Application of Matlab for Finance

September 7, 201

Today's Class

- User-defined functions

Logical Expressions

- A logical expression tests a conditional statement among arguments by using relational operator or logical functions
- There are six basic relational operators:

$>=$

  - equal to: $==$
  - not equal to: $\sim=$

- Logical function: `isempty`, `isnan`, `a`
- Logical expressions have only two possible out
  - TRUE with value of 1
  - FALSE with value of 0

Logical Operators

The logical operators operate between simple expressions in order to creat

-

l

- |

evaluate to true, and logical 0 (false) if neither of the

- NOT ∼ A : returns logical 1 (true) if A evaluat logical 0 (false) if it evaluates to true.

Logical Indexing

```
A = 100;
B = 10
res = A > B % 1 as true
```

-

```
A = [
B = A > 2 % [
A(B) % 5, the 3rd element
A(B) = 10 % replaces the 3rd element with 10
```

- B = [0,0,1,0] is a matrix of logical output f n each element of A.
- A(B) indexes the matrix A with the logical matrix B, returning a vector of elements where A>2 is true. → element A(3)=5.
- A(B)=10 reassigns the true element with new value 10. It changes matrix A into A = [1,−5,10,2].

Logical Indexing

```
A = [1,-5,5,2];
I = find(A>2) % returns 3
A(I) % returns 5
A
```

- condition.
- `I = find(A>2)` returns the index value element of matrix A is true for the judgement

- `A(I)` index the matrix `A` with `I` to read `A(3)`, which is 5;
- `A(I)=10` reassigns the true element with new value 10, `A(3)=10`. It changes matrix A into `A = [1,-5,10,2]`.

Logical Indexing

```
A = [1,-5,5,2];
B = A>0 & A<10; % B = [1,0,1,1
A(B) % returns [1,5,2]
A(B) = NaN % change A = [NaN,-5,NaN,NaN]
i
A
% A = A
```

- Now it is a compounded logical expression for 0<A<10
- A(B)=NaN replaces the logical true element Not-a-Number. NaN is often seen for missing da
- isnan(A) return logical output 1 if any eleme zero if it contains a number.
- A(isnan(A))=[] deletes all NaN element, leading A = -5
- A = A(∼isnan(A)) does the same thing by reassigning A as the element(s) that is not NaN.

Flow Control

-
-
- Main flow control structures:
  - `if`
  - `switch` (Appendix)

If structure

- The if structure evaluates a logical expression and executes a group of statements based on the value of the expression.

```
if x < 0
    disp('x is negative');
end
```

- If the logical expression is TRUE, then all the state the if and end lines are EXECUTED.
- If the logical expression is FALSE, then all the statements between the if and end lines are SKIPPED.

else and elseif

- **if..else** ⇒ There is no logical expression behind else. The statements associated with else are executed only if the preceding logical expression behind if is false
- **if..elseif** There is a logical expression behind the elseif, and will be
- ne

```
if logical_expression1
    statement1
elseif logical_expression2
    statement2
else
    statement3
end
```

else and elseif examples

- **x = input(.)** asks the user to input a value via a message printed in the command window ('Enter a number: ')
- The value of your input is assigned to variable x for the following operation
- Note: Interactive operations cannot be operated in live editors.

```
x = i
i

e

end
```

```
x = input('Enter a number: ');
if x < 0
   disp('x is negative');
elseif x > 0
   disp('x is positive');
else
   disp('x is zero');
end
```

Exercises a.1

- Create a simple if structure that returns the square root of a random number x.
  - use function randn() to generate the random number x

```
x = randn(1) %generate a random number

if x ≥ 0
    res = sqrt(x)
else % x<0
    error('Negative number has no square root.')
end
```

Exercises a.2

- For a random number x, let the user choose 1 of the following 3 calculations on x: (1) the absolute value; (2) the square; (3)the

- [illegible] ntrol
- display error messages if:
  - x is negative in the square root calculation
  - the user input a control value is not 1, 2 or 3

- display the value of `control, x, res`

Exercises a.2

```
control = input('Please choose your calculation: 1 for ...
    abs, 2 for square, 3 for square root:  ');
x = randn(1);
i

e

elseif control == 3
    if x > 0
        res = sqrt(x);
    else
        error('Negative number has no squ
    end
else
    error('Invalid control variable.')
end
```

Display Results

- disp(.) prints contents in the command window. The content can either be
  - a variable or a matrix disp(X)
  - or a text message with quotation disp('Hello World')

-

  - num2str(.) converts numbers to strin
    together with texts inside ' '.

```
clc % clear commond window
disp(['Your choice of calculation is ', num2str(control)])
disp(['x is ', num2str(x), ' and the result is ' ...
    ,num2str(res)])
```

Display Results

- fprintf(formatSpec, A1, A2, ...) also displays results A1, A2, .., but with a format defined in formatSpec

- '%' starts the formatting operator.
- 3 '%' marks mean 3 variables inputs to displa
- '%d', '%.2f' and '%.2f' print the firs second and third values in the output as a floating ith 2 digits '.2'.
- '\n' starts a new line.

