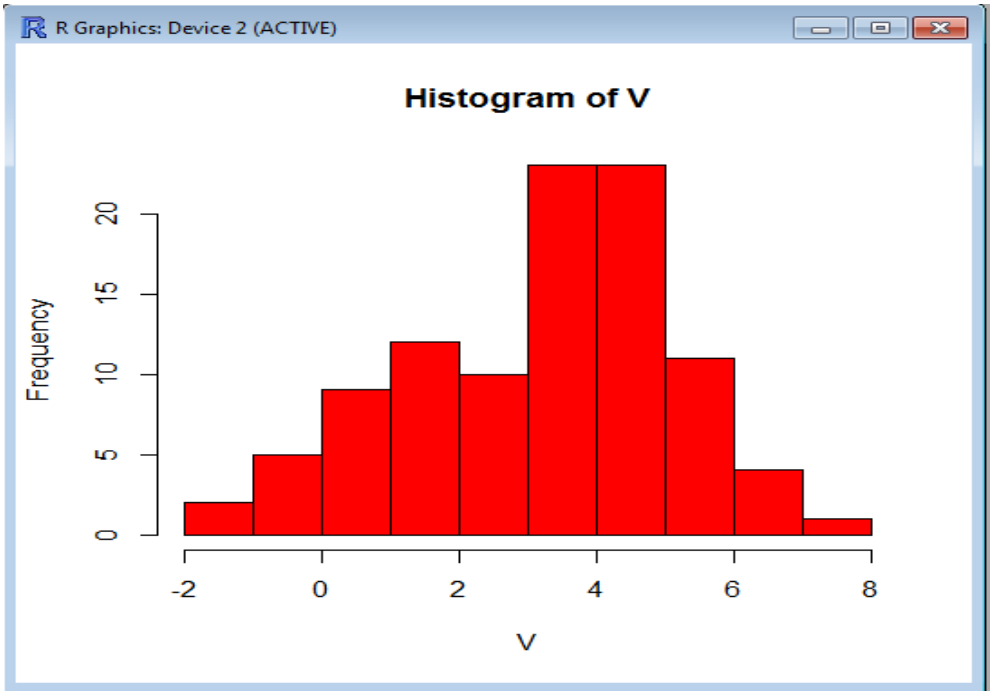

Рис. 11.1. **Гістограма вибірки**

Тут *hist* – процедура пакета R, що підраховує частоту попадання елементів вибірки V у кожний з інтервалів, на які поділена область побудови гістограм, від мінімального (min(V)) до максимального (max(V)) значення та будує стовпцевий графік.

З вигляду гістограми можна зробити висновок, що закон розподілу можливо нормальний. Знайдемо оцінки параметрів цього розподілу.

Оскільки передбачається, що закон розподілу є нормальним, а параметрами нормального закону є математичне очікування та середнє квадратичне відхилення, то знайдемо їх оцінки: вибіркове середнє *Vmean* і вибіркове середнє квадратичне відхилення *Vsd.*

```
> Vmean = sum(V)/N
> Vmean
[1] 3.011119
>
> S = 0
> for(i in 1:N){
+ S = S + (V[i]-Vmean)^2
+ }
> Vdisp = S/(N-1)
> Vdisp
[1] 0.2092162
>
> Vsd = sqrt(Vdisp)
> Vsd
[1] 0.4574015
```

Надалі висуваємо гіпотезу: закон розподілу випадкової величини, для якої в нас є вибірка V, є нормальним з параметрами a = Vmean, σ = Vsd.

Перевіряємо цю гіпотезу за критерієм Пірсона з рівнем значущості $\alpha = 0{,}95$.

```
m = 10
h = (max(V)- min(V))/m
int = c(1:(m+1))
for(i in 1:(m+1)){
  int[i] = min(V) + (i-1)*h
}
int
n = c(1:m)
p = c(1:m)
for(i in 1:m){
  p[i] = pnorm(int[i+1], Vmean, Vsd) - pnorm(int[i], Vmean, Vsd)
  n[i] = 0
```

```
  for(j in 1:N){
        if((V[j] >= int[i]) && (V[j] < int[i+1])){
              n[i] = n[i] + 1
        }
  }
}
p
n

HiSq = 0
for(i in 1:m){
  HiSq = HiSq + (n[i]-N*p[i])^2/(N*p[i])
}
HiSq

alfa = 0.95
HiSqAlfa = qchisq(alfa, m-1)
HiSqAlfa
```

Тут p_i – теоретична ймовірність попадання значення випадкової величини, розподіленої за нормальним законом з параметрами a = Vmean, σ = Vsd, в кожний з m інтервалів $(\mathrm{int_i}, \mathrm{int_{i+1}})$ на які поділена область від min(V) до max(V);

n_i – частота попадання елементів вибірки в ті самі інтервали $(\mathrm{int_i}, \mathrm{int_{i+1}})$;

pnorm – функція розподілу для нормального закону з пакета R;

qchisq – функція, обернена до функції χ^2-розподілу;

HiSq – значення критерію Пірсона;

HiSqAlfa – критичне значення для рівня значущості α і $(\mathrm{m}-1)$ степенів свободи.

```
> m = 10
> h = (max(V)- min(V))/m
> int = c(1:(m+1))
> for(i in 1:(m+1)){
+ int[i] = min(V) + (i-1)*h
+ }
> int
```

```
 [1] 1.814360 2.048153 2.281946 2.515739 2.749532 2.983325
3.217118 3.450911
 [9] 3.684704 3.918497 4.152290
> n = c(1:m)
> p = c(1:m)
> for(i in 1:m){
+ p[i] = pnorm(int[i+1], Vmean, Vsd) - pnorm(int[i], Vmean, Vsd)
+ n[i] = 0
+ for(j in 1:N){
+ if((V[j] >= int[i]) && (V[j] < int[i+1])){
+ n[i] = n[i] + 1
+ }
+ }
+ }
> p
 [1] 0.01318995 0.03781680 0.08394748 0.14429800 0.19207753
0.19800432
 [7] 0.15807259 0.09772550 0.04678411 0.01734135
> n
 [1]  1  3  8 18 20 19 11 13  3  3
>
> HiSq = 0
> for(i in 1:m){
+ HiSq = HiSq + (n[i]-N*p[i])^2/(N*p[i])
+ }
> HiSq
[1] 5.259694
>
> alfa = 0.95
> HiSqAlfa = qchisq(alfa, m-1)
> HiSqAlfa
[1] 16.91898
```

Оскільки HiSq ≤ HiSqAlfa, то висунута гіпотеза не відкидається. Таким чином, можна з імовірністю 95 % вважати, що випадкова величина, для якої в нас є вибірка, розподілена за нормальним законом з параметрами: середнє значення дорівнює 2.966, середнє квадратичне відхилення – 0.593.

11.4. Порядок виконання роботи та варіанти завдань

11.4.1. Зміст звіту

У практичній частині роботи необхідно:
виконати приклади, що наведені в підрозділі 11.3;
виконати завдання 1 і 2, наведені у 11.4.2;
надати тексти складених програм і результати їх виконання.

11.4.2. Варіанти індивідуальних завдань

Завдання 1. Згенеруйте вибірки для трьох випадкових величин, що мають такі закони розподілу: нормальний, рівномірний, пуасонівський із заданими параметрами розподілу (табл. 11.1).

Завдання 2. Використовуючи згенеровані вибірки як вихідні дані експерименту, визначте вид і параметри закону розподілу для відповідних випадкових величин.

Таблиця 11.1

Варіанти завдань

Варіант	Нормальний	Рівномірний	Пуасонівський
1	$\mu = 10, \sigma = 0.1$	$a = 5, b = 15$	$\lambda = 5$
2	$\mu = 11, \sigma = 0.2$	$a = 4, b = 14$	$\lambda = 4$
3	$\mu = 12, \sigma = 0.3$	$a = 6, b = 13$	$\lambda = 7$
4	$\mu = 13, \sigma = 0.4$	$a = 7, b = 14$	$\lambda = 8$
5	$\mu = 14, \sigma = 0.15$	$a = 8, b = 15$	$\lambda = 9$
6	$\mu = 15, \sigma = 0.25$	$a = 3, b = 12$	$\lambda = 10$
7	$\mu = 16, \sigma = 0.2$	$a = 9, b = 17$	$\lambda = 11$
8	$\mu = 17, \sigma = 0.21$	$a = 10, b = 20$	$\lambda = 12$
9	$\mu = 18, \sigma = 0.3$	$a = 11, b = 22$	$\lambda = 13$
10	$\mu = 19, \sigma = 0.5$	$a = 12, b = 23$	$\lambda = 14$
11	$\mu = 20, \sigma = 0.45$	$a = 13, b = 25$	$\lambda = 15$
12	$\mu = 21, \sigma = 0.6$	$a = 14, b = 26$	$\lambda = 16$
13	$\mu = 22, \sigma = 0.65$	$a = 15, b = 25$	$\lambda = 17$
14	$\mu = 9, \sigma = 0.26$	$a = 4, b = 10$	$\lambda = 6$
15	$\mu = 23, \sigma = 0.5$	$a = 16, b = 27$	$\lambda = 18$

11.5. Контрольні запитання

1. Який закон розподілу випадкової величини називають нормальним?
2. Який закон розподілу випадкової величини називають рівномірним?
3. Який закон розподілу випадкової величини називають пуасонівським?
4. Опишіть методику побудови гістограми відносних частот для вибірки.
5. Для чого використовується критерій згоди Пірсона? У чому полягає його сутність?

Лабораторна робота 12. Ідентифікація параметрів моделей методом найменших квадратів

12.1. Мета роботи

Вивчення методу найменших квадратів (МНК) для практичного розв'язання задач ідентифікації параметрів моделей; набуття навичок використання методу для розв'язання задачі ідентифікації параметрів моделей із застосуванням комп'ютера.

12.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* загальне формулювання задачі ідентифікації параметрів моделі; *уміти* розв'язувати задачі ідентифікації параметрів моделі методом найменших квадратів [6]; *уміти* застосовувати процедури пакета **R** для розв'язання задачі ідентифікації параметрів моделі [3, 5].

Задача ідентифікації параметрів моделі в загальному вигляді формулюється таким чином.

Нехай деяка система описується вхідними x і вихідними y змінними і яким-небудь чином обрана структура моделі (тобто залежність y від x):

$$y = G(x; a) + e, \tag{12.1}$$

де $a \in R^k$, $a = (a_1, a_2, \dots a_k)^T$ – деякі параметри моделі;

e – помилка моделі (враховуючи випадкові помилки експерименту).

Необхідно на основі результатів спостереження за вхідними та вихідними змінними системи (даних експерименту) й обраного критерію знайти оцінку параметрів моделі, тобто побудувати оптимальну в деякому розумінні математичну модель.

Найбільш відомим і ефективним із методів розв'язання задачі ідентифікації параметрів моделі є **метод найменших квадратів**, сутність якого полягає в такому. Розглянемо випадок, коли вхідних змінних *x* може бути декілька ($x \in R^m$), а вихідна змінна *y* одна ($y \in R^1$). Нехай є дані експериментів $(\mathrm{x}^{(\mathrm{i})}, \mathrm{y_i})$, $\mathrm{i} = \overline{1,\mathrm{n}}$, причому значення y_i містять випадкову помилку. Вводиться функція (від параметрів a) виду:

$$\Phi(\mathrm{a}) \equiv \sum_{\mathrm{i}=1}^{\mathrm{n}} \left[(\mathrm{y_i} - \mathrm{G}(\mathrm{x}^{(\mathrm{i})}; \mathrm{a})\right]^2, \qquad (12.2)$$

яку можна розглядати як міру відхилення функції моделі $\mathrm{G}(\mathrm{x};\mathrm{a})$ від даних експерименту $\mathrm{y_1}, \mathrm{y_2}, \ldots, \mathrm{y_n}$. Тоді оцінку параметрів a моделі $\mathrm{G}(\mathrm{x};\mathrm{a})$ можна визначити з умови найменшого відхилення $\mathrm{G}(\mathrm{x};\mathrm{a})$ від даних експерименту. Тобто оцінку параметрів a знаходять як точку, в якій функція $\Phi(\mathrm{a})$ досягає по $\mathrm{a} \in \mathrm{R^k}$ мінімального значення (точка мінімуму). Нехай a^* – шукана оцінка параметрів. Тоді величини $\mathrm{r_i} \equiv \mathrm{y_i} - \mathrm{G}(\mathrm{x}^{(\mathrm{i})}; \mathrm{a}^*)$, $\mathrm{i} = \overline{1,\mathrm{n}}$, називають **залишками** (залишковими відхиленнями), які використовуються для аналізу отриманих рішень.

Для пошуку точки мінімума функції $\Phi(\mathrm{a})$ можна застосувати функцію **optim**(fn, par) пакета **R**, де fn – ім'я цільової функції, par – початкове наближення точки мінімума.

12.3. Контрольні приклади

Приклад 1. Знайдіть оцінку параметрів моделі з однією вхідною й однією вихідною змінними, обраної у вигляді полінома другого ступеня. Для оцінки параметрів моделі застосуйте метод найменших квадратів. Дані експерименту подані в табл. 12.1.

Таблиця 12.1

Дані експерименту

X	0.5	2.5	4.5	6.5	8.5
Y	4.0	205	124	384	875

Розв'язання. Виконаємо завдання з використанням процедури *optim* пакета R.

```
n = 5
x = c(0.5, 2.5, 4.5, 6.5, 8.5)
y = c(4, 205, 124, 384, 875)
# Задаємо вид функції моделі
FunModel = function(x,a) { a[1] + a[2]*x + a[3]*x^2 }
# Визначаємо цільову функцію метода МНК
FunMNK = function(a){
        S = 0
        for(i in 1:n) { S = S + (y[i]-FunModel(x[i],a))^2 }
        return(S)
}
# Задаємо початкове значення для параметрів а функції моделі
a0 = c(0, 0, 0)
# Знаходимо точку мінімуму функції FunMNK
res = optim(fn=FunMNK, par=a0)
a1 = res$par
a1
```

Результат виконання програми:

```
[1]  87.64231 -51.96668  16.44629
```

Тобто, a[1] = 87.64231; a[2] = -51.96668; a[3] = 16.44629. Для перевірки правильності отриманого розв'язку, а також наочної інтерпретації результатів моделювання побудуємо графіки (рис. 12.1):

```
# Обчислюємо значення функції моделі в токах x
ym = FunModel(x,a1)
# Обчислюємо залишки
r = y - ym
# Обчислюємо 100 точок на відрізку [0,40]
a = 0; b = 40; m = 100
h = (b-a)/(m-1)
x1 = c(1:m)
for(i in 1:m){ x1[i] = a + h*(i-1) }
# Обчислюємо значення функції моделі  в токах x1
y1m = FunModel(x1,a1)
par(mfrow=c(1, 2))          # задаємо дві графічні підобласті
```

```
plot(x,y,type="p",col="red") # виводимо перший графік у вигляді точок
lines(x1,y1m,col="blue")   # додаємо у 1-шу графічну підобласть ще лінію
plot(x,r,type="p")          # виводимо другий графік у вигляді точок
lines(x,r,col="blue")       # додаємо у 2-гу графічну підобласть ще лінію
```

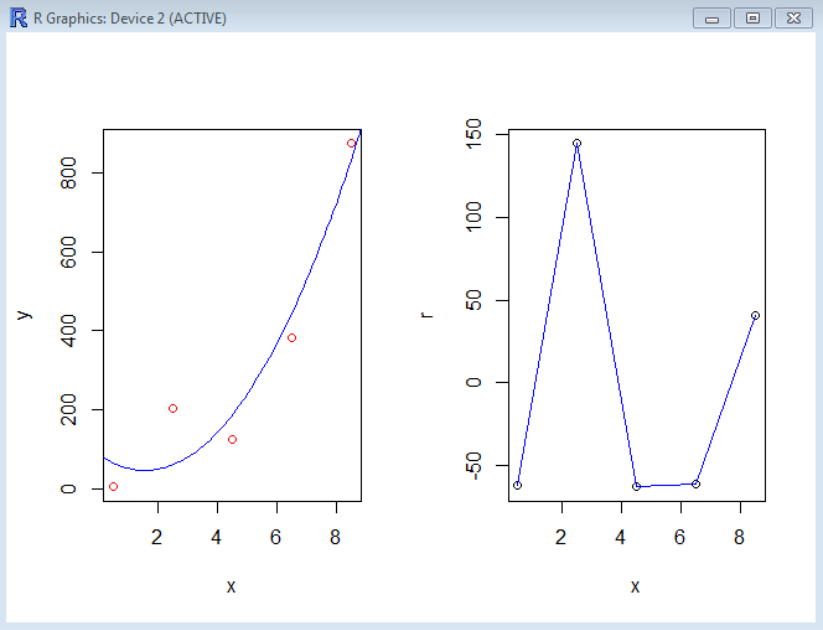

Рис. 12.1. **Графік результатів оцінки параметрів моделі**

Якщо структура моделі (12.1) була обрана правильно, то графік залишків повинен мати хаотичний вигляд, тобто бути схожим на графік поведінки випадкової величини. У нашому випадку видно, що графік залишків містить у собі деяку закономірність (можливо, кубічну залежність). Тобто структуру моделі (12.1), можливо, треба змінити.

Приклад 2 (модель роботи насосу). Залежність (HQ-характеристика насосу) створюваного насосом напору (H) від витрати води (Q) описується функцією $\mathbf{H = H_F - S_F Q^{\beta}}$. У результаті проведення натурних експериментів отримані дані, подані в табл. 12.2.

Таблиця 12.2

Дані натурних експериментів

Q, м^3/год.	200	500	900	1300	2000	2600
H, м вод. ст.	82	81	79.5	77	69	61.5

Необхідно знайти параметри (H_F, S_F, β) HQ-характеристики насоса і побудувати її графік.

Розв'язання. Виконаємо завдання з використанням процедур пакета **R**.

Оскільки розглядається залежність напору (H) від витрати води (Q), то Q – вхідна, а H – вихідна змінні. Тут функція $H = H_F - S_F Q^{\beta}$ – це структура моделі, а (H_F, S_F, β) – параметри моделі.

```
# Вводимо початкові данні
n = 6
x = c(200, 500, 900, 1300, 2000, 2600)
y = c(82, 81, 79.5, 77, 69, 61.5)
# Задаємо вид функції моделі
FunModel = function(x,a) { a[1] - a[2]*x^a[3] }
# Визначаємо цільову функцію метода МНК
FunMNK = function(a){
  S = 0
  for(i in 1:n) { S = S + (y[i]-FunModel(x[i],a))^2 }
  return( S )
}
# Задаємо початкове значення для параметрів а функції моделі
a0 = c(82, 0, 2)
# Знаходимо точку мінімуму функції FunMNK
res = optim(fn=FunMNK, par=a0)
a1 = res$par
a1
```

Результат виконання програми:

```
[1]  8.197831e+01 4.457792e-06 1.952304e+00
```

Для перевірки правильності отриманого розв'язку, а також наочної інтерпретації результатів моделювання побудуємо графіки (рис.12.2):

```
# Обчислюємо значення функції моделі в токах x
ym = FunModel(x,a1)
# Обчислюємо залишки
r = y - ym
```

```
# Обчислюємо 100 точок на відрізку [100,2700]
a = 100; b = 2700; n = 100
h = (b-a)/(n-1)
x1 = c(1:n)
for(i in 1:n){ x1[i] = a + h*(i-1) }
# Обчислюємо значення апроксимуючої функції в токах x1
y1m = FunModel(x1,a1)
par(mfrow=c(1, 2))          # задаємо дві графічні підобласті
plot(x,y,type="p",col="red") # виводимо перший графік у вигляді точок
lines(x1,y1m,col="blue")   # додаємо у 1-шу графічну підобласть ще лінію
plot(x,r,type="p")          # виводимо другий графік у вигляді точок
lines(x,r,col="blue")       # додаємо у 2-гу графічну підобласть ще лінію
```

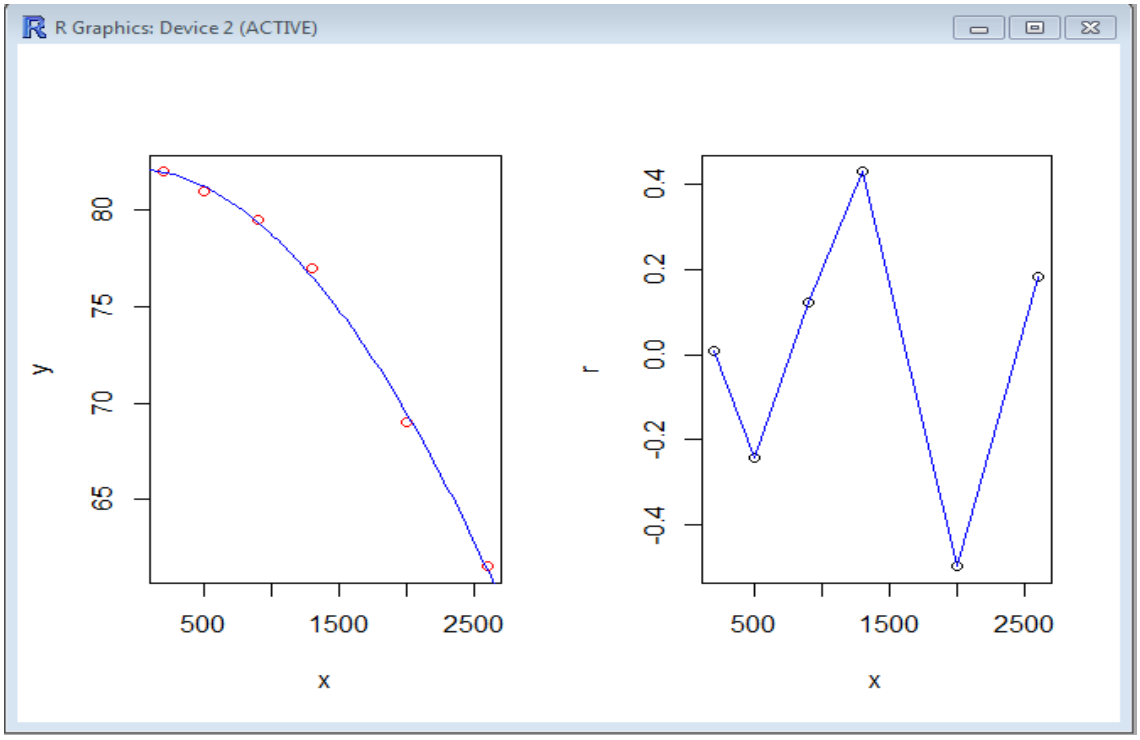

Рис. 12.2. **Графік результатів оцінки параметрів моделі**

З графіків видно, що модель добре лягає на дані експерименту, а графік залишків має хаотичний вигляд. Тому є сенс оцінити дисперсію та середнє квадратичне відхилення залишків:

```
> S = 0
> for(i in 1:n){
+ S = S + r[i]^2
+ }
```

```
> Vdisp = S/(n-1)
> Vdisp
[1] 0.1165765
>
> Vsd = sqrt(Vdisp)
> Vsd
[1] 0.341433
```

Як видно, оцінка середнього квадратичного відхилення залишків досить мала відносно значень вихідної змінної H, тому можна вважати, що модель (HQ-характеристика насосу) побудована добре.

12.4. Порядок виконання роботи і варіанти завдань

12.4.1. Зміст звіту

У практичній частині роботи необхідно:
виконати приклади, що наведені в підрозділі 12.3;
виконати завдання 1 і 2, наведені в 12.4.2;
надати тексти складених програм та результати їх виконання.

12.4.2. Варіанти індивідуальних завдань

Завдання 1. Згенеруйте дані експерименту, використовуючи гіпотетичну залежність $y = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + \varepsilon$ між вхідною змінною x і вихідною змінною y, де випадкові помилки ε розподілені за рівномірним законом розподілу. Значення змінної x у кількості n візьміть на відрізку від x_1 до x_n з постійним кроком. Дані для значень n, x_1, x_n і параметрів залежності b_j візьміть з відповідного варіанта завдань. Вважайте, що випадкова помилка ε розподілена на інтервалі (-a, a), де a дорівнює $0{,}05\, y_{сер}$, $y_{сер}$ – середнє значення змінної y в даних експерименту.

Завдання 2. Використовуючи процедури пакета R, знайдіть параметри моделі (12.1) за згенерованими в першому завданні даними експерименту. В якості структури моделі візьміть поліноми першого, другого

та третього степеня. Побудуйте графіки функції моделі та відповідних залишків. Зробіть висновки відносно обраної структури моделі.

