KU LEUVEN

MATLAB

relational and logical operators

Topics

- Use relational operators to test two values
- Compare relationships using logical operators
- Use logical expressions to find specific elements in an array
- Searching

KU LEUVEN

Logical data type

- MATLAB has a logical data type, with the possible values:
  - 1, representing true,
  - 0, representing false.
- Logicals are produced by relational and logical operators/functions and by the functions `true` and `false`, or the `logical` class cast
- `a = true`
- `b = false`
- `c = logical(variable)`

KU LEUVEN

is* logical functions

- many useful logical functions whose names begin with `is` (check with tab)

isempty	Test for empty array
isequal	Test if arrays are equal
isinf	Detect infinite array elements
isinteger	Test for integer array
islogical	Test for logical array
isscalar	Test for scalar array
issorted	Test for sorted vector

KU LEUVEN

Relational Operators

- Used to compare two numeric values
- Returns a value of true or false.
- In MATLAB,
  - 1 = true (any non-zero number);
  - 0 = false;
  - Logical data type

Relational Operators	
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	equals to
~ =	not equal to

KU LEUVEN

Relational Operators

- Comparisons between scalars produce logical 1 if the relation is true and logical 0 if it is false.
- Comparisons are also defined between arrays of the same dimension and between an array and a scalar.
- For array-array comparisons corresponding pairs of elements are compared, while for array–scalar comparisons the scalar is compared with each array element.

KU LEUVEN

Relational Operators

- The MATLAB relational operators compare corresponding elements of arrays with equal dimensions.
- Relational operators always operate element-by-element.
- example

```
A = [2 7 6;9 0 5;3 0.5 6];
B = [8 7 0;3 2 5;4 -1 7];
A == B
ans =
0 1 0
0 0 1
0 0 0
```

KU LEUVEN

Operator Precedence

1. Parentheses ()
2. Transpose (.'), power (.^), complex conjugate transpose ('), matrix power (^)
3. Unary plus (+), unary minus (-), logical negation (~)
4. Multiplication (.*), right division (./), left division (.\), matrix multiplication (*), matrix right division (/), matrix left division (\)
5. Addition (+), subtraction (-)
6. Colon operator (:)
7. Less than (<), less than or equal to (<=), greater than (>), greater than or equal to (>=), equal to (==), not equal to (~=)
8. Element-wise AND (&)
9. Element-wise OR (|)
10. Short-circuit AND (&&)
11. Short-circuit OR (||)

Good Practice: use parentheses to make the intention completely clear

KU LEUVEN

Operator Precedence

- When relational operators are present:
  - All arithmetic operations are performed first (in their particular order)
  - Then the relational operators are evaluated.
- Example 1

```
(2*3) > (4+1);
```

  - The multiplication and addition are first:

```
6 > 5
```

  - The relational operator is evaluated:

```
6 is greater than 5, so this returns 1 (true)
```

  - Result is the logical value, 1, is returned

KU LEUVEN

Logical Operators

- Logical Operators:
  - Provide a way to combine results from Relational Expressions or between logical values
  - Returns a value of true or false.
- Evaluated after all other operators have been performed (lowest precedence priority)

Logical Operators		
&	AND	true if and only if its arguments are true
|	OR	true if and only if either argument is true
~	NOT	inverse
xor	XOR	true if and only if its arguments differ

KU LEUVEN

Truth table

a	b	T_{AND}
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

AND Operation

a	b	T_{OR}
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

OR Operation

a	b	T_{xor}
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

XOR Operation

a	T_{NOT}
0	**1**
1	**0**

NOT Operation

KU LEUVEN

Logical Operators

- AND: &
  - Returns true if two expressions being compared are true.
  - Returns false if any of the two is false.
- OR: |
  - Returns true if any of the two expressions is true.
  - Returns false only if the two are both false.
- NOT: ~
  - Returns true if the single expression is false.
  - Returns false if the single expression is true.

KU LEUVEN

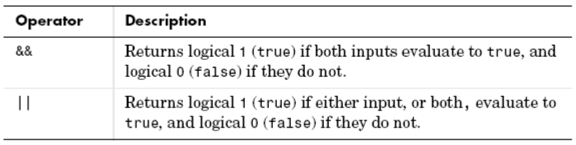
Examples:

- Assume: `a=7; b=4; c=3;`
- `~(a==3*b)`
  - Evaluates: `3*b = 12`
  - Evaluates: `(a==12)` and result is `false`
  - Evaluates `~(false)` and result is `true`
  - Returns *`ans = 1 (true)`*
- `a > 5 & b > 5`
  - Evaluates `(a>5)` and `(b>5)` separately.
  - One returns `true`, the other returns `false`.
  - Since both are not true, the expression returns `false`.
- `a == 7 | b ==1`
  - Evaluates `(a==7)` and `(b==1)` separately
  - One returns `true` and the other returns `false`
  - Since at least one is true, the expression returns `true`

Logical Operators: more

Short-Circuit Operators

- They are *short-circuit* operators in that they evaluate their second operand only when the result is not fully determined by the first operand.
- Works only with scalar logical values.



Operator	Description
`&&`	Returns logical `1` (`true`) if both inputs evaluate to `true`, and logical `0` (`false`) if they do not.
`||`	Returns logical `1` (`true`) if either input, or both, evaluate to `true`, and logical `0` (`false`) if they do not.

