

Практическая работа 7.

Работа с системой MatLab. Операции с векторами и матрицами.

Цель работы. изучение интерфейса пользователя системы MatLab и основ работы с системой в режиме прямых вычислений, освоение основных операций с векторами и матрицами в системе MatLab, приобретение навыков построения двумерных графиков в системе MatLab.

Теоретические сведения

Операции с векторами и матрицами в режиме прямых вычислений

Основные матричные операции

В среде MatLab широко используются векторы и матрицы. Отметим, что необходимо различать заглавные и прописные буквы, так, например, k и K являются разными переменными. При вводе векторов или матриц их элементы требуется заключить в квадратные скобки. На рисунке 1.1 приведена команда для ввода вектора-строки размером 1×3 , при этом элементы строки отделены пробелами. На рисунке 1.2 показан ввод вектора-столбца размером 1×3 , при этом элементы разделяют точкой с запятой.

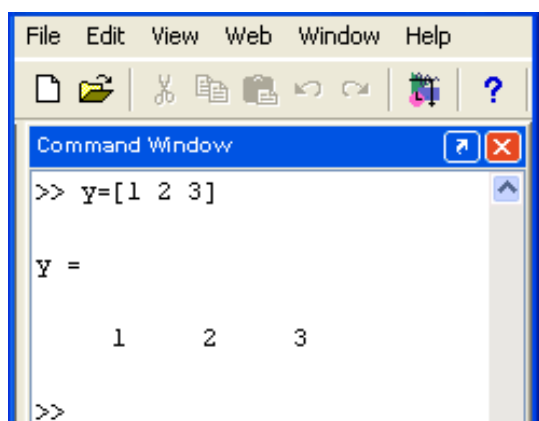


Рис. 1.1. Ввод вектора-строки

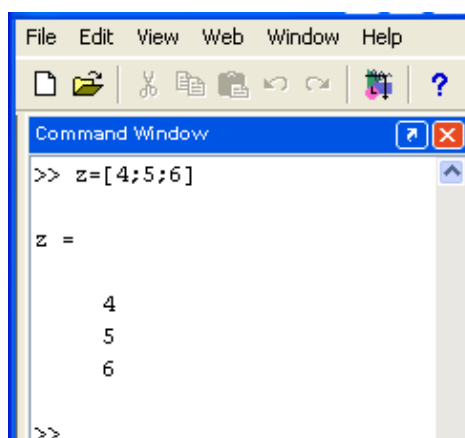


Рис. 1.2. Ввод вектора-столбца

Из командной строки, как правило, осуществляют ввод небольших по размеру матриц.