Таблиця 12.3

Варіанти завдань

Варіант	n	x_1	x_n	Вектор b
1	20	-3	3	(-4; -0,8; 1,6; 2,3)
2	20	0,5	4	(0,4; 0,3; 1,0; 1,7)
3	25	4,5	8	(7,7; 9,4; 11,4; 13,6)
4	25	0,4		(0,43; 0,94; 1,91; 3.6)
5	15	4	6,8	(13,88; 16,93; 20,47; 24.15)
6	15	1	6	(2,11; 2,45; 2,61; 2.73)
7	30	0,3	1,8	(4,39; 3,75; 4,98; 5.11)
8	30	1	6	(0,1; 0,21; 0,43; 0.5)
9	10	1	4	(4,11; 4,16; 3,23; 3.29)
10	15	0.5	4	(2,47; 2,86; 3,01; 2.91)
11	22	0,68	0,9	(0,8; 0,89; 1,02; 0.06)
12	23	0,43	0,8	(1,63; 1,73; 1,87; 0.7)
13	21	0,02	0,3	(1,02; 1,09; 1,14; 1.21)
14	19	0,11	0,5	(9,05; 6,61; 4,69; 1.6)
15	24	0,35	0,7	(2,47; 2,91; 2,01; 2.1)

12.5. Контрольні запитання

1. У чому полягає сутність метода найменших квадратів для ідентифікації параметрів моделі?
2. Що є вихідними даними при застосуванні метода найменших квадратів для ідентифікації параметрів моделі?
3. Для чого будують графік моделі?
4. Для чого будують графік залишків?
5. Який характер поведінки графіка залишків відповідає правильно обраній структурі моделі?

Лабораторна робота 13.
Побудова багатофакторної регресійної моделі

13.1. Мета роботи

Вивчення методу найменших квадратів (МНК) для практичного розв'язання задачі побудови багатофакторної регресійної моделі; набуття навичок використання цього методу для розв'язання задачі множинної лінійної регресії із застосуванням комп'ютера.

13.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* загальне формулювання задачі множинної регресії; *уміти* розв'язувати задачу побудови багатофакторної лінійної регресійної моделі методом найменших квадратів [7; 13].

Загальна задача множинної регресії полягає в знаходженні за даними експерименту залежності (у загальному випадку нелінійної) деякої змінної y від декількох факторів (змінних) $x_1, x_2, \dots, x_m$ ($x \in R^m$). У випадку розв'язання задачі лінійної множинної регресії структура моделі розглядається (у загальному випадку) у вигляді залежності:

$$y = b_0 + b_1\varphi_1(x) + \dots + b_{k-1}\varphi_{k-1}(x) + e, \qquad (13.1)$$

де $\varphi_j(x), j = \overline{1, k-1}$ – деякі задані функції від $x \in R^m$ (узагальнені регресори);

$b_j, j = \overline{0, k-1}$ – деякі коефіцієнти (параметри регресійної моделі);

e – помилка моделі (враховуючи і випадкові помилки експерименту).

Зазначимо, що залежність (13.1) має вид лінійної комбінації функцій $\varphi_j(x), j = \overline{1, k-1}$. Структура моделі лінійно залежить від параметрів $b_j, j = \overline{0, k-1}$.

Модель (13.1) є частковим випадком моделі (12.1), а саме:

$$G(x, a) = a_1 + a_2\varphi_1(x) + \dots + a_k\varphi_{k-1}(x) + e, \qquad (13.2)$$

де $a \in R^k$, $a_j = b_{j+1}$, $j = 1, \dots, k$.

Тоді для даних експерименту $(x^{(i)}, y_i)$, $i = \overline{1,n}$, $(x^{(i)} \in R^m)$, оцінки параметрів моделі за методом МНК треба знаходити з умови мінімуму за a функції:

$$\Phi(a) \equiv \sum_{i=1}^{n} \left[(y_i - G(x^{(i)}; a) \right]^2. \qquad (13.3)$$

Введемо позначення:

$$Y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, F = \begin{pmatrix} 1 & \varphi_1(x^1) & \dots & \varphi_{k-1}(x^1) \\ 1 & \varphi_1(x^2) & \dots & \varphi_{k-1}(x^2) \\ \dots & \dots & \dots & \dots \\ 1 & \varphi_1(x^n) & \dots & \varphi_{k-1}(x^n) \end{pmatrix}.$$

Тоді, згідно з (13.2) та (13.3), функція $\Phi(a)$ може бути записана в матричному вигляді:

$$\Phi(a) = \|Y - Fa\|^2.$$

Відповідно до необхідної умови мінімуму першого порядку для функції $\Phi(a)$ оцінки параметрів a можуть бути знайдені з рівняння:

$$\Phi'(a) = 2\, F^T (Y - Fa) = 0,$$

звідки – за умови, що $\mathrm{rank}(F) = k$, тобто матриця $(F^T F)$ не вироджена:

$$a^* = (F^T F)^{-1} F^T Y. \qquad (13.4)$$

Таким чином, параметри b_j, $j = \overline{0, k-1}$ моделі (13.1) дорівнюють $b_{j-1} = a_j^*$, $j = \overline{1,k}$. Вектор залишків $r = Y - Fa^*$.

13.3. Контрольні приклади

Приклад 1. Необхідно побудувати двофакторну лінійну регресійну модель виду:

$$y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_1 x_2 + b_4 x_1^2 + b_5 x_2^2 \qquad (13.5)$$

за даними експерименту (табл. 13.1).

Таблиця 13.1

Дані експерименту

№ експерименту	x_1	x_2	Y
1	−1	−1	4,7
2	1	−1	8,6
3	−1	1	11,2
4	1	1	20,9
5	0	0	7,35
6	0	1	17,0
7	0	−1	7,4
8	1	0	10,2
9	−1	0	3,4

Розв'язання. У цьому випадку, відповідно до (13.1), $\mathbf{k = 6}$, $\boldsymbol{\varphi_0(x) = 1}$, $\boldsymbol{\varphi_1(x) = x_1, \varphi_2(x) = x_2, \varphi_3(x) = x_1x_2, \varphi_4(x) = x_1^2, \varphi_5(x) = x_2^2}$.

Введемо відправні дані задачі:

```
m = 2 # кількість факторів
k = 6 # кількість функцій моделі регресії
n = 9 # кількість експериментів

X=matrix(c(-1, 1, -1, 1, 0, 0, 0, 1, -1, -1, -1, 1, 1, 0, 1, -1, 0, 0), nrow=n, ncol=m)
Y=c(4.8, 8.7, 11.2, 20.9, 7.3, 17.0, 7.4, 10.2, 3.4)

FunFi = function(x){
    Fi = c(1:k)
    Fi[1] = 1
    Fi[2] = x[1]
    Fi[3] = x[2]
    Fi[4] = x[1]*x[2]
    Fi[5] = x[1]*x[1]
    Fi[6] = x[2]*x[2]
    return (Fi)
}
```

Визначимо й обчислимо матрицю F:

```
> F = matrix(0, nrow=n, ncol=k)
> for(i in 1:n){
+ Fi = FunFi(X[i,])
```

```
+ for(j in 1:k){
+ F[i,j] = Fi[j]
+ }
+ }
> F
     [,1] [,2] [,3] [,4] [,5] [,6]
 [1,]    1   -1   -1    1    1    1
 [2,]    1    1   -1   -1    1    1
 [3,]    1   -1    1   -1    1    1
 [4,]    1    1    1    1    1    1
 [5,]    1    0    0    0    0    0
 [6,]    1    0    1    0    0    1
 [7,]    1    0   -1    0    0    1
 [8,]    1    1    0    0    1    0
 [9,]    1   -1    0    0    1    0
>
```

Знайдемо оцінки параметрів моделі та залишки:

```
> b = solve(t(F)%*%F)%*%t(F)%*%Y
> b
          [,1]
[1,]  7.433333
[2,]  3.400000
[3,]  4.700000
[4,]  1.450000
[5,] -0.700000
[6,]  4.700000
>
> r = Y - F%*%b
> r
             [,1]
 [1,]  0.01666667
 [2,]  0.01666667
 [3,] -0.08333333
 [4,] -0.08333333
 [5,] -0.13333333
 [6,]  0.16666667
 [7,] -0.03333333
 [8,]  0.06666667
 [9,]  0.06666667
```

Для перевірки правильності отриманого розв'язку, а також наочної інтерпретації результатів моделювання побудуємо графік залишків (рис. 13.1) і знайдемо оцінки дисперсії та середньоквадратичного відхилення залишків:

```
> plot(r, type = "l", col = "red")
>
>
> S = 0
> for(i in 1:n){
+ S = S + (r[i])^2
+ }
> Vdisp = S/(n-1)
> Vdisp
[1] 0.00875
>
> Vsd = sqrt(Vdisp)
> Vsd
[1] 0.09354143
```

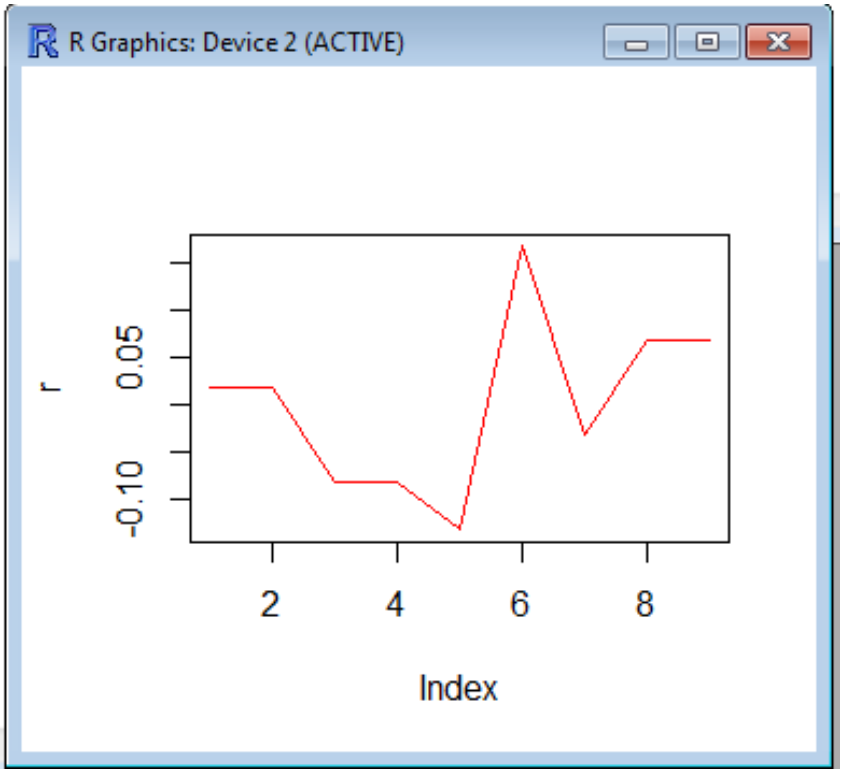

Рис. 13.1. **Графік залишків**

З вигляду графіка залишків і значення оцінки середньоквадратичного відхилення залишків Vsd можна зробити висновок про те, що лінійна

регресійна модель (13.5) досить добре описує залежність вихідної змінної y від факторів x_1 і x_2, хоча деяка залежність у залишках і спостерігається.

Приклад 2. Треба знайти функцію, що описує залежність показника міцності сталі (y) від %-го складу вуглецю в сталі (x1) і температури відпуску сталі протягом години (x2) за такими даними експерименту:

$$X := \begin{pmatrix} 6 & 133 \\ 10 & 122 \\ 23 & 112 \\ 24 & 103 \\ 15 & 116 \\ 24 & 99 \\ 57 & 83 \\ 54 & 69 \\ 65 & 76 \\ 66 & 74 \\ 68 & 61 \\ 62 & 38 \\ 83 & 40 \\ 100 & 47 \\ 116 & 31 \end{pmatrix} \qquad Y := \begin{pmatrix} 555 \\ 499 \\ 588 \\ 559 \\ 608 \\ 507 \\ 603 \\ 653 \\ 661 \\ 678 \\ 661 \\ 708 \\ 724 \\ 703 \\ 749 \end{pmatrix}$$

У першому стовпці матриці X знаходиться %-й склад вуглеця в сталі, помножений на 100, а в другому – температура відпуску сталі протягом години (в F^0), помножена на 10. У векторі Y знаходяться відповідні значення показника міцності.

Розв'язання. Застосуємо також модель (13.5):

```
m = 2 # кількість факторів
k = 6 # кількість функцій моделі регресії
n = 15 # кількість експериментів

X1 = c(6, 10, 23, 24, 15, 24, 57, 54, 65, 66, 68, 62, 83, 100, 116)
X2 = c(133, 122, 112, 103, 116, 99, 83, 69, 76, 74, 61, 38, 40, 47, 31)
X = cbind(X1, X2)
```

```
Y = c(555, 499, 588, 559, 608, 507, 603, 653, 661, 678, 661, 708, 724, 703, 749)
C = cbind(X,Y)

# завдання регресорів для моделі
FunFi = function(x){
    Fi = c(1:k)
    Fi[1] = 1
    Fi[2] = x[1]
    Fi[3] = x[2]
    Fi[4] = x[1]*x[2]
    Fi[5] = x[1]*x[1]
    Fi[6] = x[2]*x[2]
    return(Fi)
}
# обчислення матриці F
F = matrix(0, nrow=n, ncol=k)
for(i in 1:n){
    Fi = FunFi(X[i,])
    for(j in 1:k){
        F[i,j] = Fi[j]
    }
}
```

Знайдемо оцінки параметрів моделі за формулою (13.4).

```
> # знаходження оцінки параметрів моделі
> b = solve(t(F)%*%F)%*%t(F)%*%Y
> b
             [,1]
[1,] 529.37979667
[2,]   7.99654964
[3,]  -4.83972877
[4,]  -0.01246989
[5,]  -0.04126188
[6,]   0.03620605
```

Для перевірки правильності отриманого розв'язку, а також наочної інтерпретації результатів моделювання побудуємо графік залишків (рис. 13.2)

і знайдемо оцінки дисперсії та середньоквадратичного відхилення залишків:

```
> # обчислення залишків моделі
> r = Y - F%*%b
> # побудова графіка залишків моделі
> plot(r, type = "l", col = "red")
>
> # обчислення оцінки дисперсії залишків моделі
> S = 0
> for(i in 1:n){
+ S = S + (r[i])^2
+ }
> Vdisp = S/(n-1)
> Vdisp
[1] 705.283
> # обчислення оцінки середньоквадратичного відхилення залишків моделі
> Vsd = sqrt(Vdisp)
> Vsd
[1] 26.55716
```

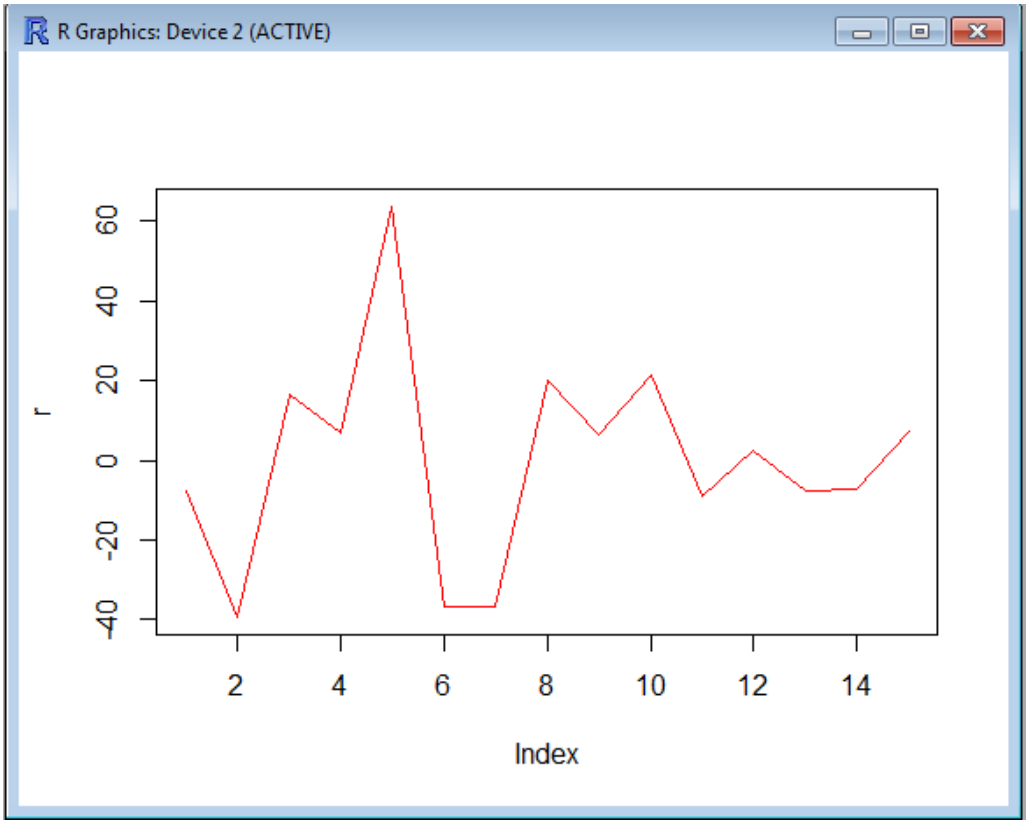

Рис. 13.2. **Графік залишків**

Для ілюстрації залежності показника міцності сталі від %-го складу вуглецю в сталі та температури відпуску сталі протягом години побудуємо тривимірний графік (рис.13.3).

```
# задання функції багатофакторної лінійної регресійної моделі
FunModel = function(x, a){
        Fi = FunFi(x)
        S = 0
        for(j in 1:k){
                S = S + a[j]*Fi[j]
        }
        return(S)
}

# задання точок сітки для побудови графіка
N1 = 10
N2 = 10
x1 = c(1:N1)
x2 = c(1:N2)
y = matrix(0, nrow=N1, ncol=N2)
h1 = (max(X1) - min(X1))/(N1-1)
h2 = (max(X2) - min(X2))/(N2-1)
minX1 = min(X1)
minX2 = min(X2)
for(i in 1:N1){
        x1[i] = minX1 + h1*(i-1)
}
for(i in 1:N2){
        x2[i] = minX2 + h2*(i-1)
}
# обчислення значень функції багатофакторної регресійної моделі на сітці
xv = c(1:2)
for(i in 1:N1){
        xv[1] = x1[i]
        for(j in 1:N2){
                xv[2] = x2[j]
                y[i,j] = FunModel(xv, b)
        }
}
```

```
# побудова графіка залежності показника міцності сталі від %-го складу
# вуглецю в сталі та температури відпуску сталі протягом години
library(rgl)
ylim = range(y)
ylen = ylim[2] - ylim[1] + 1
colorlut = terrain.colors(ylen) # height color lookup table
col = colorlut[ y-ylim[1]+1 ]   # assign colors to heights for each point
#rgl.viewpoint(zoom = 0.5)

rgl.surface(x1, x2, y, coords=1:3, color=col, back="fill")
axes3d()                # виведення осей на графік
title3d('Dependence of the strength of steel','','x1','y','x2') # виведення назв осей
```

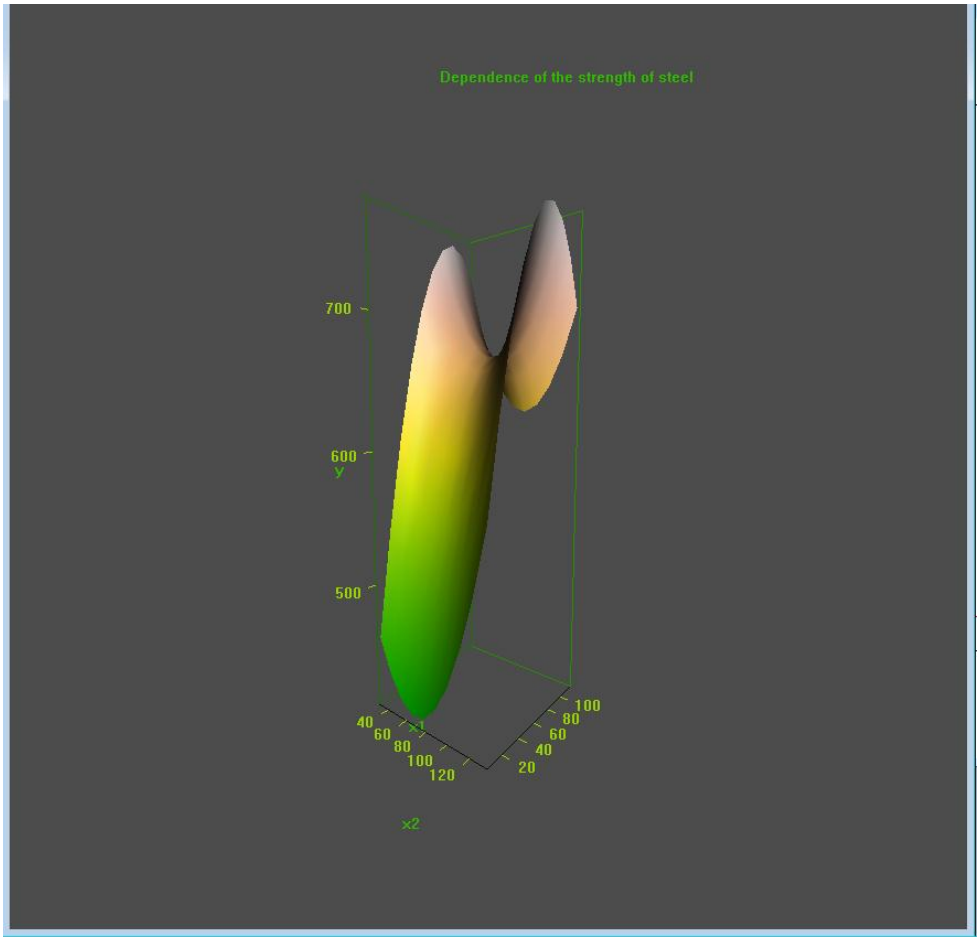

Рис. 13.3. **Графік залежності показника міцності сталі від %-го складу вуглецю в сталі та температури відпуску сталі протягом години**

Як видно з графіка (рис.13.3), залежність показника міцності сталі від %-го складу вуглецю в сталі та температури відпуску сталі протягом години має вигляд "сідла".

13.4. Порядок виконання роботи і варіанти завдань

13.4.1. Зміст звіту

У практичній частині роботи необхідно:
виконати приклади, що наведені в підрозділі 13.3;
виконати індивідуальне завдання з пункту 13.4.2;
надати тексти складених програм і результати їх виконання.

13.4.2. Варіанти індивідуальних завдань

Побудувати двофакторну лінійну регресійну модель виду:

$$y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_1 x_2 + b_4 x_1^2 + b_5 x_2^2$$

за даними експерименту ($n = 9$, фактор x_1 приймає значення в першому рядку таблиці, фактор x_2 приймає значення у першому стовпці таблиці, значення змінної y перебувають на перетині).

Варіант 1			
2ф/1ф	0	1	2
20	14	19,1	21,5
10	4	7,9	9,5
0	5,8	8,6	8,7

Варіант 2			
2ф/1ф	0	10	20
0	13	18,1	20,5
1	3	6,9	8
2	4,8	7,6	7,7

Варіант 3			
2ф/1ф	0	10	20
1	4,8	7,4	8,6
2	3	7,2	10
3	11	17	21,5

Варіант 4			
2ф/1ф	1	2	3
20	4,8	7,4	8,6
10	3,5	7	10
0	11,1	17	21,5

Варіант 5			
2ф/1ф	0	10	20
1	6,5	8,5	9,8
2	4,4	9,2	13
3	9,2	17	24

Варіант 6			
2ф/1ф	0	1	2
0	13,1	18,1	20,5
10	3	6,9	8,1
20	4,8	7,6	7,7

Варіант 7			
2ф/1ф	1	2	3
0	13	18,1	20,2
10	3,5	6,9	8
20	4,8	7,6	7,7

Варіант 8			
2ф/1ф	0	1	2
0	6,8	9,3	10,5
10	5,5	9,2	12
20	13,1	19	23,4

Варіант 9			
2ф/1ф	1	2	3
20	5,9	8,4	9,6
10	4,5	8,2	11
0	12,1	18	22,5

Варіант 10			
2ф/1ф	0	10	20
1	6,5	8,5	9,8
2	4,4	9,2	13
3	9,2	16,6	24

Варіант 11			
2ф/1ф	0	10	20
0	13	18,2	20,5
1	3	6,9	8,1
2	4,8	7,6	7,7

Варіант 12			
2ф/1ф	1	2	3
0	13	18,1	20,4
10	3,1	6,9	8
20	4,8	7,6	7,7

Варіант 13			
2ф/1ф	0	1	2
20	14	19,1	21,5
10	4	7,9	9,1
0	5,8	8,6	8,8

Варіант 14			
2ф/1ф	0	10	20
0	13	18,1	20,5
1	3	6,9	8,1
2	4,9	7,6	7,7

Варіант 15			
2ф/1ф	0	10	20
1	4,8	7,3	8,6
2	3,5	7,2	10
3	11	17	21,5

13.5. Контрольні запитання

1. У чому полягає загальна задача множинної регресії?
2. Які моделі називають лінійними відносно параметрів моделі?
3. Який метод найчастіше застосовують для оцінювання параметрів моделі?
4. Як знаходять параметри моделі методом МНК, якщо модель лінійна відносно цих параметрів?

5. Для чого будують графік моделі?

6. Для чого будують графік залишків?

7. Який характер поведінки графіка залишків відповідає правильно обраній структурі моделі?

Лабораторна робота 14. Дослідження імітаційних моделей

14.1. Мета роботи

Вивчення способів проведення експериментів із динамічними стохастичними моделями із застосуванням комп'ютера.

