

# Линейная алгебра

# Основы матричной алгебры

Матрицы и векторы имеют следующую структуру:

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \quad A' = \begin{pmatrix} a_{1,1} & a_{2,1} & a_{3,1} \\ a_{1,2} & a_{2,2} & a_{3,2} \\ a_{1,3} & a_{2,3} & a_{3,3} \end{pmatrix}$$

$$x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \quad x' = (x_1 \quad x_2 \quad x_3)$$

# Матричные операции

Пусть матрица  $A(M \times N)$  умножается на матрицу  $B(N \times K)$ .

Результат – матрица  $C$   $C = A * B$

В покомпонентной записи:

$$c_{i,j} = \sum_{k=1}^N a_{i,k} \cdot b_{k,j}$$

Для того, чтобы умножить матрицу  $A$  на матрицу  $B$  необходимо, чтобы число столбцов матрицы  $A$  равнялось числу строк матрицы  $B$ .

## Специальные операции

$$C_{i,j} = X_i \cdot Y_j$$

На Матлабе это выглядит как:

$$C = X * Y';$$

Здесь X и Y – вектор-столбцы

Аналогично выглядят операции:

$$C_{i,j} = X_i \pm Y_j$$

# Задание

Дана покомпонентная запись. Записать в формате

$$\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} K_{i,j} \cdot x_i \cdot y_j$$

Матлаба:

**A** – матрица, **x** и **y** – вектор-столбцы.

Записать в покомпонентной форме:

$$A \cdot x \cdot x' \cdot A' \quad x' \cdot A' \cdot A \cdot x$$

Какова размерность матрицы и векторов?

Что собой представляет результат?

Записать в формате

Матлаба: (j – мнимая

единица)

$$y_k = \sum_{i=0}^{N-1} x_i \cdot e^{-j \cdot 2\pi \frac{i \cdot k}{N}}$$

# Логическая индексация массивов

Логические вектора, построенные с помощью логических выражений могут служить индексами при доступе к массивам.

Пусть

```
>>x=[1,2,3,4,5,6,7,8];
```

```
>>y=x(x>3)
```

```
Y= 4  5  6  7  8
```

Пусть функция задается разными выражениями при разных значениях аргумента:

$$y = \begin{cases} 0, & x < -5 \\ \sin(x), & x \in [-5, 1] \\ x^2 - 3, & x \in (1, 4] \\ \ln(x), & x > 4 \end{cases}$$

```
>> y=@(x)[0*x((x<-5)),sin(x(a1)),x(a2).^2-3,log(x(a3))]
```

```
>>x=-10:0.1:10;
```

```
>>plot(x,y(x))
```

Обратите внимание как записывается функция, равная нулю на некотором интервале. Как формируется функция из отрезков.

**Постройте график этой функции.**

# Создание матриц с заданными свойствами

**`eye(n)`** – возвращает единичную матрицу размера  $n \times n$ ; **`eye(m,n)`** – размера  $m \times n$ .

**`eye(size(A))`** – возвращает единичную матрицу того же размера, что и **A**.

**`ones(n)`**, **`ones(m,n)`**, **`ones(size(A))`** – возвращает матрицу, все элементы которой единицы.

**`zeros(n)`**, **`zeros(m,n)`**, **`zeros(size(A))`** –



Функция **linspace** формирует *линейный массив равноотстоящих узлов*. Это подобно оператору **:**, но дает прямой контроль над числом точек. Применяется в следующих формах:

- **linspace(a,b)** – возвращает линейный массив из **100** точек, равномерно распределенных между **a** и **b**;
- **linspace(a,b,n)** – генерирует **n** точек, равномерно распределенных в интервале от **a** до **b**.

```
>> linspace(1,10,5)
```

```
ans =
```

```
1.0000 3.2500 5.5000 7.7500 10.0000
```

