ЖИТОМИРСЬКА ПОЛІТЕХНІКА
ДЕРЖАВНИЙ УНІВЕРСИТЕТ

@ZTUEDUUA

Основи цифрової обробки зображень з вимірювальною інформацією

Лекція 16
Тема:Функції обробки даних в Матлаб

1. Функції обробки та їх застосування для векторів та матриць.
2. Поелементні операції з векторами та матрицями
3. Побудова таблиць значень функцій в Матлаб.
4. Висновки та рекомендації.

1. Функції обробки та їх застосування для векторів та матриць.

Matlab – система, спеціально призначена для проведення складних обчислень над векторами, матрицями та масивами – в томі числі багатовимірними. При цьому вона по замовчуванню вважає, що кожна задана змінна – це вектор, матриця або масив. Все визначається конкретним значенням змінної. Наприклад, якщо задано x=1, то це значить, що x – це вектор з єдиним елементом, що приймає значення 1. Якщо треба задати вектор із декількох елементів, то їх значення варто перечислити в квадратних дужках, відокремлюючи пробілами.

Функції обробки для векторів і матриць є ключовими для роботи з даними, особливо у програмуванні та чисельних методах. Основні напрямки які використовуються поділяються на:

Арифметичні операції: Додавання, віднімання, множення та ділення векторів і матриць.

Трансформації: Обертання, масштабування та віддзеркалення матриць, корисні у комп'ютерній графіці.

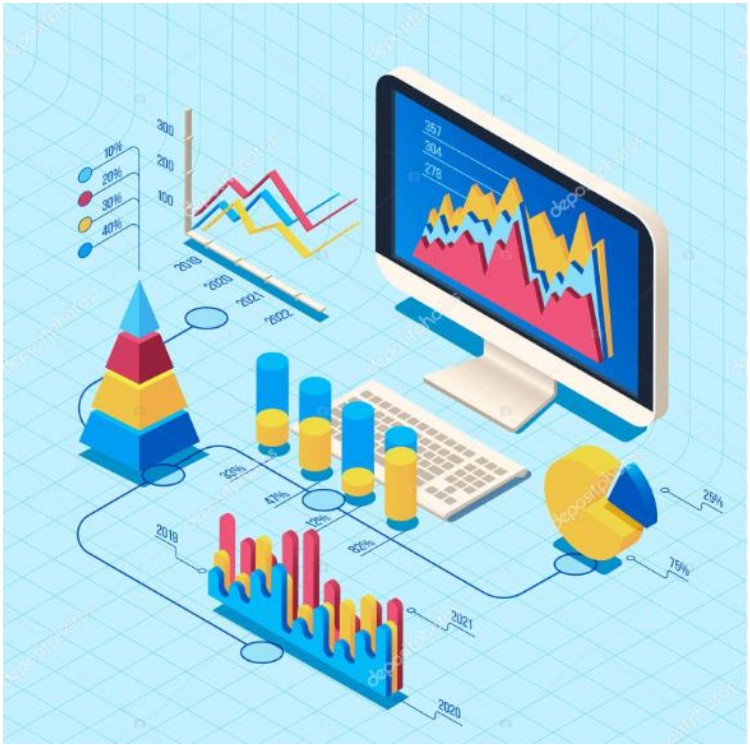
Розв'язок систем рівнянь: Використання матриць для знаходження невідомих у рівняннях.

Лінійна алгебра: Обчислення визначників, власних значень та власних векторів.

Статистичний аналіз: Знаходження середнього, медіани, коваріаційних матриць.

Оптимізація: Використання матричних перетворень для знаходження оптимальних рішень у задачах.

Обробка сигналів: Використання матриць у цифровій обробці зображень та звуків.



Приклади функції обробки векторів та матриць

Основним типом даних в є матриця, тобто двовимірний масив, який можна подати у вигляді таблиці з декількома рядками та стовпцями. Окремим випадком матриці є одновимірні масиви (вектор-рядок та вектор-стовпець) та скаляри, тобто одиночні дані.