14.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати,* що таке динамічна стохастична модель і закони розподілу випадкових величин; *уміти* оцінювати закон розподілу випадкової величини за вибіркою її значень [3; 5; 8; 10; 12; 13], розв'язувати нелінійні рівняння із застосуванням процедур математичного пакета R.

14.3. Контрольні приклади

Розглянемо декілька прикладів дослідження імітаційних моделей.

Приклад 1 [5]**.** Нехай на деякому підприємстві для водопостачання використовується резервуар, об'єм якого становить **W** тис. л. Рівень споживання води – $\mathbf{V_C}$ тис. л за добу, а швидкість наповнення резервуара насосами – $\mathbf{V}_\mathrm{H}$ тис. л за добу. Необхідно знайти час **T**, за який буде заповнено резервуар (спочатку пустий), за таких умов:

1) рівень споживання та швидкість наповнення резервуара V_H – постійні ($V_C = 480$ тис. л/доб., $V_H = 500$ тис. л/доб., $W = 1\ 000$ тис. л);

2) рівень споживання води на підприємстві має ймовірнісний характер і змінюється згідно з рівномірним розподілом ймовірності в межах $\mathrm{V_C} \pm \Delta\mathrm{V_C}$ ($\Delta\mathrm{V_C} = 50$ тис. л/доб.). Більш того, можливі перебої в роботі насосів під час подавання води, тобто швидкість наповнення резервуара розподілена за рівномірним законом із середнім значенням $\mathrm{V_H}$ і середнім відхиленням $\Delta\mathrm{V_H}$ ($\Delta\mathrm{V_H} = 50$ тис. л/доб.).

Розв'язання за умови 1. Оскільки рівень споживання $\mathbf{V_C}$ і швидкість наповнення резервуара $\mathbf{V_H}$ постійні, то час заповнення резервуара буде дорівнювати:

$$T = W/(V_H - V_C). \quad (14.1)$$

Виконаємо завдання з використанням пакета **R**:

```
> # Вводимо відправні данні
> Wr = 1000
> VC = 480
> VH = 500
> # Обчислюємо час, за який буде заповнено резервуар
> T = Wr/(VH - VC)
> T
[1] 50
```

Як видно, заповнення резервуара з незмінним рівнем споживання V_C і швидкості наповнення резервуара V_H відбудеться через 50 діб.

Розв'язання за умови 2. Розв'язок задачі знайдемо шляхом побудови імітаційної моделі.

Позначимо через:

t_i – моменти модельного часу, що змінюється з постійним кроком Δt. Тобто:

$$t_i = t_{i-1} + \Delta t \quad (i = 1,2,\dots), \quad t_0 = 0;$$

V_{Hi} – поточну швидкість наповнення резервуара в моменти t_i;

V_{Ci} – поточний рівень споживання в моменти t_i;

W_i – поточний стан резервуара в моменти t_i.

Тоді:

$$W_i = W_{i-1} + (V_{Hi} - V_{Ci})\,\Delta t. \quad (14.2)$$

Оскільки рівень споживання води та швидкість наповнення резервуара мають імовірнісний характер, то у деякий момент часу t_i поточні

значення V_{Ci}, V_{Hi} можуть бути змодельовані за допомогою генератора псевдовипадкових чисел рівномірного розподілу на відрізку [0,1]:

$$V_{Ci} = V_C - \Delta V_C + 2\,\Delta V_C\, r_i,\ V_{Hi} = V_H - \Delta V_H + 2\,\Delta V_C\, r_i.$$

Імітаційна модель (14.2) є стохастичною. Тому для визначення часу T заповнення резервуара треба виконати кілька прогонів моделі. Один прогін моделі (процес моделювання) закінчується, якщо на деякому кроці i виконується умова $W_i \geq W$. Точність результату залежить від значення Δt. Після кожного прогону моделі отримаємо випадкові значення T_j, де j – номер прогону $(j = 1,\ 2,\ 3,\ \ldots)$.

Виконаємо завдання з використанням пакета R. Один прогін моделі реалізуємо процедурою:

```
# Процедура, що реалізує прогін моделі
ProgonMod = function(dt, Wr, VH, dVH, VC, dVC){
  t = c(100)
  W = c(100)
  i = 1
  t[i] = 0
  W[i] = 0
  while(W[i] < Wr){
          i = i + 1
          t[i] = t[i-1] + dt
          VHi = (VH - dVH) + 2*dVH*runif(1, 0, 1)
          VCi = (VC - dVC) + 2*dVC*runif(1, 0, 1)
          W[i] = W[i-1] + (VHi - VCi)*dt
  }
  return(list(i=i, t=t, W=W))
}
```

Виконаємо спочатку один прогін моделі з $\Delta t = 1$:

```
# Доповнимо відправні данні
dVC = 50
dVH = 10
dt = 1
res = ProgonMod(dt, Wr, VH, dVH, VC, dVC)
res
```

```
i = res$i
t = res$t
W = res$W
```

Отримуємо результат прогону моделі:

```
> res = ProgonMod(dt, Wr, VH, dVH, VC, dVC)
> res
$`i`
[1] 47

$t
 [1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
[26] 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46

$W
 [1]   0.000000 -18.975806  -3.936096  69.583193  77.897756  40.702591
 [7] 101.544423 117.080082 145.568955 159.307428 196.718430 251.218145
[13] 257.621733 314.242469 316.631726 380.475282 421.943336 445.762424
[19] 509.252067 533.527025 543.207877 561.510549 583.165288 576.186728
[25] 595.651407 660.889493 725.822097 737.815695 746.464447 772.242225
[31] 766.043518 753.837476 822.199854 824.954573 814.962660 796.166933
[37] 829.854172 836.990616 903.018277 930.268372 944.297190 950.079052
[43] 950.601771 969.699120 962.762483 965.082154 1027.743126
```

Як видно з результатів одного прогону моделі, заповнення резервуара відбудеться через 47 діб. Графік заповнення резервуара порівняно з детермінованим випадком зображено на рис. 14.1.

```
plot(t, W, type="l", col="red")

F = function(t){
  (VH - VC)*t
}

# Обчислюємо 100 точок на відрізку [0,tmax]
a = 0; b = t[i]; m = 100
h = (b-a)/(m-1)
t1 = c(1:m)
for(i in 1:m){
```

```
  t1[i] = a + h*(i-1)
}
W1 = F(t1)
lines(t1,W1, col="blue")     # додаємо ще лінію
```

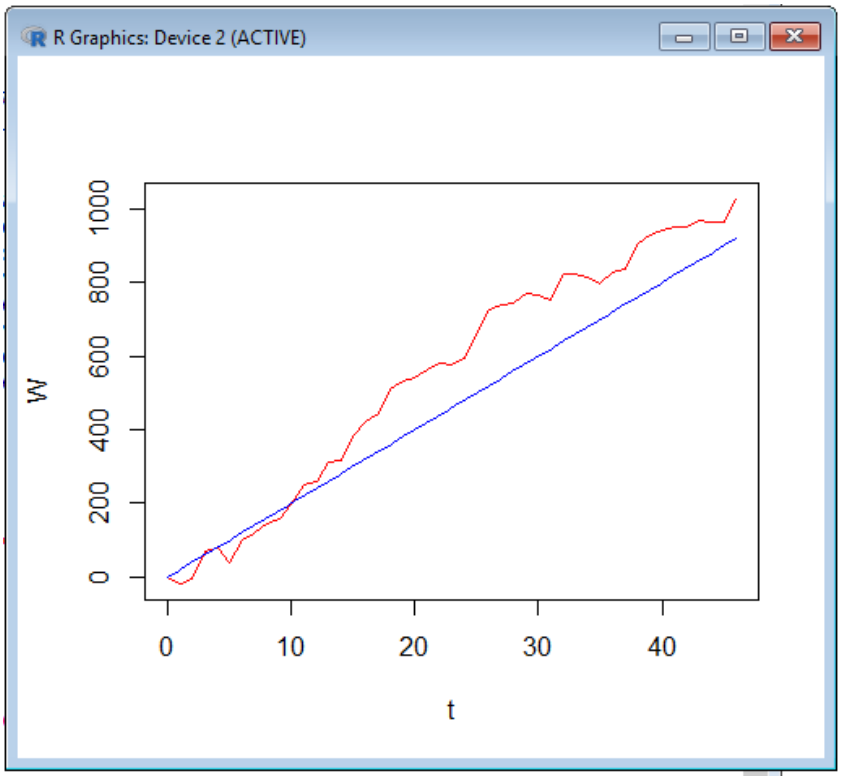

Рис. 14.1. **Графік заповнення резервуара порівняно з детермінованим випадком**

Тепер виконаємо 100 прогонів моделі:

```
N = 100
Tp = c(1:N)
for(j in 1:N){
        res = ProgonMod(dt, Wr, VH, dVH, VC, dVC)
        Tp[j] = res$t[res$i]
}
Tpmean = mean(Tp)
Tpmean
TpDisp = var(Tp)
Tpsd = sqrt(TpDisp)
Tpsd
```

Отримуємо результат прогонів моделі:

```
> Tpmean
[1] 50.39
> TpDisp = var(Tp)
> Tpsd = sqrt(TpDisp)
> Tpsd
[1] 11.52678
```

Як видно з результатів прогонів імітаційної моделі, заповнення резервуара може відбутися за різний час T_j, але у середньому приблизно через $T_{cp} = 50.39$ діб (≈ 50 діб). Водночас вибіркове відхилення $s = 11.52678$.

Знайдемо тепер ΔT (точність оцінювання T), за якого з рівнем довіри $\alpha = 0{,}95$ можна буде гарантувати, що у 95 випадках із 100 час заповнення резервуара буде знаходитись у межах $T_{cp} \pm \Delta T$. Для цього оцінимо закон розподілу для T_j (рис. 14.2). Побудуємо гістограму:

```
hist(Tp, col="red")
```

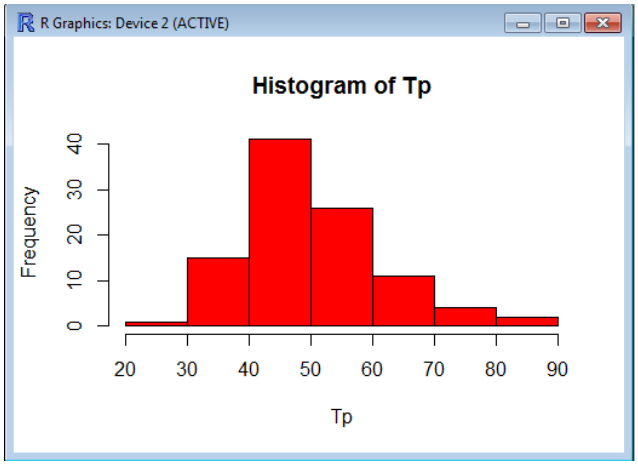

Рис. 14.2. **Гістограма часу заповнення резервуара**

З вигляду гістограми можна зробити висновок, що закон розподілу можливо нормальний із параметрами $\mu = T_{cp}$, $\sigma = s$. Перевіримо цю гіпотезу за критерієм Пірсона з рівнем значущості $\alpha = 0.95$:

```
V = Tp
Vmean = Tpmean
Vsd = Tpsd
```

```
m = 10
h = (max(V)- min(V))/m
int = c(1:(m+1))
for(i in 1:(m+1)){
  int[i] = min(V) + (i-1)*h
}

n = c(1:m)
p = c(1:m)
for(i in 1:m){
  p[i] = pnorm(int[i+1], Vmean, Vsd) - pnorm(int[i], Vmean, Vsd)
  n[i] = 0
  for(j in 1:N){
        if((V[j] >= int[i]) && (V[j] < int[i+1])){
              n[i] = n[i] + 1
        }
  }
}

HiSq = 0
for(i in 1:m){
  HiSq = HiSq + (n[i]-N*p[i])^2/(N*p[i])
}
HiSq

alfa = 0.95
HiSqAlfa = qchisq(alfa, m-1)
HiSqAlfa
```

Отримуємо результат обчислень:

```
> HiSq
[1] 6.285277
>
> alfa = 0.95
> HiSqAlfa = qchisq(alfa, m-1)
> HiSqAlfa
[1] 16.91898
```

Тобто гіпотеза підтверджується. Тому вірогідність того, що час заповнення резервуара буде у межах $[T_{cp} - \Delta T, T_{cp} + \Delta T]$ для деякого ΔT обчислюється функцією:

```
Palfa = function(dT){
  pnorm(Tpmean+dT, Tpmean, Tpsd) - pnorm(Tpmean-dT, Tpmean, Tpsd)
}
```

Знайдемо ΔT з рівняння $P\alpha(\Delta T) = \alpha$ за допомогою процедури ***uniroot*** пакета R:

```
alfa = 0.95
Froot = function(dT){
  Palfa(dT) - alfa
}
Froot(0)
Froot(30)
rc = uniroot(Froot, lower=0, upper=30)
rc$root
```

Отримуємо результат обчислень:

```
> alfa = 0.95
> Froot = function(dT){
+ Palfa(dT) - alfa
+ }
> Froot(0)
[1] -0.95
> Froot(30)
[1] 0.04673725
>
> rc = uniroot(Froot, lower=0, upper=30)
> rc$root
[1] 19.98718
```

Таким чином, $\Delta T \approx 19.99$ діб. Тобто резервуар буде заповнено з вірогідністю 0,95 не раніше $T_{ср} - \Delta T = 30.4$ та не пізніше $T_{ср} + \Delta T = 70.38$ діб.

Приклад 2. Розглянемо деяку дискретну динамічну систему, схема функціонування якої подана на рис. 14.3.

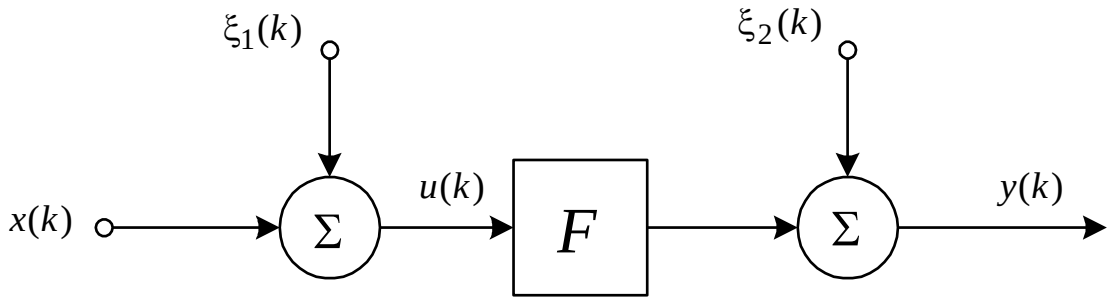

Рис. 14.3. **Схема функціонування дискретної динамічної системи**

Тут k – дискретний час, що приймає послідовні значення з ряду натуральних чисел; $x(k)$ – вхідний сигнал (може бути векторним); $\xi_1(k)$, $\xi_2(k)$ – випадкові сигнали (перешкоди), що діють як складові; $u(k)$ – проміжний сигнал (може бути векторним); $y(k)$ – вихідний сигнал; F – передавальна функція, що зв'язує вхідний та вихідний сигнали.

Характеристики сигналів $x(k)$, $\xi_1(k)$, $\xi_2(k)$, а також вид передавальної функції F, мають такий вид:

$$x(k) = \left[y(k-1),\ y(k-2),\ y(k-3), \frac{\sin(2\pi k)}{250}\right), \sin(2\pi(k-1)/250)],$$
$$\xi_1 = N(0, 0.05), \quad \xi_2 = U(-0.5, 0.5),$$
$$F(k) = (u_1(k)u_2(k)u_3(k)u_5(k)(u_3(k)-1) + u_4(k))/(1 + u_3^2(k) + u_2^2(k)).$$

Дискретний час k приймає значення від 1 до 450.

Необхідно провести чисельне моделювання цієї системи, тобто сформувати вхідні та вихідні вектори сигналів x, y та побудувати їх графіки.

Розв'язання. Проведемо чисельне моделювання функціонування цієї системи, тобто сформуємо вхідні та вихідні вектори сигналів $\mathbf{x}$, $\mathbf{y}$.

Оскільки вхідний сигнал x є векторним розміру 5 (тобто вхідних сигналів на самому ділі 5), то будемо накопичувати їх значення для кожного

дискретного часу k у матриці. Рядки матриці будуть відповідати дискретному часу, стовпці – значенням сигналів. Вихідний сигнал y є скалярним, тому будемо накопичувати його значення для кожного дискретного часу k у векторі. Для моделювання випадкових перешкод для вхідних і вихідного сигналів скористаємося відповідно до умов задачі генераторами для нормального та рівномірного законів розподілу.

Звернемо увагу на те, що першими трьома вхідними сигналами x є вихідний сигнал y із запізненням у часі. Такі системи називають *системами зі зворотним зв'язком*. Тому моделюємо ці вхідні сигнали тільки з моменту, коли вони вже існують.

```
nT = 450
m = 5
x = matrix(0, nrow=nT, ncol=m)
u = x
y = rep(0, nT)
for (k in 1:nT)
{
        x[k,4] = sin(2*pi*k/250)
        x[k,5] = sin(2*pi*(k-1)/250)
        if (k > 1)
                x[k,1] = y[k - 1]
        if (k > 2)
                x[k,2] = y[k - 2]
        if (k > 3)
                x[k,3] = y[k - 3]
        for (i in 1:m)
                u[k,i] = x[k,i] + rnorm(1, 0, 0.05)
        y[k] = (u[k,1]*u[k,2]*u[k,3]*u[k,5]*(u[k,3]-1) + u[k,4])/(1 + u[k,3]*u[k,3] +
+ u[k,2]*u[k,2])
        y[k] = y[k] + runif(1, -0.5, 0.5)
}
```

Побудуємо графік п'ятого вхідного та вихідного сигналів (рис. 14.4).

```
plot(x[,5], type="l", col="blue")
lines(y, type="l", col="red")
```

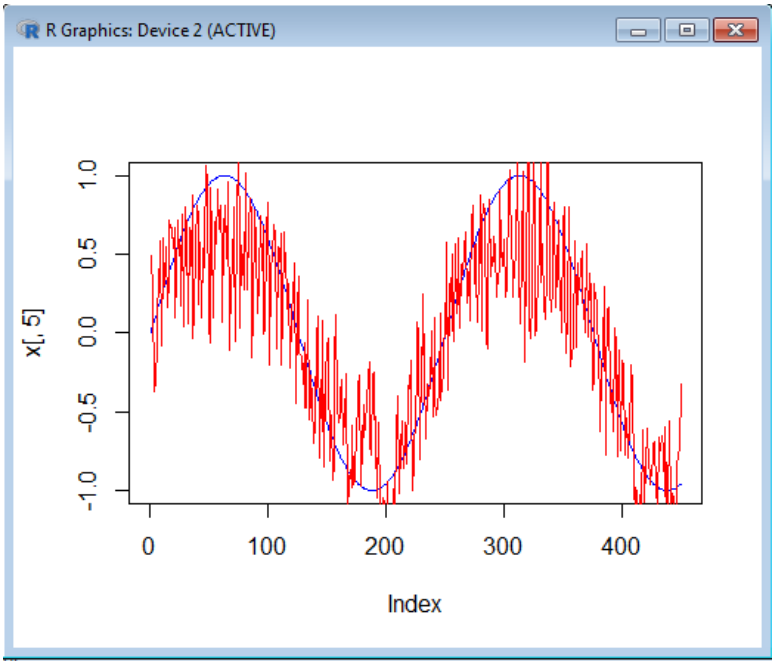

Рис. 14.4. **Графік п'ятого вхідного та вихідного сигналів**

14.4. Порядок виконання роботи та варіанти завдань

14.4.1. Зміст звіту

У практичній частині роботи необхідно:
виконати приклади, що наведені в підрозділі 14.3;
виконати завдання із пункту 14.4.2;
надати тексти складених програм та результати їх виконання.

14.4.2. Варіанти індивідуальних завдань

Варіант 1. Визначте з рівнем довіри 95 %, на скільки часу вистачить бензину на бензоколонці, якщо: запас його дорівнює 5 000 літрів, автомобілі під'їжджають приблизно кожні 30 ± 10 хвилин і заправляються по 15 ± 10 літрів.

Варіант 2. Визначте з рівнем довіри 99 %, на скільки діб вистачить газу на підприємстві, якщо: запас його в газосховищах дорівнює 17 000 м^3, рівень споживання газу – 200 ± 20 м^3 на добу, рівень власного видобутку газу (з відходів) – 60 ± 5 м^3 на добу.

Варіанти 3 – 12 мають такий загальний вид.

Розглянемо деяку дискретну динамічну систему, схема функціонування якої подана на рис. 14.3 зі значеннями:

k – дискретний час, що приймає послідовні значення з ряду натуральних чисел; $x(k)$ – вхідний сигнал (може бути векторним): $\xi_1(k)$, $\xi_2(k)$ – випадкові сигнали (перешкоди), що діють як складові; $u(k)$ – проміжний сигнал (може бути векторним); $y(k)$ – вихідний сигнал; F – передавальна функція, яка сполучає вхідний із вихідним сигналом.

Характеристики сигналів $x(k)$, $\xi_1(k)$, $\xi_2(k)$, а також вид передавальної функції F, вказані в індивідуальних варіантах 3 – 12.

Необхідно провести чисельне моделювання цієї системи, тобто сформувати вхідні та вихідні вектори сигналів x, y та побудувати їх графіки.

Варіант 3. $x(k) = \sin\frac{k}{10} + \frac{30}{k+10}\sin k$, $\xi_1(k) = N(0,0.2)$, $\xi_2(k) = N(0.5;2)$, $F(k) = 0.6u(k) + 0.4u(k-1)$, $k = 1\ldots500$.

Варіант 4. $x(k) = [\sqrt{k/10}; \sin(k/20)]$, $\xi_1(k) = U(-0.35,\ 0.45)$, $\xi_2(k) = 0$, $F(k) = \cos(u_1(k)u_2(k))$, $k = 1\ldots600$.

Варіант 5. $x(k) = [u(k-1), \sin(k)]$, $\xi_1(k) = N(0;0.35)$, $\xi_2(k) = N(0.1;0.2)$, $F(k) = 1/(1 - e^{-u_1(k)u_2(k)})$, $k = 1\ldots500$.

Варіант 6. $x(k) = \sin(2\pi k/250)$, $\xi_1(k) = N(0,0.15)$, $\xi_2(k) = U(-0.1,\ 0.45)$, $F(k) = 0.6\sin(\pi u(k)) + 0.3\sin(3\pi u(k)) + 0.1\sin(5\pi u(k))$, $k = 1\ldots500$.

Варіант 7. $x(k) = [\,y(k-1),\ y(k-2),\ y(k-3),\ \sin(2\pi k/250)$, $\xi_1(k) = N(0,0.05)$, $\xi_2(k) = 0$, $F(k) = \big(u_1(k)u_2(k)u_3(k)u_5(k)(u_3(k)-1) + u_4(k)\big) / / (1 + u_3^2(k) + u_2^2(k))$, $k = 1\ldots1000$.

Варіант 8. $x(k) = \sin(2\pi k/250)$, $\xi_1(k) = U(-0.2,\ 0.2)$, $\xi_2(k) = 0$,

$$F(k) = \begin{cases} 2u(k)/(2-1) + 1/(2-1), u(k) < -0.5 \\ -2 * 0.5/(2-1) + 1/(2-1), -0.5 \le u(k) < -0.5, \\ 2u(k)/(2-1) + 1/(2-1), 0.5 \le u(k) \end{cases} k = 1\ldots500.$$

Фактично, розглядувана функція $F(k)$ є так званою "зоною нечутливості", достатньо неприємною особливістю багатьох об'єктів управління. Загальна формула такої функції має такий вигляд:

$$f(x) = \begin{cases} 2x/(2-L) + L/(2-L), & u < S \\ 2S/(2-L) + L/(2-L), & S \le u < S + L, \\ 2x/(2-L) - L/(2-L), & S + L \le u \end{cases}$$

де S – початок зони нечутливості, а L – її ширина. У тому випадку, коли значення сигналу знаходиться в цій зоні, вихід функції – константа. У даному конкретному випадку $F(k)$ – лінійна функція із зоною нечутливості від −0.5 до 0.5.

Варіант 9. $x(k) = 0.5\,sin(2\pi k/250) + 0.5\,sin(2\pi k/25)$, $\xi_1(k) = U(-0.05, 0.05)$, $\xi_2(k) = N(0.1,0.35)$, $F(k) = 0.6\,sin(\pi u(k))0.3\,sin(3\pi u(k)) + 0.1\,sin(5\pi u(k))$, $k = 1\ldots750$.

Варіант 10. $\mathrm{x(k) = [lg(k), arctg(k)]}$, $\xi_1(\mathrm{k}) = \mathrm{N}(0.1,0.25)$, $\xi_2(\mathrm{k}) = 0$, $\mathrm{F(k) = cos(u_1(k) + u_2(k))}$, $\mathrm{k} = 1\ldots500$.