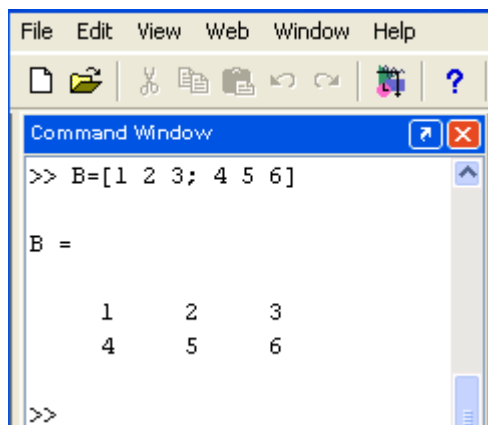


Рис. 1.3. Матрица как вектор-столбец

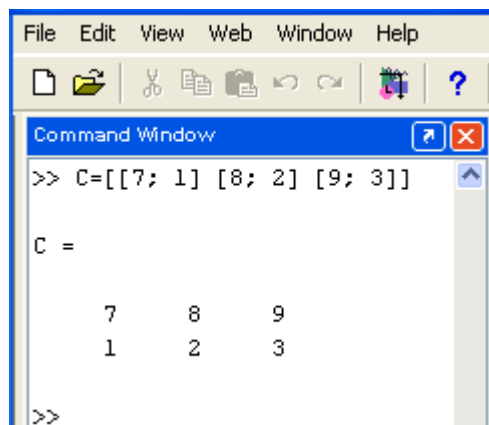


Рис. 1.4. Матрица как вектор-строка

На рисунке 1.3 осуществляется ввод матрицы из командной строки. Сама матрица рассматривается как вектор-столбец, каждый элемент которого является вектор-строкой. На рисунке 1.4 матрица трактуется как вектор-строка, каждый элемент которой является вектор-столбцом. Доступ к элементам матриц выполняется при указании двух индексов — номеров строки и столбца, заключенных в круглые скобки. Для выделения из матрицы столбца или строки требуется указать в качестве одного из индексов номер столбца или номер строки матрицы, а в качестве другого индекса используется двоеточие. На рисунке 1.5 в качестве вектора x используется третья строка матрицы B . На рисунке 1.6 выполнено выделение блоков матрицы с помощью двоеточия.

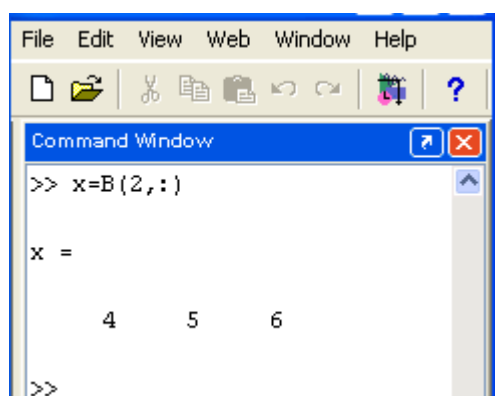


Рис. 1.5. Выделение строки матрицы

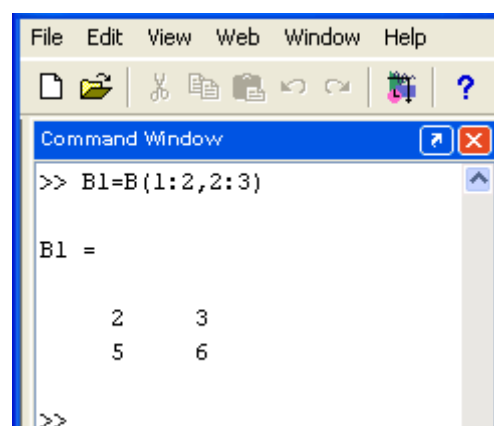
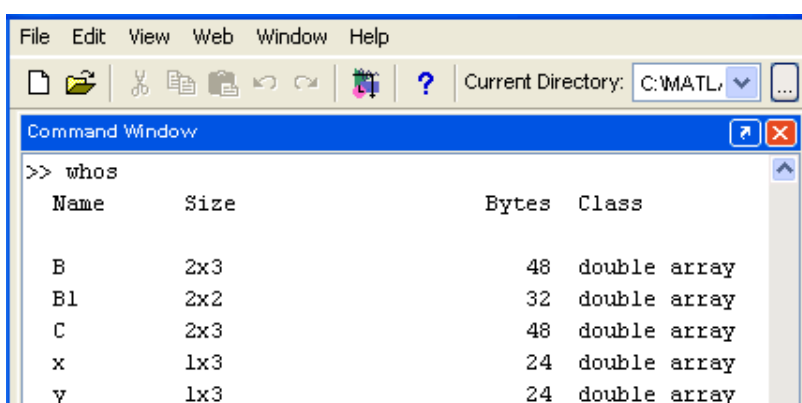
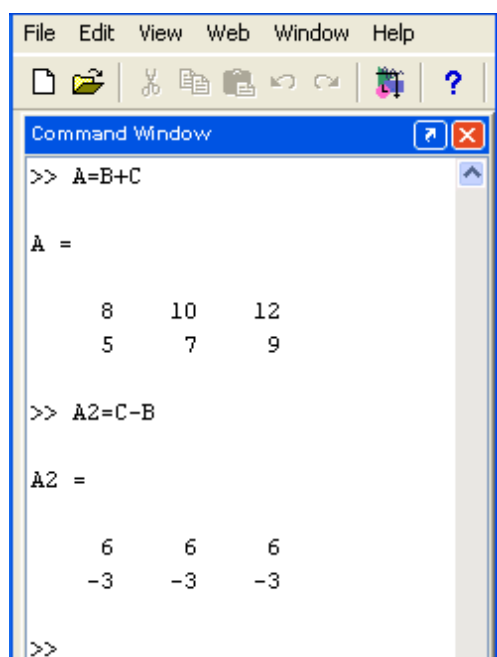


Рис. 1.6. Выделение блоков матрицы



На рисунке 1.7 изображены переменные рабочей среды, выведенные с помощью команды whos.

