Lecture 3: Element-wise Operations, Indexing

A8B17CAS

Miloslav Čapek

Department of Electromagnetic Field
Czech Technical University in Prague
Czech Republic
miloslav.capek@fel.cvut.cz

September 20
Winter semester 2022/23

Outline

1. Element-wise Operations
2. Indexing

Warm Up: Complex Power Delivered To a Circuit

Consider the impedance matrix $\mathbf{Z}$ and feeding voltage vector $\mathbf{V}$ are known.

Evaluate:

- Current:

$$\mathbf{I} = \mathbf{Z}^{-1}\mathbf{V}$$

- Total power delivered to the system:

$$P = \frac{1}{2}\mathbf{I}^{\mathrm{H}}\mathbf{V}.$$

- Is the circuit, represented by $\mathbf{Z}$, active or passive? Judge from the real part of P…

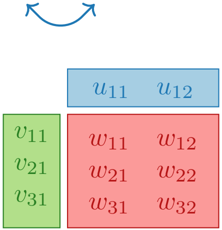
$$\mathbf{Z} = Z_0 \begin{bmatrix} 1+1\mathrm{j} & 0 & 2 \\ 0 & 2-1\mathrm{j} & -1\mathrm{j} \\ 2 & -1\mathrm{j} & 3 \end{bmatrix}, \quad \mathbf{V} = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$$

Exercise

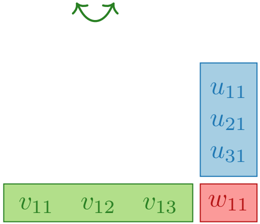
Vector and Matrix Operations

- Remember that matrix multiplication is not commutative, i.e. $\mathbf{AB} \neq \mathbf{BA}$.
- Remember that vector-vector multiplication results in

$$\mathbf{v}_{M,1}\mathbf{u}_{1,N} = \mathbf{w}_{M,N}$$



$$\mathbf{v}_{1,M}\mathbf{u}_{M,1} = \mathbf{w}_{1,1}$$



…pay attention to the dimensions of matrices!

Element-by-element Vector Product

- It is possible to multiply arrays of the same size in the element-by-element manner in MATLAB.
  - Result of the operation is an array.
  - Size of all arrays are the same, *e.g.*, in the case of 1×3 vectors:

$$\mathbf{a} = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}$$

```
>> a*b
```

$\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} * \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix} \rightarrow$

```
Error using *
(Inner matrix dimensions must agree.)
```

```
>> a.*b
```

$\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} .* \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix} \rightarrow \begin{bmatrix} a_1 b_1 & a_2 b_2 & a_3 b_3 \end{bmatrix} = [a_i b_i]$

Element-by-element Matrix Product

- If element-by-element multiplication of two matrices of the same size is needed, use the `.*` operator.
  - It is so called *Hadamard product/element-wise product/Schur product*: $\mathbf{A} \circ \mathbf{B}$.
  - These two cases of multiplication are distinguished:

```
>> A*B
```

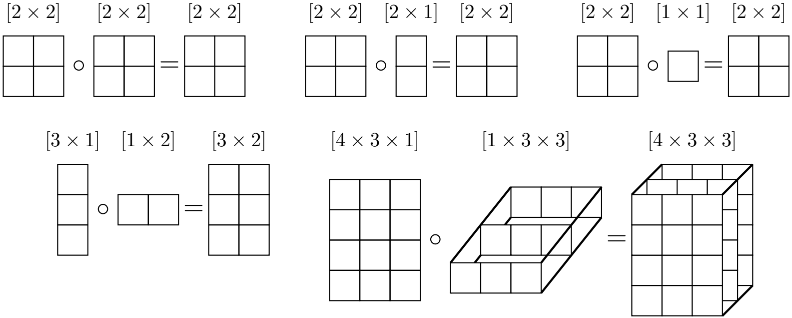
$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} * \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \rightarrow \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{bmatrix}$$

```
>> A.*B
```

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} .* \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \rightarrow \begin{bmatrix} A_{11}B_{11} & A_{12}B_{12} \\ A_{21}B_{21} & A_{22}B_{22} \end{bmatrix}$$

Compatible Array Size

- Since MATLAB version R2016b most two-input (binary) operators support arrays that have *compatible sizes*.
  - Variables have compatible sizes if their sizes are either the same or one of them is 1 (for all dimensions).
- Examples:
  - ∘ represents arbitrary two-input element-wise operator (+, −, .*, ./, &, <, ==, ...).