Варіант 11. $\mathrm{x(k) = [y(k-1), y(k-2), y(k-3), sin(2\pi k/250), sin(2\pi(k-1)/250)]}$, $\xi_1(\mathrm{k}) = 0$, $\xi_2(\mathrm{k}) = \mathrm{U}(-0.1,\ 0.1)$, $\mathrm{F(k) = (u_1(k)u_2(k)u_3(k)u_5(k)(u_3(k)-1) + u_4(k))/(1 + u_3^2(k) + u_2^2(k))}$, $\mathrm{k} = 1\ldots1000$.

Варіант 12. $\mathrm{x(k) = [sin(2\pi k/30), sin(2\pi k/40)]}$, $\xi_1(\mathrm{k}) = \mathrm{N}(0,0.2)$, $\xi_2(\mathrm{k}) = \mathrm{U}(-0.01,\ 0.01)$, $\mathrm{F(k) = Exp[-(u(k) - [1\ 1])(u(k) - [1\ 1])'/200)]}$, $\mathrm{k} = 1\ldots500$.

14.5. Контрольні запитання

1. Яку модель називають динамічною?
2. Яку модель називають стохастичною?
3. Які закони розподілу випадкових величин ви знаєте?
4. Як згенерувати значення випадкової величини на комп'ютері?

Лабораторна робота 15.
Дослідження моделей, заснованих на диференціальних рівняннях (D-схемах)

15.1. Мета роботи

Вивчення способів проведення експериментів з динамічними моделями, що описуються диференціальними рівняннями; набуття навичок розв'язання диференціальних рівнянь із застосуванням математичних пакетів.

15.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати*, як записується динамічна модель за допомогою звичайних диференціальних рівнянь [5]; *уміти* розв'язувати задачу Коші для звичайних диференціальних рівнянь із застосуванням процедур математичного пакета **R**.

Динамічна модель, у загальному випадку, з декількома виходами $y_1 = y_1(t), \dots, y_k = y_k(t)$ описується за допомогою системи звичайних диференціальних рівнянь першого порядку (нормальна форма):

$$\begin{aligned} y_1' &= f_1(t, y_1, y_2, \dots, y_k), \\ y_2' &= f_2(t, y_1, y_2, \dots, y_k), \\ &\dots\dots\dots\dots\dots\dots\dots\dots \\ y_k' &= f_k(t, y_1, y_2, \dots, y_k). \end{aligned} \qquad (15.1)$$

Якщо відомі значення $y = (y_1, y_2, \dots, y_k)^T$ у деякий момент часу t_0, $y(t_0) = y^{(0)}$, то в результаті проведення експерименту з динамічною моделлю (15.1) буде знайдена траєкторія поводження вихідних змінних $y = y(t)$ на заданому інтервалі часу $[t_0, t_1]$. Зазначимо, що це є задачею Коші для системи звичайних диференціальних рівнянь (15.1).

15.3. Контрольні приклади

Приклад 1. Проведіть експеримент із динамічною моделлю, що описується диференціальним рівнянням:

$$y' = \sin(t) + \frac{1}{y} \qquad (15.2)$$

на інтервалі [0, 8π] за умови, що в початковий момент часу $t = 0$ $y(0) = 1$.

Розв'язання (у середовищі R). Динамічна модель (15.2) описується одним рівнянням і має одну вихідну змінну $y \in R^1$. Визначимо параметри моделі, виходячи із загального виду (15.1), тобто функцію правої частини диференціального рівняння, та відправні дані експерименту за моделлю:

```
# Залаємо функцію правої частини диференціального рівняння (моделі)
FunRight = function(t, y, parms) {
    dy = c(1)
    dy[1] = sin(t) + 1/y[1]
    return(list(dy))
}
# Залаємо відправні дані експерименту з моделлю
t0 = 0      # початкове значення часу
t1 = 8*pi   # кінцеве значення часу
y0 = 1      # початкове значення змінної y
```

Для розв'язання диференціального рівняння (15.2) скористаємося процедурою *rk4* пакету ***deSolve***, що призначена для розв'язання задачі Коші (15.1) методом Рунге – Кутта четвертого порядку (див. методичні вказівки до лабораторної роботи 5). Задамо параметри процедури *rk4*:

```
library(deSolve)
n = 100       # кількість поділень часового відрізка
h = (t1 - t0)/n          # шаг поділення часового відрізка
time = seq(t0, t1, h) # сітка на відрізку
parms = c(1:1)        # невикористовувані параметри моделі
# rk4 - метод Рунге-Кутта
out = rk4(y0, time, FunRight, parms)
out
```

Для побудови графіка визначимо масиви: T – значення часу t на інтервалі [0, 8π] і Y – відповідні модельовані значення вихідної змінної y.

```
T = out[, 1]
Y = out[, 2]
plot(T, Y, type = "l", col = "red")
```

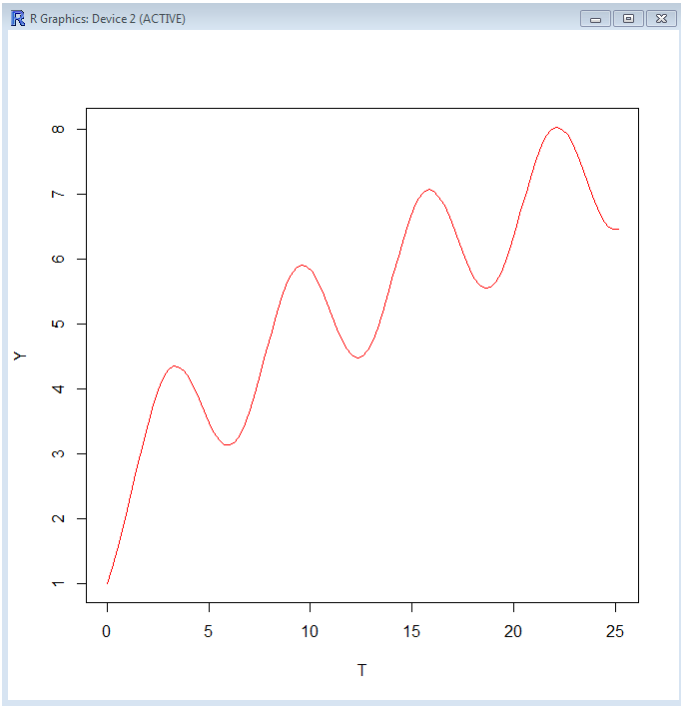

Рис. 15.1. **Графік поведінки змінної моделі *y* в часі**

Побудований графік (рис. 15.1) дозволяє побачити поведінку змінної y в часі.

Приклад 2. Припустимо, що автомобіль почав рух від пункту А; його швидкість у будь-який момент t була у півтора рази більша за середню швидкість, відносно попереднього руху. На яку відстань від'їде автомобіль від пункту А через 4 години, якщо відомо, що через 15 хвилин він вже був на відстані 19 км?

Розв'язання. Позначимо через $\mathrm{y} = \mathrm{y}(\mathrm{t})$ відстань від пункту А у момент t з початку руху. Тоді y' – швидкість автомобіля у момент t (миттєва швидкість); $\mathrm{y}(\mathrm{t})/\mathrm{t}$ – середня швидкість, відносно попереднього руху. Оскільки швидкість автомобіля у будь-який момент t була у півтора рази більша за середню швидкість, то справедливе відношення:

$$\mathrm{y}' = \frac{1.5\mathrm{y}}{\mathrm{t}}. \qquad (15.3)$$

Тобто записано диференціальне рівняння, що описує рух автомобіля. Відомо, що через 15 хвилин він вже був на відстані 19 км. Це є початковою умовою: $\mathrm{y}(0,25) = 19$ (15 хвилин – це 0,25 години). Треба знайти

відстань, на яку від'їде автомобіль від пункту А через 4 години, тому інтервал $[t_0, t_1]$ = [0.25; 4].

Реалізуємо розв'язання задачі у середовищі R:

```
# Залаємо функцію правої частини диференціального рівняння (моделі)
FunRight = function(t, y, parms) {
      dy = c(1)
      dy[1] = 1.5*y[1]/t
      return(list(dy))
}
# Задаємо відправні дані експерименту з моделлю
t0 = 0.25    # початкове значення часу
t1 = 4       # кінцеве значення часу
y0 = 19      # початкове значення змінної y

library(deSolve)
n = 100      # кількість поділень часового відрізка
h = (t1 - t0)/n          # шаг поділення часового відрізка
time = seq(t0, t1, h) # сітка на відрізку
parms = c(1:1)       # невикористовувані параметри моделі
# rk4 - метод Рунге-Кутта
out = rk4(y0, time, FunRight, parms)
Y = out[, 2]
Y[n+1]
```

Отримуємо результат роботи програми:

```
> Y[n+1]
[1] 1215.993
```

Розв'язок: автомобіль за 4 години від'їде від пункту А на 1 216 км.

15.4. Порядок виконання роботи та варіанти завдань

15.4.1. Зміст звіту

У практичній частині роботи необхідно:
виконати приклади, що наведені в підрозділі 15.2;
виконати індивідуальне завдання з пункту 15.4.2;
надати тексти складених програм та результати їх виконання.

15.4.2. Варіанти індивідуальних завдань

Проведіть експеримент із динамічною моделлю, що описується диференціальним рівнянням ($y \in R^1$):

$$y' = f(t, y)$$

на інтервалі $[t_0, t_1]$ за умови, що в початковий момент часу t_0 $y(t_0) = y_0$. Побудуйте графік поводження вихідної змінної y на інтервалі $[t_0, t_1]$.

Таблиця 15.1

Варіанти завдань

Варіант	Рівняння	Інтервал	Умова
1	$y' = t^2 + y^2$	$[0,1]$	$y(0) = 0$
2	$y' = 1 + ty^2$	$[0,1]$	$y(0) = 0$
3	$y' = \frac{y}{t+1} - y^2$	$[0,1]$	$y(0) = 1$
4	$y' = \frac{y}{t}$	$[1,25]$	$y(1) = 1$
5	$y' = \frac{tx^2 - y^2}{ty}$	$[2,25]$	$y(2) = 1$
6	$y' = \frac{1}{2}ty$	$[0,1]$	$y(0) = 1$
7	$y' = \frac{ty}{t^2 - y^2}$	$[2,25]$	$y(2) = 1$
8	$y' = \frac{t + y - 3}{t - y - 1}$	$[2,25]$	$y(2) = 1$
9	$y' = \frac{2ty - 1}{4t + 2y + 5}$	$[2,25]$	$y(2) = 1$
10	$y' = \frac{3t - y + 2}{2t - 3y + 1}$	$[3,30]$	$y(3) = 5$
11	$y' = \frac{ty^3}{t^2 + y^2}$	$[5,25]$	$y(5) = 7$
12	$y' = \frac{y^4 + y}{t^2}$	$[2,34]$	$y(2) = 10$
13	$y' = \frac{2y + y^2}{t}$	$[2,22]$	$y(2) = 8$
14	$y' = \frac{3y^2 - y}{2 + t}$	$[3,33]$	$y(3) = 7$
15	$y' = \frac{5y + y^2}{22 - t}$	$[1,21]$	$y(1) = 18$

15.5. Контрольні запитання

1. Яку модель називають динамічною?
2. Як записується динамічна модель за допомогою диференціальних рівнянь?
3. До чого зводиться проведення експерименту із динамічною моделлю, що описується диференціальними рівняннями?
4. У чому полягає задача Коші для системи звичайних диференціальних рівнянь?

Лабораторна робота 16. Середовище імітаційного моделювання *GPSS World.* Моделювання найпростіших та багатоканальних систем масового обслуговування

16.1. Мета роботи

Навчитися моделювати роботу найпростіших і багатоканальних систем масового обслуговування; забезпечувати пріоритетне обслуговування та зміну дисципліни обслуговування, використовуючи інструментарій системи GPSS World [10; 14].

16.2. Методичні рекомендації щодо організації самостійної роботи

16.2.1. Моделювання найпростіших систем масового обслуговування

Для моделювання найпростіших систем масового обслуговування використовуються такі блоки.

GENERATE A, B, C, D, E. Блок GENERATE вводить транзакти в модель. Операнди:

A – середнє значення інтервалу часу, через який кожний наступний транзакт надходить до моделі;

B – поле допуску для значення операнда A;

C – задає час появи в моделі першого транзакта;

D – задає максимальне число транзактів, яке має бути згенероване в блоці GENERATE;

E – визначає пріоритет транзактів.

TERMINATE A. Блок TERMINATE видаляє транзакти з моделі. Операнд:

A – число, що віднімається з лічильника завершень.

SEIZE A

Блок SEIZE моделює зайняття пристрою. Операнд:

A – ім'я або номер пристрою.

RELEASE A

Блок RELEASE призначений для вивільнення пристрою тим транзактом, яким він був зайнятий. Операнд:

A – ім'я або номер пристрою.

ADVANCE A, B. Блок ADVANCE реалізує затримку транзакта на час обслуговування. Операнди:

A – середній час перебування транзакта в блоці;

B – поле допуску для операнда A.

Для збирання статистики по черзі використовуються блоки QUEUE (став в чергу) і DEPART (покинув чергу).

QUEUE A, B. Блок QUEUE реєструє, що транзакт став у чергу.

DEPART A, B. Блок DEPART реєструє, що транзакт покинув чергу. Операнди:

A – ім'я або номер черги;

B – число одиниць, на яке збільшується (зменшується) довжина черги.

Пріоритет назначається транзактам під час генерації (операнд E блока GENERATE) або блоком PRIORITY у будь-якому місці моделі.

PRIORITY A, B. Блок встановлює пріоритет A активному транзакту. Якщо в полі B записано BUFFER, то призначений пріоритет діє негайно.

Переривання та подальше відновлення початого обслуговування організується блоками PREEMPT і RETURN. Заявка, яка є причиною переривання, сама не може бути перерваною.

PREEMPT A, B, C, D, E. Операнди:

A – ім'я або номер пристрою, що переривається;

B – режим зайняття пристрою. Операнд (необов'язковий) має бути PR або Null;

C – блок нового призначення транзакта, який у цей час володіє пристроєм;

D – номер параметра перерваного транзакта, який запам'ятовує недоданий час обслуговування, якщо перерваний транзакт видаляється зі списку майбутніх подій;

E – при значенні RE виключає перерваний транзакт з боротьби за пристрій (тоді операнд C необов'язковий).

RETURN A. Блок RETURN звільняє пристрій A або видаляє перерваний транзакт з боротьби за пристрій.

Пристрої, окрім ступеня зайнятості, характеризуються поточною готовністю. Режим готовності пристрою вводиться блоком FAVAIL, режим неготовності – блоком FUNAVAIL. Ця властивість дозволяє моделювати поломки техніки, регламентні роботи, перерви у касирів на здачу грошей, у продавців – на прийом товару, роботу з перервами на обід, тижневий і річний ритми. Часи поломок генеруються аналогічно моментам прибуття, інтервали між ними зазвичай підпорядковані розподілу Вейбулла.

FAVAIL A Блок установлює вказаний операндом A пристрій у стан готовності.

FUNAVAIL A, B, C, D, E, F, G, H. Блок переводить пристрій у стан неготовності для вхідних транзактів. Операнди:

A – ім'я або номер пристрою;

B – розпорядження подальших дій для транзакта, що займав пристрій. Операнд необов'язковий, може бути RE, CO (REmove – видалити або COntinue – продовжити) або Null;

C – наступний блок для транзакта, що займав пристрій;

D – номер параметра для реєстрації часу, що залишився до закінчення обслуговування, якщо транзакт, що займав пристрій, видаляється зі списку майбутніх подій;

E – розпорядження подальших дій для транзактів, обслуговування яких було перерване до виконання блока FUNAVAIL. Операнд має бути RE, CO або Null;

F – новий блок для транзактів, виконання яких було перерване раніше;

G – подальші дії для транзактів, що очікували можливості зайняти пристрій до виконання блока FUNAVAIL. Операнд має бути RE, CO або Null;

H – новий блок, до якого перейдуть транзакти, що очікували можливості зайняти пристрій до виконання блока FUNAVAIL.

Транзакти, що надійшли в період неготовності, будуть затримані та не допущені до пристрою. Якщо в операндах записана опція RE, транзакти перестають претендувати на зайняття пристрою. Якщо в полі G для транзактів, що очікували можливості зайняти пристрій, вказано RE, то для подальшого переспрямування транзактів треба заповнити поле H.

Якщо використовується опція CO, то транзакти, що займали пристрій, можуть продовжувати його займати навіть у період неготовності. У статистичних даних з використання пристрою цей час враховуватиметься.

Якщо вказується блок нового призначення, транзакти звільняються від раніше встановлених зв'язків і прямують до нового блока. Затримані та перервані транзакти, керовані операндами G і H, не можуть бути переспрямовані без опції RE. Транзакт, що зайняв пристрій і є керованим операндами B – D, а також перервані транзакти, керовані операндами E та F, можуть знову претендувати на заняття пристрою, навіть будучи скеровані за новим призначенням. Це здійснюється вказівкою зміненого призначення без використання опції RE.

Операнди B – D використовуються для управління транзактом, що володів пристроєм. Якщо B=CO, цей транзакт не позбавляється володіння пристроєм, але може отримати нове призначення через операнд C. Якщо B=RE він видаляється і може продовжувати шлях у моделі, не заходячи в RETURN або RELEASE для даного пристрою. Тут операнд C обов'язковий.

16.2.2. Моделювання багатоканальних систем масового обслуговування

Часто в реальних системах обслуговувальні пристрої працюють паралельно, опрацьовуючи деяку кількість вимог, що надходять у систему. Такі пристрої характеризуються однаковою інтенсивністю обслуговування. Наприклад, автоматична телефонна станція, автобус, ПК, що працює в мультипрограмному режимі, тощо. Такі обслуговувальні пристрої в GPSS називають *багатоканальними пристроями* (БКП).

Оператор опису БКП має вигляд: **<ім'я БКП> STORAGE A**. Операнд A визначає місткість БКП, під якою розуміють кількість обслуговувальних каналів у БКП або кількість пристроїв, об'єднаних у БКП.

Для моделювання багатоканальних систем використовуються такі блоки ENTER (увійти в БКП) і LEAVE (вийти з БКП).

ENTER A, B. Операнди:

A – ім'я або номер БКП;

B – обсяг запиту транзакта (число одиниць, на яке зменшується місткість доступного БКП).

LEAVE A, B. Операнди:

A – ім'я або номер БКП;

B – кількість пристроїв, що вивільняється.

БКП, окрім ступеня зайнятості, характеризується поточною готовністю. На відміну від сукупності пристроїв, БКП може виявитися неготовим лише в цілому. Режим готовності БКП вводиться блоком SAVAIL, режим неготовності – блоком SUNAVAIL.

Недоступність тільки забороняє вхід транзактів у БКП, їх обробка продовжується. Здійснити імітацію виходу БКП з ладу, за якого всі транзакти, що перебувають в БКП на обслуговуванні, втрачаються, блоками SUNAVAIL і SAVAIL не можна.

SAVAIL A. Блок SAVAIL установлює БКП A в доступний стан. Якщо які-небудь транзакти очікують у списку затримки для цього БКП, вони отримують можливість увійти до БКП згідно з уже обговореним правилом. Ті транзакти, запити яких не можуть бути задовільнені, залишаються в списку затримки.

SUNAVAIL A. Блок SUNAVAIL переводить БКП A в стан неготовності. Усі транзакти, що претендують на вхід у БКП, поміщатимуться в список затримки для цього БКП.

Усі зміни враховуються в статистиці, що збирається про функціонування БКП.

16.2.3. Використання функцій у моделях СМО

Генерація випадкових чисел. GPSS/W має датчики рівномірно розподілених псевдовипадкових чисел {RNj}, які використовуються в блоках GENERATE для генерації інтервалів між транзактами, в ADVANCE – для формування затримок у пристроях і БКП, у блоці TRANSFER (статистичний режим) – для випадкового вибору напряму подальшого руху. Розподіл датчиків випадкових чисел (ДВЧ) за згаданими типами блоків задається в меню Edit – Settings на вкладці Random Numbers.

Потоки рівномірно розподілених псевдовипадкових чисел генеруються за допомогою мультиплікативного генератора з довжиною періоду $2^{31} – 2$ (нульове значення виключається). Початкове значення за замовчуванням збігається з номером відповідного ДВЧ.

Команда **RMULT A, B, C, D, E, F, G** дозволяє задати нестандартні початкові значення перших семи ДВЧ. Початкові установки перших семи ДВЧ за допомогою команди RMULT можна задавати довільно, що дозволяє управляти парами потоків випадкових чисел через підбір початкових установок. Якщо для одного генератора випадкових чисел вибрана установка M, то у виборі множника для іншого генератора $N = 2^{31} – M = 2147483648 – M$. Останній формуватиме випадкові числа, додаткові до чисел першого ДВЧ.

СЧА класу RN повертають цілі випадкові числа в інтервалі [0, 999]. За необхідності їх можна привести до будь-якого іншого цілочисельного діапазону, визначивши відповідну змінну за допомогою операції віднімання за модулем. Наприклад, конструкція RN3@24+40 дасть випадкові цілі числа з інтервалу [40, 63].

У GPSS є сім датчиків рівномірно розподілених псевдовипадкових чисел, що позначаються RN1 – RN7. Якщо не використовується оператор RMULT, то всі датчики генерують однакові послідовності чисел. Оператор RMULT дозволяє отримувати різні послідовності. Наявність семи незалежних датчиків допомагає відтворювати певні послідовності значень у декількох прогонах. Наприклад, для порівняння декількох варіантів модельованої системи в одному вхідному потоці.

Примітка. Оператори CLEAR і RESET не змінюють значень індексів і множників датчиків.

16.2.3.1. Табличні функції

Табличну функцію GPSS/W визначає оператор FUNCTION:

<ім'я функції> FUNCTION A, B. Ім'я функції записується в поле мітки оператора опису FUNCTION. Операнд A задає аргумент функції. 0якої іншої функції. Якщо як аргумент функції використовується випадкове число RNj, то значеннями аргумента будуть числа, рівномірно розподілені в інтервалі 0<=RNj<1. Операнд B задає тип функції і кількість координат (пара точок аргумент-функція X[i] й Y[i]).

Розрізняють **п'ять типів функцій**:

C – неперервна числова функція. Значення функції визначають методом лінійної інтерполяції, беручи як ціле число;

D – дискретна числова функція. Значення функції береться дорівненим значенню на правому кінці інтервалу як ціле число;

E – розширення поняття дискретної функції; у ній можна використати будь-який СЧА, крім матриць;

L – спискова числова функція, аргументом якої є відрізок натурального ряду, починаючи з 1;

M – розширення поняття спискової функції; значенням функції може бути будь-який СЧА.

За кожним оператором опису FUNCTION ідуть оператори для задання координат (X[i] й Y[i]) функції. Не допускається використання коментарів між оператором опису FUNCTION і операторами, що задають значення функції.

Під час написання операторів, що задають значення координат функції, необхідно дотримуватися таких правил:

запис має починатися в позиції 1;

значення координат X[i] і Y[i] однієї точки функції розділяють комою;

набори координат розділяють знаком "/";

координати X[i] і Y[i], що належать до однієї точки, задаються одним оператором;

кожне наступне значення X[i] має бути більше попереднього;

кожна функція повинна мати принаймні дві описані точки.

Звернення до функції можна записувати в операнді B блоків ADVANCE або GENERATE. У цьому випадку її називають "функцією-модифікатором". Час затримки в цих блоках обчислюється як добуток операнда A на функцію-модифікатор.

Неперервні числові функції (Cn). Коли значення аргумента неперервної числової функції попадає в інтервал між двома заданими значеннями (X[i],X[i+1]), програма виконує лінійну інтерполяцію для визначення значення функції FN, що знаходиться в інтервалі між (Y[i],Y[i+1]). Результат береться як ціле число (десяткові знаки відкидаються).

Дискретні числові функції (Dn). Дискретні числові функції задають те саме значення функції FN=Y[i] для всіх значень аргумента X[i–1]<X<=X[i]. Інтерполяція не проводиться, значення функції береться рівним значенню в правому кінці інтервалу. Нецілі значення функцій приводяться до цілих шляхом виділення цілої частини.

Спискові числові функції (Ln). У багатьох випадках значеннями аргумента X[i] є неперервні послідовності цілих чисел 1,2,3...n.

Машинний час, необхідний для обчислення значень таких функцій, можна значно зменшити, якщо вони описані як табличні. У цьому випадку значення аргумента X у операторах, що задають значення координат функції, програмою введення GPSS не розглядаються і приймаються рівними X[1]=1, X[2]=2,..., X[i]=i,..., X[n]=n.

Інтерпретатор використовує аргумент функції для прямого звертання до масиву заданих величин функції. Таким чином, не потрібен послідовний перегляд таблиці, пов'язаної з функцією.

Значення Y[i] мають бути записані у відповідних полях операторів, що задають значення координат. Для зручності програміста в операторах можуть бути записані значення X, хоча вони ніколи не переглядаються. Якщо значення аргумента виходить за межі інтервалу (1,n), то видається повідомлення про помилку на етапі виконання.

Дискретні (En) і спискові (Mn) атрибутивні функції. Дискретні атрибутивні функції (En) подібні дискретним числовим функціям (Dn); спискові атрибутивні функції (Mn) подібні списковим числовим функціям (Ln).