**logspace(a,b)** – возвращает вектор-

строку из **50** равноотстоящих в

логарифмическом масштабе точек

между декадами  **$10^a$**  и  **$10^b$** ;

**logspace(a,b,n)** – возвращает **n** точек

между декадами  **$10^a$**  и  **$10^b$** ;

```
>> logspace(1,10,5)
```

```
ans =
```

```
Columns 1 through 4:
```

```
10.00000    1778.27941
```

```
316227.76602    56234132.51903
```

```
Column 5:
```

```
100000000000.00000
```

Функция *rand* генерирует массивы случайных чисел, значения элементов которых *равномерно распределены* в промежутке *(0,1)*

Функция *randn* генерирует массив со случайными элементами, распределенными по нормальному закону с нулевым математическим ожиданием и средним квадратическим отклонением, равным 1.

*randn(n), randn(m,n), randn(size(A))*

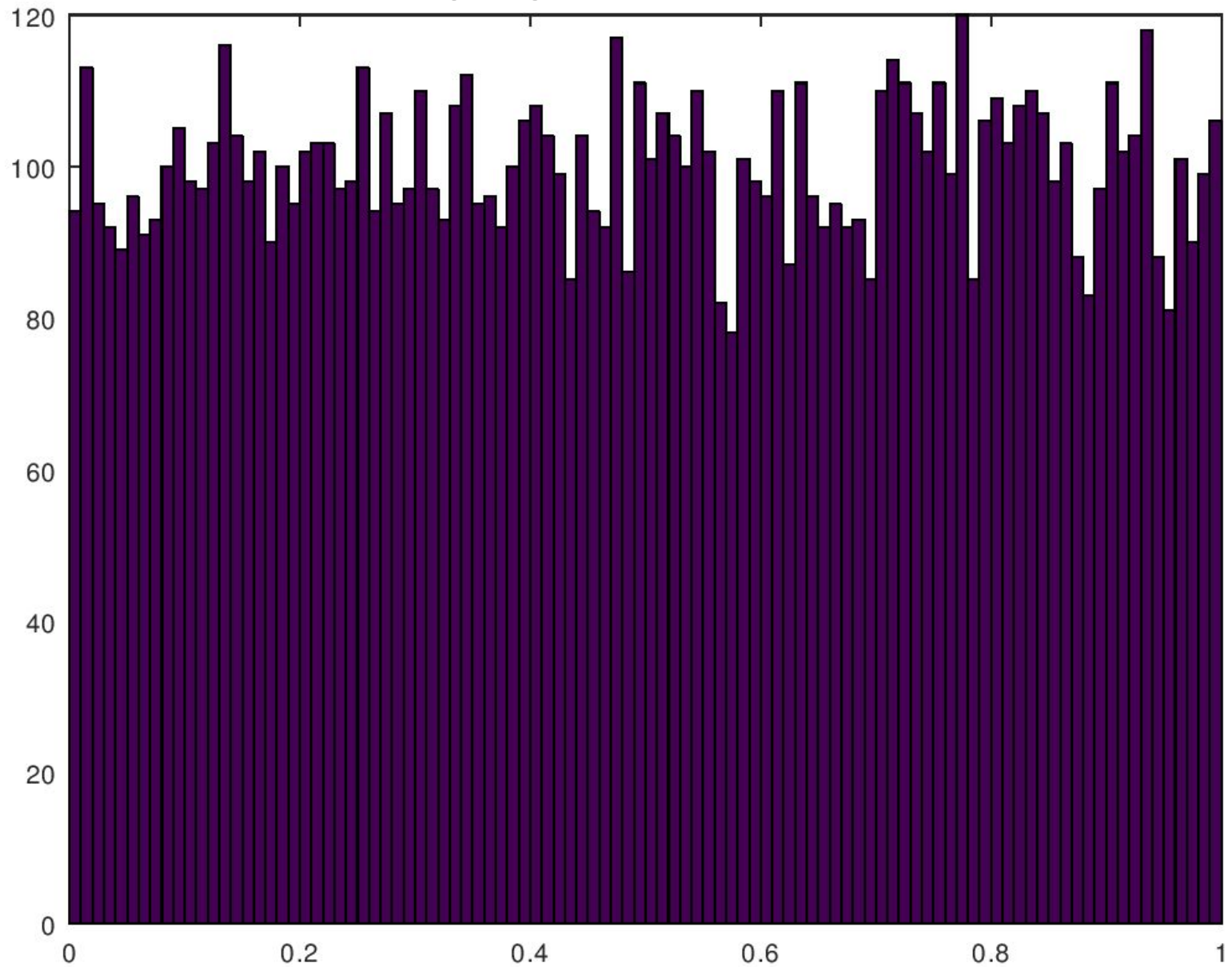
*rand(n), rand(m,n), rand(size(A))*

Проверить распределение случайных чисел можно, построив гистограмму распределения большого количества чисел.

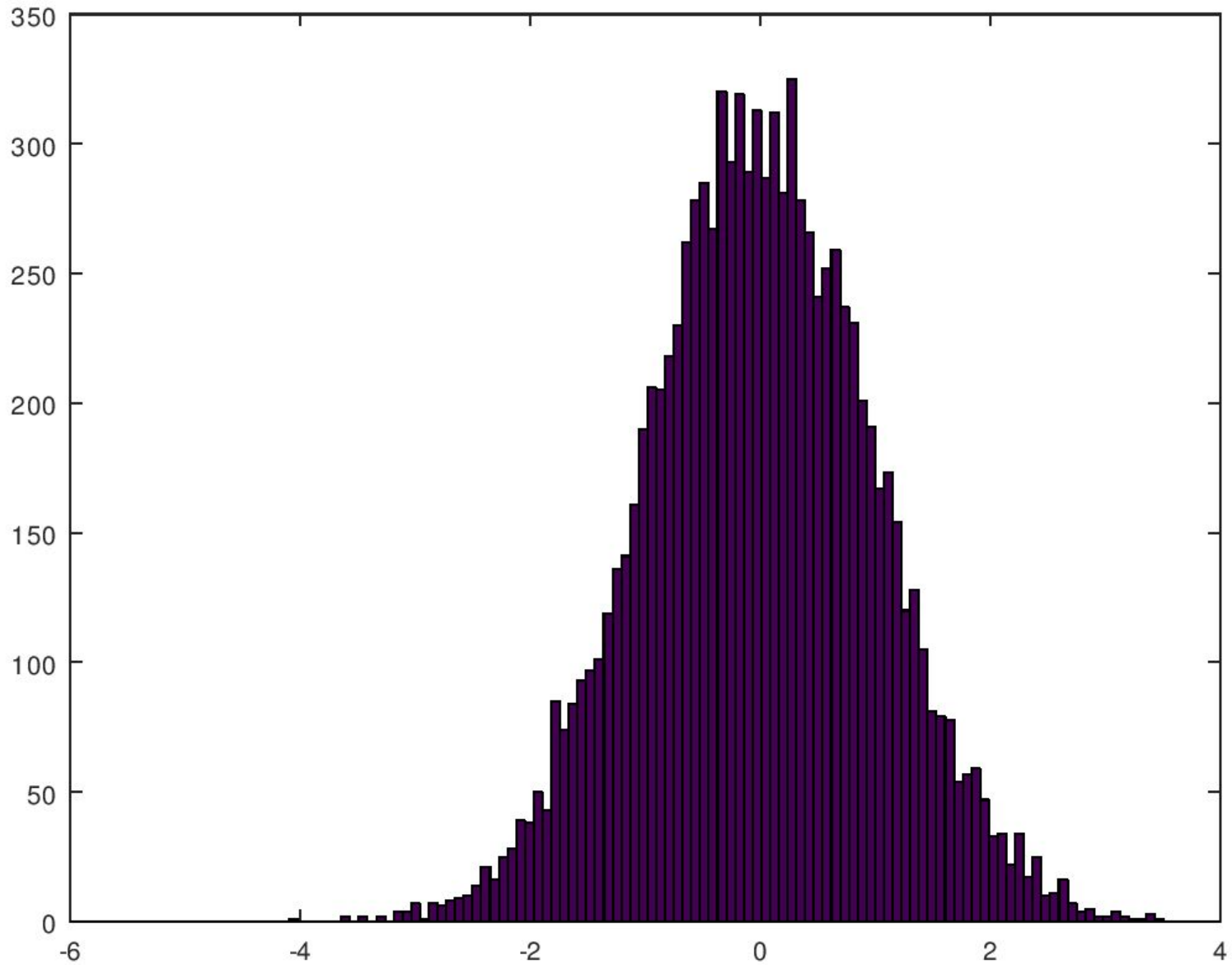
```
>> Y=rand(10000,1); hist(Y,100)
```

```
>> Y=randn(10000,1); hist(Y,100)
```