При поелементному формуванні векторов і матриць їх елементи поміщають у квадратні дужки [ ], елементи одного рядка відокремлюють один від одного комами «,» або пробілами «␣«, а рядки – крапкою з комою «;» або переводом рядку клавішею Enter, наприклад,

```
» x=[2 3 –8]; % Вектор-рядок
» y=[5; 6.5; -2.23; 0]; % Вектор-стовпець
» A=[1,2,3; 4 5 6]; % Матриця 2*3
» B=[1 2; 3 4; 5 6]; % Матриця 3*2
```

Вектори-рядки з рівномірним розподіленням елементів можна утворювати двома способами.

1) Оператори

VarName = InitValue : Step : EndValue та VarName = InitValue : EndValue

створюють вектори-рядки за заданими значеннями першого InitValue та останнього EndValue елементів та різницею Step між двома сусідніми елементами. У другому випадку за замовчанням (default) Step=1. Наприклад,

```
» z=2:5
z = 2 3 4 5
» w=-1:0.5:1 w = -1.0000 -0.5000 0 0.5000 1.0000
```

Кількість елементів у рядку Number можна підрахувати за формулою:

Number = ((EndValue – InitValue) / Step)+1

Рядки з однаковою кількістю елементів можна поєднувати у матриці:

```
» c=[1:4; 2:5]
c = 1 2 3 4 2 3 4 5
```

2) Оператор

VarName = linspace (InitValue, EndValue, Number)

створюють рівномірно розподілені вектори-рядки за заданими значеннями першого **InitValue** і останнього **EndValue** елементів та їх кількістю **Number**. За замовчанням **Number = 100**. Оператори **linspace** зручно використовувати при завданні діапазону зміни аргументу при побудові графіків, наприклад,

```
>> a=linspace(3,7,10)
a = 3.0000 3.4444 3.8889 4.3333 4.7778 5.2222 5.6667 6.1111 6.5556 7.0000
>> x=linspace(0,2.5)
x =
Columns 1 through 10
 0 0.0253 0.0505 0.0758 0.1010 0.1263 0.1515 0.1768 0.2020 0.2273
Columns 11 through 20
 0.2525 0.2778 0.3030 0.3283 0.3535 0.3788 0.4040 0.4293 0.4545 0.4798
Columns 21 through 30
 0.5051 0.5303 0.5556 0.5808 0.6061 0.6313 0.6566 0.6818 0.7071 0.7323
Columns 31 through 40
 0.7576 0.7828 0.8081 0.8333 0.8586 0.8838 0.9091 0.9343 0.9596 0.9848
Columns 41 through 50
 1.0101 1.0354 1.0606 1.0859 1.1111 1.1364 1.1616 1.1869 1.2121 1.2374
```

Columns 51 through 60

1.2626 1.2879 1.3131 1.3384 1.3636 1.3889 1.4141 1.4394 1.4646 1.4899

Columns 61 through 70

1.5152 1.5404 1.5657 1.5909 1.6162 1.6414 1.6667 1.6919 1.7172 1.7424

Columns 71 through 80

1.7677 1.7929 1.8182 1.8434 1.8687 1.8939 1.9192 1.9444 1.9697 1.9949

Columns 81 through 90

2.0202 2.0455 2.0707 2.0960 2.1212 2.1465 2.1717 2.1970 2.2222 2.2475

Columns 91 through 100

2.2727 2.2980 2.3232 2.3485 2.3737 2.3990 2.4242 2.4495 2.4747 2.5000

У таких рядках крок між двома сусідніми елементами Step можна підрахувати за формулою:

Step = (EndValue – InitValue) / (Number -1)

Вектори рядки, рівномірно розподілені впродовж логарифмічної осі, можна сформувати оператором

VarName = logspace (pInit, pEnd, Number)

який створює вектори-рядок кількістю Number елементів (за замовчанням Number=50),

2. Поелементні операції з векторами та матрицями.

