Лекція 8
Функції аналізу і обробки даних в MATLAB

1. Обробка статистичних даних

1.1 Знаходження максимального та мінімального елементів масиву

Будь-який найпростіший аналіз даних, що містяться у деякому масиві, передбачає визначення його елементів з максимальними та мінімальними значеннями. Для цього у MATLAB існують такі функції знаходження максимальних і мінімальних елементів масиву:

- **max(A), min(A)** – повертає максимальний (мінімальний) елемент, якщо A – вектор. Якщо A – матриця, то повертається вектор-рядок, що містить максимальні (мінімальні) значення кожного стовпчика;
- **max(A,B), min(A,B)** – повертає матрицю тієї ж розмірності, що A і B, кожен елемент якої є максимальним (мінімальним) з відповідних елементів цих матриць;
- **max(A,[],dim), min(A,[],dim)** – повертає максимальний (мінімальний) елемент по стовпчиках або по рядках у залежності від значення скаляра dim (dim=1 – пошук екстремальних елементів іде по стовпчиках, dim=2 – пошук екстремальних елементів іде по рядках);
- **[C,I]=max(A), [C,I]=min(A)** – крім максимальних (мінімальних) значень додатково виводиться рядок індексів елементів з цими значеннями.

Приклади:

```
>> A=magic(7)
A =

   30   39   48    1   10   19   28
   38   47    7    9   18   27   29
   46    6    8   17   26   35   37
    5   14   16   25   34   36   45
   13   15   24   33   42   44    4
   21   23   32   41   43    3   12
   22   31   40   49    2   11   20
>> C = max(A)
C =
   46   47   48   49   43   44   45
>> C = max(A,[],1)
C =
   46   47   48   49   43   44   45
>> C = max(A,[],2)
```

```
C =

    48
    47
    46
    45
    44
    43
    49
>> [C,I] = max(A)
C =

    46    47    48    49    43    44    45
I =

    3    2    1    7    6    5    4
```

1.2 Сортування елементів масиву

Багато операцій статистичної обробки даних виконуються простіше, швидше і точніше, коли дані попередньо впорядковані або відсортовані. Крім того, це може бути корисним для їх подальшої інтерпретації. Для сортування елементів масивів використовуються такі функції:

- **sort(A)** – у випадку одновимірного масиву A сортує і повертає елементи за зростанням їх значень, у випадку двовимірного масиву відбувається сортування і повернення елементів кожного стовпчика. Допускаються дійсні, комплексні та рядкові елементи;
- **[B,INDEX]=sort(A)** – разом з відсортованим масивом повертає масив індексів INDEX. Він має розмірність size(A), за допомогою нього можна відновити структуру початкового масиву;
- **sort(A,dim)** – для матриць сортує елементи по рядках або стовпчиках у залежності від значення змінної dim.

Приклади:

```
>> A=magic(5)
A =

    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
```

```
>> [B,INDEX] = sort(A)
B =

     4     5     1     2     3
    10     6     7     8     9
    11    12    13    14    15
    17    18    19    20    16
    23    24    25    21    22
INDEX =

     3     2     1     5     4
     4     3     2     1     5
     5     4     3     2     1
     1     5     4     3     2
     2     1     5     4     3
```

1.3 Обчислення описових статистик масиву

Для визначення описових статистик масиву (випадкової вибірки), таких як середнє значення, медіана, середнє квадратичне відхилення (у статистиці воно має назву стандартного відхилення), використовуються функції:

- **mean(A)** – повертає середнє арифметичне значення елементів масиву, якщо A – вектор. Якщо A – матриця, то повертається вектор-рядок, що містить середні арифметичні значення кожного стовпчика;
- **mean(A,dim)** – повертає середні значення елементів по стовпчиках або по рядках у залежності від значення скаляра dim (dim=1 – по стовпчиках, dim=2 – по рядках);

Приклади:

```
>> A = [1 2 6 4 8; 6 7 13 5 4; 7 9 0 8 12; 6 6 7 1 2]
A =
     1     2     6     4     8
     6     7    13     5     4
     7     9     0     8    12
     6     6     7     1     2
>> mean(A)
ans =
    5.0000    6.0000    6.5000    4.5000    6.5000
>> mean(A,2)
ans =
    4.2000
    7.0000
    7.2000
    4.4000
```

