KU LEUVEN

MATLAB

Array indexing

1

Topics

- Indexing
  - How to select elements

KU LEUVEN

2

Indexing: referencing elements

- Extract individual entries by specifying the indices inside **round** brackets **()**.
- Extract several entries at once by specifying
  - An array,
  - use the : operator to extract all entries along a certain dimension.

KU LEUVEN

3

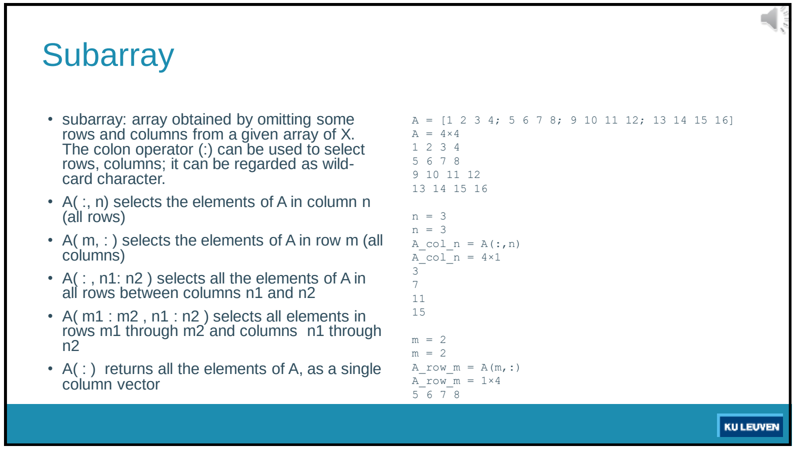
Subvector

- **(m:n)** refers to elements m through n
- **([n1, n2, …])** specify the elements to select in a vector
- **(:)** refers to all elements of the vector

```
vec_1 = rand(1,7)
vec_1 = 1×7
0.9831 0.3015 0.7011 0.6663 0.5391 0.6981 0.6665

vec_select = vec_1(2:4)
vec_select = 1×3
0.3015 0.7011 0.6663

vec_select_multi_1 = vec_1([1,3,7])
vec_select_multi_1 = 1×3
0.9831 0.7011 0.6665

vec_select_multi_2 = vec_1([1,3,5:7])
vec_select_multi_2 = 1×5
0.9831 0.7011 0.5391 0.6981 0.6665

vec_all = vec_1(:)
vec_all = 7×1
0.9831
0.3015
0.7011
0.6663
0.5391
0.6981
0.6665
```

KU LEUVEN

4

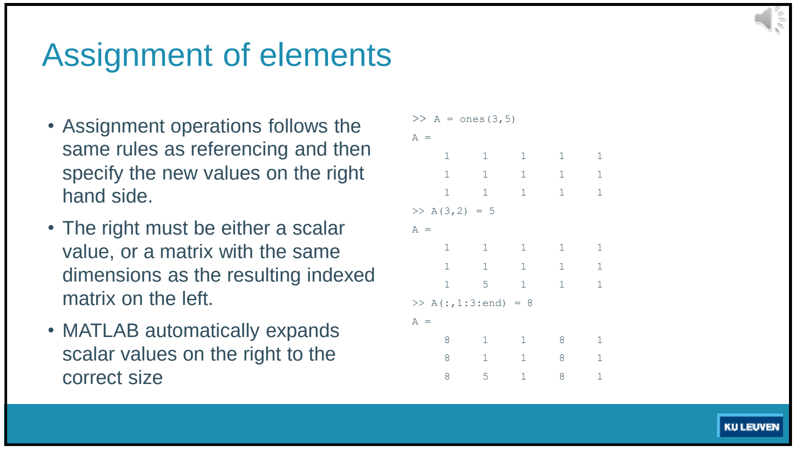


Subarray

- subarray: array obtained by omitting some rows and columns from a given array of X. The colon operator (:) can be used to select rows, columns; it can be regarded as wild-card character.
- A(:, n) selects the elements of A in column n (all rows)
- A(m, :) selects the elements of A in row m (all columns)
- A(: , n1: n2) selects all the elements of A in all rows between columns n1 and n2
- A(m1 : m2 , n1 : n2) selects all elements in rows m1 through m2 and columns n1 through n2
- A(:) returns all the elements of A, as a single column vector

```
A = [1 2 3 4; 5 6 7 8; 9 10 11 12; 13 14 15 16]
A = 4×4
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16

n = 3
n = 3
A_col_n = A(:,n)
A_col_n = 4×1
3
7
11
15

m = 2
m = 2
A_row_m = A(m,:)
A_row_m = 1×4
5 6 7 8
```