Поелементні операції в MATLAB виконуються між відповідними елементами матриць або векторів і дозволяють ефективно працювати з масивами без необхідності використання циклів.
Ось основні операції:
•**Арифметичні**: +, -, .*, ./, .^ (додавання, віднімання, множення, ділення, степінь).
•**Логічні**: &, |, ~ (логічне "і", "або", заперечення).
•**Транспонування**: .' (переведення рядків у стовпці).
•**Векторизація**: Використання цих операцій покращує швидкість обчислень.
Приклад коду:

1.Додавання та віднімання:
matlab
A = [1 2 3]; B = [4 5 6]; C = A + B; % [5 7 9] D = A - B; % [-3 -3 -3]
2.Поелементне множення (.*):
matlab
A = [2 3 4]; B = [3 2 1]; C = A .* B; % [6 6 4]
3.Поелементне ділення (./):
matlab
A = [10 20 30]; B = [2 4 6]; C = A ./ B; % [5 5 5]
4.Поелементне піднесення до степеня (.^):
matlab
A = [2 3 4]; B = [3 2 1]; C = A .^ B; % [8 9 4]
5.Логічні операції:
matlab
A = [1 0 1]; B = [0 1 1]; C = A & B; % [0 0 1] (логічне "і") D = A | B; % [1 1 1] (логічне "або")
6.Транспонування (.'):
matlab
A = [1 2 3; 4 5 6]; B = A.'; % [1 4; 2 5; 3 6]

Приклади обрахування векторів

```
A = [2, 4, 6];
B = [1, 3, 5];

C = A + B; % Векторна сума
disp('Vector Sum:');
disp(C);
```

```
Command Window
>> untitled5
Vector Sum:
     3     7    11
```

Сума двох векторів

```
A = [2, 3, 4];
B = [1, 0, 5];

dot_product = dot(A, B);
disp('Dot Product:');
disp(dot_product);
```

```
Command Window
>> untitled5
Dot Product:
    22
```

Скалярний добуток

```
A = [3, 4, 12];

length_A = norm(A);
disp('Vector Length:');
disp(length_A);
```

```
Command Window
>> untitled5
Vector Length:
    13
```

Довжина(модуль)вектора

```
A = [1, 2, 3];
B = [4, 5, 6];

theta_rad = acos(dot(A, B) / (norm(A) * norm(B)));
theta_deg = rad2deg(theta_rad); % Перевід у градуси

disp('Angle between vectors (degrees):');
disp(theta_deg);
```

```
Command Window
>> untitled5
Angle between vectors (degrees):
   12.9332
```

Кут між двома векторами (в градусах)

```
A = [1, 0, 0];
B = [0, 1, 0];

cross_product = cross(A, B);
disp('Cross Product:');
disp(cross_product);
```

```
Command Window
>> untitled5
Cross Product:
     0     0     1
```

Векторний добуток (лише для 3D векторів)

3. Побудова таблиць значень функцій в Матлаб.

З усіх способів задання функції найбільш зручним у багатьох випадках є аналітичний спосіб у вигляді формули. Цей спосіб дає можливість обчислити значення функції для будь-якого фіксованого значення аргументу, а отже, і скласти таблицю її значень у деяких точках. Складання таблиці значень функції називають **табулюванням функції.**

У практичних задачах значення функції, що представляють деяку фізичну величину, часто одержують у результаті експерименту у вигляді таблиці або графіка. Досить часто виникає необхідність знайти значення функції при значеннях аргументів, що відсутні в таблиці. Така задача, яку образно можна назвати задачею «читання таблиці між рядками», й одержала назву **задачі інтерполювання** (inter – між, polio – прикладати).

Задача інтерполювання функції розв'язується шляхом побудови деякого аналітичного виразу, який співпадає зі значеннями таблично заданої функції в скінченній кількості табличних значень аргументу.