Як і у випадку дискретних, неперервних і табличних функцій, першим елементом упорядкованої пари точок, використовуваної для задання такої функції, є константа. Другим елементом кожної впорядкованої пари є СЧА. Це означає, що кожне значення функції задається шляхом визначення функції.

16.2.3.3. Оператори опису деяких імовірнісних розподілів

Неперервні розподіли.

Рівномірний розподіл:

$$\text{Real} = \text{UNIFORM}(\text{RNj},a,b).$$

Характеристики:

щільність розподілу на інтервалі (a,b) : $f(x) = \frac{1}{b-a}$;

середнє значення: $\nu = \frac{a+b}{2}$;

дисперсія: $D = \frac{(b-a)^2}{12}$.

Експонентний розподіл:

$$\text{Real} = \text{EXPONENTIAL}(\text{RNj},\text{m},\text{s}),$$

де m – зсув розподілу;

s – масштабний параметр.

Характеристики:

щільність розподілу: $\text{f}(\text{x}) = \frac{1}{\text{s}} \exp\left(-\frac{\text{x}-\text{m}}{\text{s}}\right)$, $\text{x} \geq \text{m}$;

середнє значення: $\nu = \text{m} + \text{s}$;

дисперсія: $\text{D} = \text{s}^2$.

Нормальний розподіл:

$$\text{Real} = \text{NORMAL}(\text{RNj},\text{m},\text{s}).$$

Характеристики:

щільність розподілу: $\text{f}(\text{x}) = \frac{1}{\text{s}\sqrt{2\pi}} \exp\left(-\frac{(\text{x}-\text{m})^2}{2\text{s}^2}\right)$;

середнє значення: $\nu = \text{m}$;

дисперсія: $\text{D} = \text{s}^2$.

Дискретні розподіли.

Дискретний рівномірний розподіл:

$$\mathbf{Integer} = \mathbf{DUNIFORM}(\mathbf{RNj}, \mathbf{min}, \mathbf{max}).$$

Характеристики:

імовірність: $\text{p}(\text{x}) = \frac{1}{\max - \min + 1}$, $\text{x} \in \{\min, \min + 1, \ldots, \max\{\}\}$;

середнє значення: $\nu = \frac{\min + \max}{2}$;

дисперсія: $\text{D} = \frac{(\max - \min + 1)^2 - 1}{12}$.

Розподіл Пуассона:

$$\mathbf{Integer} = \mathbf{POISSON}(\mathbf{RNj}, \boldsymbol{\nu}),$$

де ν – середнє значення.

Характеристики:

імовірність: $p(x) = \frac{\nu^k}{k!} e^{-\nu}$, $k = 0,1, \ldots$;

дисперсія: $D = \nu^2$.

16.2.4. Використання змінних, виразів, збережуваних величин і матриць у моделюванні СМО

16.2.4.1. Змінні та вирази.

Алфавіт мови GPSS складається з латинських букв, цифр і спеціальних символів. Великі та малі букви розрізняються тільки в рядкових константах і коментарях. Рекомендується службові слова набирати великими літерами, а індивідуальні імена об'єктів – малими, починаючи із великої.

У записі виразів використовують такі знаки операцій (у порядку убування пріоритету):

^ – піднесення до степеня;
# – множення;
/ – ділення;
\ – ділення націло;
@ – цілочисельний залишок;
+,– – додавання й віднімання;
>=, 'GE' – більше або дорівнює;
<=, 'LE' – менше або дорівнює;
>, 'G' – більше;
<, 'L' – менше;
=, 'E' – дорівнює;
!=, 'NE' – не дорівнює;
&, 'AND' – логічне множення;
|, 'OR' – логічне додавання;
'NOT' – заперечення.

З математичних функцій дозволені ABS, ATN (арктангенс), COS, SIN, TAN, EXP, LOG (натуральний), SQR (корінь), INT (виділення цілого числа). Аргументи математичних функцій повинні бути взяті в дужки. Усі кути задаються в радіанній мірі.

Обчислювальні вирази є комбінацією математичних операторів, бібліотечних функцій, СЧА і констант, що задовольняють правилам елементарної алгебри. Вони обчислюються згідно з ієрархією розглянутих операторів у напрямку зліва направо. Порядок обчислень можна змінити за допомогою дужок.

Арифметичні змінні дозволяють обчислювати арифметичні вирази, що складаються із стандартних числових атрибутів (СЧА). У виразі змінної використовують оператори, арифметичні дії і виклики бібліотечних функцій.

Команда VARIABLE задає арифметичну змінну (правило для отримання числового значення):

<ім'я> VARIABLE <вираз>

Арифметичні змінні з плаваючою комою аналогічні розглянутим вище арифметичним змінним, за винятком того, що всі операції над операндами виразів змінних з плаваючою комою виконуються без перетворення операндів і проміжних результатів до цілого виду. Лише остаточний результат обчислення перетвориться до цілого числа.

Для змінних із плаваючою комою недопустима операція ділення за модулем. Тільки для опису змінних з плаваючою комою допускається застосування дробових констант. Стандартний числовий атрибут V використовується для звернення як до арифметичних змінних, так і до змінних із плаваючою комою. Спосіб обчислення змінної V визначається оператором опису цією змінної.

Команда FVARIABLE задає арифметичну змінну з плаваючою комою:

<ім'я> FVARIABLE <вираз>

Булеві змінні дають можливість приймати рішення залежно від стану та значення багатьох об'єктів GPSS, використовуючи для цього лише один блок. Булевими змінними є логічні вирази, складені з різних стандартних числових атрибутів, у тому числі й інших булевих змінних.

У булевій змінній перевіряється одна або декілька логічних умов. Результатом перевірки є 1, якщо задані умови задовольняються, і 0, якщо вони не задовольняються.

У булевих змінних допускаються три типи операторів: логічні, булеві й оператори відношення. Логічні оператори пов'язані з об'єктами пристроїв і використовуються для визначення стану цих об'єктів. Оператори відношення проводять алгебраїчні порівняння операндів. Операндами можуть бути цілі константи або стандартні числові атрибути. Усі оператори відношення записуються в лапках. Булевих оператора два: оператор 'OR' (''або'') і оператор 'AND' (''і'').

Команда BVARIABLE задає булеву змінну:

<ім'я> BVARIABLE <вираз>

З посиланням на змінну вираз обчислюється за поточних значень параметрів активного транзакта, збережуваних значень тощо.

Команда EQU створює іменований об'єкт (змінну користувача) і привласнює йому обумовлене виразом числове значення:

<ім'я> EQU <вираз>

16.2.4.2. Збережувані величини і матриці

Транзакти не можуть безпосередньо посилатися один на одного. Їхнє спілкування реалізується через збережувані величини. Збережувані величини можуть бути скалярними та матричними.

SAVEVALUE A, B. Блок змінює значення комірки А. Операнд А може супроводжуватися знаком "+" або "−" для вказівки додавання або віднімання від існуючого значення. Операнд В вказує значення, що повинне бути занесене, додане або відняте.

ASSIGN A, B, C. Блок використовується для задавання або зміни значення параметра транзакта. Операнди:

А – номер параметра активного транзакта;

В – привласнюване значення;

С – номер функції (не обов'язковий).

PLUS A, B. Блок оцінює вираз А і поміщає результат в операнд В.

MARK A. Блок зчитує абсолютний час із системного годинника у параметр А активного транзакта.

INDEX A, B. Блок додає до параметра А активного транзакта числове значення, що зазначене у В. Результат поміщається в перший параметр активного транзакта.

SELECT O A, B, C, D, E, F. Блок вибирає об'єкт за умовою та поміщає його номер у параметр активного транзакта. Операнди:

О – логічний оператор або показчик відношення;

А – операнд для запису номера обраного елемента;

В – нижня межа діапазону пошуку;

С – верхня межа діапазону пошуку;

D – опорне значення (база порівняння) з операндом Е;

Е – ім'я класу СЧА, обов'язковий тільки для умовної форми;

F – номер блока призначення, якщо не відібраний жоден об'єкт.

COUNT O A, B, C, D, E. Блок заносить число об'єктів у параметр активного транзакта. Операнди:

О – логічний оператор або покажчик відношення;

А – операнд для прийняття результату;

B – нижня межа діапазону об'єктів, що перевіряються;

C – верхня межа діапазону;

D – опорне значення для операнда E, обов'язкове тільки за наявності умови;

E – специфікатор класу об'єктів (для умовного режиму перевірки).

LOGIC O A. Блок призначений для вмикання, вимикання або інвертування стану логічного ключа. Якщо логічний оператор O дорівнює S або R, то ключ A встановлюється в стан "увімкнений" (Set) або "вимкнений" (Reset), відповідно. Якщо логічний оператор – I, то стан ключа інвертується. Блок LOGIC у режимі інвертування дозволяє чергувати альтернативи різного роду – маршрути, функції, числові значення тощо.

Матриця – це прямокутний масив чисел. Розмір матриці повинен бути попередньо оголошений у пропозиції MATRIX:

<ім'я> MATRIX ,B,C

Мітка <ім'я> визначає ім'я матриці. Значення поля повинне бути ім'ям або номером.

Операнд A не використовується в даній версії GPSS, оскільки немає необхідності специфікувати точність збережуваних елементів матриці. Він залишений для сумісності з попередніми версіями GPSS.

Наступні операнди (до шести) задають довжину за вимірами. За потребою у векторі останній оголошується як двовимірна матриця з одним рядком (стовпцем).

У поле B задається число рядків матриці, у поле C – число стовпців. Операнди B і C можуть бути додатними цілими.

Індексування елементів за всіма вимірами починається з одиниці. Команда MATRIX повинна передувати першому посиланню на цю матрицю.

Оператор MATRIX створює матрицю в поточній моделі. Матриця не може бути видалена з поточної моделі. Якщо оператор MATRIX видаляється з виконуваної програми, зв'язок із матрицею в поточній моделі залишається. Матриця може бути перевизначена або ініціалізована повторно іншим оператором MATRIX з тією самою міткою. Перевизначення, за якого розмір матриці змінюється, викликає виділення пам'яті під нову матрицю. Проте раніше розподілена оперативна пам'ять залишається зайнятою.

Комірки та матриці використовують для зберігання і подальшого використання значень стандартних числових атрибутів. Початкові значення комірок і матриць можна задати за допомогою оператора опису INITIAL.

INITIAL A, B. Операнди:

A – елемент матриці або збережувана величина;

B – значення, що записується в зазначену комірку.

Оператор INITIAL привласнює елементам матриці початкові значення. Наступні зміни значень здійснюються блоком MSAVEVALUE.

MSAVEVALUE A, B, C, D

Блок змінює значення елемента матриці (довільний доступ можливий тільки за першими двома вимірами, інші індекси беруться одиничними). Операнди:

A – ім'я або номер матриці з необов'язковим знаком "+" або "−";

B, C – номери рядка й стовпця, відповідно;

D – значення, що повинне бути записане, додане або відняте.

16.2.5. Моделювання систем зі зміненою дисципліною обслуговування

Послідовність проходження транзактами блоків у моделі можна змінити за допомогою ряду блоків.

TRANSFER A, B, C, D. Блок змінює траєкторію руху активного транзакта. Операнди:

A – режим блока, обраний з множини {BOTH, ALL, PICK, FN, P, SBR, SIM}, може бути також дробовим числом, ім'ям, константою, СЧА;

B – номер або ім'я блока (у режимі P – ім'я параметра);

C – номер або ім'я блока. У режимі функції (FN) або параметра (P) – збільшення;

D – збільшення номера блока для режиму ALL. За замовчуванням дорівнює 1.

Блок TRANSFER може діяти в одному з дев'яти режимів. За потребою у випадковому виборі номер ДВЧ установлюється на вкладці Random Numbers у меню Edit – Settings….

Режим безумовного переходу. За відсутності операнда A активний транзакт переходить за адресою, специфікованою операндом B.

Статистичний режим. Якщо операнд A не є ключовим словом, то транзакт переходить за адресою, вказаною операндом C, з імовірністю, заданою операндом A. Якщо останній – ціле число, воно інтерпретується як кількість тисячних і перетвориться в дробову ймовірність. Альтернативне призначення специфікується операндом B. Його значення за замовчуванням – блок, наступний за TRANSFER.

Режим BOTH. Тут спочатку перевіряється блок, заданий операндом B. Якщо він не може прийняти активний транзакт, то перевіряється зазначений у C. Якщо обидва блоки відмовляться прийняти транзакт, то він залишиться у блоці TRANSFER, поки не зможе ввійти в один із блоків, що перевіряються.

Режим ALL. Перевіряється послідовність блоків між операндами B і C включно до першого успіху. Адреса кожного наступного блока, що перевіряється, обчислюється додаванням операнда D (за замовчуванням 1) до адреси попереднього блоку. Якщо операнд C не використовується, перевіряється тільки один блок. Якщо жоден із блоків не зможе прийняти транзакт, він залишається в поточному блоці.

Режим PICK вибирає подальше призначення транзакта в інтервалі від B до C (послідовні номери) з рівними ймовірностями незалежно від можливого блокування.

Режим функції діє, коли операнд A є FN. Нове призначення вибирається обчисленням заданої операндом B цілочисельної функції, яка потім додається до необов'язкового зсуву, заданого операндом C.

Режим параметра задає операнд A = P. Активний транзакт переходить за адресою, що дорівнює сумі значення параметра й операнда C. Ім'я або номер параметра зазначені в операнді B.

Режим підпрограми реалізується, якщо A = SBR. У цьому випадку активний транзакт завжди переходить за адресою, заданою операндом B. Адреса поточного блока поміщається в параметр транзакта, зазначений операндом C. Ця технологія дозволяє організувати перехід транзакта з поверненням, аналогічний переданню управління на підпрограму з поверненням на продовження основної програми. Повернення забезпечується блоком TRANSFER у параметричній моді.

У **режимі SIM** перехід залежить від "індикатора затримки" транзакта, безпосередньо не доступного програмістові. Цей індикатор вмикається, коли транзакт отримує відмову на вхід у який-небудь блок, і може бути вимкнений блоком TRANSFER. З увімкненням індикатора (Set) транзакт переходить за адресою, зазначеною в C; індикатор переводиться в Reset. У протилежному разі виконується перехід за адресою, зазначеною у B. Цей режим є рудиментом ранніх версій GPSS. Координоване оцінювання стану декількох об'єктів простіше організувати через булеву змінну.

LOOP A, B. Блок LOOP (зациклити) модифікує параметр та управляє рухом активного транзакта, виходячи з результату цієї модифікації.

Операнд A містить посилання на параметр транзакта, що відіграє роль лічильника, і значення параметра зменшується на 1. Якщо нове значення параметра не дорівнює нулю і операнд B специфікований, то планується перехід транзакта за адресою, зазначеною в B. Інакше транзакт переходить до наступного блока. Таким чином, блок LOOP можна використати для задавання циклічних маршрутів (не обчислень) із відомим числом повторень. Цикли виходять з від'ємним кроком і нижнім закінченням (обов'язковим проходженням тіла циклу). Допускаються вкладені цикли.

TEST O A, B, C. Блок порівнює значення зазначених СЧА й управляє напрямком руху транзакта, ґрунтуючись на результаті порівняння. Його операнди:

O – перевіряє відношення числових значень операндів A та B;

A, B – значення, що зіставляються;

C – альтернативний блок призначення.

Оскільки логічні значення в GPSS/W кодуються нулем і одиницею, у блоці TEST можна зіставляти логічні значення.

При невиконанні умови O, якщо операнд C не використовується, то TEST діє в режимі відмови – блокує вхід активного транзакта, а якщо операнд C задано – робить перехід по цьому операнду.

Із заданим операндом C реалізується режим альтернативного виходу. За неуспішного порівняння транзакт планується до передавання згідно з операндом C і поміщається в ЛПП у початок своєї пріоритетної групи.

Блоки TEST і GATE мають великі можливості, але неуспішні перевірки умови можуть викликати великі витрати машинного часу. Для скорочення їхньої частоти можна помістити транзакти в список користувача за допомогою блоків LINK й UNLINK.

GATE O A, B. Блок GATE (впустити) змінює рух транзакта за станом об'єкта, який перевіряється, – логічного ключа, пристрою або БКП, але не черги. Стан завжди має логічне значення. Операнди:

O – вид умови, що перевіряється. З виконанням умови активний транзакт переходить до наступного блоку.

Допустимі значення:

для пристроїв: **U/NU** (зайнятий/вільний), **I/NI** (обслуговує переривання/неперерваний), **FV/FNV** (доступний/недоступний);

для БКП: **SE/SNE** (порожній/непорожній), **SF/SNF** (заповнений/незаповнений), **SV/SNV** (доступний/недоступний);

для логічних ключів: **LS/LR** (включений/выключений);

для транзактів: **M** – у блоці, заданому в полі A, у стані синхронізації перебуває транзакт, який належить до тієї самої сім'ї, що і транзакт, який перебуває в блоці GATE або робить спробу увійти до нього; **NM** – у блоці, заданому в полі A, у стані синхронізації немає жодного транзакта, який належить до тієї самої сім'ї, що і транзакт, який робить спробу увійти до блока GATE;

A – ім'я або номер об'єкта, який перевіряється. Тип елемента повинен бути погоджений з умовою;

B – номер блока, до якого спрямовується транзакт в разі невиконання умови.

Якщо операнд B не використовується, GATE діє в режимі відмови та затримує транзакт до виконання умови. У поєднанні з БКП GATE може використовуватися для формування пачок постійного обсягу, що дорівнює ємності БКП.

Для розгалуження маршруту на два напрямки використовуються двопозиційні логічні ключі: замок є/немає, червоний або зелений сигнал світлофора, табличка "Місць немає", "Перерва". У випадку триколірного світлофора використовують блок TEST і числову змінну (VARIABLE).

16.3. Контрольні приклади

Приклад 1. Модель системи з одним пристроєм і чергою.

1. *Постановка задачі.* Необхідно промоделювати роботу верстата, що обробляє деталі. Деталі до верстата надходять рівномірно в інтервалі від 3 до 7 хвилин. Час обробки деталі складає 6 ± 4 хв. Деталі надходять до верстата, обробляються в порядку "першим прийшов – першого обслугували", а потім залишають його.

Побудувати модель із позачерговим збиранням статистики. Визначити для однієї робочої зміни (8 годин) характеристики роботи системи. Проаналізувати ефективність роботи системи та внести пропозиції щодо покращення роботи системи.

2. *Метод побудови моделі.* У цій моделі деталі ототожнюються з транзактами, а верстат – з пристроєм. Середній час надходження деталей складає 5 хв, половина поля допуску часу надходження складає 2 хв. Аналогічно, середній час обробки складає 6 хв, а половина поля допуску часу обробки – 4 хв. Деталі надходять до верстата та приєднуються

до черги на обробку. Якщо верстат у момент приходу деталі вільний, то обробка починається відразу.

Надходження деталей моделюється блоком GENERATE, видалення деталей моделюється блоком TERMINATE. Блоки SEIZE та RELEASE моделюють зайняття і звільнення пристрою. Моделювання затримки на обробку здійснюється блоком ADVANCE. Збирання статистики за чергою здійснюється за допомогою пари блоків QUEUE і DEPART. Пара блоків GENERATE і TERMINATE моделюють таймер. Час роботи системи – 480 хвилин.

3. *Таблиця визначень*. Одиниця часу в моделі – 1 хвилина.

Таблиця 16.1

Таблиця визначень

Елементи GPSS	Призначення
Транзакти	
1-й сегмент моделі	Обробка деталей
2-й сегмент моделі	Таймер
Пристрій	
VERSTAT	Верстат
Черга	
Q_ VERSTAT	Черга до верстата

4. *Схематичне подання роботи системи* (рис. 16.1).

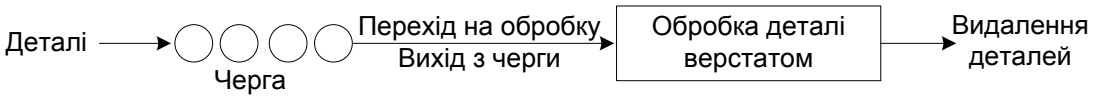

Рис. 16.1. **Схематичне подання роботи системи**

5. *Роздруківка програми*.

```
*Сегмент 1: Обробка деталей
        GENERATE  5,2
        QUEUE     Q_VERSTAT
        SEIZE     VERSTAT
        DEPART    Q_VERSTAT
```

```
            ADVANCE    6,4
            RELEASE    VERSTAT
            TERMINATE
*Сегмент 2: Таймер
            GENERATE  480
            TERMINATE 1
*Запуск імітації
            START      1
```

6. *Вихідні дані програми.* Результати моделювання роботи верстата подані у вигляді стандартного звіту:

```
          GPSS World Simulation Report - Model_8_1.2.1

           Sunday, January 18, 2009 13:53:45

        START TIME        END TIME  BLOCKS  FACILITIES  STORAGES
           0.000          480.000    9       1          0

          NAME                VALUE
        Q_VERSTAT             10000.000
        VERSTAT               10001.000

 LABEL          LOC  BLOCK TYPE     ENTRY COUNT CURRENT COUNT RETRY
                 1   GENERATE          98           0      0
                 2   QUEUE             98           26     0
                 3   SEIZE             72           0      0
                 4   DEPART            72           0      0
                 5   ADVANCE           72           1      0
                 6   RELEASE           71           0      0
                 7   TERMINATE         71           0      0
                 8   GENERATE          1            0      0
                 9   TERMINATE         1            0      0

FACILITY        ENTRIES  UTIL.   AVE. TIME AVAIL. OWNER PEND INTER RETRY
DELAY
 VERSTAT           72    0.987     6.578    1      73    0    0     0     26

QUEUE            MAX CONT. ENTRY ENTRY(0) AVE.CONT. AVE.TIME   AVE.(-0)
RETRY
 Q_VERSTAT         26   26     98      1    12.371     60.593     61.217   0
```

```
FEC XN  PRI      BDT     ASSEM  CURRENT  NEXT  PARAMETER   VALUE
 73   0       480.964    73      5      6
100   0       482.112   100      0      1
101   0       960.000   101      0      8
```

7. *Обговорення.* Протягом робочої зміни до верстата надійшло 98 деталей, з них 71 деталь була оброблена, одна залишилася в процесі обробки. На кінець робочої зміни на обробку залишилось чекати черги 26 деталей. Верстат працював над деталями 72 рази і був зайнятий 98,7 % робочого часу. На обробку однієї деталі в середньому витрачалося 6,6 хв. Максимальний вміст черги протягом періоду моделювання – 26 деталей. Поза чергою на обслуговування потрапила тільки одна деталь (та, що надійшла до верстата першою). Оскільки максимальний і поточний вміст черги збігаються, без черги пройшла на обробку одна деталь, то можна зробити висновок, що черга постійно збільшується. Середній час перебування деталі в черзі склав 60,6 хв з урахуванням усіх входів у чергу і 61,2 хв без урахування нульових входів у чергу.

Для покращення роботи системи можна збільшити кількість верстатів, що дозволить обробити більшу кількість деталей, скоротити час перебування деталей у черзі.

Приклад 2. Модель системи з одним пристроєм і перервами в роботі пристрою.

1. *Постановка задачі*. Візьмемо за основу приклад 1. Додамо ще одну умову: кожні 3 год проводиться технічний огляд верстата 5 хв і його налагодження 10 ± 3 хв.

Побудувати модель роботи верстата з перервами на технічний огляд і налагодження. Зібрати статистику за чергою. Визначити для однієї робочої зміни (8 год) характеристики роботи системи. Проаналізувати ефективність роботи системи та внести пропозиції щодо покращення роботи системи. Порівняти отримані результати в прикладах 1 і 2.

2. *Метод побудови моделі*. Візьмемо за основу приклад 1.

Перерви в роботі верстата моделюються блоками FUNAVAIL (переведення верстата в режим неготовності для обробки деталей) і FAVAIL (повернення в режим готовності).

3. *Таблиця визначень* (табл. 16.2). Одиниця часу в моделі – 1 хв.

Таблиця 16.2

Таблиця визначень

Елементи GPSS	Призначення
Транзакти	
1-й сегмент моделі	Обробка деталей
2-й сегмент моделі	Технічний огляд і налагодження верстата
3-й сегмент моделі	Таймер
Пристрій	
VERSTAT	Верстат
Черга	
Q_ VERSTAT	Черга до верстата

4. *Схематичне подання роботи системи.* За зразком прикладу 1 (рис. 16.1).

