MATLAB

Unit 6-Lecture 20

SVKM'S NMIMS Deemed to be UNIVERSITY | NAVI MUMBAI

BTech (CSBS) -Semester VII

30 Septemer 2022, 09:35AM

Control Flow and Operators

1) relational and logical operators
2) “if ... end" structure
3) “for ... end" loop
4) “while ... end" loop
5) other flow structures
6) operator precedence
7) saving output to a file

Relational operators

Relational operators compare the elements in two arrays and return logical true or false values to indicate where the relation holds.

`==`	Determine equality
`>=`	Determine greater than or equal to
`>`	Determine greater than
`<=`	Determine less than or equal to
`<`	Determine less than
`~=`	Determine inequality
`isequal`	Determine array equality
`isequaln`	Determine array equality, treating NaN values as equal

Relational operators: Examples

Examples: If $x = [1 \;\; 5 \;\; 3 \;\; 7]$ and $y = [0 \;\; 2 \;\; 8 \;\; 7]$, then

`k = x < y`	results in $k = [0\ 0\ 1\ 0]$	because $x_i < y_i$ for $i = 3$,
`k = x <= y`	results in $k = [0\ 0\ 1\ 1]$	because $x_i \leq y_i$ for $i = 3$ and 4,
`k = x > y`	results in $k = [1\ 1\ 0\ 0]$	because $x_i > y_i$ for $i = 1$ and 2,
`k = x >= y`	results in $k = [1\ 1\ 0\ 1]$	because $x_i \geq y_i$ for $i = 1, 2,$ and 4,
`k = x == y`	results in $k = [0\ 0\ 0\ 1]$	because $x_i = y_i$ for $i = 4$, and
`k = x ~= y`	results in $k = [1\ 1\ 1\ 0]$	because $x_i \neq y_i$ for $i = 1, 2,$ and 3.

Logical operators

`&`	Find logical AND
`~`	Find logical NOT
`|`	Find logical OR
`xor`	Find logical exclusive-OR
`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)

Logical operators: Examples

Examples: For two vectors $x = [0\ \ 5\ \ 3\ \ 7]$ and $y = [0\ \ 2\ \ 8\ \ 7]$,

`m = (x>y)&(x>4)`	results in $m = [0\ 1\ 0\ 0]$, because the condition is true only for x_2,
`n = x|y`	results in $n = [0\ 1\ 1\ 1]$, because either x_i or y_i is nonzero for $i = [2\ 3\ 4]$,
`m = ~(x|y)`	results in $m = [1\ 0\ 0\ 0]$, which is the logical complement of `x|y`, and
`p = xor(x,y)`	results in $p = [0\ 0\ 0\ 0]$, because there is no such index i for which x_i or y_i, but not both, is nonzero.

Logical operators: builtin functions

In addition to these logical operators, there are many useful built-in logical functions, such as:

`all`	true (= 1) if all elements of a vector are true, *Example:* `all(x<0)` returns 1 if each element of x is negative.
`any`	true (= 1) if any element of a vector is true, *Example:* `any(x)` returns 1 if any element of x is nonzero.
`exist`	true (= 1) if the argument (a variable or a function) exists,
`isempty`	true (= 1) for an empty matrix,
`isinf`	true for all infinite elements of a matrix,
`isfinite`	true for all finite elements of a matrix,
`isnan`[3]	true for all elements of a matrix that are not a number (NaN), and
`find`	finds indices of nonzero elements of a matrix. *Examples:* `find(x)` returns [2 3 4] for `x=[0 2 5 7]` and `[r,c]=find(A>100)` returns the row and column indices i and j of A, in vectors r and c, for which $A_{ij} > 100$.

To complete this list of logical functions, we just mention `isreal`, `issparse`, `isstr`, and `ischar`.