5

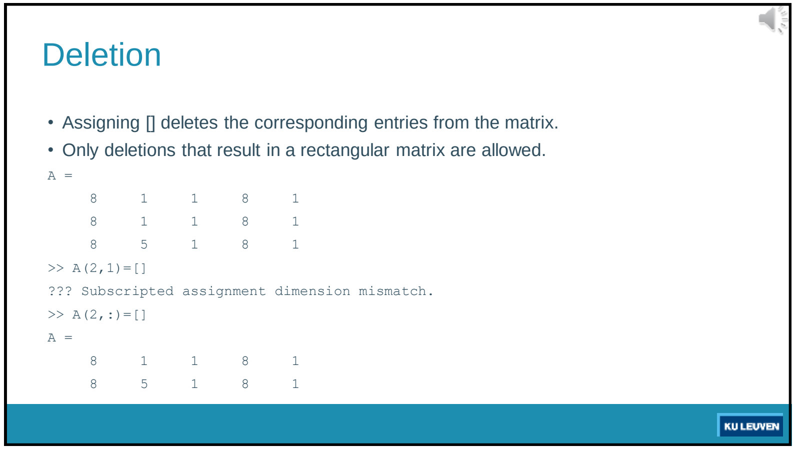


Assignment of elements

- Assignment operations follows the same rules as referencing and then specify the new values on the right hand side.
- The right must be either a scalar value, or a matrix with the same dimensions as the resulting indexed matrix on the left.
- MATLAB automatically expands scalar values on the right to the correct size

```
>> A = ones(3,5)
A =
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
>> A(3,2) = 5
A =
     1     1     1     1     1
     1     1     1     1     1
     1     5     1     1     1
>> A(:,1:3:end) = 8
A =
     8     1     1     8     1
     8     1     1     8     1
     8     5     1     8     1
```

7

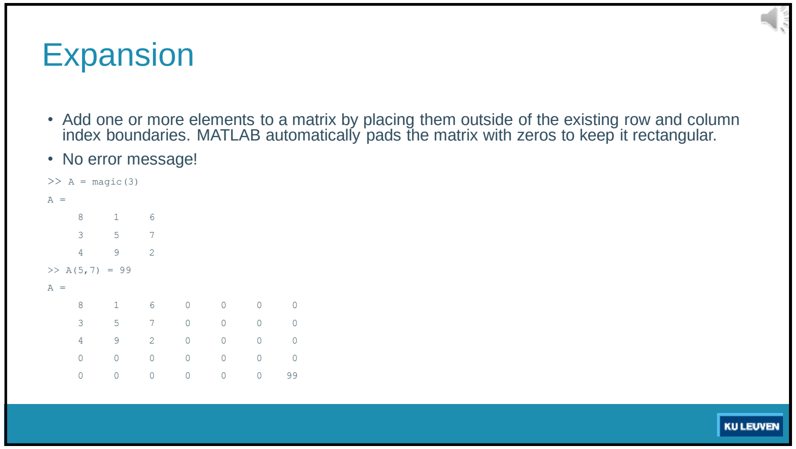


Deletion

- Assigning [] deletes the corresponding entries from the matrix.
- Only deletions that result in a rectangular matrix are allowed.

```
A =
     8     1     1     8     1
     8     1     1     8     1
     8     5     1     8     1
>> A(2,1)=[]
??? Subscripted assignment dimension mismatch.
>> A(2,:)=[]
A =
     8     1     1     8     1
     8     5     1     8     1
```

8



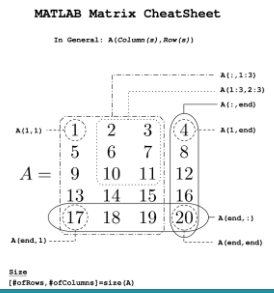
Expansion

- Add one or more elements to a matrix by placing them outside of the existing row and column index boundaries. MATLAB automatically pads the matrix with zeros to keep it rectangular.
- No error message!

```
>> A = magic(3)
A =
     8     1     6
     3     5     7
     4     9     2
>> A(5,7) = 99
A =
     8     1     6     0     0     0     0
     3     5     7     0     0     0     0
     4     9     2     0     0     0     0
     0     0     0     0     0     0     0
     0     0     0     0     0     0    99
```

9

Matrix cheat sheet



MATLAB Matrix CheatSheet

In General: A(*Column(s)*,*Row(s)*)

$$A = \begin{matrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \\ 17 & 18 & 19 & 20 \end{matrix}$$

Size

[#ofRows,#ofColumns]=size(A)

http://alexandergutenkunst.net/blog/ accessed 2016

Demo / recap

- *File: array_indexing.mlx used in screencast matlab_array_indexing*