3. Лабораторна робота. Застосування системи комп'ютерної алгебри Maxima до розв'язання задач лінійної алгебри

Мета: засвоїти можливості роботи СКА Maxima до розв'язання задач лінійної алгебри, функції для роботи з матрицями, матричні операції та роботу з масивами.

Теоретичні відомості та методичні рекомендації

У СКА Maxima є вбудовані засоби для роботи з матрицями та векторами, які дозволяють виконувати як базові операції, так і лінійну алгебру (визначники, обернені матриці, власні значення). Для створення матриці з елементами, які задані у вигляді списку `list`, у Maxima використовується функція `matrix(list)` (див. рис. 3.1):

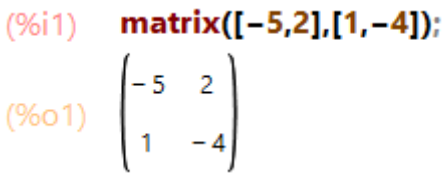

Рисунок 3.1 – Приклад форми виклику функції `matrix()`

Вектор задається як список (див. рис. 3.2):

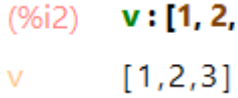

Рисунок 3.2 – Завдання вектору у СКА Maxima

Нехай задано дві матриці A і B. Базові операції над матрицями наступні:

1) $A + B$ – додавання;
2) $A - B$ – віднімання;
3) $A.B$ – матричне множення (див. рис. 3.3);
4) $k * A$ – множення числа на матрицю (див. рис. 3.3);
5) $A\text{^^} - 1$ – обчислення оберненої матриці до матриці A;
6) функція `invert(A)` – другий спосіб обчислення оберненої матриці до матриці A (див. рис. 3.4);
7) функція `transpose(A)` – транспонування матриці A;
8) функція `determinant(A)` – обчислення визначника матриці A;
9) функція `eigenvalues(A)` – обчислення власних значень матриці A (див. рис. 3.5);
10) функція `minor(A)` – обчислення мінору матриці A;
11) функція `rank(A)` – обчислення рангу матриці A;
12) функція `submatrix()` використовується для створення нової матриці з існуючої, вилучаючи вказані рядки та стовпці;
13) функція `echelon()` перетворює матрицю у форму рядків, що ступінчасто зменшуються, подібно до методу Гауса.

Щоб використовувати розширену лінійну алгебру, треба підключити модуль `load("linearalgebra")`. Щоб отримати у відповіді наближені значення, використовуйте команду `float(…)`.

