Операции с матрицами и свойства матриц

Создайте скрипт-файл, включите все предлагаемые примеры в качестве кода, разберите особенности работы команд, конструкторов и функций.

Пример 1. Определение размера матрицы и формы матрицы

```
clear
b=zeros(8)
ndims(eye(2,3))
ndims(rand(2,3,4))
[m,n]=size(ones(3,4))
r=ones(3,4,5)
maxdim=length(zeros(7,5,3)) % определяется длина большей
размерности
numel(ones(7,5,3)) % общее количество элементов
Объясните назначение конструкторов: eye,ones,zeros,rand
b(1:3,1:4)=r(3,4,1), b(end:-1:end-2, end:-1:end-3)=r(3,4,3), %
редактирование b –заполнение 1-м и 3-м «листами» r
spy(b)
Объясните назначение конструкторов: eye,ones,zeros
```

Пример 2. Контроль типов

```
clear
a=rand(2,3)
b=isnumeric(a), c=islogical(b) % b –логическая единица
isvector(a(:))
a(2,2)=0/0, isfinite(a)
b=isnumeric(a), c=islogical(b) % b –логическая единица
d=isreal(c) % логическая единица не имеет мнимой части
isinf(1/0), isnan(0/0) % объясните нюансы (/ - slash)
```


Пример 3. Редактирование строк и блоков матриц

```
clc
clear
A=ones(5)% матрица 5 порядка из единиц
A(3:end, 3:end)=5% элементы блока третьего порядка равны пяти
A(1:2, 1:2)=[1,2;3,4] % блок второго порядка в матрице A заменен на
матрицу правой части
C=blkdiag(A,eye(3),rand(7))% построение диагональной матрицы
spy(C) % графическая визуализация ненулевых элементов
C(1:2, :)=[], spy(C)% редуцированы первые две строки
C(:, 2:2:end)=0, spy(C)% обнулены четные столбцы
```

Пример 4. Функции min, max, sum, prod

```
clear
A=diag([0 1 2])+ones(3)
B=prod(A)% определите тип результата
prod(prod(A))% определите тип результата
sum(sum(A))
min(min(A))
max(max(A))
B=[1 -2; 3+i 4]
abs(B), abs(B(2,1))
```

```
sum(B) % по результатам команд
prod(B) % объясните,
prod(prod(B)),%как работают функции min, max, sum, prod, abs,
сравните случаи, когда аргумент является вектором и матрицей
```

Пример 5. Сортировка - sort

```
clear
clc
A =[3 2 1; 7 8 9; 5 4 6]
B = sort(A) % сортирует по столбцам по возрастанию
C = sort(A,'descend') % по убыванию
D = sort(A,2) % вдоль строк (по столбцам) по возрастанию
E = sort(A,2, 'descend') % вдоль строк (по столбцам) по убыванию
```

Пример 6. find поиск элементов ненулевых или соответствующих заданному условию

```
clear
clc
A =[3 0 0; 0 7 0; 0 0 256]
[i, j, x] = find(A) % определяются индексы [i,j] и ненулевые
%значения x матрицы A(i,j), записанные в вектор x
[i, j, x] = find(A>7); % определяются векторы индексов i и j для
значений A(i,j) >7, вектор x состоит из логических единиц
```

Элементы линейной алгебры

Пример 7. Обращение матриц

Если матрица A является квадратной и невырожденной, уравнения AX =E и XA =E (E – единичная матрица) имеют одинаковое решение X . Это решение называется матрицей обратной к A , обозначается через A^{-1} и вычисляется с помощью функции `inv (A)` или `A^(-1)`

```
clear
A= pascal(5) % матрица Паскаля, см. help
d = det (A) % вычисляется определитель A
X = inv (A) % вычисляется обратная матрица к A
A*X == eye(size(A)) % объясните результат (eye - конструктор
единичной матрицы)
B=A*X - eye(size(A)) % самостоятельно постройте, используя
подходящий конструктор, матрицу B
```

Пример 8. Собственные значения и собственные векторы матрицы A – eig(A)

Собственным значением и собственным вектором квадратной матрицы A называются скаляр λ и вектор v, удовлетворяющие условию $Av = \lambda v$

```
clear
A = [4 1 2; ...
3 7 1; ...
2 2 8];
lambda = eig(A) % здесь собственные значения различны
% и выходной параметр - вектор
```

```
[V,D] = eig(A) % собственные значения на главной диагонали
диагональной матрицы D,   и все собственные векторы (с.в.) в
матрице V записаны по столбцам, с. в. независимы
V\A*V % равносильно решению D=A*V*(inv(V)) матричного уравнения
AV=DV
```

Пример 9. Норма и число обусловленности матрицы

```
clear
A= rand(1,4)
norm(A) % норма вектора, без второго уточняющего (тип нормы)
аргумента  совпадает с Евклидовой нормой
%самостоятельно примените остальные предлагаемые нормы, сравните
результат
B=A+A ' % придумайте проверку симметрии матрицы B
cond(B) % norm(B)*norm(inv(B))число обусловленности – характеризует
устойчивость решения системы
rank(B) % ранг матрицы; важно при оценке вырожденности системы
```

Факторизация матриц и решение систем линейных алгебраических уравненийСЛАУ

Систему называют определенной, если она имеет единственное решение.
X = A\B (\ - backslash) - является решением матричного уравнения AX = B (\ - backslash)
X = B/A - является решением матричного уравнения XA = B.

Пример 10. Решение СЛАУ методом Гаусса

```
clear
A = [4 1 2; ...
3 7 1; ...
2 2 8];
b = [7; 11; 12];
x = A\b % в ML для вычисления x по умолчанию реализован алгоритм
Гаусса
% функция mldivide равносильна операции деления (\)
x = mldivide(A, b)
% Проверка точности решения: вычисление невязки или нормы вектора
невязки
b - A*x, norm(b - A*x)
```

Пример 11. Факторизация матриц

```
clear
A = [4 1 2; ...
3 7 1; ...
2 2 8];
b = [7; 11; 12];
[l,u]=lu(A) % факторизация A на произведение l – нижней треугольной
и u – верхней треугольной матриц, такая что l*u=A
[Q,R]=qr(A) % факторизация A; R – верхняя треугольная, Q –
унитарная или в вещественном случае ортогональная (Q*Q'=E) матрицы
такие, что Q*R=A
B=A*A' % B – симметричная матрица с положительными элементами
all(all(B==B'))% объясните, что же здесь проверили (может быть, для
понимания поможет реструктурирование команды)
```

```
R=chol(B) % если B – положительно определенная матрица, R'*R = B, и
R – верхняя треугольная матрица с положительными элементами на
диагонали
```

Пример 12. Решение СЛАУ с помощью процедуры linsolve (см.подробно с.8)

```
clc
clear
eig( [4 1 2; ...
3 7 1; ...
2 2 8];
b = [7; 11; 12];
x=linsolve(A,b) % изучите справку linsolve – как управлять выбором
решателей в зависимости от свойств матрицы системы; предложите
опциональные варианты
```