Loop structure

-
-
    - `while`

The for loop

```
for index = start_val:increment:end_val
    statements % condition on the value of index
end
```

- index
  - index starts at value start_val ch time until reach the end_val when t
  - increment can be omitted if increase ste

```
for index = start_val:end_val
    statements
end
```

The for loop

- Create a 100 × 1 matrix A, whose row element is its row number plus 5.

```
A = z
f

e
```

- `index = 1, 2..., 100`
- for `index=1`, do `A(index,1)=inde`
- move to `index=2`, do `A(index,1)=` 7
- move to `index=3`, do `A(index,1)=index+5` ⇒ `A(3,1)=3+5=8`
- repeat the same calculation for each value of `index` until it reaches `100`

The while loop

The while loop structure executes a statement or group of statements repeatedly, as long as the controlling logical expression is true (that is, if it evaluates to logical 1).

```
w [illegible]
[illegible]
e [illegible]

i = 0; % initial value of i
while (i < 10)
    i = i + 1; %overwrite i = i+1
    disp(i)
end
```

As long as i<10, repeat operation i=i+1 until i=10

Exercises b.1

- Generate a 1000 × 1 vector of a random variable x.
-
-
  -
  - function log() calculate the natural lo
- With the calculated ln price, calculate the log ret
  - log return in finance as $r_t = ln(P_t)$
  - return of the starting date is zero $r_0 = 0$.

Exercise b.1

```
x = randn(1000,1);
price = x(x>0); % price is elements of x that are positive

T = l
l

% c
for t = 1: T;
    ln_p(t,1) = log(price(t,1));
end

% calculate log return
ln_r = zeros(T,1);
for t = 2: T; %1st day return = 0
    ln_r(t,1) = ln_p(t)-ln_p(t-1);
end
```

Exercise b.2

- Use a while loop to count the number of uniformly distributed realizations (use the rand(.)) between 0 and 1 that it takes to add

```
%
m
count = 0;
while (my_sum < 20)
    temp = rand(1);
    my_sum = my_sum +temp;
    count = count + 1;
end
disp(count);
```

User-defined functions

```
function [ out1, out2] = my_func( input1, input2 )
    % function body, do some stuff here
e
```

- on
- [out1, out2] are the declared outputs of the fu
- my_func is the name of the function, so as the name of the .
- input1, input2 are the required inputs the fu
- The function body performs calculations and produce
- The function needs to be located in the current working dictionary to be called.

Exercises c

(c.1) Create a function called `myabs(.)` that returns the absolute value of a input.

(c.2) Cr
i o get n!.

(c.3) Cr at both
output the maximum, minimum and average o

- `stats_1(.)` outputs three values resp
- `stats_2(.)` outputs one vector that st

- Note: Distinguish what is the function file and wh
command.

Exercise c.1

Function

```
function [ res ] = myabs( x )
% This function computes the absolute value
% X: the input can be scalar, vector or matrix
% r
r
e
```

Main C

```
%% Exericse c.1
% here is the main command to call the function
% call function on input -10
myabs(-10)

% call function on input t and store output y
t = -0.8
y = myabs(t)
```

Exercise c.2

Function

```
function [ res ] = myfact( n )
% This function performs factorial calculation
% n: the input can be scalar
% r
r

end
```

Main Command

```
% here is the main command to call the function
% call function on input 10
myfact(10)

% call function on input n and store output y
n = 77
y = myfact(n)
```

Exercises c.3

Function

```
function [max_val, min_val, avg_val ] = stats_1( vec )
max_val = max(vec);
min_val = min(vec);
avg_val = mean(vec);
end
```

```
f
r
end
```

Main Command

```
% here is the main command to call the function
% stats_1: three outputs for max, min and avg
x = [1,2,3,4,5];
[a, b, c] = stats_1(x)

% stats_2: 1 output stores the max, min and avg
res = stats_2(x)
```

TakeAway

- Logical expression
-
-

  - Note: `log(.)` can be perform on the entir than using the loop.

- Function:
  - It is a separate .m file from your main code file.
  - It has to be shown in the current folder to be called.

Appendix: Switch structure

The switch structure executes certain statements based on the value of the expression, which takes only the values specified in difference case scenario

```
s[illegible]
[illegible]
[illegible]
[illegible]
        statement2
        ...
    case value10
        statement10

    % executes if expression does not match any case
    otherwise
        default_statement
end
```

Appendix: Switch structure

```
mynumber = input('Enter a number:');

s[illegible]
[illegible]
[illegible]
[illegible]
        disp('zero');
    case 1  % mynumber == 1
        disp('positive one');
    otherwise % mynumber is not -1 or 0 or 1
        disp('other value');
end
```

Appendix: Exercise a.3

Perform the same tasks as in exercise a.2 with a `switch` structure.

- but set `x=0; res=0` if the user gives invalid control input

```
control = input('Please choose 1 for abs, 2 for squared, ...
    3 for square root: ');
x = randn(1);
s




    case 3
        if x > 0
            res = sqrt(x)
        else
            error('Negative number ha
        end
    otherwise
        error('Invalid control variable.')
        x = 0;
        res = 0;
end
```