- **median(A)** – повертає медіану елементів масиву, якщо A – вектор. Якщо A – матриця, то повертається вектор-рядок, що містить медіани кожного стовпчика;
- **median(A,dim)** – повертає значення медіан для стовпчиків або для рядків у залежності від значення скаляра dim;

Приклади:

```
>> A = magic(6)
A =

    35     1     6    26    19    24
     3    32     7    21    23    25
    31     9     2    22    27    20
     8    28    33    17    10    15
    30     5    34    12    14    16
     4    36    29    13    18    11
>> M = median(A)
M =

    19.0000  18.5000  18.0000  19.0000  18.5000  18.0000
>> M = median(A,2)
M =

    21.5000
    22.0000
    21.0000
    16.0000
    15.0000
    15.5000
```

- **std(X)** – повертає стандартне відхилення елементів масиву, якщо X – вектор. Якщо X – матриця, то повертається вектор-рядок, що містить стандартні відхилення кожного стовпчика;
- **std(X,flag)** – повертає те саме значення, що і std(X), якщо flag=0, тобто обчислює незміщену оцінку $s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$; якщо flag=1, то обчислюється зміщена оцінка $s = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$;
- **std(X,flag,dim)** – повертає оцінку стандартного відхилення по стовпчиках або рядках матриці X у залежності від значення змінної dim.

Приклад:

```
>> X = linspace(0,3*pi,10)
X =
Columns 1 through 6
    0   1.0472   2.0944   3.1416   4.1888   5.2360
Columns 7 through 10
    6.2832   7.3304   8.3776   9.4248
>> s = std(X)
s =
    3.1705
```

1.4 Обчислення коефіцієнтів кореляції та коваріацій

Під *кореляцією* мають на увазі статистичний взаємозв'язок між деякими величинами, значення яких представлені векторами або матрицями. Загальноприйнятою мірою лінійної кореляції випадкових величин x і y є *коефіцієнт кореляції*

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2 (y_i - \bar{y})^2}{n\sigma_x\sigma_y}.$$

Значення коефіцієнта кореляції лежить в діапазоні $-1 \le r_{xy} \le 1$. Чим ближче абсолютне значення цього коефіцієнта до одиниці, тим сильнішим є взаємозв'язок між величинами x і y.

Для обчислення коефіцієнтів кореляції та коваріації вхідного масиву даних використовуються функції:

- **corrcoef(X)** – повертає матрицю коефіцієнтів кореляції для вхідної матриці X, рядки якої розглядаються як спостереження, а стовпчики – як змінні. Матриця S=corrcoef(X) зв'язана з матрицею коваріацій C=cov(X) співвідношенням S(i,j)=C(i,j)/sqrt(C(i,i)*C(j,j));
- функція **corrcoef(x,y)**, де x і y – вектори-стовпчики, аналогічна функції **corrcoef([x y])**;

Приклад:

```
>> M=magic(5)
M =

    17   24    1    8   15
    23    5    7   14   16
     4    6   13   20   22
    10   12   19   21    3
    11   18   25    2    9
```

```
>> S=corrcoef(M)
S =

    1.0000    0.0856   -0.5455   -0.3210   -0.0238
    0.0856    1.0000   -0.0981   -0.6731   -0.3210
   -0.5455   -0.0981    1.0000   -0.0981   -0.5455
   -0.3210   -0.6731   -0.0981    1.0000    0.0856
   -0.0238   -0.3210   -0.5455    0.0856    1.0000
```

- **cov(X)** – повертає зміщену дисперсію елементів масиву X, якщо X – вектор. Для матриці X, рядки якої розглядаються як спостереження, а стовпчики – як змінні, cov(X) повертає матрицю коваріацій. Тоді diag(cov(X)) – вектор дисперсій кожного стовпчика, sqrt(diag(cov(X))) – вектор стандартних відхилень;
- функція **cov(x,y)**, де x і y – вектори-стовпчики однакової довжини, аналогічна функції **cov([x y])**.