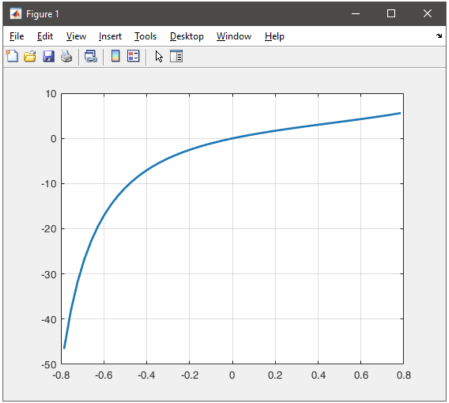


Element-wise Operations I.

- Elements-wise operations can be applied to vectors as well in MATLAB. Element-wise operations can be usefully combined with vector functions.
- It is possible, quite often, to eliminate 1 or even 2 for-loops!!!
- These operations are extremely efficient → allow use of so called vectorization (*see later*).

$$f(x) = \frac{10}{(x+1)}\tan(x), \quad x \in \left[-\frac{\pi}{4}, \frac{\pi}{4}\right]$$

```
x = -pi/4:pi/100:pi/4;
fx = 10 ./ (1 + x) .* tan(x);
plot(x, fx)
grid on
```



Element-wise Operations II.

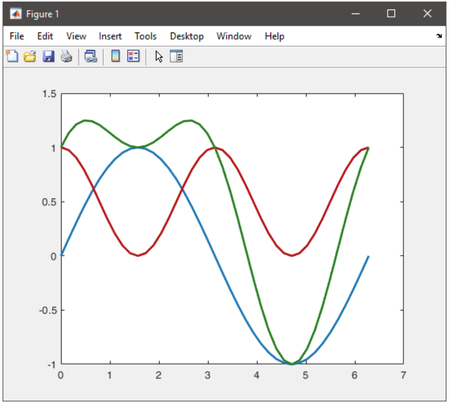
- Evaluate functions of the variable $x \in [0, 2\pi]$:

$$f_1(x) = \sin(x)$$
$$f_2(x) = \cos^2(x)$$
$$f_3(x) = f_1(x) + f_2(x)$$

- Evaluate the functions in evenly spaced points of the interval, the spacing is $\Delta x = \pi/20$.

- For verification use:

```
plot(x, f1, x, f2, x, f3)
```



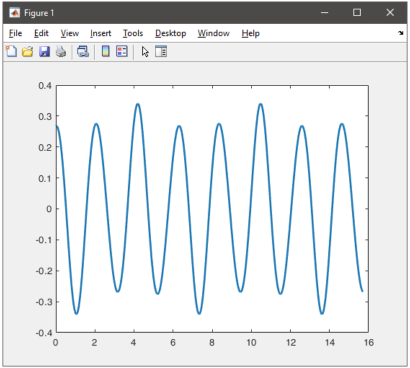
Exercise

Element-wise Operations III.

- Depict graphically following functional dependency in the interval $x \in [0, 5\pi]$.
- Plot the result using the following function:

$$f_4(x) = \frac{-\cos(3x)}{\cos(x)\sin\left(x - \frac{\pi}{5}\right) - \pi}$$

```
plot(x, f4)
```



- Explain the difference in the way of multiplication of matrices of the same size.

```
>> A*B
```

```
>> A.*B
```

```
>> A'.*B
```

Exercise

What Element-wise Operation is Correct?

- Consider the operation `a1^a2`. Is this operation applicable to the following cases?

`a1` – matrix	`a2` – scalar
`a1` – matrix	`a2` – matrix
`a1` – matrix	`a2` – vector
`a1` – scalar	`a2` – scalar
`a1` – scalar	`a2` – matrix
`a1`, `a2` – matrix	`a1.^a2`

You can always create the matrices `a1`, `a2` and make a test ...

Exercise

Indexing in MATLAB

- Mastering indexing is crucial for efficient work with MATLAB.
- Up to now, we have been working with entire matrices, quite often we need, however, to access individual elements of arrays.
- Two ways of accessing matrices/vectors are distinguished.
  - Access using round brackets "( )".
    - Matrix indexing: refers to position of elements in a matrix.
  - Access using square brackets "[ ]".
    - Matrix concatenation: refers to element's order in a matrix.

Indexing in MATLAB I.