# rand



# randn



# *Конкатенация матриц*

**C = cat(dim,A,B)** – объединяет массивы

**A** и **B** в соответствии со

спецификацией размерности **dim** и

возвращает объединенный массив;

**dim=1** – горизонтальная конкатенация;

**dim=2** – вертикальная конкатенация;

**dim=3** – многомерный массив

размерности 3 и т.д.



$X = \text{diag}(v, k)$  – для вектора  $v$ ,  
состоящего из  $n$  компонентов,  
возвращает  
квадратную матрицу  $X$  порядка  
 $n + \text{abs}(k)$  с элементами  $v$  на  $k$ ой  
диагонали, при  $k=0$  это главная  
диагональ (из левого верхнего угла  
матрицы в правый нижний угол),  
при  $k>0$  – одна из диагоналей выше  
главной диагонали, при  $k<0$  – одна из  
нижних диагоналей. Остальные  
элементы матрицы – нули;

- **prod(A)** – возвращает произведение элементов массива, если **A** – вектор, или вектор-строку, содержащую произведения элементов каждого столбца, если **A** – матрица;
- **prod(A,dim)** – возвращает вектор (строку или столбец) с произведением элементов массива **A** по столбцам (**dim=1**), по строкам (**dim=2**).

**Пример:**

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

```
A =
```

```
1 2 3 4
```

```
2 4 5 7
```

```
6 8 3 4
```

```
>> B=prod(A)
```

```
B = 12 64 45 112
```

```
>> B=prod(A,2)
```

```
B = 24
```

```
280
```

```
576
```

- **sum(A)** – возвращает сумму элементов массива, если **A** – вектор, или вектор-строку, содержащую сумму элементов каждого столбца, если **A** – матрица;
- **sum(A,dim)** – возвращает сумму элементов массива по столбцам (**dim=1**), строкам (**dim=2**) или иным размерностям, в зависимости от значения скаляра **dim**.

```
>> X=[1 2;3 4]
```

```
X =
```

```
1 2
```

```
3 4
```

```
>> sum(X)
```

```
ans =
```

```
4 6
```

**Возможно создание *пустых*  
матриц, например:**

```
>> M=[]
```

```
M = [](0x0)
```

```
>> M=[M [1,2;3,4]]
```

```
M =
```

```
 1  2
```

```
 3  4
```

# Матричные функции

$\text{expm}(X)$  – возвращает  $\text{exp}(X)$  от квадратной матрицы  $X$ .

$\text{sqrtm}(X), \text{logm}(X)$  – квадратный корень и логарифм от матрицы  $X$ .

