Logical Arrays and Masks

Section 4.3 Textbook by Chapra

Exercise in document: “003-Logical Arrays & Masks.pdf”

Read: “LogicalArrays.pdf”

Boolean data have two possible values

Logical Data Type

true—(1)

false—(0)

```
a=2; b=1;
T = b<a;
F = a<b;
FF= a==b
tf=[ T, F, FF ]
```

```
x=[2 1]
tf=[x(2)<x(1), x(1)<x(2),x(1)==x(2)];
```

```
x=[1,0,0]
tf=logical(x)
```

```
tf=[true,false,false]
```

Initialize Logical Arrays

```
>> A = [false, true; false, false] <E>
A =

   0   1
   0   0
```

these are not numbers

```
>> A = [1, 0, 2; -1, 0, 0; 0, 1.5, -10.0];
>> B = logical(A) <E>

B =
   1   0   1
   1   0   0
   0   1   1
```

Array A contains numbers
Array B contains logical data

```
>>A = false(3,2)

A=

   0   0
   0   0
   0   0
```

```
>> A= true(3)     % same as true(3,3)

A =

   1   1   1
   1   1   1
   1   1   1
```

A Logical Array is a collection of logical values, i.e., a set of true and false represented by 0s and 1s. These 0s and 1s are not numbers. They are Boolean values

Logical Array Example:

Cronstruct the following logical array:
c = [0 1 0 1 0 1 0 1 0 1 0]

```
clc, clear
 n=11;
for ii=1:n
    if mod(ii,2)==0
        c(ii)=true;
    else
        c(ii)=false;
    end
end
disp('c=');
disp(c);
```

```
clc, clear
 n=11;
for ii=1:n
        c(ii)=mod(ii,2)==0;
end
disp('c=');
disp(c);
```

'if' is GONE!

```
clc, clear
n=11;
ii=[1:1:n];
c= mod(ii,2)==0;
disp('c=');
disp(c);
```

'loop' is GONE!

Logical Arrays and Numbers

Example:

```
clc, clear
n = 11;
ii= [1:1:n];
c = 2.*(mod(ii,2)==0)   % number= (number . * logical)
```

OUTPUT

```
c =

    0    2    0    2    0    2    0    2    0    2    0
```

The logical array (mod(ii,2)==0) converts to numbers after their multiplication by 2 (a number). Therefore, c is a numerical array, not longer a logical array

Masks

- Logical arrays have a very important special property—they serve as masks for arithmetic operations
- A mask is an array that selects particular elements of another array for use in an operation
- The specified operation will be applied to the selected elements, and not to the remaining elements
- To create a mask use the logical array as index of the targeted array, whose elements you want to manipulate

Mascaras sirven para “enmascarar” los elementos que no queremos que entren en efecto y permitir que actuen los que si queremos.

Mask, example:

Square root the values greater than 5 in the numerical array **a**

Consider the following solution:

a = [4 5 6 7 8]; ← numerical array

b = a>5 ← b is logical array, b = [0 0 1 1 1];

a(b) = sqrt(a(b)) % Last two: a(a>5)=sqrt(a(a>5))

Same number of elements as array a

<u>OUTPUT</u>

a = [4 5 $\sqrt{6}$ $\sqrt{7}$ $\sqrt{8}$]

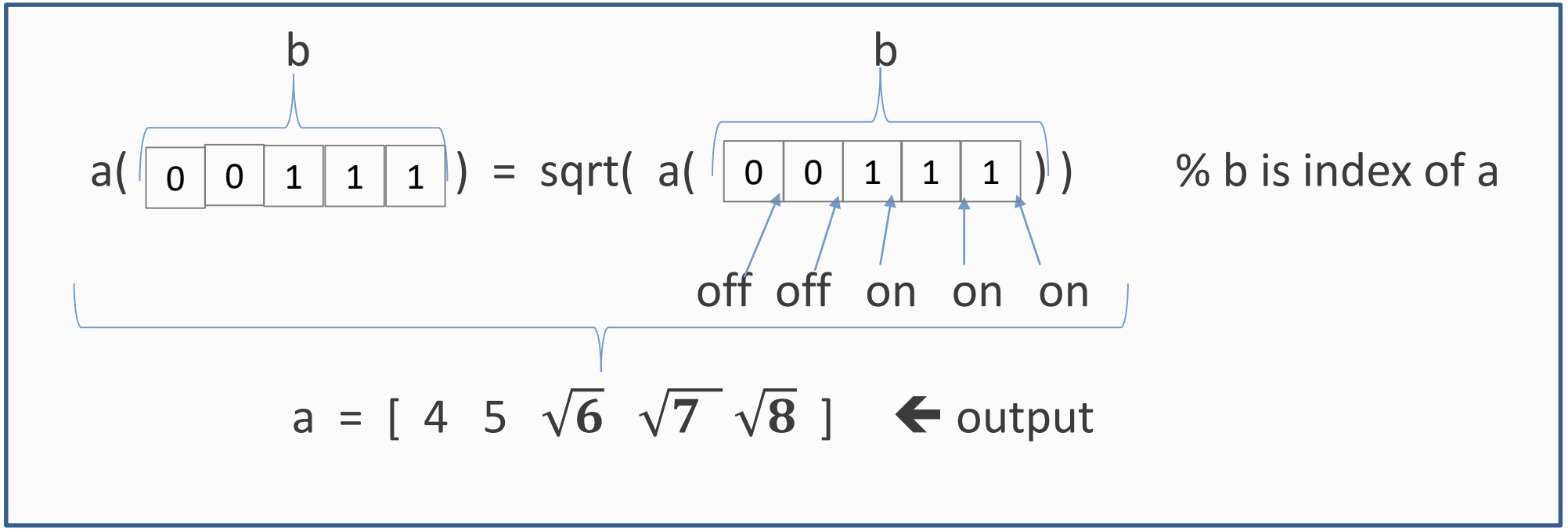
Code:

```
a = [ 4   5   6   7   8 ];     ← numerical array

b = a>5;                       ← b is logical array, b = [ 0   0   1   1   1 ];

a(b) = sqrt(a(b));             ← now b is a mask
```