При выполнении матричных операций необходимо учитывать, что при сложении или вычитании матрицы должны иметь одинаковый размер, а при перемножении матриц число столбцов первой матрицы равняется числу строк второй матрицы. На рисунке 1.8 показано сложение и вычитание матриц, на рисунке 1.9 – результат умножения матриц.



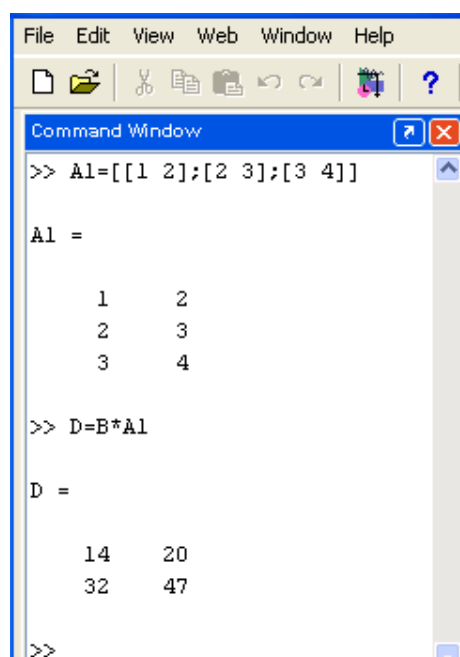
```
File Edit View Web Window Help
A =
     8     10     12
     5      7      9

>> A2=C-B

A2 =
      6      6      6
     -3     -3     -3

>>
```

Рис. 1.8. Сложение и вычитание матриц



```
File Edit View Web Window Help
A1 =
      1      2
      2      3
      3      4

>> D=B*A1

D =
     14     20
     32     47

>>
```

Рис. 1.9. Умножение матриц

Если задано несколько матриц с одинаковым количеством строк, то из них можно построить единую матрицу, располагая матрицы горизонтально в одну линию, например, $F4=[F1\ F2\ F3]$. Такую операцию принято называть горизонтальной конкатенацией матриц или сцеплением матриц. Пример горизонтальной конкатенации матриц изображен на рисунке 1.10. Вертикальную конкатенацию матриц можно выполнить аналогично, при условии, что все составные матрицы имеют одинаковое ко-

личество столбцов, например, с помощью команды $F5=[F1; F2; F3]$. На рисунке 1.11 показан процесс выполнения вертикальной конкатенации матриц.

```
>> F1=[1 2; 2 3; 3 4];
>> F2=[4 5; 5 6; 6 7];
>> F3= [7 8; 8 9; 0 1];
>> F4=[F1 F2 F3]

F4 =

     1     2     4     5     7     8
     2     3     5     6     8     9
     3     4     6     7     0     1

>>
```

Рис. 1.10. Горизонтальная конкатенация

```
>> F5=[F1; F2; F3]

F5 =

     1     2
     2     3
     3     4
     4     5
     5     6
     6     7
     7     8
     8     9
     0     1

>>
```

Рис. 1.11. Вертикальная конкатенация

На рисунке 1.12 приведен пример возведения квадратной матрицы в целую степень с использованием оператора «^».

```
>> A3=[1 2 3; 4 5 6; 7 8 9]

A3 =

     1     2     3
     4     5     6
     7     8     9

>> A4=A3^2

A4 =

    30    36    42
    66    81    96
   102   126   150

>>
```

Рис. 1.12. Возведение матрицы в степень

Создание матриц специального вида

Для заполнения прямоугольной матрицы нулями можно использовать встроенную функцию `zeros`. На рисунке 1.13 показан результат работы функции `zeros`.