  Приклад:

```
>> D=[2 -3 6;3 6 -1;9 8 5]; C=cov(D)
C =

    14.3333    16.3333    3.6667
    16.3333    34.3333  -10.3333
     3.6667   -10.3333   14.3333
>> diag(cov(D))
ans =

    14.3333
    34.3333
    14.3333
>> sqrt(diag(cov(D)))
ans =

    3.7859
    5.8595
    3.7859
>> std(D)
ans =
    3.7859   5.8595   3.7859
```

2. Математичні операції з поліномами

Поліноми (які називають також *степеневими многочленами*) – широко відомий об'єкт математичних обчислень та обробки даних. В MATLAB поліном $p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$ задається у вигляді вектора-рядка `p=[a_n a_n-1 … a_1 a_0]`.

Широке застосування поліномів зумовлено їх гарними апроксимуючими властивостями для подання різноманітних функціональних залежностей. В MATLAB передбачена ціла група функцій для роботи з поліномами.

2.1 Обчислення значень та коренів поліномів

Для обчислення значень поліному в заданих точках та знаходження коренів полінома (які можуть бути як дійсними, так і комплексними) використовуються такі функції:

- **polyval(p,x)** – повертає значення полінома p, обчислені в точках, заданих у масиві x. Поліном p – вектор-рядок, елементи якого є коефіцієнтами полінома у порядку зменшення степенів. x може бути матрицею або вектором, у будь-якому випадку функція polyval обчислює значення полінома p для кожного елемента x;

  Приклад:

```
>> p=[3,0,4,3]; d=polyval(p,[2 6])
d =
    35   675
```

- **poly(A)** – для квадратної матриці A розмірності $n \times n$ повертає вектор-рядок розміром $n+1$, елементи якого є коефіцієнтами характеристичного полінома $\det(A - sI)$, де I – одинична матриця і s – оператор Лапласа. Коефіцієнти впорядковані за спаданням степенів. Якщо вектор складається з $n+1$ компонентів, то йому відповідає поліном виду $c_1 s^n + c_2 s^{n-1} + \ldots + c_n s + c_{n+1}$;

  Приклад:

```
>> A=[3,6,8;12,23,5;11,12,32]
A =
    3    6    8
   12   23    5
   11   12   32
>> poly(A)
ans =
    1.0000   -58.0000   681.0000   818.0000
```

- **poly(r)** – для вектора-стовпчика r повертає вектор-рядок з коефіцієнтами полінома, коренями якого є елементи вектора r;

Приклад:

```
>> poly([3;-8;2])
ans =

    1    3  -34   48
```

- **roots(p)** – повертає вектор-стовпчик, елементи якого є коренями полінома p; Функція roots(p) виконує зворотне перетворення по відношенню до функції poly(r);

Приклади:

```
>> roots([1 3 -34 48])
ans =

  -8.0000
   3.0000
   2.0000
>> p=[7 45 12 23]; d=roots(p)
d =

  -6.23820 + 0.00000i
  -0.09519 + 0.71948i
  -0.09519 - 0.71948i
```

2.2 Обчислення похідної полінома

Для обчислення похідної полінома використовується функція

- **polyder(p)** – повертає похідну полінома p;
- **polyder(a,b)** – повертає похідну від добутку поліномів a і b;
- **[q,d]=polyder(b,a)** – повертає чисельник q і знаменник d похідної від частки поліномів b/a;

Приклади:

```
>> a=[3,5,8];b=[5,3,8];dp=polyder(a)
dp = 6 5
>> dt=polyder(a,b)
dt =  60  102  158  64
>> [q,p]=polyder(b,a)
q = 16 32 -16
p =  9  30  73  80  64
```

2.3 Розкладання відношення поліномів на прості дроби

Дробово-раціональну функцію, яка є відношенням двох поліномів, можна розкласти на суму простих дробів

$$\frac{b(s)}{a(s)} = \frac{r_1}{s-p_1} + \frac{r_2}{s-p_2} + \cdots + \frac{r_n}{s-p_n} + k(s).$$

В MATLAB цю та зворотну до неї операцію можна виконати за допомогою функцій:

- **[r,p,k] = residue(b,a)** – повертає відрахування, полюси і многочлен цілої частини відношення двох поліномів b(s) і a(s);
- **[b,a] = residue(r,p,k)** – виконує зворотну згортку суми простих дробів у дробово-раціональну функцію відношення двох поліномів b(s) і a(s).