Комплексный результат получается, если  $X$  имеет неположительные собственные значения. Пусть

$X = [1, 2; 3, 4]$ ;

Найти:  $\text{exp}(X)$ ;  $\text{expm}(X)$ ;  $\text{sqrt}(X)$ ;  $\text{sqrtm}(X)$ ;

Объяснить отличия.

```
>> S=[1,0,3;1,3,1;4,0,0]
```

```
>> a=expm(S)
```

```
a =
```

31.22028	0.00000	23.37787
38.96594	20.08554	30.05928
31.17049	0.00000	23.42766

```
>> b=logm(a)
```

```
b =
```

1.00000	0.00000	3.00000
1.00000	3.00000	1.00000
4.00000	0.00000	-0.00000



# Умножение матриц

$$C_{i,k} = \sum_{j=1}^N A_{i,j} \cdot B_{j,k} \rightarrow C = A * B;$$

*Это запись в Матлабе*

$$C_{i,j} = X_i \otimes Y_j \rightarrow C = X \otimes Y';$$

*Если X вектор – столбец*

*Y вектор – столбец*

$\otimes$  – любой оператор : + – \*

$$\sum_{i=1}^N X_i \cdot Y_i \rightarrow X' * Y$$

*Скалярное произведение*

# Вычисление нормы и чисел обусловленности матрицы

Норма вектора  $X$  (или, точнее, его  $p$ -норма) задается выражением и вычисляется функцией  $\text{norm}(x,p)$ .

Заметим, что  $\text{norm}(x,2)=\text{sqrt}(x'*x)$

$$\|X\|_p = \left( \sum_k |x_k|^p \right)^{1/p}$$

Пусть  $A$  – матрица. Тогда  $n = \text{norm}(A)$  эквивалентно  $n = \text{norm}(A, 2)$  и возвращает вторую норму, то есть самое большое сингулярное число матрицы  $A$ .

$$n = \text{norm}(A, 1) = \max_i \sum_j |A_{i,j}|$$

$$n = \text{norm}(A, \text{inf}) = \max_j \sum_i |A_{i,j}|$$

В общем случае  $p$ -норма матрицы  $A$  вычисляется как  $\max_x \frac{\|Ax\|_p}{\|x\|_p}$

```
>> A=[2,3,1;1,9,4;2,6,7]
```

```
A =
```

```
 2  3  1
```

```
 1  9  4
```

```
 2  6  7
```

**Проверьте!**

```
>> norm(A)
```

```
ans = 13.735
```

```
>> norm(A,1)
```

```
ans = 18
```

```
>> norm(A,Inf)
```

```
ans = 15
```

Число обусловленности матрицы определяет чувствительность решения системы линейных уравнений к погрешностям исходных данных. Следующая функция позволяет найти число обусловленности матриц:

**cond(X)** – возвращает число обусловленности, основанное на второй норме.

$$\text{cond}(A) = (\max \|Ax\| / \|x\|) / (\min \|Ax\| / \|x\|)$$

```
>> A=hilb(4)
```

```
A =
```

1.00000	0.50000	0.33333	0.25000
0.50000	0.33333	0.25000	0.20000
0.33333	0.25000	0.20000	0.16667
0.25000	0.20000	0.16667	0.14286

```
>> cond(A)
```

```
ans = 15513.73874
```

Для нахождения определителя (детерминанта) и ранга матриц в MATLAB имеются следующие функции:

- **det(X)** – возвращает определитель квадратной матрицы **X**. Если **X** содержит только целые элементы, то результат – тоже целое число. Использование **det(X)=0** как теста на вырожденность матрицы действительно только для матрицы малого порядка с

**Ранг матрицы определяется количеством **сингулярных** чисел, превышающих порог **tol**.**

**Для вычисления ранга**

**используется функция **rank**:**

**$\text{rank}(A)$  – возвращает количество**

****сингулярных** чисел, которые**

**являются**

**большими, чем заданный по**

**умолчанию допуск;**

**$\text{rank}(A, \text{tol})$  – возвращает количество**

****сингулярных** чисел, которые**



```
>> A=hilb(11);
```

```
>> rank(A)
```

```
ans = 10
```

```
>> cond(A)
```

```
ans = 5.2237e+14
```

Вычисление **ортонормированного** базиса матрицы обеспечивают следующие функции:

- **$V = \text{orth}(A)$**  – возвращает ортонормированный базис матрицы **A**. Столбцы **V** определяют то же пространство, что и столбцы матрицы **A**, но столбцы **V** ортогональны, то есть  **$V' * V = \text{eye}(\text{rank}(A))$** . Количество столбцов матрицы **V** равно рангу матрицы **A**.