A Common Mistake

- You will not get into trouble if you make sure that Logical Operators are always used with logical values.
- `A > B & C` (where `A=10, B=5, C=0`)
  - This looks like a relational expression asking if `A` is greater than both `B` and `C` which should be true for these values.
  - Here is what really happens:
    - **`A>B`** is evaluated as true
    - result (**`true`**) is logically **`AND`**ed with **`C`**
    - Since MATLAB treats any zero numeric as false, it will mistakenly treat C as a logical and the result will be **`false`**
  - The CORRECT form is: `(A > B) & (A > C)` and this returns a `true` result.

KU LEUVEN

Logical Values in Assignments

- True/False values can be assigned to variables and then treated numerically in MATLAB.
- The variables will be assigned the value that is returned from relational and/or logical operators.
- The variables will thus have a value of 1 or 0.

  Example: `a=7; b=4; c=3;`
  - `x = a > 2;`
    - Then x = 1;
  - `y = b==5;`
    - Y will be equal to 0.

KU LEUVEN

Some Other Warnings…

- Using numeric values to represent logicals can have some strange repercussions…
- Never try to use NaN in a relational or logical expression because NaN has no value (can be considered to have all values)

```
aa = [ 10 16 16 16 17 9 11]
>> aa > 15
ans =     0     1     1     1     1     0     0
>> aaa = ans
>> aa(aaa)
ans =
    16    16    16    17
>> aaai = uint16(aaa)
aaai =
      0      1      1      1      1      0      0
>> aa(aaai)
??? Subscript indices must either be real positive
integers or logicals.
```

```
>> nan==nan
ans =
     0
>> inf==inf
ans =
     1
```

KU LEUVEN

Reduce Logical Arrays to a Single Value

- Aggregating logical values
  - `any( )`
  - `all( )`
- all:
  - returns 1 if all the elements of the vector are nonzero and 0 otherwise
  - matrix:
    - operates on columns of A, returning a row vector of 1s and 0s
    - returns 1 if all elements of the column are logical true
- any:
  - returns 1 is at least 1 element in the vector is nonzero
  - matrix:
    - operates on columns of A, returning a row vector of 1s and 0s
    - Returning logical true if any element of that column is true.

KU LEUVEN

Logical Operators: more

```
ages = [10 62 18 27]
anyKids = any(ages <= 12)
anySeniors = any(ages >= 65)
anyKids =

     1

anySeniors =

     0
```

```
allAdults = all(ages >= 18)
noSeniors = all(ages <= 65)
allAdults =

     0

noSeniors =

     1
```

KU LEUVEN

Logical Conditions and arrays

- **Find Array Elements That Meet a Condition**
- filter the elements of an array by applying one or more conditions to the array

```
>> rng(0)
A = randi(15,3)

A =

    13    14     5
    14    10     9
     2     2    15

>> A > 5

ans =

     1     1     0
     1     1     1
     0     0     1

>> A(A>5)

ans =

    13
    14
    14
    10
     9
    15
```

KU LEUVEN

Logical Conditions and arrays

- `find( )`
- `find` returns the indices corresponding to the nonzero elements of a vector.
- find applied to a matrix A, the index vector corresponds to A regarded as a vector of the columns stacked one on top of the other (that is, A(:)), and this vector can be used to index into A
- information about the **locations** of the array elements that meet a condition rather than their actual values.

```
>> A

A =

    13    14     5
    14    10     9
     2     2    15

>> find(A>5)

ans =

     1
     2
     4
     5
     8
     9
```

KU LEUVEN

Searching

- The find command “finds” members of an array that meet a criteria. The result of the command is a list of element numbers.
- http://blogs.mathworks.com/loren/2009/01/20/more-ways-to-find-matching-data/

```
>> grades = -5: 10: 105
grades =
    -5     5    15    25    35    45    55    65
   75    85    95   105
>> set1 = find(grades>100 | grades <0)
set1 =
     1    12
>> set2 = find(grades>=0 & grades <=100)
set2 =
     2     3     4     5     6     7     8     9
   10    11
>> grades(set1)
ans =
    -5   105
>> grades(set2)
ans =
     5    15    25    35    45    55    65    75
   85    95
```

KU LEUVEN

Sorting

- Sort array elements in ascending or descending order
- Syntax
  - `B = sort(A)`
  - `B = sort(A,dim)`
  - `B = sort(...,mode)`
  - `[B,IX] = sort(...)`
- `B = sort(A)` sorts the elements along different dimensions of an array, and arranges those elements in ascending order.
- If A is a ...sort(A) ...
  - Vector: Sorts the elements of A.
  - Matrix: Sorts each column of A.

```
>> a = randperm(8)

a =

     8     2     7     4     3     6     5     1

>> b = sort(a)

b =

     1     2     3     4     5     6     7     8

>> A

A =

     3     7     5

     0     4     2

>> sort(A)

ans =

     0     4     2

     3     7     5

>> sort(A,1)

ans =

     0     4     2

     3     7     5

>> sort(A,2)

ans =

     3     5     7

     0     2     4
```

KU LEUVEN

Demo / recap

- *File: arrays_logical.mlx*
- logical expressions on arrays

Relational Operators	
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	equals to
~ =	not equal to

Logical Operators	
&	AND
|	OR
~	NOT
xor	XOR

KU LEUVEN