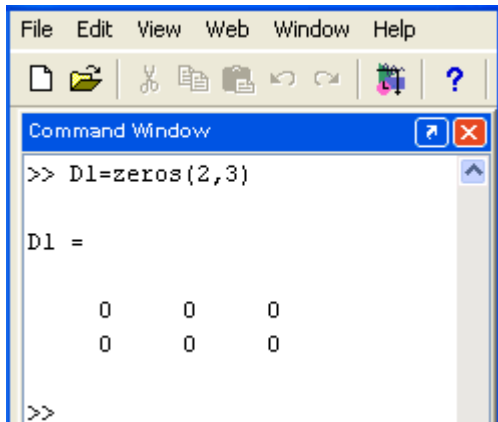


Рис. 1.13. Использование функции `zeros`

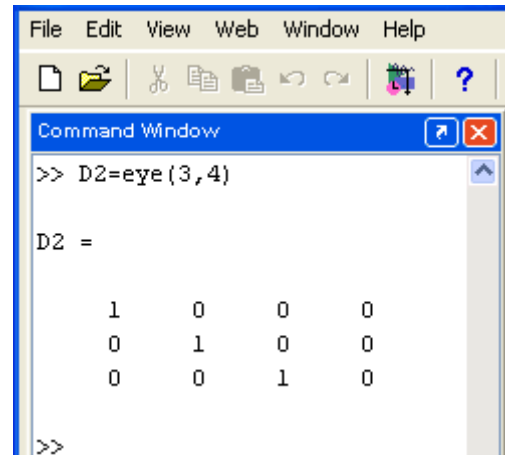


Рис. 1.14. Использование функции `eye`

На рисунке 1.14 изображено формирование единичной матрицы с помощью функции `eye`. На рисунке 1.15 образована матрица, состоящая из единиц, в результате работы функции `ones`.

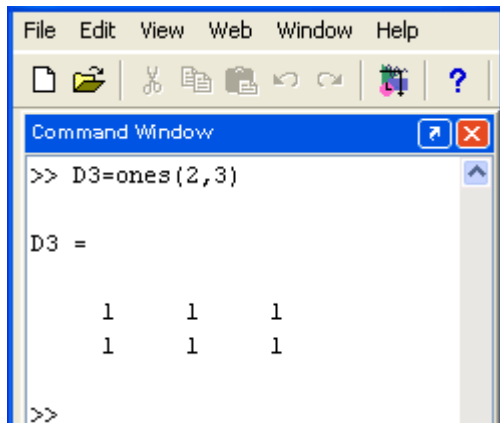


Рис. 1.15. Функция `ones`

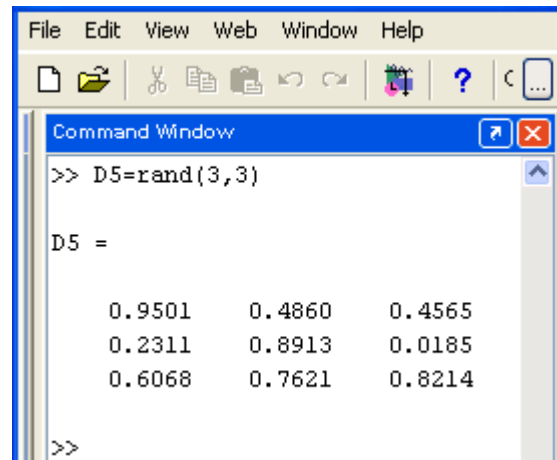
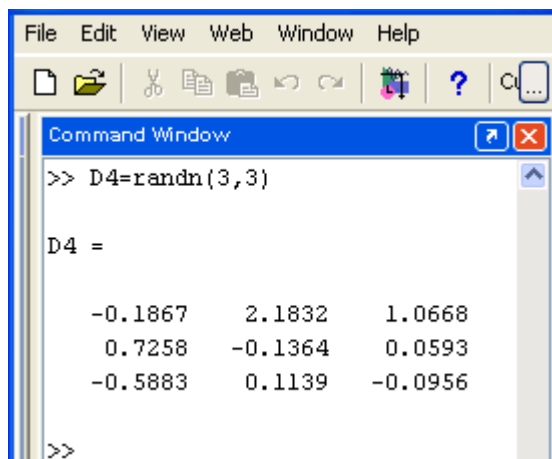


Рис. 1.16. Функция `rand`

Для заполнения матриц случайными числами, равномерно распределенных между нулем и единицей, можно использовать функцию `rand` (рисунок 1.16). Функция `randn` используется для формирования матрицы чисел, распределенных по нормальному закону с нулевым средним и единичной дисперсией (рисунок 1.17).