# Собственные значения и собственные векторы квадратной матрицы

Задача на собственные значения для  
квадратной матрицы имеет вид:

$$A\psi = \lambda\psi$$

Или в покомпонентной записи:

$$\sum_{j=1}^n A_{i,j} \psi_{j,k} = \lambda_k \psi_{i,k}$$

**Совокупность всех собственных векторов, относящихся к одному и тому же собственному значению, вместе с нулевым вектором, образует линейное подпространство.**

**Если вектора  $x_1, x_2, \dots, x_n$  являются собственными и относятся к разным собственным значениям, то векторы  $x_1, x_2, \dots, x_n$  – линейно независимы.**

**Матрица  $A$  приводима к диагональному виду тогда и только тогда, когда существует базис в  $n$ -мерном пространстве, состоящий из собственных векторов. Факторизация матрицы  $A$  — матричное представление уравнения на собственные значения.  $\Lambda$  – диагональная матрица, состоящая из собственных значений;  $\Psi$  – матрица собственных векторов.**

$$A\Psi = \Psi\Lambda \rightarrow A = \Psi\Lambda\Psi^{-1} \quad \text{Факторизация матрицы}$$

Матрица **A** называется неотрицательно определенной, если  $X^H A X \geq 0$  Для любого вектора **X**

Матрица **A** называется симметрической, если:  $A^H = A$  (**n** – символ эрмитова транспонирования, т.е. транспонирования и комплексного сопряжения. **В Матлабе символ '.**)  
Для симметрической и неотрицательно определенной матрицы собственные вектора –

**$[V, \lambda] = \text{eig}(A)$  – вычисление матрицы собственных векторов ( $V$ ) и диагональной матрицы собственных значений ( $\lambda$ ) от матрицы  $A$**   
 **$\gg [V,D]=\text{eig}(A2)$**

1	2	3	V =		
4	5	6	-0.231971	-0.785830	0.408248
7	8	9	-0.525322	-0.086751	-0.816497
			-0.818673	0.612328	0.408248

**$\gg \text{diag}(D)$**  **Каков ранг этой матрицы?**

ans =

1.6117e+01 -1.1168e+00 -1.3037e-15

**>> V'\*V**

**ans =**

**1.0000e+00 -2.7343e-01 5.5511e-17  
-2.7343e-01 1.0000e+00 -9.9920e-16  
5.5511e-17 -9.9920e-16 1.0000e+00**

**Матрица A2 – не симметрическая и  
не является неотрицательно  
определенной, поэтому  
собственные вектора не  
ортономированные**



# Теплицева матрица

$$A = \begin{pmatrix} a & b & c & d \\ e & a & b & c \\ f & e & a & b \\ g & f & e & a \end{pmatrix}$$