Elementary math functions

Trigonometric functions

`sin`, `sind`	sine,	`sinh`	hyperbolic sine,
`asin`, `asind`	inverse sine,	`asinh`	inverse hyperbolic sine,
`cos`, `cosd`	cosine,	`cosh`	cyperbolic cosine,
`acos`, `acosd`	inverse cosine,	`acosh`	inverse hyperbolic cosine,
`tan`, `tand`	tangent,	`tanh`	hyperbolic tangent,
`atan`, `atand`	inverse tangent,	`atanh`	inverse hyperbolic tangent,
`atan2`	four-quadrant $\tan^{-1}$,		
`sec`, `secd`	secant,	`sech`	hyperbolic secant,
`asec`, `asecd`	inverse secant,	`asech`	inverse hyperbolic secant,
`csc`, `cscd`	cosecant,	`csch`	hyperbolic cosecant,
`acsc`, `acscd`	inverse cosecant,	`acsch`	inverse hyperbolic cosecant,
`cot`, `cotd`	cotangent,	`coth`	hyperbolic cotangent,
`acot`, `acotd`	inverse cotangent, and	`acoth`	inverse hyperbolic cotangent.

Elementary math functions

The angles given to these functions as arguments must be in *radians* for `sin`, `cos`, etc., and in *degrees* for `sind`, `cosd`, etc. Thus, `sin(pi/2)` and `sind(90)` produce the same result. All of these functions, except `atan2`, take a single scalar, vector, or matrix as input argument. The function `atan2` takes two input arguments, `atan2(y,x)`, and produces the four-quadrant inverse tangent such that $-\pi \leq \tan^{-1}\frac{y}{x} \leq \pi$. This gives the angle a rectangular to polar conversion.

Examples: If `q=[0 pi/2 pi]`, `x=[1 -1 -1 1]`, and `y=[1 1 -1 -1]`, then

`sin(q)`	gives `[0 1 0]`,
`sinh(q)`	gives `[0 2.3013 11.5487]`,
`atan(y./x)`	gives `[0.7854 -0.7854 0.7854 -0.7854]`, and
`atan2(y,x)`	gives `[0.7854 2.3562 -2.3562 -0.7854]`.

Exponential Functions

`exp`	exponential, *Example:* `exp(A)` produces a matrix with elements $e^{(A_{ij})}$. So how do you compute e^A? See the next section.
`log`	natural logarithm, *Example:* `log(A)` produces a matrix with elements $\ln(A_{ij})$.
`log10`	base 10 logarithm, *Example:* `log10(A)` produces a matrix with elements $\log_{10}(A_{ij})$.
`sqrt`	square root, *Example:* `sqrt(A)` produces a matrix with elements $\sqrt{A_{ij}}$. But what about $\sqrt{A}$? See the next section.
`nthroot`	real nth root of real numbers, *Example:* `nthroot(A,3)` produces a matrix with elements $\sqrt[3]{A_{ij}}$.

Complex Functions

`abs`	absolute value, *Example:* `abs(A)` produces a matrix of absolute values $\lvert A_{ij} \rvert$.
`angle`	phase angle, *Example:* `angle(A)` gives the phase angles of complex A.
`complex`	constructs complex numbers from given real and imaginary parts, *Example:* `complex(A,B)` produces $A + Bi$.
`conj`	complex conjugate, *Example:* `conj(A)` produces a matrix with elements $\bar{A}_{ij}$.
`imag`	imaginary part, *Example:* `imag(A)` extracts the imaginary part of A.
`real`	real part, *Example:* `real(A)` extracts the real part of A.

Round off Functions

`fix`	round toward 0, *Example:* `fix([-2.33 2.66])` $= [-2 \quad 2]$.
`floor`	round toward $-\infty$, *Example:* `floor([-2.33 2.66])` $= [-3 \quad 2]$.
`ceil`	round toward $+\infty$, *Example:* `ceil([-2.33 2.66])` $= [-2 \quad 3]$.
`mod`	modulus after division; `mod(a,b)` is the same as `a-floor(a./b)*b`, *Example:* `mod(26,5)` $= 1$ and `mod(-26,5)` $= 4$.
`round`	round toward the nearest integer, *Example:* `round([-2.33 2.66])` $= [-2 \quad 3]$.
`rem`	remainder after division, `rem(a,b)` is the same as `a-fix(a./b)*b`, *Example:* If `a=[-1.5 7]`, `b=[2 3]`, then `rem(a,b)` $= [-1.5 \ 1]$.
`sign`	signum function, *Example:* `sign([-2.33 2.66])` $= [-1 \quad 1]$.

Matrix Functions

`expm(A)`	finds the exponential of matrix A, e^A,
`logm(A)`	finds $\log(A)$ such that $A = e^{\log(A)}$, and
`sqrtm(A)`	finds $\sqrt{A}$.