Тому, задача інтерполювання функції в деякому розумінні обернена до задачі табулювання функції: при табулюванні від аналітичного способу задання функції переходять до табличного, а при інтерполюванні – за табличними значеннями функції будується деякий аналітичний вираз, тобто формула, що задає шукану функцію наближено.

Постановка задачі. Нехай деяка функція задана таблично, тобто для заданих значень аргументу x_i задані значення функції $y_i = f(x_i)(i = 0,1,2,...,n)$.

Треба знайти аналітичний вираз деякої функції F (x) n , котра наближала би дану функцію f (x) , тобто у точках x_i приймала би значення $y_i(0,1,2,...,n) : F_n(x_i) = f(x_i)$, i = 0,1,...,n.

У MATLAB можна легко створювати таблиці значень функцій, використовуючи вектори для аргументів та обчислюючи відповідні значення функцій. Ось базовий підхід:

Приклад: Таблиця значень для функції y = sin(x)

```
x = 0:0.1:2*pi; % Вектор аргументів від 0 до 2π з кроком 0.1
y = sin(x); % Обчислення значень функції
% Побудова таблиці
T = table(x', y', 'VariableNames', {'x', 'sin(x)'});
disp(T);
```

Створення таблиці для декількох функцій

```
x = -5:0.5:5; % Вектор аргументів
y1 = x.^2; % Квадратична функція
y2 = exp(x); % Експоненціальна функція
y3 = log(abs(x+1)); % Логарифмічна функція
% Побудова таблиці
T = table(x', y1', y2', y3', 'VariableNames', {'x', 'x^2', 'exp(x)', 'log(|x+1|)'});
disp(T);
```

Експорт таблиці в файл

```
writetable(T, 'function_table.csv'); % Збереження таблиці у CSV-файл
```

Цей підхід дозволяє швидко створювати таблиці та зберігати їх для подальшого аналізу.

T = 5×3 table

	Age	Weight	Height
1 A	18	55	71
2 B	19	63	69
3 C	16	56	64
4 D	17	49	67
5 E	18	59	64

4. Висновки та рекомендації.

1. Функції обробки та їх застосування для векторів та матриць

Функції обробки даних у MATLAB дозволяють ефективно працювати з великими обсягами чисел, виконувати арифметичні та статистичні операції, а також проводити лінійні перетворення. Застосування таких функцій широко використовується у науці, техніці та машинному навчанні. MATLAB оптимізує ці процеси завдяки векторизації, що значно підвищує продуктивність.

Рекомендації:

•Використовуйте векторизовані операції (sum, mean, std) для швидшого обчислення статистичних характеристик.

•Застосовуйте reshape, transpose, diag для зручної модифікації матриць.

•Використовуйте inv, eig, det для роботи з лінійною алгеброю.

2. Поелементні операції з векторами та матрицями

Поелементні операції дозволяють виконувати арифметичні та логічні дії між відповідними елементами матриць. Вони ефективніші за традиційні for-цикли завдяки векторизації MATLAB.

Рекомендації:

•Застосовуйте .*, ./, .^ для поелементного множення, ділення та степеня.

•Використовуйте логічні оператори &, |, ~ для фільтрації даних.

•Виконуйте транспонування (.') для зміни структури матриць.

3. Побудова таблиць значень функцій в MATLAB

MATLAB дозволяє створювати таблиці значень функцій, що корисно для аналізу та представлення даних.

Рекомендації:

•Використовуйте table для створення таблиць.

•Застосовуйте writetable для збереження таблиці у файл.

•Будуйте графіки значень функцій (plot, surf, bar) для візуального аналізу.

@ZTUEDUUA

ЖИТОМИРСЬКА ПОЛІТЕХНІКА

ДЕРЖАВНИЙ УНІВЕРСИТЕТ

- **Розвиваємо лідерів**
- **Створюємо інновації**
- **Змінюємо світ на краще**