5. *Роздруківка програми.*

```
*Сегмент 1: Обробка деталей
        GENERATE  5,2
        QUEUE     Q_VERSTAT
        SEIZE     VERSTAT
        DEPART    Q_VERSTAT
        ADVANCE   6,4
        RELEASE   VERSTAT
        TERMINATE
*Сегмент 2: Технічний огляд і налагодження верстата
        GENERATE  180
        FUNAVAIL  VERSTAT
        ADVANCE   5
        ADVANCE   10,3
        FAVAIL    VERSTAT
        TERMINATE
*Сегмент 3: Таймер
        GENERATE  480
        TERMINATE 1
*Запуск імітації
        START     1
```

6. *Вихідні дані програми.* Результати моделювання роботи верстата представлені нижче у вигляді стандартного звіту:

```
          GPSS World Simulation Report - Model_8_3.8.1
              Sunday, January 18, 2009 14:05:58
        START TIME          END TIME  BLOCKS  FACILITIES  STORAGES
            0.000           480.000    15        1          0

          NAME                    VALUE
       Q_VERSTAT                10000.000
       VERSTAT                  10001.000

 LABEL            LOC  BLOCK TYPE     ENTRY COUNT CURRENT COUNT RETRY
                    1    GENERATE          97            0       0
                    2    QUEUE             97           27       0
                    3    SEIZE             70            0       0
                    4    DEPART            70            0       0
                    5    ADVANCE           70            1       0
                    6    RELEASE           69            0       0
                    7    TERMINATE         69            0       0
                    8    GENERATE           2            0       0
                    9    FUNAVAIL           2            0       0
                   10    ADVANCE            2            0       0
                   11    ADVANCE            2            0       0
                   12    FAVAIL             2            0       0
                   13    TERMINATE          2            0       0
                   14    GENERATE           1            0       0
                   15    TERMINATE          1            0       0
FACILITY         ENTRIES  UTIL.   AVE. TIME AVAIL. OWNER PEND INTER RETRY
DELAY
 VERSTAT            70    0.916     6.280    1      73    0    0     0     27
QUEUE             MAX CONT. ENTRY ENTRY(0) AVE.CONT. AVE.TIME   AVE.(-0)
RETRY
 Q_VERSTAT          28   27     97      1    13.375     66.185     66.874   0

FEC XN   PRI        BDT      ASSEM  CURRENT  NEXT  PARAMETER    VALUE
  102    0        484.303    102      0      1
   73    0        486.469     73      5      6
   78    0        540.000     78      0      8
  103    0        960.000    103      0     14
```

7. *Обговорення*. Протягом робочої зміни до верстата надійшло 97 деталей, із них 69 деталей було оброблено, одна залишилася в процесі обробки. На кінець робочої зміни на обробку залишилось чекати в черзі

27 деталей. Верстат працював над деталями 70 разів і був зайнятий обробкою деталей 91,6 % робочого часу. На кінець моделювання верстат перебуває в стані готовності для обробки деталей. На обробку однієї деталі в середньому витрачалося 6,3 хв. Максимальний вміст черги протягом періоду моделювання – 26 деталей. Середній час перебування деталі в черзі склав 66,2 хв з урахуванням усіх входів у чергу і 66,8 хв без урахування нульових входів у чергу. За зміну двічі проводився технічний контроль і налагодження верстата.

Порівняння результатів моделювання, отриманих в прикладах 1 і 2:
середній час очікування у черзі збільшився на 5,6 хв;
кількість оброблених деталей зменшилась на дві деталі;
верстат простоював 8,4 % робочого часу (проти 1,3 % у прикладі 1).

Приклад 3. Модель системи обслуговування із забезпеченням пріоритетного обслуговування і переривань в обслуговуванні.

1. Постановка задачі. Телевізійна майстерня найняла одного майстра для капітального ремонту наданих в оренду телевізорів, сервісного обслуговування техніки клієнтів і виконання дрібного негайного ремонту. Необхідність у капітальному ремонті телевізорів, що належать компанії, виникає кожні 20 ± 8 год; ремонт займає 10 ± 1 год. Дрібний ремонт (наприклад, заміна плавкого запобіжника, налаштування каналів і налаштування телевізорів) виконується негайно. Необхідність у дрібному ремонті виникає кожні 90 ± 10 хв, ремонт триває 15 ± 5 хв. Телевізори, що належать клієнтам та вимагають звичайного обслуговування, прибувають кожні 5 ± 1 год, їх ремонт триває 150 ± 30 хв. Звичайне обслуговування телевізорів має вищий пріоритет, ніж капітальний ремонт техніки, що здається в оренду, та техніки, що перебуває у власності компанії.

Необхідно промоделювати роботу майстерні протягом 50 днів, врахувавши, що за цей період можна прийняти на капітальний ремонт 15 одиниць техніки, а робочий день триває 8 год.

2. Метод побудови моделі. У телевізійній майстерні майстер надає три типи послуг: капітальний ремонт телевізорів, сервісне обслуговування техніки клієнтів і дрібний ремонт. Моделювання надання кожного типу послуг проводитиметься за окремим сегментом моделі. У майстра формується одна черга на обслуговування. Позачергово проводитиметься звичайне обслуговування телевізорів (вищий пріоритет). Дрібний ремонт

техніки проводиться негайно, а отже, має найвищий пріоритет. "Негайно" – означає, що, як тільки надійде замовлення на цей тип ремонту, майстер перериває виконувану на той момент роботу та береться за це замовлення. Якщо на момент надходження нового замовлення на ремонт майстер вільний, то виконання замовлення починається відразу. У зв'язку з наявністю замовлень із різними пріоритетами, дисципліна обслуговування – "першим прийшов – першого обслугували" дещо зміниться.

Надходження замовлень на ремонт моделюється блоком GENERATE, покидання системи виконаним замовленням моделюється блоком TERMINATE. Блоки SEIZE та RELEASE моделюють зайняття і звільнення пристрою (майстра). Блоки PREEMPT і RETURN моделюють зайняття і звільнення пристрою (майстра) в режимі переривання. Моделювання затримки на обслуговуванні здійснюється блоком ADVANCE. Збирання статистики за чергою здійснюється за допомогою пари блоків QUEUE і DEPART. Обмеження на приймання техніки на капітальний ремонт задається операндом D у блоці GENERATE відповідного сегмента моделі. Пара блоків GENERATE та TERMINATE моделюють таймер. Час роботи системи – 50 робочих днів по 480 хв (у таймері кожен транзакт моделює один день).

3. *Таблиця визначень* (табл. 16.3). Одиниця часу в моделі – 1 хв.

Таблиця 16.3

Таблиця визначень

Елементи GPSS	Призначення
Транзакти	
1-й сегмент моделі	Капітальний ремонт техніки
2-й сегмент моделі	Звичайний ремонт техніки клієнтів
3-й сегмент моделі	Дрібний ремонт
4-й сегмент моделі	Таймер
Пристрій	
MASTER	Майстер
Черги	
Q_MASTER	Загальна черга замовлень до майстра
Q_KAP_REM	Черга із замовлень на капітальний ремонт
Q_TEX_KLIENT	Черга із замовлень на ремонт техніки клієнтів
Q_DRIB_REM	Черга із замовлень на дрібний ремонт

4. Схематичне подання роботи системи (рис. 16.2).

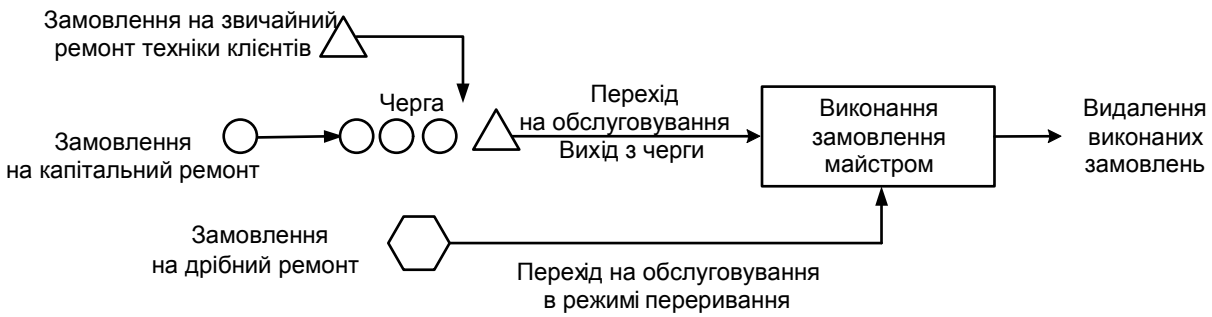

Рис. 16.2. **Схематичне подання роботи системи**

5. Роздруківка програми.

```
*Сегмент 1: Капітальний ремонт орендованої техніки
            GENERATE  1200,480,,15,1
            QUEUE     Q_KAP_REM
            QUEUE     Q_MASTER
            SEIZE     MASTER
            DEPART    Q_KAP_REM
            DEPART    Q_MASTER
            ADVANCE   600,60
            RELEASE   MASTER
            TERMINATE

* Сегмент 2: Звичайний ремонт техніки клієнтів
            GENERATE  300,60,,,2
            QUEUE     Q_TEX_KLIENT
            QUEUE     Q_MASTER
            SEIZE     MASTER
            DEPART    Q_TEX_KLIENT
            DEPART    Q_MASTER
            ADVANCE   150,30
            RELEASE   MASTER
            TERMINATE
*Сегмент 3: Дрібний ремонт
            GENERATE  90,10,,,3
            QUEUE     Q_DRIB_REM
            QUEUE     Q_MASTER
            PREEMPT   MASTER,PR
            DEPART    Q_DRIB_REM
```

```
            DEPART      Q_MASTER
            ADVANCE     15,5
            RETURN      MASTER
            TERMINATE
*Сегмент 4: Таймер
            GENERATE  480
            TERMINATE 1
* Запуск імітації
            START       50
```

6. Вихідні дані програми. Результати моделювання роботи телевізійної майстерні представлені нижче у вигляді стандартного звіту.

```
          GPSS World Simulation Report - Model_8_4.34.1
              Thursday, January 15, 2009 14:06:57
        START TIME          END TIME  BLOCKS  FACILITIES  STORAGES
            0.000         24000.000    29        1         0
          NAME                  VALUE
       MASTER                 10002.000
       Q_DRIB_REM               10000.000
       Q_KAP_REM                10004.000
       Q_MASTER                 10001.000
       Q_TEX_KLIENT              10003.000
 LABEL            LOC  BLOCK TYPE     ENTRY COUNT CURRENT COUNT RETRY
                    1    GENERATE          15            0       0
                    2    QUEUE             15            0       0
                    3    QUEUE             15            2       0
                    4    SEIZE             13            0       0
                    5    DEPART            13            0       0
                    6    DEPART            13            0       0
                    7    ADVANCE           13            0       0
                    8    RELEASE           13            0       0
                    9    TERMINATE         13            0       0
                   10    GENERATE          80            0       0
                   11    QUEUE             80            0       0
                   12    QUEUE             80            0       0
                   13    SEIZE             80            0       0
                   14    DEPART            80            0       0
                   15    DEPART            80            0       0
                   16    ADVANCE           80            1       0
                   17    RELEASE           79            0       0
                   18    TERMINATE         79            0       0
                   19    GENERATE         266            0       0
```

```
              20   QUEUE          266          0      0
              21   QUEUE          266          0      0
              22   PREEMPT         266          0      0
              23   DEPART          266          0      0
              24   DEPART          266          0      0
              25   ADVANCE         266          0      0
              26   RETURN          266          0      0
              27   TERMINATE       266          0      0
              28   GENERATE         50          0      0
              29   TERMINATE        50          0      0
FACILITY      ENTRIES  UTIL.   AVE. TIME AVAIL. OWNER PEND INTER RETRY
DELAY
 MASTER         359    0.970     64.854  1     404    0    0     0     2

QUEUE           MAX CONT. ENTRY ENTRY(0) AVE.CONT. AVE.TIME   AVE.(-0)
RETRY
 Q_DRIB_REM       1    0    266    266     0.000     0.000     0.000   0
 Q_MASTER         8    2    361    271     3.503   232.919   934.266   0
 Q_TEX_KLIENT     3    0     80      4     1.104   331.171   348.601   0
 Q_KAP_REM        5    2     15      1     2.400  3839.346  4113.585   0

FEC XN   PRI       BDT     ASSEM  CURRENT  NEXT  PARAMETER    VALUE
  410    2      24035.902    410      0     10
  413    3      24051.356    413      0     19
  404    2      24147.240    404     16     17
  414    0      24480.000    414      0     28
```

7. Обговорення. За 50 робочих днів у майстерню надійшло 15 замовлень на капітальний ремонт (з них 13 було виконано), 80 замовлень на ремонт техніки клієнтів (з них виконано 79, одне замовлення на кінець моделювання залишилося на виконанні) і 266 замовлень на дрібний ремонт були виконані повністю. Усі замовлення на дрібний ремонт виконувалися одразу, про що свідчить статистика за чергами. У 80 замовлень на звичайний ремонт техніки клієнтів 4 не очікували в черзі, а останні затрималися в ній у середньому на 348 хв (майже 6 год). Майстер був зайнятий ремонтом 97 % всього робочого часу та виконав 359 замовлень (одне не було закінчено).

У цілому система працює добре, але можна дещо змінити дисципліну обслуговування – переривати роботу з капітального ремонту, щоб ремонтувати техніку клієнтів, оскільки за умовою цей вид робіт є пріоритетнішим. Це значно скоротить час перебування таких замовлень у черзі.

Приклад 4. Модель системи масового обслуговування з багатоканальним пристроєм.

1. *Постановка задачі.* До перукарні, в якій працює два перукарі, приходять клієнти двох типів – чоловіки та жінки. Клієнти переходять на обслуговування до вільного перукаря. Інтервали приходу клієнтів розподілені рівномірно в інтервалі 30 ± 10 хв (для жінок) і 50 ± 15 хв (для чоловіків). Час обслуговування також розподілений рівномірно в інтервалі 40 ± 15 хв (для жінок) і 30 ± 5 хв (для чоловіків). Дисципліна обслуговування у перукарні "першим прийшов – першого обслугували". Після обслуговування клієнти залишають перукарню.

Промоделювати роботу перукарні протягом одного робочого дня (8 год.) зі збиранням статистики за чергами. Проаналізувати ефективність роботи системи та внести пропозиції щодо покращення роботи системи.

2. *Метод побудови моделі*. У цій моделі клієнти ототожнюються з транзактами, а перукарі – з багатоканальним пристроєм. Клієнти приходять до перукарні та приєднуються до черги на обслуговування. У порядку черги клієнти переходять на обслуговування до вільного перукаря.

Прихід клієнтів моделюється блоком GENERATE, вихід клієнтів із перукарні моделюється блоком TERMINATE. Блоки ENTER і LEAVE моделюють вхід і вихід у БКП. Моделювання затримки на обслуговуванні здійснюється блоком ADVANCE. Збирання статистики за чергою здійснюється за допомогою пари блоків QUEUE та DEPART. Пара блоків GENERATE та TERMINATE моделюють таймер. Час роботи системи – 480 хв.

3. Таблиця визначень (табл. 16.4). Одиниця часу в моделі – 1 хв.

Таблиця 16.4

Таблиця визначень

Елементи GPSS	Призначення
Транзакти	
1-й сегмент моделі	Клієнти першого типу (чоловіки)
2-й сегмент моделі	Клієнти другого типу (жінки)
3-й сегмент моделі	Таймер
БКП	
PERUKARI	БКП Перукарі
Черга	
Q_PERUKAR	Загальна черга до перукарів

4. Схематичне представлення роботи системи (рис. 16.3).

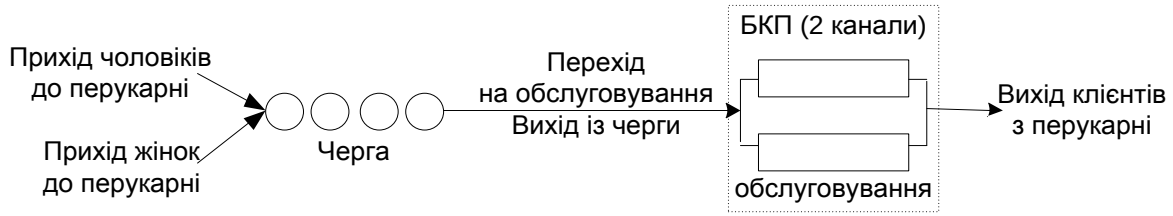

Рис. 16.3. **Схематичне подання роботи системи**

5. Роздруківка програми.

```
*Багатоканальний пристрій
PERUKARI              STORAGE          2
*Сегмент 1: Обслуговування клієнтів 1-го типу (чоловіки)
              GENERATE  50,15
              QUEUE     Q_PERUKARI
              ENTER     PERUKARI
              DEPART    Q_PERUKARI
              ADVANCE   30,5
              LEAVE     PERUKARI
              TERMINATE
*Сегмент 2: Обслуговування клієнтів 2-го типу (жінки)
              GENERATE  30,10
              QUEUE     Q_PERUKARI
              ENTER     PERUKARI
              DEPART    Q_PERUKARI
              ADVANCE   40,15
              LEAVE     PERUKARI
              TERMINATE
* Сегмент 3: Таймер
              GENERATE  480
              TERMINATE 1
* Запуск імітації
              START     1
```

6. Вихідні дані програми. Результати моделювання роботи перукарні представлені нижче у вигляді стандартного звіту.

```
          GPSS World Simulation Report - Model_9_1.1.1
              Thursday, January 15, 2009 21:18:25
        START TIME          END TIME  BLOCKS  FACILITIES  STORAGES
            0.000           480.000    16       0          1
```

```
          NAME                VALUE
        PERUKARI              10000.000
        Q_PERUKARI            10001.000

 LABEL          LOC  BLOCK TYPE     ENTRY COUNT CURRENT COUNT RETRY
                 1   GENERATE           9           0       0
                 2   QUEUE              9           0       0
                 3   ENTER              9           0       0
                 4   DEPART             9           0       0
                 5   ADVANCE            9           1       0
                 6   LEAVE              8           0       0
                 7   TERMINATE          8           0       0
                 8   GENERATE          15           0       0
                 9   QUEUE             15           0       0
                10   ENTER             15           0       0
                11   DEPART            15           0       0
                12   ADVANCE           15           0       0
                13   LEAVE             15           0       0
                14   TERMINATE         15           0       0
                15   GENERATE           1           0       0
                16   TERMINATE          1           0       0

QUEUE            MAX CONT. ENTRY ENTRY(0) AVE.CONT. AVE.TIME   AVE.(-0)
RETRY
 Q_PERUKARI        2    0    24     9    0.332     6.641     10.626   0

STORAGE          CAP. REM. MIN. MAX.  ENTRIES AVL.  AVE.C. UTIL. RETRY DELAY
 PERUKARI          2    1   0    2      24   1    1.726  0.863    0    0

FEC XN   PRI       BDT      ASSEM  CURRENT  NEXT  PARAMETER    VALUE
   26    0        485.182    26      0      8
   25    0        500.706    25      5      6
   27    0        503.526    27      0      1
   28    0        960.000    28      0     15
```

7. Обговорення. Протягом робочого дня до перукарні прийшло дев'ять чоловіків і 15 жінок (усього 24 клієнти). На кінець періоду моделювання майже всі клієнти були обслуговані (один залишився в процесі обслуговування). Дев'ять осіб не затрималися в черзі, тобто в момент приходу їх до перукарні один із перукарів був вільний, і клієнти відразу проходили на обслуговування. Максимальний вміст черги за період моделювання складав дві особи. На кінець періоду моделювання в черзі

клієнтів не було. Середній час очікування клієнтів у черзі (без нульових входів) склав 10,6 хв. Багатоканальний пристрій був зайнятий 86,3 % робочого періоду. Максимальна кількість одночасно зайнятих каналів – два (отже обидва перукарі працювали).

Приклад 5. Модель системи масового обслуговування з багатоканальним пристроєм і перервою в роботі.

1. Постановка задачі. За основу візьмемо задачу з прикладу 4 цієї лабораторної роботи. Укажемо додатково, що перукарня працює з 08:00 до 16:00, а перерва з 12:00 до 13:00.

2. Метод побудови моделі. За основу взяти приклад 4.

Для моделювання перерви в роботі перукарні використовуються блоки SUNAVAIL і SAVAIL. Перерва одна на робочий день, починається через 4 год (240 хв) після початку роботи та триває 1 год (60 хв).

3. Таблиця визначень (табл. 16.5). Одиниця часу в моделі – 1 хв.

Таблиця 16.5

Таблиця визначень

Елементи GPSS	Призначення
Транзакти	
1-й сегмент моделі	Клієнти першого типу (чоловіки)
2-й сегмент моделі	Клієнти другого типу (жінки)
3-й сегмент моделі	Моделювання перерви в роботі
4-й сегмент моделі	Таймер
БКП	
PERUKARI	БКП Перукарі
Черга	
Q_PERUKAR	Загальна черга до перукарів

4. Схематичне представлення роботи системи. За зразком п. 4 прикладу 4.

5. Роздруківка програми.

```
*Багатоканальний пристрій
PERUKARI        STORAGE     2
```

```
*Сегмент 1: Обслуговування клієнтів 1-го типу (чоловіки)
            GENERATE  50,15
            QUEUE     Q_PERUKARI
            ENTER     PERUKARI
            DEPART    Q_PERUKARI
            ADVANCE   30,5
            LEAVE     PERUKARI
            TERMINATE
*Сегмент 2: Обслуговування клієнтів 2-го типу (жінки)
            GENERATE  30,10
            QUEUE     Q_PERUKARI
            ENTER     PERUKARI
            DEPART    Q_PERUKARI
            ADVANCE   40,15
            LEAVE     PERUKARI
            TERMINATE
*Сегмент 3: Перерва в роботі
            GENERATE  240,,,1
            SUNAVAIL  PERUKARI
            ADVANCE   60
            SAVAIL    PERUKARI
            TERMINATE
*Сегмент 4: Таймер
            GENERATE  480
            TERMINATE 1
*Запуск імітації
            START     1
```

6. Вихідні дані програми. Результати моделювання роботи перукарні подані у вигляді стандартного звіту.

```
          GPSS World Simulation Report - Model_9_3.1.1
              Sunday, January 18, 2009 16:50:24
       START TIME        END TIME  BLOCKS  FACILITIES  STORAGES
          0.000          480.000    21       0         1

          NAME                VALUE
       PERUKARI             10000.000
       Q_PERUKARI           10001.000

 LABEL          LOC  BLOCK TYPE     ENTRY COUNT CURRENT COUNT RETRY
                 1   GENERATE           9           0      0
```

2	QUEUE	9	1	0
3	ENTER	8	0	0
4	DEPART	8	0	0
5	ADVANCE	8	1	0
6	LEAVE	7	0	0
7	TERMINATE	7	0	0
8	GENERATE	15	0	0
9	QUEUE	15	2	0
10	ENTER	13	0	0
11	DEPART	13	0	0
12	ADVANCE	13	1	0
13	LEAVE	12	0	0
14	TERMINATE	12	0	0
15	GENERATE	1	0	0
16	SUNAVAIL	1	0	0
17	ADVANCE	1	0	0
18	SAVAIL	1	0	0
19	TERMINATE	1	0	0
20	GENERATE	1	0	0
21	TERMINATE	1	0	0

QUEUE	MAX	CONT.	ENTRY	ENTRY(0)	AVE.CONT.	AVE.TIME	AVE.(-0)	RETRY
Q_PERUKARI	5	3	24	5	1.978	39.559	49.970	0

STORAGE	CAP.	REM.	MIN.	MAX.	ENTRIES	AVL.	AVE.C.	UTIL.	RETRY	DELAY
PERUKARI	2	0	0	2	21	1	1.534	0.767	0	3

FEC XN	PRI	BDT	ASSEM	CURRENT	NEXT	PARAMETER	VALUE
28	0	481.780	28	0	8		
22	0	486.924	22	12	13		
27	0	506.116	27	0	1		
23	0	510.914	23	5	6		
29	0	960.000	29	0	20		

7. Обговорення. Протягом робочого дня до перукарні прийшло дев'ять чоловіків і 15 жінок (усього 24 клієнти). На кінець періоду моделювання одна жінка і один чоловік залишилися на обслуговуванні. П'ять осіб не затрималися в черзі, тобто в момент приходу їх до перукарні один з перукарів був вільний, і клієнти відразу проходили на обслуговування. Максимальний вміст черги за період моделювання складав п'ять осіб. На кінець періоду моделювання в черзі очікувало три клієнта. Середній

час очікування клієнтів у черзі (без нульових входів) склав 49,97 хв. Багатоканальний пристрій був зайнятий 76,7 % робочого періоду. Максимальна кількість одночасно зайнятих каналів – два (отже, обидва перукарі працювали).

Передбачена перерва майже в п'ять разів збільшила середній час перебування клієнтів у черзі (не враховуючи тих, що пройшли на обслуговування без очікування). Як наслідок, зменшилась кількість обслугованих клієнтів (на чотири особи) і збільшився простій БКП – до 24,3 % робочого часу (проти 13,7 % при роботі без перерви).

Приклад 6. Використання функцій у GPSS-моделях.

1. Постановка задачі. Двоє викладачів приймають екзамен у студентів. Час приходу студентів в аудиторію рівномірно розподілений на інтервалі від 5 до 10 хв (рівномірний закон розподілу). Взявши екзаменаційний білет, студент готується протягом 20 ± 5 хв і йде відповідати одному з викладачів. В аудиторії може знаходитися одночасно не більше 10 студентів. Час відповіді на основні запитання розподілений таким чином: з імовірністю 0,2 – 3 хв, з імовірністю 0,3 – 5 хв, з імовірністю 0,4 – 8 хв і з імовірністю 0,1 – 12 хв. Кожен студент отримує від 3 до 10 додаткових запитань, відповідь на кожне з яких триває 1,5 хв.

Після складання екзамену студент чекає 2 хв (час, потрібний викладачеві для виставлення оцінки в екзаменаційну відомість і в залікову книжку) та залишає аудиторію.