**На всех диагоналях одинаковые значения**

`toeplitz (c)`

`toeplitz (c, r)`

**Возвращает матрицу Теплица  
созданную из вектора  $c$  (в первом  
случае).**

**Во втором случае верхняя  
треугольная из вектора  $c$ , а нижняя  
треугольная из вектора  $r$ .**

**Создадим матрицу  $K_{i,j} = \rho^{(i-j)^2}$ ,  $i, j \in \overline{0, n-1}$**

**>> r=0.9;**

**>> n=5;a=(0:n-1).^2;**

**>> c=r.^a;**

**>> K=toeplitz (c)**

**K =**

<b>1.00000</b>	<b>0.90000</b>	<b>0.65610</b>	<b>0.38742</b>	<b>0.18530</b>
<b>0.90000</b>	<b>1.00000</b>	<b>0.90000</b>	<b>0.65610</b>	<b>0.38742</b>
<b>0.65610</b>	<b>0.90000</b>	<b>1.00000</b>	<b>0.90000</b>	<b>0.65610</b>
<b>0.38742</b>	<b>0.65610</b>	<b>0.90000</b>	<b>1.00000</b>	<b>0.90000</b>
<b>0.18530</b>	<b>0.38742</b>	<b>0.65610</b>	<b>0.90000</b>	<b>1.00000</b>

# Найдем собственные вектора и собственные значения этой матрицы

```
>> [V,D]=eig(K);
```

```
V =
```

```
-1.6166e-01  3.8166e-01  5.7288e-01 -5.9526e-01  3.8168e-01  
4.9416e-01 -5.9526e-01 -1.7637e-01 -3.8166e-01  4.7402e-01  
-6.7775e-01  7.8822e-16 -5.3048e-01  1.9868e-17  5.0916e-01  
4.9416e-01  5.9526e-01 -1.7637e-01  3.8166e-01  4.7402e-01  
-1.6166e-01 -3.8166e-01  5.7288e-01  5.9526e-01  3.8168e-01
```

```
>> diag(D)'
```

```
ans =
```

```
0.00057261  0.01525073  0.18139281  1.14334725  3.65943661
```

```
>> V'*V
```

```
ans =
```

```
1.0000e+00  1.1102e-16  6.9389e-17 -4.1633e-17 -9.0206e-17  
1.1102e-16  1.0000e+00 -1.6653e-16  5.5511e-17  5.5511e-17  
6.9389e-17 -1.6653e-16  1.0000e+00 -1.1102e-16  5.5511e-17  
-4.1633e-17  5.5511e-17 -1.1102e-16  1.0000e+00 -2.2204e-16  
-5.5511e-17  5.5511e-17  2.7756e-17 -2.2204e-16  1.0000e+00
```

**Так как эта теплицева матрица симметрическая и неотрицательно определенная, то собственные векторы ортогональны.**

# Скорость выполнения

**Файл test1**

```
A=rand(1000,1000);
```

```
B=rand(1000,1000);
```

```
for i=1:1000
```

```
    for j=1:1000
```

```
        A(i,j)=A(i,j).^B(i,j);
```

```
    end
```

```
end
```

**Файл test2**

```
A=rand(1000,1000);
```

```
B=rand(1000,1000);
```

```
A=A.^B;
```

```
A=rand(1000,1000);
B=rand(1000,1000);
for i=1:1000
    for j=1:1000
        A(i,j)=A(i,j).^B(i,j);
    end
end
-----
>> clear
>> tic,test1,toc
Elapsed time is 25.106
seconds.
```

```
A=rand(1000,1000);
B=rand(1000,1000);
A=A.^B;
-----
-
>> clear
>> tic,test2,toc
Elapsed time is
0.224 seconds.
Выигрыш во
времени
выполнения 100
раз!
```

## Тест в MatLab 7.5.0

```
>> clear
```

```
>> tic,test1,toc
```

```
Elapsed time is 0.340472 seconds.
```

```
>> clear
```

```
>> tic,test2,toc
```

```
Elapsed time is 0.254457 seconds.
```

**В Матлабе интерпретатор  
оптимизирован лучше, чем в Octave.**



# Запрограммировать на матлабе:

Пусть A и B – квадратные матрицы,  
имеющие обратные.

Записать на Матлабе задачу на  
обобщенные собственные значения  
(использовать eig() )

$$\sum_{j=1}^N A_{ij} \cdot X_{j,k} = \lambda_k \sum_{i=1}^N B_{i,j} \cdot X_{j,k}$$

$$A=[1,2;2,3] \quad B=[25,5.5;5.5,121]$$