Приклади:

```
>> b=[4 3 1]; a=[1 3 7 1]; [r,p,k]=residue(b,a)
r =

    1.9484 + 0.8064i
    1.9484 - 0.8064i
    0.1033
p =

   -1.4239 + 2.1305i
   -1.4239 - 2.1305i
   -0.1523
k =

    [ ]
>> [b1,a1]=residue(r,p,k)
b1 =

    4.0000   3.0000   1.0000
a1 =

    1.0000   3.0000   7.0000   1.0000
```

2.4 Апроксимація функціональних залежностей поліномами

Одним з найпоширеніших видів апроксимації є поліноміальна апроксимація. В системі MATLAB визначено функції апроксимації даних поліномами за методом найменших квадратів. Це досить універсальний вид апроксимації. Наприклад, при степені полінома 1 ми отримуємо лінійну залежність, при степені полінома 2 – квадратичну і т.д.

Поліноміальну апроксимацію здійснює функція:

- **polyfit(x,y,n)** – повертає вектор коефіцієнтів полінома p(x) степеню n, який з найменшою середньоквадратичною похибкою апроксимує функцію y(x). Результатом є вектор-рядок довжиною n+1, що містить коефіцієнти полінома у порядку зменшення степенів. Якщо масиви x і y мають довжину n+1, то реалізується звичайна поліноміальна апроксимація, при якій графік полінома точно проходить через вузлові точки з координатами (x,y), що зберігаються в масивах x і y. У противному випадку точного збігу графіка з вузловими точками не буде.

  Приклад (поліноміальна апроксимація функції sin(*x*)):

```
>> x=-3:0.2:3; y=sin(x); p=polyfit(x,y,3)
p =

    -0.0953    0.0000    0.8651   -0.0000
>> x=-4:0.2:4; y=sin(x);
>> f=polyval(p,x); plot(x,y,'o',x,f)
```

Результат виконання цих команд наведено на рис. 8.1. З графіка видно, що апроксимаційний поліном 3-го степеню досить точно наближує функцію sin(*x*) на проміжку –3…3, але за його межами похибка апроксимації суттєво зростає.

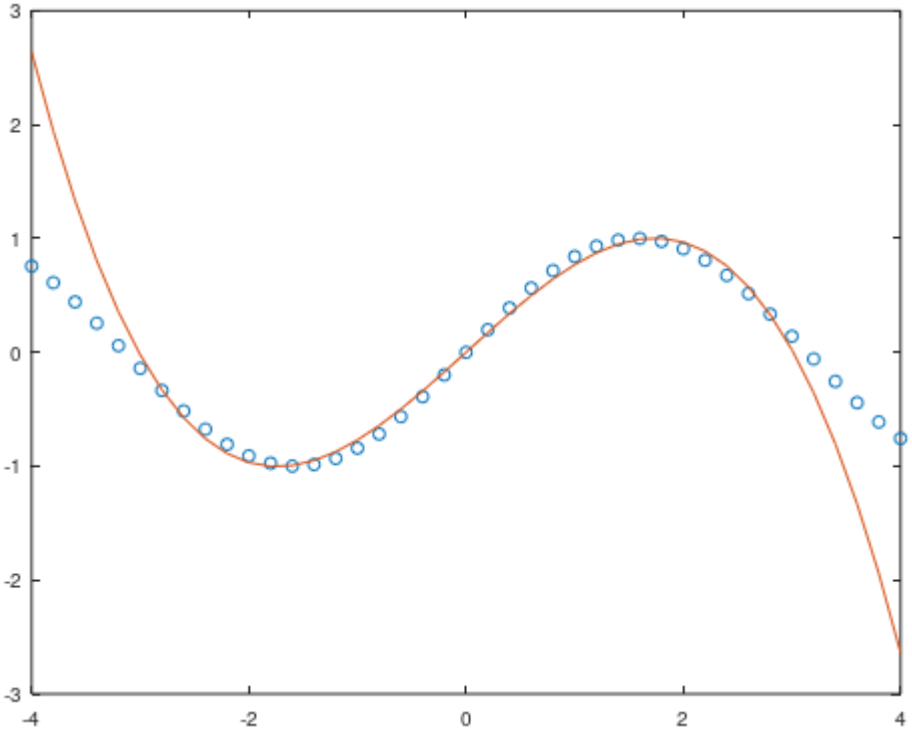

Рисунок 8.1 – Приклад використання функції polyfit