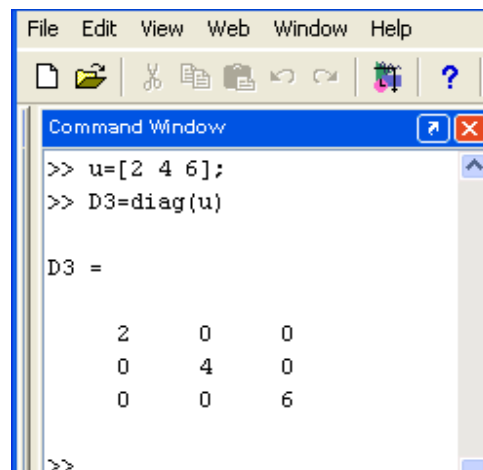
```
File Edit View Web Window Help
>> D4=randn(3,3)

D4 =

    -0.1867    2.1832    1.0668
     0.7258   -0.1364    0.0593
    -0.5883    0.1139   -0.0956

>>
```

Рис. 1.17. Функция randn



```
File Edit View Web Window Help
>> u=[2 4 6];
>> D3=diag(u)

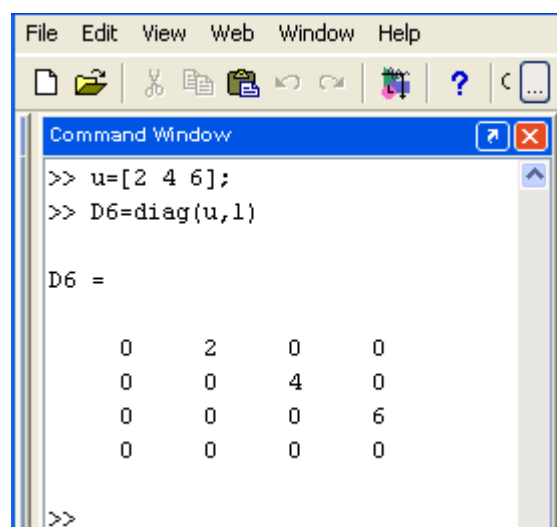
D3 =

     2     0     0
     0     4     0
     0     0     6

>>
```

Рис. 1.18. Функция diag

Функция `diag` формирует диагональную матрицу из вектора, располагая элементы по диагонали (рисунок 1.18). Для того чтобы элементы вектора размещались на другой диагонали, необходимо указать номер диагонали, выраженный целым числом (рисунок 1.19). Отметим, что диагонали отсчитываются по направлению вверх от главной диагонали. Функцию `diag` можно использовать для создания вектора-столбца, состоящего из элементов главной диагонали заданной матрицы (рисунок 1.20).



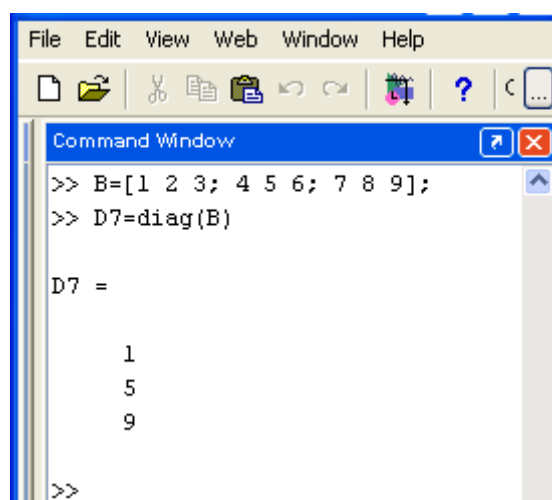
```
File Edit View Web Window Help
>> u=[2 4 6];
>> D6=diag(u,1)

D6 =

     0     2     0     0
     0     0     4     0
     0     0     0     6
     0     0     0     0

>>
```