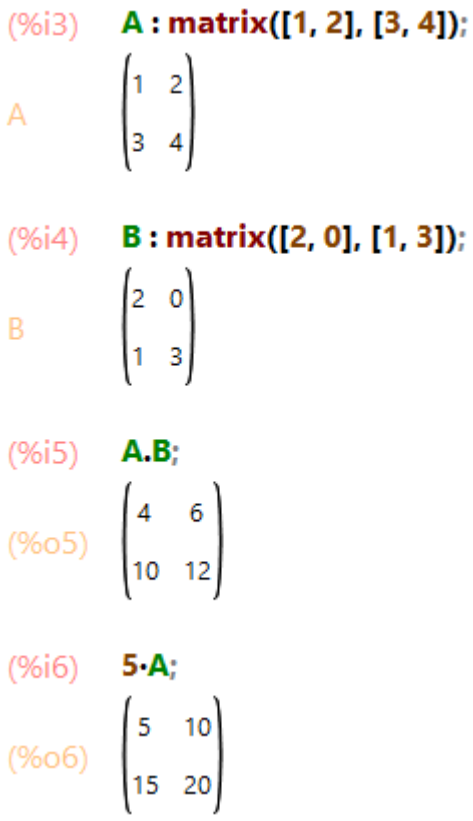

Рисунок 3.3 – Матричне множення і множення числа на матрицю

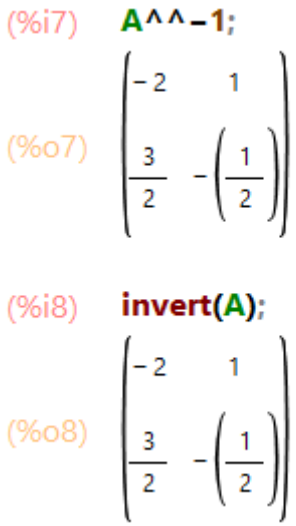

Рисунок 3.4 – Обчислення оберненої матриці

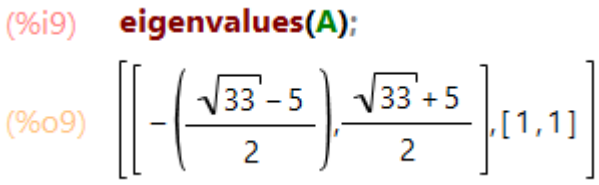

Рисунок 3.5 – Обчислення власних значень матриці

У СКА Maxima є три типу даних: списки, множини та масиви.

Список – це впорядкована сукупність довільних об’єктів, у тому числі, можливо, і самих списків. Список може бути пустим. Щоб створити список, необхідно через кому перерахувати всі об’єкти у квадратних дужках. До елементів списку можна звертатися після його створення, вказавши у квадратних дужках його порядковий номер (див. рис. 3.6):

```
(%i26) n:[2,4,6,1,3,5];
n      [2,4,6,1,3,5]

(%i27) n[6];
(%o27) 5
```

Рисунок 3.6 – Приклад списку та звертання до елемента списку

Список можна створювати за допомогою функції:

`makelist(form,i,a,b),`

де `form` – список, який залежить від параметру `i` (належить цілим числам); `i` – індекс; `a` та `b` – діапазон зміни параметру `I`(див. рис. 3.7).

```
(%i28) L:makelist(i^^2,i,1,5);
L      [1,4,9,16,25]
```

Рисунок 3.7 – Приклад форми виклику функції `makelist()`

До елементів масиву можна звертатися після його створення, вказавши у квадратних дужках його порядковий номер елемента, який називається індексом.

Масив – це впорядкована послідовність величин. Якщо виникає необхідність використовувати індексовану змінну `x[k]`, не визначаючи `x`, то `x` буде масивом у Maxima. Масив можна створювати за допомогою функції:

`array(name,n,k),`

де `name` – ім’я масиву; `n,k` (належать цілим числам) – розмірність масиву.

```
(%i29) array(a,1,1);
(%o29) a

(%i33) a[0,0]:-1;a[0,1]:3;a[1,0]:5;a[1,1]:-2;
(%o30) -1
(%o31) 3
(%o32) 5
(%o33) -2

(%i34) a[1,0];
(%o34) 5
```

Рисунок 3.8 – Приклад форми виклику функції `array()`

До елементів масиву можна звертатися після його створення, вказавши у квадратних дужках його порядковий номер елемента, який називається індексом (див. рис. 3.8).

У деяких випадках потрібно масив перетворити у список, для цього використовується функція `listarray()` (див. рис. 3.9):

```
(%i35) listarray(a);
(%o35) [-1,3,5,-2]
```

Рисунок 3.9 – Приклад форми виклику функції `listarray()`

Умовні оператори.

У Maxima умовні оператори дозволяють виконувати різні дії залежно від виконання певної умови. Вони використовуються як у виразах, так і у визначенні функцій.

Основна форма умовного оператора:

```
if cond_1 then expr_1 else expr_0.
```

Якщо умова `cond_1` виконується, то виконується вираз `expr_1`, інакше – виконується вираз `expr_0`. Пакет Maxima надає змогу використати різні форми оператора `if`, наприклад:

```
if cond_1 then expr_1 elseif cond_2 then expr_2 elseif
                    ... else expr_0
```

Якщо виконується умова `cond_1`, то виконується вираз `expr_1`, інакше – перевіряється умова `cond_2`, і якщо вона виконується – виконується вираз `expr_2`, тощо. Якщо жодна з умов не виконується – виконується вираз `expr_0`.