Same number of elements as array a



Masks in 2D array

$$a = [1,2,3; 4,5,6; 7,8,9] \quad \Rightarrow \quad a = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Square root the values greater than 5 in the numerical array **a**

$$\mathrm{b} = \mathrm{a} > 5 \qquad \mathrm{b} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

```
a =   [1, 2, 3;
       4, 5, 6;
       7, 8, 9];
b=a>5;
a(b)= sqrt(a(b))
```

$$a(b) \qquad a = \begin{bmatrix} \mathrm{off} & \mathrm{off} & \mathrm{off} \\ \mathrm{off} & \mathrm{off} & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

$$sqrt\big(a(b)\big) \qquad a = \begin{bmatrix} \mathrm{off} & \mathrm{off} & \mathrm{off} \\ \mathrm{off} & \mathrm{off} & \sqrt{6} \\ \sqrt{7} & \sqrt{8} & \sqrt{9} \end{bmatrix}$$

run it and explain it

$$a(b) = sqrt\big(a(b)\big) \qquad a = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & \sqrt{6} \\ \sqrt{7} & \sqrt{8} & \sqrt{9} \end{bmatrix}$$

Mask: Square elements greater than 5 in the 2D a array

```
a =    [1, 2, 3;
        4, 5, 6;
        7, 8, 9];
for ii=1:1:3
        for jj=1:1:3
            if a(ii,jj)>5
                    a(ii,jj)= sqrt(a(ii,jj));
            end
        end
 end
```

With loops and if statement

Please don't be confused with the code above it doesn't use logical arrays and masks at all. This is just to show you how long the procedure is without logical arrays/masks.

Example

% Find the sum of all values multiples of 3 or 5 below 100.

Standard code

```
clc,clear
total = 0;
for ii = 1:100
   if mod(ii,3) == 0 || mod(ii,5) == 0
      total = total + ii;
   end
end
fprintf('\nTotal: %d \n', total );
```

Logical Arrays & Masks

```
clc,clear
ii = [1:100]';
LA = mod(ii,3) == 0 | mod(ii,5) == 0;
m3or5=ii(LA);
disp(['Total: ',num2str(sum(m3or5))]);
```

Example

1. Create a 100-elements array containing the values, 1, 2,…, 100. Then square root all elements whose values are greater than 50 using for loop and if constructs
2. Create a 100-elements array containing the values, 1, 2,…, 100. Then square root all elements whose value is smaller than 50 using a logical array & masks
3. Create a 100-elements array containing the values, 1, 2,…, 100. Then square root all elements whose value is smaller than 50 using a logical array & masks. Then square all elements whose value is greater than or equal to 50.
4. Create a 100-elements array containing the values, 1, 2,…, 100. Then take the square of all elements whose values are between 50 and 75 using logical arrays

(see solutions on notes page)

Example

5. We want square root element in a two-dimensional array “a” whose values are greater than 5, and to square the remaining elements. The code for this operation using loops and branches is shown in the following slide and in next one it’s a solution with logical arrays and masks.

$$a = \begin{bmatrix} 1 & 3 & 4 & 7 \\ 7 & 8 & 2 & 3 \\ 5 & 2 & 9 & 6 \end{bmatrix}$$

Logical Functions

all	Determine if all array elements are nonzero or true
any	Determine if any array elements are nonzero
false	Logical 0 (false)
find	Find indices and values of nonzero elements
islogical	Determine if input is logical array
logical	Convert numeric values to logicals
true	Logical 1 (true)