Рис. 1.19. Функция diag с параметром



```
File Edit View Web Window Help
>> B=[1 2 3; 4 5 6; 7 8 9];
>> D7=diag(B)

D7 =

     1
     5
     9

>>
```

Рис. 1.20. Выделение главной диагонали

```
File Edit View Web Window Help
[Icons] ? Currer ...

Command Window

A7 =

     1     2     3     4     5
     6     7     8     9     0
     8     7     6     5     4

>> A8=fliplr(A7)

A8 =

     5     4     3     2     1
     0     9     8     7     6
     4     5     6     7     8

>>
```

Рис. 1.21. Функция fliplr

```
File Edit View Web Window Help
[Icons] ? Currer ...

Command Window

A7 =

     1     2     3     4     5
     6     7     8     9     0
     8     7     6     5     4

>> A9=flipud(A7)

A9 =

     8     7     6     5     4
     6     7     8     9     0
     1     2     3     4     5

>>
```

Рис. 1.22. Функция flipud

Функция `fliplr` позволяет сформировать матрицу путем перестановки столбцов заданной матрицы относительно вертикальной оси (рисунок 1.21). Функция `flipud` осуществляет перестановку заданной матрицы относительно горизонтальной оси (рисунок 1.22).

```

File Edit View Web Window Help
[Icons] ? Current ...

Command Window

C1 =

     1     2     3     4
    11    22    33    44
     5     6     7     8

>> C3=reshape(C1,2,6)

C3 =

     1     5    22     3     7    44
    11     2     6    33     4     8

>>

```

Рис. 1.23. Функция reshape

```

File Edit View Web Window Help
[Icons] ? Current ...

Command Window

C1 =

     1     2     3     4
    11    22    33    44
     5     6     7     8

>> C2=rot90(C1)

C2 =

     4    44     8
     3    33     7
     2    22     6
     1    11     5

>>

```

Рис. 1.24. Функция rot90

Функция reshape дает возможность построить матрицу размером $m \times n$, задавая из столбцов элементы заданной матрицы и распределяя их по n столбцам, каждый из которых содержит m элементов (рисунок 1.23). Функция rot90 переворачивает заданную матрицу на 90° против часовой стрелки (рисунок 1.24)

В среде MatLab есть множество способов для работы с матрицами. На рисунке 1.25 выполнено транспонирование матрицы с помощью оператора «'». На рисунке 1.26 обратная матрица находится с использованием встроенной функции inv для квадратных матриц. Псевдообратную матрицу можно найти с помощью функции pinv.

```

File Edit View Web Window Help
[Icons] ?

Command Window

>> B2=B'

B2 =

     1     4
     2     5
     3     6

>>

```

Рис. 1.25. Транспонирование

```

File Edit View Web Window Help
[Icons] ? Current ...

Command Window

>> A5=[10 20 30; 25 15 35; 35 45 50];
>> A6=inv(A5)

A6 =

    -0.0892    0.0378    0.0270
    -0.0027   -0.0595    0.0432
     0.0649    0.0270   -0.0378

>>

```

Рис. 1.26. Обратная матрица

На рисунке 1.27 приведена команда, которая позволяет поместить заданное число на определенное место в матрице. В языке MatLab есть функции деления матриц слева направо (символ «/») и деление матриц справа налево (символ «\»). Операция B/A эквивалентна операции $B \cdot \text{inv}(A)$, где функция inv осуществляет обращение матрицы (рисунок 1.28). Операция B/A используется для решения матричного уравнения $X \cdot A = B$. Операция $A \setminus B$ эквивалентна операции $\text{inv}(A) \cdot B$, используется при решении матричного уравнения $A \cdot X = B$.