Необхідно записати GPSS-модель роботи екзаменаційної комісії. Визначити час, потрібний комісії, щоб прийняти екзамен у 50 студентів.

Зробити висновки та внести пропозиції щодо поліпшення роботи приведеної системи масового обслуговування.

2. *Метод побудови моделі*. Викладачі розглядаються як багатоканальний пристрій, оскільки студенти йдуть відповідати одному з двох викладачів. В аудиторії може одночасно знаходитися не більше 10 студентів, тому аудиторію можна розглядати як багатоканальний пристрій з 10 каналами обслуговування (місцями).

Прихід студентів на екзамен моделюється блоком GENERATE. Час приходу студентів описується рівномірним законом розподілу, який реалізований системі GPSS, і задається в операнді A. Операнд D – задає максимальну кількість студентів, що має прийти на екзамен (за умовою – 50 осіб). Час відповіді на основні запитання описується дискретною

функцією і задається чотирма точками. Кількість додаткових запитань, що їх може отримати кожен студент, задається неперервною функцією з двома координатами: значення аргумента x1 = 0 указує на мінімальну кількість запитань – 3 запитання, тобто y1 = 3; x2 = 1 вказує на максимальну кількість запитань – 10 запитань (оскільки ДВЧ генерує значення в інтервалі від 0 до 0.999999, то, щоб отримати 10 запитань, треба задати y2 = 11). Посилання на цю функцію вказується в операнді D блока ADVANCE, і ця функція виступає як модифікатор. Управляючою картою START задається кількість студентів, що мають скласти екзамен.

3. *Таблиця визначень* (табл. 16.6). Одиниця часу в моделі – 1 хв.

Таблиця 16.6

Таблиця визначень

Елементи GPSS	Призначення
Транзакти	
1-й сегмент моделі	Проведення екзамену
БКП	
AUDITOR	БКП Аудиторія
VIKLADACH	БКП Викладачі
Черги	
Q_AUDITOR	Черга студентів до аудиторії
Q_VIKLADACH	Черга студентів до викладачів
Функції	
UNIFORM	Вбудована, рівномірний розподіл
VIDP_OSN	Час відповіді на основні запитання
KILK_DOD	Кількість додаткових запитань

4. Схематичне представлення роботи системи (рис. 16.4).

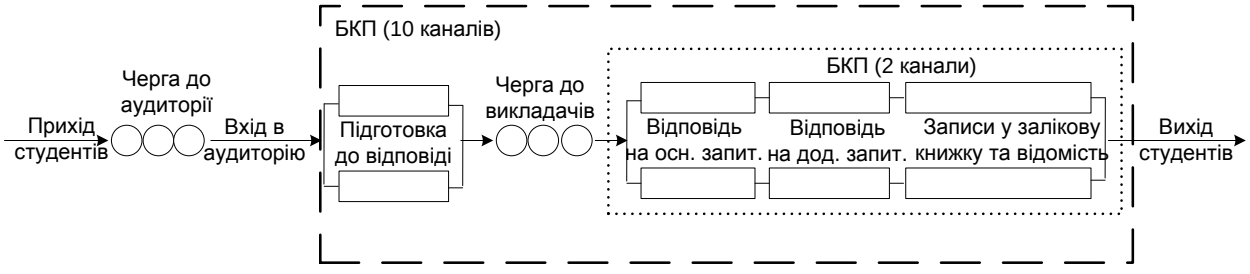

Рис. 16.4. **Схематичне подання роботи системи**

5. Роздруківка програми.

```
*Багатоканальні пристрої
AUDITOR         STORAGE      10
VIKLADACH       STORAGE      2
*Функція розподілу часу відповіді на основні запитання
VIDP_OSN                FUNCTION            RN1,D4
.2,3/.5,5/.9,8/1,12
*Функція для визначення кількості додаткових запитань
KILK_DODFUNCTION   RN1,C2
0,3/1,11
*Проведення екзамену
                GENERATE  (UNIFORM(RN1,5,10)),,,50
                QUEUE       Q_AUDITOR
                ENTER       AUDITOR
                DEPART      Q_AUDITOR
                ADVANCE     20,5
                QUEUE       Q_VIKLADACH
                ENTER       VIKLADACH
                DEPART      Q_VIKLADACH
                ADVANCE     FN$VIDP_OSN
                ADVANCE     1.5,FN$KILK_DOD
                ADVANCE     2
                LEAVE       VIKLADACH
                LEAVE       AUDITOR
                TERMINATE 1
* Запуск імітації
                START       50
```

6. Вихідні дані програми. Результати моделювання проведення екзамену подані у вигляді стандартного звіту.

```
          GPSS World Simulation Report - Model_11_1.1.1
              Friday, January 16, 2009 11:43:43
       START TIME          END TIME  BLOCKS  FACILITIES  STORAGES
          0.000           515.684    14       0         2

          NAME                VALUE
       AUDITOR              10000.000
       KILK_DOD             10003.000
       Q_AUDITOR              10004.000
       Q_VIKLADACH              10005.000
       VIDP_OSN             10002.000
       VIKLADACH              10001.000
```

LABEL	LOC	BLOCK TYPE	ENTRY COUNT	CURRENT COUNT	RETRY
	1	GENERATE	50	0	0
	2	QUEUE	50	0	0
	3	ENTER	50	0	0
	4	DEPART	50	0	0
	5	ADVANCE	50	0	0
	6	QUEUE	50	0	0
	7	ENTER	50	0	0
	8	DEPART	50	0	0
	9	ADVANCE	50	0	0
	10	ADVANCE	50	0	0
	11	ADVANCE	50	0	0
	12	LEAVE	50	0	0
	13	LEAVE	50	0	0
	14	TERMINATE	50	0	0

QUEUE	MAX	CONT.	ENTRY	ENTRY(0)	AVE.CONT.	AVE.TIME	AVE.(-0)	RETRY
Q_AUDITOR	6	0	50	24	0.951	9.807	18.860	0
Q_VIKLADACH	8	0	50	2	4.167	42.973	44.764	0

STORAGE	CAP.	REM.	MIN.	MAX.	ENTRIES	AVL.	AVE.C.	UTIL.	RETRY	DELAY
AUDITOR	10	10	0	10	50	1	7.930	0.793	0	0
VIKLADACH	2	2	0	2	50	1	1.844	0.922	0	0

7. Обговорення. Щоб прийняти екзамен у 50 студентів, екзаменаційна комісія витратила майже 516 хв. Усього в аудиторію увійшло 50 студентів. На кінець періоду моделювання в аудиторії студентів не було. За такої організації приходу студентів на екзамен у черзі до аудиторії було максимум шість студентів, а до викладачів на відповідь очікувало максимум вісім студентів. Без черги в аудиторію увійшло 24 студенти, а до викладачів – двоє. За час проведення екзамену аудиторія використовувалася протягом 79,3 % часу, а викладачі були зайняті протягом 92,2 % часу.

Приклад 7. Зміна дисципліни обслуговування.

1. Постановка задачі. Продовольчий магазин складається з трьох прилавків та однієї каси на виході з магазину. Вхідний потік покупців має пуасонівський характер із середнім інтервалом приходу покупців 75 с.

Увійшовши до магазину, кожен покупець бере кошик і може обійти один або кілька прилавків, вибираючи продукти. Імовірність виконання купівель для першого прилавка – 0,75, для другого – 0,55, для третього – 0,82. Час, потрібний для того, щоб обійти прилавок, і число покупок, вибраних біля прилавка, розподілені рівномірно. Час обходу першого прилавка – 120 ± 60 с, другого – 150 ± 30 с, третього – 120 ± 45 с. Число покупок, зроблених біля першого прилавка, – 3 ± 1 шт., другого – 4 ± 1 шт., третього – 5 ± 1 шт.

Після того як товар вибрано, покупець стає в кінець черги до каси. Стоячи в черзі, покупець може зробити ще 2 ± 1 покупок. Час обслуговування покупця в касі пропорційний до числа зроблених покупок, на одну купівель витрачається 3 с. Після оплати товару покупець залишає кошик і покидає магазин.

Необхідно побудувати модель, що описує процес покупок у магазині протягом 8-годинного робочого дня. Слід вважати, що число кошиків необмежене. Визначити навантаження касира, максимальну довжину черги протягом робочого дня, максимальне число кошиків, що перебували у покупців одночасно.

2. Метод побудови моделі. Щоб показати необмежене число кошиків, необхідно задати в моделі багатоканальний пристрій з дуже великою місткістю. Процедура взяття кошика моделюється як вхід у багатоканальний пристрій. Кількість покупок, зроблених біля кожного прилавка та біля каси моделюється за даними задачі за допомогою функцій.

Підхід до кожного прилавка моделюється з використанням блока TRANSFER у статистичному режимі. Якщо покупець вирішує не робити покупок біля одного з прилавків, то він проходить далі до нового блока TRANSFER. Якщо покупець залишається щоб зробити покупку, то він затримується біля прилавка на деякий час. Кількість покупок, зроблених біля кожного прилавка моделюється з використанням неперервної функції. Число покупок, починаючи з початку моделі, накопичується (використовується блок ASSIGN у режимі приросту). Зробивши покупки, покупці приєднуються до черги в касу. Стоячи в черзі, покупці можуть зробити ще покупки, кількість яких моделюється неперервною функцією. Час обслуговування касиром визначається розрахунком функції, аргументом якої є параметр, що містить значення загальної кількості зроблених покупок. Час роботи магазину – 8 годин або 28 800 с.

3. *Таблиця визначень* (табл. 16.7). Одиниця часу в моделі – 1 секунда.

Таблиця 16.7

Таблиця визначень

Елементи GPSS	Призначення
Транзакти	
1-й сегмент моделі	Робота магазину
2-й сегмент моделі	Таймер
БКП	
CARTS	Кошики в магазині
Пристрій	
KASSA	Каса в магазині
Функції	
PRIL1	Кількість покупок, зроблених біля прилавка 1
PRIL2	Кількість покупок, зроблених біля прилавка 2
PRIL3	Кількість покупок, зроблених біля прилавка 3
POK_KASSA	Кількість покупок, зроблених біля каси
OPLATA	Час обслуговування покупця в касі
Параметри транзактів	
1	Для прийняття значень: кількість зроблених покупок

4. Схематичне представлення роботи системи (рис. 16.5).

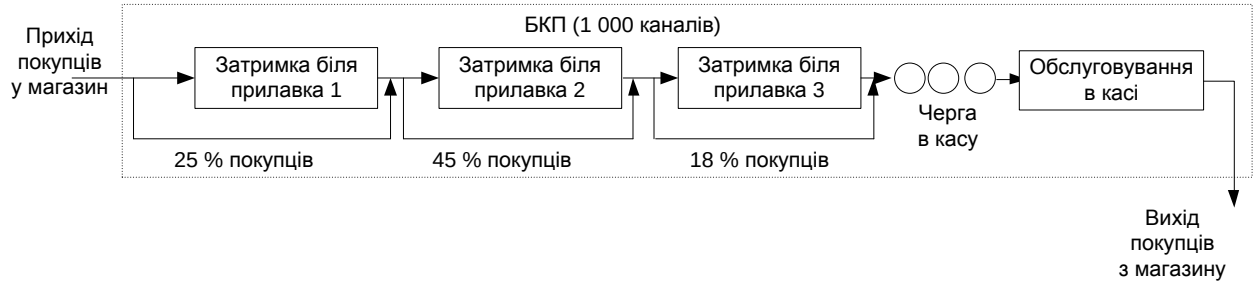

Рис. 16.5. **Схематичне подання роботи системи**

5. Роздруківка програми.

```
*Багатоканальний пристрій - кошики в магазині
CARTS STORAGE 1000
*Функції, що визначають кількість покупок, зроблених біля кожного з прилавків
PRIL1 FUNCTION RN2,C2
0,2/1,5
PRIL2 FUNCTION RN2,C2
0,3/1,6
PRIL3 FUNCTION RN2,C2
0,4/1,7
```

```
*Функція, що визначає час обслуговування покупця в касі
OPLATA FUNCTION P1,C2
0,3/18,54
*Функція, що визначає кількість покупок, зроблених біля каси
POK_KASSA FUNCTION RN1,C2
0,1/1,4
*Обслуговування покупців в магазині
                GENERATE (POISSON(RN2,75))
                ENTER CARTS
                TRANSFER .25,,MET1
                ADVANCE 120,60
                ASSIGN 1,FN$PRIL1
        MET1    TRANSFER .45,,MET2
                ADVANCE 150,30
                ASSIGN 1+,FN$PRIL2
        MET2    TRANSFER .18,,OUT
                ADVANCE 120,45
              ASSIGN 1+,FN$PRIL3
        OUT     QUEUE KASSA
                ASSIGN 1+,FN$POK_KASSA
                SEIZE KASSA
                DEPART KASSA
                ADVANCE FN$OPLATA
                RELEASE KASSA
                LEAVE CARTS
                TERMINATE
*таймер
                GENERATE 28800
                TERMINATE 1
*Запуск імітації
              START       1
```

6. Вихідні дані програми. Результати моделювання роботи магазину подані у вигляді стандартного звіту:

```
          GPSS World Simulation Report - 14_1.1.1
             Monday, January 19, 2009 02:50:33
       START TIME          END TIME  BLOCKS  FACILITIES  STORAGES
          0.000          28800.000    21       1         1

          NAME                VALUE
       CARTS                10000.000
       KASSA                10006.000
       MET1                     6.000
```

MET2	9.000
OPLATA	10004.000
OUT	12.000
POK_KASSA	10005.000
PRIL1	10001.000
PRIL2	10002.000
PRIL3	10003.000

LABEL	LOC	BLOCK TYPE	ENTRY COUNT	CURRENT COUNT	RETRY
	1	GENERATE	389	0	0
	2	ENTER	389	0	0
	3	TRANSFER	389	0	0
	4	ADVANCE	297	1	0
	5	ASSIGN	296	0	0
MET1	6	TRANSFER	388	0	0
	7	ADVANCE	231	2	0
	8	ASSIGN	229	0	0
MET2	9	TRANSFER	386	0	0
	10	ADVANCE	311	1	0
	11	ASSIGN	310	0	0
OUT	12	QUEUE	385	0	0
	13	ASSIGN	385	0	0
	14	SEIZE	385	0	0
	15	DEPART	385	0	0
	16	ADVANCE	385	1	0
	17	RELEASE	384	0	0
	18	LEAVE	384	0	0
	19	TERMINATE	384	0	0
	20	GENERATE	1	0	0
	21	TERMINATE	1	0	0

FACILITY	ENTRIES	UTIL.	AVE. TIME	AVAIL.	OWNER	PEND	INTER	RETRY	DELAY
KASSA	385	0.498	37.278	1	387	0	0	0	0

QUEUE	MAX	CONT.	ENTRY	ENTRY(0)	AVE.CONT.	AVE.TIME	AVE.(-0)	RETRY
KASSA	2	0	385	229	0.157	11.747	28.990	0

STORAGE	CAP.	REM.	MIN.	MAX.	ENTRIES	AVL.	AVE.C.	UTIL.	RETRY	DELAY
CARTS	1000	995	0	7	389	1	4.310	0.004	0	0

FEC	XN	PRI	BDT	ASSEM	CURRENT	NEXT	PARAMETER	VALUE
387	0		28817.420	387	16	17	1	11.290
391	0		28826.000	391	0	1		
386	0		28837.458	386	10	11	1	9.455
390	0		28848.729	390	4	5		
388	0		28855.832	388	7	8	1	2.199
389	0		28870.889	389	7	8	1	3.225
392	0		57600.000	392	0	20		

7. Обговорення. Протягом робочого дня до магазину ввійшло 389 покупців. 296 покупців зробили покупку і один покупець ще затримався біля першого прилавка, 229 покупців зробили покупку і ще двоє затрималися біля другого прилавка, 310 покупців зробили покупку і один покупець затримався біля третього прилавка. До каси підійшло 385 покупців. На кінець моделювання 384 покупця вже вийшли з магазину, один покупець обслуговується касиром. Касир був зайнятий майже 50 % робочого часу. У середньому на обслуговування одного покупця касир витрачав близько 37,3 с. Протягом усього періоду моделювання було взято одночасно сім кошиків, ресурс багатоканального пристрою використовувався на 0,4 %.

16.4. Порядок виконання роботи і варіанти завдань

16.4.1. Зміст звіту

У практичній частині роботи необхідно:
виконати приклади, що наведені в підрозділі 16.3;
виконати індивідуальне завдання пункту 16.4.2;
надати тексти складених програм і результати їх виконання.

16.4.2. Варіанти індивідуальних завдань

Кожен варіант має 4 завдання.

Варіант 1.

1.1. На одноканальний комутатор автоматизованої телефонної станції надходять замовлення на телефонні розмови кожні 1 – 3 хв. Середня тривалість однієї розмови – 2 хв. Комутатор час від часу виходить із ладу.

Середній час напрацювання на відказ дорівнює 1 000 хв. Для усунення несправності у середньому потрібно 100 хв. Якщо замовлення на розмову надійде в момент, коли комутатор несправний, то воно загубиться.

Побудувати GPSS-модель роботи станції протягом семи днів. Визначити коефіцієнт завантаження комутатора.

1.2. На склад по запасні частини приходять робітники. По запасні частини першого типу робітники приходять із періодичністю 10 ± 5 хв і обслуговуються протягом 7 ± 4 хв, по запасні частини другого типу робітники приходять кожні 7 ± 3 хв і обслуговуються 5 ± 2 хв. Оскільки важливішими є замовлення на запасні частини першого типу, то робітники, що прийшли по них, обслуговуються в першу чергу. По запасні частини третього типу робітники приходять двічі на годину й обслуговуються поза чергою.

Промоделювати роботу складу протягом 8-годинної робочої зміни, якщо на складі працює три комірники. Після 3 – 4 відпрацьованих годин дозволяється одна перерва на 30 – 40 хв.

1.3. На СТО працює три механіки, які виконують ремонт машин. Інтервали приїзду машин на ремонт розподілені за нормальним законом з параметрами (20, 5). Розподілення часу, необхідного на ремонт, показано в табл. 16.8.

Таблиця 16.8

Розподілення часу на ремонт

Вид ремонту	Частка машин, %	Час ремонту, хв
1	50	до 40
2	30	40 – 70
3	20	70 – 100

Промоделювати роботу СТО протягом 12 годин.

1.4. У художній майстерні працює два художника-портретиста й один фотограф. Час, потрібний художнику для малювання портрета, складає 1,5 ± 0,5 год. Вартість портрета складає 45 грн. Час, потрібний фотографу для проведення фотосесії, та її вартість, розподілені певним чином (табл. 16.9).

Таблиця 16.9

Розподілення часу для проведення фотосесії та її вартості

Частка замовлень, %	5	25	30	15	25
Кількість знімків	5	10	15	20	25
Час, хв	10	25	40	50	65
Вартість фотосесії, грн	35	60	75	90	110

Клієнти, які бажають замовити портрет, приходять щогодини, а якщо обидва художника зайняті, то залишають майстерню. Клієнти, які бажають замовити фотосесію, приходять кожні 40 ± 10 хв і чекають на обслуговування тільки в тому випадку, якщо в черзі не більше 5 осіб.

Промоделювати роботу художньої майстерні. Оцінити час, необхідний для обслуговування 10 клієнтів.

Варіант 2.

2.1. У ремонтний підрозділ з одним каналом обслуговування надходять засоби зв'язку, що вийшли з ладу та потребують поточного ремонту. Інтервали часу надходження несправних засобів зв'язку розподілені рівномірно в інтервалі 18 ± 6 год. Час ремонту теж розподілений рівномірно в інтервалі 16 ± 4 год. Ремонт виконується в міру надходження: "першим надійшло – першим відремонтовано".

Необхідно промоделювати функціонування ремонтного підрозділу протягом 10 діб, зібравши статистику за чергою. Урахувати можливість технічної перерви на дві години наприкінці кожної доби.

2.2. Троє викладачів проводять консультації для студентів. Вид консультацій, час приходу студентів і час консультації наведені в табл. 16.10.

Таблиця 16.10

Розподілення часу приходу студентів та часу консультації

Вид консультації	Час приходу студентів, хв	Час консультації, хв
З теоретичного матеріалу	15 ± 10	15 ± 5
З практичної роботи	20 ± 5	15 ± 10

Необхідно промоделювати роботу викладачів. Оцінити час, необхідний для консультування 10 студентів.

2.3. У відділенні банку працює два вікна на прийом платежів. Розподілення видів платежу, частоти появи та тривалості обслуговування подані в табл. 16.11.

Таблиця 16.11

Розподілення частоти появи платежів та часу обслуговування

Вид платежу	Частота появи	Час обслуговування, хв
Комунальні платежі	0,6	10 – 20
За навчання	0,25	8 – 10
За Інтернет	0,15	5 – 7

Необхідно записати GPSS-модель роботи відділення банку, якщо час приходу клієнтів на оплату розподілений за експонентним законом із середнім часом 5 хв.

2.4. У відділенні лікарні працює два лікарі. До першого лікаря приходять пацієнти трьох типів: звичайні відвідувачі, інваліди та люди похилого віку. Люди похилого віку обслуговуються поза чергою, а інваліди приєднуються до загальної черги. Час приходу звичайних відвідувачів 20 ± 2 хв, інвалідів – 30 ± 10 хв, а людей похилого віку – 40 ± 15 хв. На прийом одного пацієнта лікар витрачає 15 хв незалежно від категорії. Після огляду 20 % пацієнтів направляється на додаткове обстеження іншим лікарем, де пацієнти затримуються з імовірністю 0,5 на 30 хв, з імовірністю 0,3 на 40 хв, та з імовірністю 0,2 на 55 хв. Останні пацієнти покидають лікарню.

Необхідно записати GPSS-модель роботи двох лікарів протягом 8-годинного робочого дня.

Варіант 3.

3.1. На СТО працює механік, який може виконати три види ремонту. Розподіл інтервалів приїзду машин на ремонт і часу обслуговування (ремонту) наведені в табл. 16.12.

Таблиця 16.12

Розподілення інтервалів приїзду машин та часу ремонту

Вид ремонту	Інтервали приїзду машин	Час ремонту, хв
1	40 ± 10	30 ± 20
2	60 ± 15	50 ± 10
3	240 ± 60	110 ± 20

Необхідно промоделювати роботу СТО протягом доби, врахувавши можливість трьох перерв (по 1 год після кожних 7 – 8 відпрацьованих годин. Проаналізуйте отримані результати.

3.2. На СТО працює два механіки, кожен з яких може виконати два види ремонту (1-й і 2-й). Є ще один механік, який виконує 3-й вид ремонту. Інтервали приїзду машин на ремонт та час обслуговування (ремонту) розподілені, як показано в табл. 16.13.

Таблиця 16.13

Розподілення інтервалів приїзду машин та часу ремонту

Вид ремонту	Інтервали приїзду машин	Час ремонту, хв
1	40 ± 10	30 ± 20
2	50 ± 10	50 ± 10
3	150 ± 60	90 ± 20

Необхідно промоделювати роботу СТО протягом 16 год. Урахувати можливість двох перерв після кожних шести відпрацьованих год.

3.3. На пакування надходять дитячі іграшки. Час надходження рівномірно розподілений на інтервалі від 1 до 3 хв. Іграшки пакуються на чотирьох робочих місцях. Розподілення час пакування подано в табл. 16.14.

Таблиця 16.14

Розподілення часу пакування

Частка іграшок, %	20	30	15	35
Час пакування, хв	5	7	10	12

Після пакування іграшки передаються на склад.

Необхідно записати GPSS-модель роботи пакувальників, підраховувати кількість упакованих іграшок.

3.4. На обчислювальну систему, що містить три процесори, надходить експонентний потік завдань із середнім інтервалом надходження 140 одиниць модельного часу. Кожне завдання з імовірністю 0,2 належить до одного з п'яти видів: 1, 2, 3, 4 або 5. Середній час обслуговування завдань кожного виду складає, відповідно 90, 100, 110, 120 і 130 одиниць модельного часу. Необхідно побудувати GPSS-модель, що дозволяє оцінити середні значення часу очікування завдань кожного типу.

Варіант 4.

4.1. У магазині працює два продавці: один у відділі хлібобулочних виробів, а другий – у відділі молочних продуктів. По хлібобулочні вироби покупці приходять із періодичністю 5 ± 3 хв, по молокопродукти – кожні 10 хв. Час обслуговування покупців першим продавцем – 5 ± 2 хв, а другим – 7 ± 2 хв. Протягом години приходить три покупці, які купують спочатку хліб, а потім і молочні продукти.

Необхідно промоделювати роботу магазину протягом доби. Перерви в роботі допустимі на 10 – 20 хв, кожні 4 ± 0.5 години, але не більше п'яти перерв.

4.2. У швейному цеху працює два кравці та чотири швачки. У цеху виготовляються жіночі костюми-трійки. Кравець повністю розкроює костюм, швачка повністю шиє його. Вихідні дані задачі наведено в табл. 16.15.