Альтернативні вирази `expr_1`, `expr_2`, ... , `expr_n` – довільні вирази Maxima (зокрема вкладені оператори `if`). Умови – дійсно або потенційно логічні вирази, що зводяться до значень `true` або `false`. Спосіб інтерпретації умов залежить від значення прапорця `prederror`. Якщо `prederror=true`, виводиться помилка, якщо значення якогось із виразів `cond_1`, ... , `cond_n`, відрізняється від `true` або `false`. Якщо `prederror=false` і значення якогось із виразів `cond_1`, ... , `cond_n`, відрізняється від `true` або `false`, результат обчислення `if` – умовний вираз.

Оператори циклу.

Для виконання ітерацій використовується оператор `for`. Можуть використовуватися три варіанти його виклику, що відрізняються умовою закінчення циклу:

– for variable: init_value step increment thru limit do body;
– for variable: init_value step increment while condition do body;
– for variable: init_value step increment unless condition do body.

Тут `variable` – змінна циклу; `init_value` – початкове значення; `increment` – крок (типово дорівнює 1); `limit` – кінцеве значення змінної циклу; `body` - оператори тіла циклу.

Ключові слова `thru`, `while`, `unless` вказують на спосіб завершення циклу:

– за досягнення змінною циклу значення `limit`;
– поки виконується умова `condition`;
– поки не буде досягнута умова `condition`.

Параметри `init_value`, `increment`, `limit`, і `body` можуть бути довільними виразами. Контрольна змінна по завершенні циклу має бути додатною (при цьому початкове значення може бути і від'ємним). Вирази `limit`, `increment`, умови завершення (`condition`) обчислюються на кожному кроці циклу, тому їхня складність впливає на час виконання циклу.

При нормальному завершенні циклу величина, що повертається – `done` (див. рис. 3.10). Примусовий вихід із циклу здійснюється за допомогою оператора `return`, що може повертати довільне значення.

```
(%i2)  s:0$ for i:1 while i <= 10 do s: s+i;
(%o2)  done

(%i3)  s;
(%o3)  55
```

Рисунок 3.10 – Приклад циклу у СКА Maxima

Контрольна змінна циклу – локальна усередині циклу, тому її зміна у циклі не впливає на контекст (навіть при наявності поза циклом змінної з тією ж назвою).

Наприклад, треба написати програму за допомогою операторів циклу, яка буде розкладати функцію $e^{\sin x}$ у ряд Маклорена до 7 порядку (див. рис. 3.11):

```
(%i13) series: 1$
       term: exp(sin(x))$

       for p:1 unless p > 7 do (
         term: diff (term, x)/p,
         series: series + subst (x=0, term)·x^p
       )$;

       series;
```

$$(\%o13)\quad \frac{x^7}{90} - \frac{x^6}{240} - \frac{x^5}{15} - \frac{x^4}{8} + \frac{x^2}{2} + x + 1$$

Рисунок 3.11 – Приклад розкладання функції у ряд Маклорена за допомогою оператору циклу

То й же самий результат можна отримати, використовуючи вбудовану команду `taylor`, яка автоматично будує ряд Тейлора (Маклорена) для будь-якої

функції. Вона дозволяє перевірити результат ручного алгоритму, який написано через цикл (див. рис. 3.12):

```
(%i14) taylor(exp(sin(x)), x, 0, 7);
```

$$(\%o14)/T/\quad 1+x+\frac{x^2}{2}-\frac{x^4}{8}-\frac{x^5}{15}-\frac{x^6}{240}+\frac{x^7}{90}+\ldots$$

Рисунок 3.12 – Приклад розкладання функції у ряд Маклорена через вбудовану команду `taylor`

Таким чином, система комп'ютерної алгебри Maxima є ефективним засобом для навчання та практичного розв'язання задач лінійної алгебри, дозволяючи поєднувати аналітичні обчислення з наочністю та автоматизацією процесів.