- Let's consider following triplet of matrices.
  - Execute individual commands and find out their meaning.
  - Start from inner part of the commands.
  - Note the meaning of the pointer `end`.

$$\mathbf{N}_1 = \begin{bmatrix} -5 \\ 0 \\ 5 \end{bmatrix} \qquad \mathbf{N}_2 = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 6 & 8 & 10 \\ 2 & 3 & 5 & 7 & 11 \end{bmatrix} \qquad \mathbf{N}_3 = \begin{bmatrix} 11 & 12 & 13 & 14 \\ 22 & 24 & 26 & 28 \\ 33 & 36 & 39 & 42 \\ 44 & 48 & 52 & 56 \end{bmatrix}$$

```
N1 = (-5:5:5)';   N2 = [1:5;2:2:10;primes(11)];   N3 = (1:4)'*(11:14);
```

```
N1(1:3)
N1([1 2 3])
N1(3:-1:1)
N1([1 3])
N1([1 3].')
N1([1 3]).'
N1([1; 3])
N1([1 3],1)
```

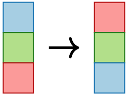
```
N2(1, 3)
N2(3, 1)
N2(1, end)
N2(end, end)
N2(1, :)
N2(1, :).'
N2(:, 2)
N2(:, 3:end)
```

```
N3(2:3, [1 1 1]) % like repmat
N3(2:3, ones(1,3))
N3(2:3, ones(3,1))
N3([N2(2,1:2)/2 4], [2 3])
N3([1 end], [1:4 1:2:end])
N3(:, :, 2) = magic(4)
N3([1 3], 3:4, 3) = ...
   [1/2 -1/2; pi*ones(1, 2)]
```

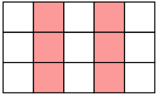
Indexing in MATLAB II.

- Remember the meaning of `end` and the application of colon operator "**:**".

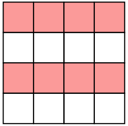
- Flip the elements of the vector $\mathbf{N}_1$ without use of `fliplr`/`flipud` functions.



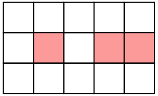
- Select only the even columns of $\mathbf{N}_2$.



- Select only the odd rows of $\mathbf{N}_3$.



- Select 2nd, 4th and 5th column of 2nd row of $\mathbf{N}_2$.



- Create matrix $\mathbf{A}$ of size 4×3 containing numbers 1 to 12 (row-wise, from left to right).

Exercise

Indexing in MATLAB IV.

- Which one of the following returns corner elements of a matrix $\mathbf{A}$ (10×10)?

```
A([1, 1], [end, end])
A({[1, 1], [1, end], [end, 1], [end, end]})
A([1, end], [1, end])
A(1:end, 1:end)
```

Exercise

Deleting and Replacing Elements of a Matrix

Empty matrix is a crucial concept in deleting elements of a matrix.

```
T = [];
```

We want to:

- Remove 2nd row of a matrix **A**.

```
A(2, :) = []
```

- Remove 1st, 2nd and 5th column of a matrix **A**.

```
A(:, [1 2 5]) = []
```

- Replace 3rd column of a matrix **A** (of size $M \times N$) by a vector $\boldsymbol{x}$ (length M).

```
A(:, 3) = x
```

- Replace 2nd, 4th and 5th row of a matrix **A** by three rows of a matrix **B** (number of columns of both **A** and **B** is the same).

```
A([2 4 5], :) = B(1:3, :)
```

Deleting, Adding and Replacing Matrices

- Which of the following deletes the first and the last column of matrix **A** (6×6)?
  - Create your own matrix and give it a try.

```
A[1, end]    = 0
A(:, 1, end)  = []
A(:, [1:end]) = []
A(:, [1 end]) = []
```

- Replace 2nd, 3rd and 5th row of matrix **A** by first row of matrix **B**.
  - Assume the number of columns of matrices **A** and **B** is the same.
  - Consider the case where **B** has more columns than **A**.
  - What happens if **B** has less columns than **A**?

Exercise

Linear Indexing

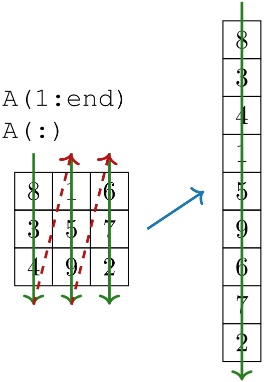
- Elements of an array of arbitrary number of dimensions and arbitrary size can be referred using simple index.
  - Indexing takes place along the main dimension (column-wise) then along the secondary dimension (row-wise) etc.

```
A = magic(3)
```

A =

8	1	6
3	5	7
4	9	2

```
A(1:end)
A(:)
```




```
A([1 5])
```

8	1	6
3	5	7
4	9	2

```
A([1 5], :)
```

Index in position 1 exceeds array bounds (must not exceed 3).

Questions?

A8B17CAS
miloslav.capek@fel.cvut.cz

September 20
Winter semester 2022/23

This document has been created as a part of A8B17CAS course.
Apart from educational purposes at CTU in Prague, this document may be reproduced, stored, or transmitted only with the prior permission of the authors.