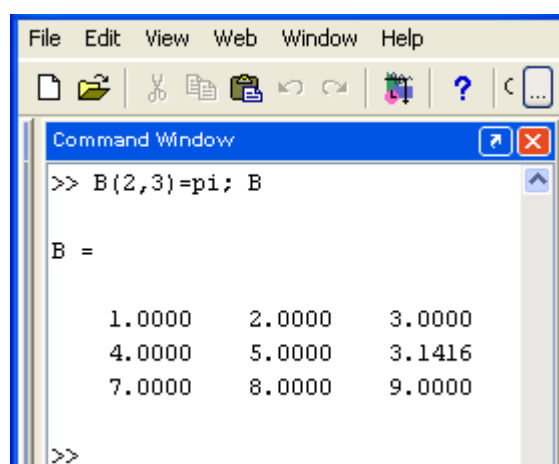


Рис. 1.27. Вставка числа

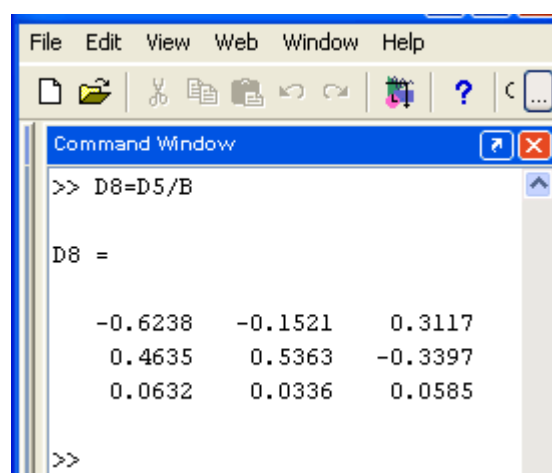


Рис. 1.28. Деление матриц

Функция $\text{size}(m)$ служит для определения числа строк и столбцов матрицы m . Она возвращает вектор $[n, p]$, содержащий эти данные. Функция $\text{max}(v)$ возвращает значение максимального по значению элемента вектора v . Если ее аргументом является матрица, например $\text{max}(m)$, то функция возвращает вектор-строку, содержащий значения максимальных элементов каждого из столбцов. Аналогично действует функция $\text{min}(m)$, выделяющая элементы с минимальными значениями.

Функция $\text{mean}(v)$ возвращает среднее значение элементов вектора v , а функция $\text{mean}(m)$ с матричным аргументом возвращает вектор-строку средних значений каждого из столбцов данных. Функция сортировки $\text{sort}(v)$ возвращает вектор, элементы которого расположены в порядке роста их значений. Для матричного аргумента эта функция возвращает матрицу, у которой отсортированы элементы каждого столбца. Функция $\text{sum}(v)$ возвращает сумму элементов вектора v , а для матричного аргумента функция $\text{sum}(m)$ возвращает вектор-строку сумм элементов по каждому из столбцов. Аналогично функция $\text{prod}(m)$ возвращает вектор произведений элементов каждого из столбцов.

Графическое оформление результатов вычислений

В среде MatLab представлены широкие возможности графического оформления информации. Это дает возможность для построения двумерных и трехмерных графиков функций, заданных как в аналитическом виде, так и в виде векторов и матриц, что позволяет построить множество функций на одном рисунке, а также иллюстрировать графики разными цветами, типами точек, линий в различных системах координат.

Пользователь среды MatLab способен строить диаграммы, гистограммы и графики специальных функций. В качестве основных функций двумерной графики используются следующие функции:

plot (*x*, *y*)

plot (*x*, *y*, *s*)

plot (*x*₁, *y*₁, *s*₁, *x*₂, *y*₂, *s*₂, ..., *x*_{*n*}, *y*_{*n*}, *s*_{*n*})

где *x* – аргумент функции, задаваемой в виде вектора;

y – функция, представленная в аналитическом виде или в виде вектора или матрицы;

s – вектор стилей графика, может быть константа, определяющая цвет линий графика, тип точек и тип линии;

*x*₁, *x*₂, ..., *x*_{*n*} – аргументы *n* функций, изображаемых на одном графике;

*y*₁, *y*₂, ..., *y*_{*n*} – функции, изображаемые на одном графике.

Функция *plot*(*x*,*y*) позволяет строить график при задании функции $y = f(x)$ в аналитическом виде, в виде вектора или матрицы.

Ниже перечислены функции, которые используются при построении графиков:

loglog – использование логарифмического масштаба при построении функции;

semilogx – использование полулогарифмического масштаба при построении функции (*log* по оси *x*);

semilogy – использование полулогарифмического масштаба при построении функции (*log* по оси *y*);

polar – использование полярной системы координат при построении функции;

mesh – изображение графика функции трехмерной поверхности;

contour – изображение графика с контурными линиями – уровнями равных высот;

bar – изображение графика функции столбцовой гистограммы;

stairstep – изображение графика функции в виде ступенчатой линии.