Таблиця 16.15

Розподілення часу на виконання роботи

Вихідні дані	Час на виконання роботи, хв	
	розкрій	Пошиття
Спідниця	10 ± 3	30 ± 10
Брюки	15 ± 5	60 ± 5
Жакет	25 ± 10	120 ± 15

Необхідно записати GPSS-модель роботи швейного цеху протягом 10-годинного робочого дня, визначити, скільки костюмів буде пошито.

4.3. Вхідний потік деталей до цеху описується нормальним законом з параметрами (7, 3). Деталі обробляються групою верстатів (20 шт.). Тривалість обробки групою верстатів розподілена так, як показано в табл. 16.16.

Таблиця 16.16

Розподілення тривалості обробки

Відносна частота, %	15	43	31	11
Тривалість обробки, хв	4	5	2	1

Після обробки деталі надходять до відділу технічного контролю, де працює один робочий. Тривалість контролю – 1 ± 0.5 хв.

Необхідно побудувати GPSS-модель обробки деталей у цеху протягом 8-годинної зміни зі збиранням статистики за чергами, визначити характеристики роботи наведеної системи масового обслуговування.

4.4. У перукарні є два зали – чоловічий і жіночий. Вихідні дані наведені в табл. 16.17. Вважатимемо, що всі клієнти бажають підстригтися, 45 % чоловіків бажають поголитися, а 30 % жінок – пофарбувати волосся.

Таблиця 16.17

Розподілення часу приходу клієнтів та вартості послуг

Вихідні дані	Чоловіки		Жінки	
	Час, хв	Вартість, грн	Час, хв	Вартість, грн
Прихід клієнтів	50 ± 10	–	40 ± 15	–
Стрижка	20 ± 5	Від 7 до 15	40 ± 10	Від 10 до 30
Гоління	15 ± 4	Від 5 до 8	–	–
Фарбування волосся	–	–	40 ± 15	Від 5 до 30

Необхідно промоделювати роботу перукарні протягом 8 годин зі збиранням статистики за чергами. Визначте, які послуги та в якій кількості були надані протягом робочого дня.

Варіант 5.

5.1. У швейному цеху працює один кравець і одна швачка. У цеху виготовляються жіночі костюми-трійки. Кравець повністю розкроює костюм, швачка повністю шиє його. Вихідні дані задачі наведені в табл. 16.18.

Таблиця 16.18

Розподілення часу на виконання роботи

Вихідні дані	Час на виконання роботи, хв	
	Розкрій	Пошиття
Спідниця	10 ± 3	30 ± 10
Брюки	15 ± 5	60 ± 5
Жакет	25 ± 10	120 ± 15

Необхідно записати GPSS-модель роботи швейного цеху, оцінити час, потрібний для пошиття 12 костюмів.

5.2. У масажний кабінет, в якому працює три масажисти, приходять клієнти двох типів. Клієнти першого типу приходять кожні 15 – 45 хв на профілактичний масаж, який триває 30 – 40 хв. Клієнти другого типу приходять з періодичністю 10 – 20 хв і обслуговуються у масажиста протягом 20 – 30 хв. Після масажу клієнти плавають протягом 30 хв у басейні та залишають салон.

Необхідно промоделювати роботу масажного кабінету протягом 6-годинного робочого дня. Слід вважати, що в басейні одночасно може знаходитись не більше 10 осіб.

5.3. Супермаркет працює 14 годин на добу. Час приходу покупців описується нормальним законом розподілу з параметрами (2; 0,25). Спочатку покупці обирають товар протягом 15 – 20 хв, а потім ідуть до каси. Час обслуговування біля каси залежить від кількості обраних товарів (2 с на одиницю товару). Розподіл кількості покупок показано в табл. 16.19.

Таблиця 16.19

Розподілення кількості покупок

Частка покупців, %	20	30	50
Кількість покупок, од.	від 2 до 7	від 7 до 10	від 10 до 20

Необхідно промоделювати роботу супермаркету протягом п'яти днів, якщо в ньому працює сім кас.

5.4. Потрібно змоделювати розв'язання завдань двопроцесорним комп'ютером зі спільною пам'яттю, розділеною на вісім блоків. Кожному завданню відводиться для його розв'язання один блок. Інтервали часу між надходженнями завдань розподілені рівномірно в інтервалі [2; 14] одиниць часу, час обробки завдання підпорядкований експонентному закону з інтенсивністю v1 = 5 у процесорі CPU1 і з v2 = 2 у процесорі CPU2.

Після обробки чергового завдання з імовірністю 0,6 можливе звернення до зовнішньої пам'яті, в якій час обслуговування розподілено рівномірно в діапазоні [2; 8]. З імовірністю 0,4 завдання є розв'язаними та залишають систему.

Моделювання слід виконати на відрізку часу, що відповідає розв'язанню ста завдань.

Варіант 6.

6.1. На автовокзалі працюють дві каси. У першій касі продаються квитки на місцеві автобуси, а в другій – на автобуси дальнього сполучення. Час обслуговування у першій касі – 2 хв, а в другій – 4 хв. До першої каси пасажири підходять кожні 3 – 5 хв, а до другої – кожні 7 хв. У другу касу перший клієнт прийде тільки об 11:00. Робочий день починається о 8:00.

Перша каса кожні 3 год, а друга – кожні 2 год зачиняються на технічну перерву (15 хв).

Необхідно промоделювати роботу кас автовокзалу протягом 12 годин робочого дня, зібравши статистику за чергами.

6.2. Обчислювальна система включає три ЕОМ. У систему в середньому через 30 с надходять завдання, які потрапляють в чергу на обробку до будь-якої з ЕОМ, де вони обробляються близько 30 с у першому режимі і 14 ± 5 с – у другому режимі.

Необхідно змоделювати 4 год роботи системи, визначити необхідну місткість накопичувачів перед ЕОМ і коефіцієнти завантаження системи.

6.3. Вхідний потік пацієнтів до терапевта має пуасонівський характер з інтенсивністю 6 осіб у годину. Тривалість терапевтичного огляду розподілена так, як показано в табл. 16.20.

Таблиця 16.20

Розподілення часу терапевтичного огляду

Відносна частота, %	15	43	31	11
Тривалість огляду, хв	10	12	20	15

Після огляду терапевт робить запис про стан здоров'я пацієнта. Тривалість запису – 5 ± 2 хв.

Необхідно побудувати GPSS-модель терапевтичного огляду протягом 8-годинної зміни зі збиранням статистики за чергою та визначити характеристики роботи наведеної системи масового обслуговування.

6.4. Обчислювальна система складається з трьох комп'ютерів. З інтервалом 3 ± 1 хв у систему надходять завдання, які з імовірністю P1 = 0,4, P2 = P3 = 0,3 адресуються одному з трьох комп'ютерів. Перед кожним комп'ютером є черга завдань, довжина якої не обмежена. Після обробки на першому комп'ютері з імовірністю P12 = 0,3 завдання надходить у чергу до другого комп'ютера та з імовірністю P13 = 0,7 – до третього. Після обробки на другому або третьому комп'ютері завдання вважається виконаним. Тривалість обробки завдань на різних комп'ютерах характеризується інтервалом часу: T1 = 7 ± 4 хв, T2 = 3 ± 1 хв, T3 = 5 ± 2 хв.

Необхідно змоделювати процес обробки двохсот завдань, визначити максимальну довжину кожної черги та коефіцієнти завантаження комп'ютерів.

Варіант 7.

7.1. Робочий день перукарні починається о 8:00 ранку. До перукаря приходять клієнти за послугами двох типів: перші – підстригтися, другі – пофарбувати волосся. Обслуговування першого типу клієнтів продовжується 20 ± 5 хв, а другого – 30 ± 10 хв. Клієнти першого типу приходять кожні 20 – 40 хв, клієнти другого типу – кожні 40 ± 10 хв, причому перший клієнт цього типу прийде тільки об 11:00 ранку.

Необхідно промоделювати роботу перукарні, якщо робочий день триває до 19:00, а перерва в роботі – з 13:00 до 14:00, і зібрати статистику за чергою.

7.2. Морський порт має два причали – один на дев'ять місць, а другий – на шість. Кожні 5 – 20 год прибувають малі судна, які швартуються біля другого причалу і за своєю довжиною займають два місця на період 2 – 10 год. Кожні 30 год прибувають великі судна, які швартуються біля першого причалу і за своєю довжиною займають три місця на 12 – 18 год.

Необхідно промоделювати роботу порту протягом трьох місяців за умови цілодобової роботи.

7.3. Вхідний потік покупців до магазину має рівномірний характер на інтервалі 5 – 30 хв. У магазині працюють три прилавки, де можна здійснити рівнозначні покупки. Тривалість обслуговування біля будь-якого з трьох прилавків розподілена експонентному закону із середнім значенням, що залежить від кількості покупок (табл. 16.21).

Таблиця 16.21

Розподілення тривалості обслуговування

Кількість покупок	4	2	5	1
Середня тривалість обслуговування, хв	8	2	9	1

Кількість покупок розподілена так, як показано у табл. 16.22.

Таблиця 16.22

Розподілення кількості покупок

Відносна частота, %	15	40	25	20
Кількість покупок	4	2	5	1

Після оплати покупці залишають магазин.

Необхідно промоделювати роботу магазину протягом 10-годинного робочого дня та проаналізувати отримані результати.

7.4. Абітурієнти приходять у приймальну комісію університету. Прийом документів на кожну спеціальність обслуговує один секретар. Процедура прийому документів така:

1) фотографування – 2 хв;

2) написання заяви – 10 хв;

3) передавання документів та їх перевірка секретарем – 15 – 25 хв.

Фотографування проводиться в окремому кабінеті, куди приходять усі абітурієнти незалежно від того, на який факультет вони подають документи. Написання заяви, передавання документів та їх перевірка відбуваються у секретаря.

Час приходу абітурієнтів розподілений за пуасонівським законом розподілу із середнім часом 30 с, причому тільки 25 % від усіх абітурієнтів бажають подати документи на факультет економічної інформатики на одну з трьох спеціальностей: економічна кібернетика (ЕК), економічна статистика (ЕС), інформаційні управляючі системи та технології (ІС) факультету економічної інформатики. 75 % подає документи на інші факультети. З тих, хто подає документи на факультет економічної інформатики, 35 % йдуть до секретаря ЕК, 25 % – до секретаря ЕС, 40 % – до секретаря ІС.

Необхідно промоделювати роботу приймальної комісії протягом 9-годинного робочого дня та визначити:

а) скільки абітурієнтів подасть заяви на кожну спеціальність;

б) яке навантаження припадає на кожного секретаря;

в) скільки абітурієнтів встигли сфотографуватися.

Варіант 8.

8.1. На склад по товар приходять продавці. По товар першого виду продавці приходять з періодичністю 20 ± 15 хв і обслуговуються протягом 25 хв, по товар другого виду продавці приходять двічі на годину і обслуговуються 25 ± 10 хв. Оскільки важливішими є замовлення на товар другого типу, то продавці, що прийшли по нього, обслуговуються в першу чергу.

Необхідно промоделювати роботу складу протягом 10-годинного робочого дня, якщо на складі працює один комірник.

8.2. У морському порту є два причали: старий і новий. Біля старого причалу одночасно можуть швартуватися два судна. Тут працюють два портові крани, що проводять розвантаження – вантаження судна за 40 ± 10 год. Біля нового причалу є місце для п'яти судів. Тут працюють три крани, що проводять розвантаження – вантаження за 20 ± 5 год. Судна прибувають в акваторію порту кожні 7 ± 3 год. А кожні 9 ± 3 год. прибувають судна, що мають пріоритет в обслуговуванні. В очікуванні

місця біля причалу судно кидає якір на рейді. Для швартовки та відходу судна від причалу треба по 1 год. часу. Суднам, що мають пріоритет в обслуговуванні, місце біля причалу надається в першу чергу. Розвантаження – вантаження судна завжди проводить один кран.

Необхідно змоделювати процес початку навігації в морському порту за умови, що в акваторію порту зайшли 150 суден; підрахувати число суден, обслугованих на кожному причалі, та зафіксувати їх максимальну кількість на рейді; визначити середній час очікування місця біля причалу окремо для суден, що мають і не мають пріоритету в обслуговуванні, а також коефіцієнти завантаження портових кранів.

8.3. На склад по запасні частини приходять робітники. По запасні частини першого типу вони приходять з інтервалом часу, рівномірно розподіленим на інтервалі 7 – 15 хв. По запасні частини другого типу – кожні 7 ± 3 хв. Оскільки важливішими є замовлення на запасні частини першого типу, то робітники, що прийшли по них, обслуговуються в першу чергу.

Розподілення часу обслуговування робітників, що прийшли по деталі першого типу, показано в табл. 16.23.

Таблиця 16.23

Розподілення тривалості обслуговування

Відносна частота, %	15	20	5	60
Тривалість обслуговування, хв	3	5	2	4

Час обслуговування робітників, що прийшли по деталі другого типу, складає 5 ± 2 хв.

Необхідно промоделювати роботу складу протягом 8-годинної робочої зміни, якщо на складі працює три комірники та зібрати статистику за чергами.

8.4. На автомийку, де надаються три послуги, приїжджають автомобілі. Час їх прибуття рівномірно розподілений на інтервалі 15 – 25 хв. Відправні дані подано в табл. 16.24.

Таблиця 16.24

Розподілення тривалості обслуговування та вартості послуг

Послуга	Мийка кузова	Полірування кузова	Чистка салону
Час обслуговування, хв.	20	30 ±10	40 ±15
Вартість послуги, грн	Від 25 до 35	Від 45 до 100	Від 30 до 70
Частка авто, що користуються даною послугою, % від усіх авто, що приїхали на автомийку	90	70	55

Необхідно: промоделювати роботу автомийки протягом 10-годинного робочого дня, зібравши статистику за чергами на кожну послугу; обчислити, в якому обсязі була надана кожна з послуг (кількість автомашин, що скористалися послугою).

Варіант 9.

9.1. На пристрій автоматизованої обробки інформації надходять заявки кожні 1 – 2 хв. Середня тривалість обробки – 2 хв. Пристрій час від часу виходить із ладу. Середній час напрацювання на відмову дорівнює 1 000 хв. Для усунення несправності в середньому потрібно 100 хв.

Необхідно побудувати модель роботи пристрою протягом 96 годин.

9.2. У ремонтний підрозділ з двома каналами обслуговування надходить техніка, що вийшла з ладу та потребує поточного ремонту. Інтервали часу надходження несправної техніки розподілені рівномірно в інтервалі 5 ± 3 години. Час ремонту теж розподілений рівномірно в інтервалі 9 ± 2 години. Ремонт виконується в міру надходження: першим надійшло – першим відремонтовано.

Необхідно промоделювати функціонування ремонтного підрозділу протягом 15 діб, причому кожні дві доби робиться технічна перерва на 4 години.

9.3. Державна екзаменаційна комісія приймає захист дипломних робіт. Час презентування дипломної роботи розподілений за нормальним законом розподілу з параметрами (10; 2). Після доповіді студент отримує ряд додаткових запитань, відповідь на кожне з яких триватиме 2,5 хв.

Кількість додаткових запитань та ймовірність, з якою вони будуть задані, подано в табл. 16.25.

Таблиця 16.25

Розподілення кількості запитань

Імовірність	0,2	0,3	0,3	0,2
Кількість запитань	3	5	8	12

Необхідно промоделювати роботу ДЕК протягом п'яти годин, відведених на захист дипломних робіт. Слід вважати, що на захист прийшли 10 студентів одночасно до початку роботи ДЕК. Зробіть висновки щодо проведених розрахунків.

9.4. Розподілений банк даних системи збирання інформації організований на базі комп'ютерів, з'єднаних дуплексним каналом зв'язку. Запит, що надходить, обробляється на першому комп'ютері; з імовірністю 50 % необхідна інформація виявляється на місці. В іншому випадку необхідна переадресація запиту до другого комп'ютера. Запити надходять через 10 ± 3 с, первинна обробка запиту триває 2 с, видача відповіді вимагає 18 ± 2 с, передавання каналом зв'язку – 3 с. Часові характеристики другого комп'ютера аналогічні першій.

Необхідно: змоделювати проходження чотирьохсот запитів; визначити необхідну місткість накопичувачів перед комп'ютерами, що забезпечує безвідмовну роботу системи.

Варіант 10.

10.1. Два суднохідні канали з'єднані між собою шлюзом, який дозволяє переправляти судна з одного каналу в інший. Час заповнення шлюзу водою складає 25 хв, час переміщення судна в шлюзі до іншого каналу – 20 хв, а час спуску води зі шлюзу – 10 хв. Шлюз працює в одному напрямку, тому для повернення шлюзу у вихідне положення відведено 10 хв.

Необхідно: промоделювати роботу шлюзу, якщо судна прибувають кожні 50 ± 10 хв; оцінити час для переправлення в другий канал п'ятнадцяти суден; зробити висновки та внести пропозиції щодо покращення роботи системи.

10.2. На обчислювальну систему, що містить три процесори, надходить потік завдань із середнім інтервалом надходження 140 одиниць модельного часу. Завдання обслуговується одним із процесорів протягом 90 – 130 одиниць модельного часу.

Необхідно побудувати модель, що дозволяє оцінити середні значення часу очікування завдань на обслуговування.

10.3. Екзаменатор приймає екзамен у студентів. Інтенсивність приходу студентів на екзамен – 10 осіб за годину. Для підготовки до відповіді студент має 20 ± 5 хв. Після відповіді на основні запитання студент отримує ряд додаткових запитань, тривалість відповіді на які розподілена таким чином, як показано в табл. 16.26.

Таблиця 16.26

Розподілення тривалості відповіді

Імовірність	0,2	0,3	0,3	0,2
Тривалість відповіді, хв	3	5	6	4

Необхідно: промоделювати роботу екзаменатора протягом 5 год; визначити, скільки студентів встигне скласти екзамен; зробити висновки; внести пропозиції щодо покращення роботи модельованої системи.

10.4. Час надходження деталей до цеху розподілений рівномірно на інтервалі від 3 до 8 хв. 40 % деталей оброблюється першою групою з п'яти верстатів (тривалість обробки – 7 ± 3 хв). Ця група дає 3 % браку, що визначається контролером (2 ± 1 хв). Усі браковані деталі повторно оброблюються на своїй групі верстатів. 60 % деталей оброблюється другою групою з трьох верстатів, причому тривалість обробки залежить від поточної довжини черги (табл. 16.27).

Таблиця 16.27

Розподілення тривалості обробки

Довжина черги	Тривалість обробки, хв
≤ 12	7
> 13	5

Після обробки деталі залишають цех.

Необхідно побудувати GPSS-модель функціонування системи протягом восьмигодинного робочого дня та зробити висновки.

16.5. Контрольні запитання

1. Дайте визначення поняттю "система".
2. Дайте визначення поняттю "модель".
3. Які системи називають системами масового обслуговування?
4. З яких елементів складається модель GPSS?
5. Який закон розподілу випадкової величини називають нормальним?
6. Який закон розподілу випадкової величини називають рівномірним?
7. Який закон розподілу випадкової величини називають пуасонівським?

Рекомендована література

Основна

1. Бахвалов Н. С. Численные методы / Н. С. Бахвалов, Н. П. Жидков, Г. М. Кобельков. – Москва : Бином, 2007. – 636 с.

2. Задачин В. М. Чисельні методи : навч. посіб. / В. М. Задачин, І. Г. Конюшенко. – Харків : Вид. ХНЕУ, 2014. – 190 с.

3. Рыжиков Ю. И. Имитационное моделирование. Теория и технологии / Ю. И. Рыжиков. – Москва : Альтекс-А, 2004. – 384 с.

4. Сухарев А. Г. Курс методов оптимизации / А. Г. Сухарев, А. В. Тимохов, В. В. Федоров. – Москва : Наука, 1986. – 328 с.

5. Томашевський В. М. Моделювання систем / В. М. Томашевський. – Київ : Видавнича група BHV, 2005. – 349 с.

6. Фельдман Л. П. Чисельні методи в інформатиці / Л. П. Фельдман, А. І. Петренко, О. А. Дмитрієва. – Київ : Видавнича група BHV. – 2006. – 480 с.

Додаткова

7. Амосов А. А. Вычислительные методы для инженеров : учеб. пособ. / А. А. Амосов, Ю. А. Дубинський, Н. В. Копченова. – Москва : Высш. шк., 1994. – 544 с.

8. Гультяев А. MATLAB 5.2. Имитационное моделирование в среде Windows : практ. пособ. / А. Гультяев. – Санкт-Петербург : КОРОНА принт, 1999. – 288 с.

9. Дэннис Дж. Численные методы безусловной оптимизации и решения нелинейных уравнений : пер. с англ. / Дж. Дэннис, Р. Шнабель. – Москва : Мир, 1988. – 440 с.

10. Кравченко П. П. Имитационное моделирование вычислительных систем средствами GPSS/PC / П. П. Кравченко, Н. Ш. Хусаинов. – Таганрог : ТРТУ, 2000. – 116 с.

11. Рыжиков Ю. И. Теория очередей и управления запасами : учеб. пособ. / Ю. И. Рыжиков. – Санкт-Петербург : Питер, 2001. – 376 с.

12. Сытник В. Ф. Имитационное моделирование : учеб.-метод. пособ. / В. Ф. Сытник, Н. С. Орленко. – Киев : КНЕУ, 1999. – 208 с.

13. Томашевский В. Н. Решение практических задач методами компьютерного моделирования / В. Н. Томашевский, Е. Г. Жданова, А. А. Жолдаков. – Киев : Изд-во "Корнійчук", 2001. – 268 с.

14. Томашевский В. Н. Имитационное моделирование в среде GPSS / В. Н. Томашевский, Е. Г. Жданова. – Москва : Бестселлер, 2003. – 416 с.

15. Шикин Е. В. Кривые и поверхности на экране компьютера. Руководство по сплайнам для пользователей / Е. В. Шикин, А. И. Плис. – Москва : ДИАЛОГ – МИФИ, 1996. – 242 с.

16. Zadachyn V. Calculation of optimal path for parallel car parking / V. Zadachyn, O. Dorokhov // Transport and Telecommunication. – Vol. 13. – 2012. – P. 303–309.

Інформаційні ресурси

17. Сайт персональних навчальних систем ХНЕУ ім. С. Кузнеця. – Режим доступу :https://pns.hneu.edu.ua.

18. Website Quick-R. – Access mode : http://www.statmethods.net/index.html.

19. Website Statistics with R. – Access mode : http://zoonek2.free.fr/UNIX/48_R/all.html.

20. Website The Comprehensive R Archive Network. – Access mode : http://cran.r-project.org.

Зміст

Загальні положення 3
Змістовий модуль 1. Чисельні методи 6
Лабораторна робота 1. Вступ у систему R 6
Лабораторна робота 2. Чисельні методи розв'язання систем лінійних алгебраїчних рівнянь 18
Лабораторна робота 3. Чисельні методи розв'язання нелінійних рівнянь з одним невідомим і систем нелінійних рівнянь 27
Лабораторна робота 4. Чисельні методи наближення функцій. Апроксимація та інтерполяція функцій 40
Лабораторна робота 5. Чисельні методи розв'язання задачі Коші для звичайних диференціальних рівнянь 52
Лабораторна робота 6. Чисельні методи розв'язання лінійної крайової задачі для звичайних диференціальних рівнянь 2-го порядку .. 61
Змістовий модуль 2. Методи оптимізації 68
Лабораторна робота 7. Чисельні методи розв'язання задач одномірної оптимізації та задач безумовної оптимізації 68
Лабораторна робота 8. Чисельні методи розв'язання задач нелінійного програмування 82
Лабораторна робота 9. Розв'язання задач лінійного програмування ... 89
Змістовий модуль 3. Моделювання систем 99
Лабораторна робота 10. Вивчення можливостей математичного пакета R 99
Лабораторна робота 11. Обробка й аналіз даних експерименту. Підбір параметрів розподілу 110
Лабораторна робота 12. Ідентифікація параметрів моделей методом найменших квадратів 119
Лабораторна робота 13. Побудова багатофакторної регресійної моделі . 127
Лабораторна робота 14. Дослідження імітаційних моделей 139
Лабораторна робота 15. Дослідження моделей, заснованих на диференціальних рівняннях (D-схемах) 152
Лабораторна робота 16. Середовище імітаційного моделювання *GPSS World.* Моделювання найпростіших та багатоканальних систем масового обслуговування 157
Рекомендована література 215
Основна 215
Додаткова 215
Інформаційні ресурси 216

НАВЧАЛЬНЕ ВИДАННЯ

МОДЕЛЮВАННЯ СИСТЕМ ТА МЕТОДИ ОПТИМІЗАЦІЙ

Методичні рекомендації до лабораторних робіт для студентів галузі знань 12 "Інформаційні технології" першого (бакалаврського) рівня

Самостійне електронне текстове мережеве видання

Укладач **Задачин** Віктор Михайлович

Відповідальний за видання *О. Г. Руденко*

Редактор *О. В. Анацька*

Коректор *О. В. Анацька*

План 2019 р. Поз. № 83 ЕВ. Обсяг 218 с.

Видавець і виготовлювач – ХНЕУ ім. С. Кузнеця, 61166, м. Харків, просп. Науки, 9-А

Свідоцтво про внесення суб'єкта видавничої справи до Державного реєстру ***ДК № 4853 від 20.02.2015 р.***