Завдання до лабораторної роботи

1. Для матриць

$$A=\begin{pmatrix}5 & 16-n & h-4\\ -2 & -h+n & n+3\\ 4 & 2h-n & 7\end{pmatrix},\ B=\begin{pmatrix}-10 & -n & -h+9\\ 11 & h & 17-n\\ 18 & n & -2\end{pmatrix},$$

де n – номер варіанту, h – остання цифра номеру вашої групи, знайти у СКА Maxima: 1) AB; 2) BA; 3) $nA-hB$; 4) $\det A$ та $\det B$; 5) A^{-1} та B^{-1}; 6) ранг матриці A, слід матриці B, власні значення матриць A та B; 7) привести матрицю A до трикутного вигляду, матрицю B до Жорданової форми.

2. Розв'язати матричне рівняння $AX=B$ трьома способами у СКА Maxima (методом Гауса, матричним методом та за формулами Крамера):

$$A=\begin{pmatrix}5 & 16-n & h-4\\ -2 & -h+n & n+3\\ 4 & 2h-n & 7\end{pmatrix},\ B=\begin{pmatrix}-10 & -n & -h+9\\ 11 & h & 17-n\\ 18 & n & -2\end{pmatrix},$$

де n – номер варіанту, h – остання цифра номеру вашої групи.

3. Знайти суму ряду у СКА Maxima, використовуючи списки:

1) $\sum_{i=0}^{10}\frac{1}{i}$.
2) $\sum_{i=-10}^{10}\frac{i!}{2i+1}$.
3) $\sum_{i=-10}^{10}(2i+1)$.
4) $\sum_{i=0}^{10}\frac{1}{k^i}$, для $k=10, k=15$.
5) $\sum_{i=0}^{5}k^{\frac{1}{i}}$, для $k=2, k=3$.
6) $\sum_{i=0}^{4}\cos^2(\pi i)$.
7) $\sum_{i=0}^{4}\left(2^i-1\right)$.
8) $\sum_{i=0}^{10}\frac{e^i}{i+1}$.
9) $\sum_{i=1}^{10}(5i^2-1)$.

10) $\sum_{i=0}^{5} \sin^2\left(\frac{\pi i}{3}\right)$.

4. Розв'язати задачу, використовуючи списки та масиви СКА Maxima:

1) Вивести значення функції $f(x) = x^2 \cos 3x\,, x \in [0; 2\pi]$ з шагом $h = 0{,}2\pi$.
2) Знайти суму елементів матриці 4×4.
3) У довільному списку, який містить 10 елементів, замінити всі елементи менші 5 на 0.
4) Знайти добуток мінімальних елементів кожного стовбця матриці 4×4.
5) У довільному списку, який містить 10 елементів, замінити всі елементи більші 3 на 1.
6) У довільному списку, який містить 10 елементів, знайти найменший.
7) Вивести значення функції $f(x) = \sqrt{x} \sin 2x\,, x \in [0; \pi]$ з шагом $h = 0{,}1\pi$.
8) У довільному списку, який містить 15 елементів, знайти найбільший.

Питання для самоконтролю

1. Охарактеризуйте можливості СКА Maxima для матричних обчислень.
2. Які Ви знаєте базові операції над матрицями?
3. Для чого потрібно підключати модуль `linearalgebra` у СКА Maxima?
4. Чи можна обернену матрицю обчислити наближено у СКА Maxima?
5. Які типи даних є у СКА Maxima?
6. Охарактеризуйте функцію `makelist()`.
7. Для чого використовується функція `array()` у СКА Maxima?
8. Розкажіть про умовні оператори у СКА Maxima.
9. Розкажіть про оператори циклу у СКА Maxima.