Нижеперечисленные операторы используются для оформления графиков:

text – размещение заданного текста на графике;

title – оформление титульного заголовка;

xlabel – размещение текста по оси *x*;

ylabel – размещение текста по оси *y*;

grid – изображение масштабной сетки.

При формировании графических иллюстраций принято использовать следующие операторы:

axis(<масштаб>) – указание масштаба при построении осей графика;

hold – предоставляет возможность сохранения предыдущих построений;

subplot(m,n,p) – данная операция осуществляет разбиение исходного окна на меньшие окна,

где *m* – количество окон по вертикали, *n* – по горизонтали, *p* – номер подокна.

В таблице приведены стили графиков системы MATLAB.

Таблица 1. Стили графиков

Тип точек		Цвет линии		Тип линии	
.	Точка	Y	Желтый	-	Сплошная
O	Окружность	M	Фиолетовый	:	Двойной пунктир
X	Крест	C	Голубой	-.	Штрих-пунктир
+	Плюс	R	Красный	--	Штриховая
*	Звездочка	G	Зеленый		
S	Квадрат	B	Синий		
D	Ромб	W	Белый		
P	Пятиугольник	K	Черный		

При задании стиля символ *s* представляется в виде вектора, элементами которого являются: тип точки, цвет линии и тип линии, разделенные запятыми и выделенные одиночными кавычками.

Пример: *plot(x, y, ['R ', ' +', ' - '])*

Это график красного цвета, точки графика в виде плюса, линия сплошная.

Функция *plot(x₁, y₁, s₁, x₂, y₂, s₂, ..., x_i, y_i, s_i, ..., x_n, y_n, s_n)* дает возможность разместить несколько функций на одном графике. Здесь приняты следующие обозначения:

x_i – массив аргументов для *i*-й функции;

y_i – массив значений для *i*-й функции;

s_i – стиль графика для *i*-й функции.

В случае, если стиль для графика не указан, то MatLab сам установит стиль. Используя команду *hold on*, можно вывести все графики на один рисунок. Отменить этот режим можно командой *hold off*.

К построенному графику можно подключить так называемую *легенду* – пояснения в виде линий с текстовой информацией. Команда *legend('x₁', 'x₂', 'x₃')* позволяет сформировать отрезки линий графиков с поясняющей текстовой информацией 'x₁', 'x₂', 'x₃', размещаемой внутри графика или око-

ло него. Если на рисунке изображено три функции, то легенда будет расположена в правом верхнем углу графика.

Для того чтобы русский текст воспроизводился необходимо задать код шрифта командой *set*. Команда *gca* позволяет выбрать шрифт для координатных осей.

В полярной системе координат выполнить построение графиков можно осуществляется с помощью следующих функций:

polar(Q, r);

polar(Q, r, s)

где Q – угол функции $r(Q)$, r – радиус, s – вектор стилей.

Для построения трехмерного графика функции $z = f(x, y)$ используются матрицы значений переменных x и y . В этом случае применяют функции:

[X, Y] = meshgrid(x, y)

[X, Y] = meshgrid(x)

[X, Y, Z] = meshgrid(x, y, z)

Функция *meshgrid(x, y)* – осуществляет преобразование областей векторов x и y в массивы значений X и Y соответственно. Эти массивы значений в дальнейшем используются при вычислении функции $z = f(x, y)$.

В результате вычисления функции *[X, Y, Z] = meshgrid(x, y, z)* получаем массив, который необходим для построения соответствующего графика.

Для построения графиков трехмерных поверхностей используются следующие функции:

plot3(x, y, z)

plot3(X, Y, Z)

plot3(X, Y, Z, s)

plot3(x₁, y₁, z₁, s₁, x₂, y₂, z₂, s₂, ..., x_n, y_n, z_n, s_n)

где x, y, z – представляют собой вектора аргументов функции, X, Y, Z – матрицы, s – указанные стили графика.

Все перечисленные функции строят точки на графике и соединяют их в соответствии с выбранным стилем.