Boolean algebra

Variable of boolean type can have only two values
- true
- false

There are three logical operators which are used in boolean algebra

- logic not, Matlab
  - true = false
  - false = true

- logic and, Matlab
  - A ∧ B = {true, if A=true and B=true, false, otherwise}

- logic or, Matlab
  - A ∨ B = {false, if A=false and B=false, true, otherwise}
Boolean algebra

Variable of boolean type can have only two values
- true (Matlab uses 1 to indicate it, actually everything but zero)
- false (Matlab uses 0)

There are three logical operators which are used in boolean algebra
- ¬ - logic not, Matlab
  - true = false
  - false = true
- ∧ - logic and, Matlab
  \[ A \land B = \begin{cases} 
  \text{true, if } A=\text{true and } B=\text{true,} \\
  \text{false, otherwise} 
\end{cases} \]
- ∨ - logic or, Matlab
  \[ A \lor B = \begin{cases} 
  \text{false, if } A=\text{false and } B=\text{false,} \\
  \text{true, otherwise} 
\end{cases} \]
Boolean operators precedence in Matlab

If $A = false$, $B = true$, $C = true$

$A | \sim B \& C$

\sim has highest precedence, then \&, and then |

Thus $A | \sim B \& C = false$
If $A = \text{false}$, $B = \text{true}$, $C = \text{true}$

$A|(\neg B \& C)\\n\neg$ has highest precedence, then $\&$, and then $|$ 

$A|((\neg B)\&C)$

Thus

$A|\neg B \& C = \text{false}$

“Cat is an animal and cat is not an animal” is false statement

$\neg Z \& Z = \text{false}$
Boolean logic examples

There is an island, which is populated by two kind of people: liars and truthlovers.

- Liars always lie and never speak a word of truth.
- Truthlovers always speak only truth.

Suppose, you are landed on this island and met a person. What will be the answer to your question “Who are you?”

- The answer always will be “Truthlover”.

Now you see a person who answers to your question. “I am a liar.” Is it possible?

This makes a paradox and should not ever happen on this island.
Matlab boolean logic examples

- $123.3 \ & \ 12 = 1$
- $\sim 1232e-6 = 0$

```
B = [1.22312, 0; 34.343, 12]
B =
    1.2231  0
    34.3430 12.0000
~B
ans =
    0  1
    0  0
B|~B
ans =
    1  1
    1  1
```
Matlab boolean logic examples

123.3 & 12 = 1
1 ~ 1232e-6 = 0

>> B=[1.22312, 0; 34.343, 12]
B =
    1.2231     0
    34.3430    12.0000

~B

ans =
     0     1
     0     0

B|~B

"To be or not to be"
Matlab boolean logic examples

```matlab
>> B=[1.22312, 0; 34.343, 12]
B =
   1.2231  0
   34.3430 12.0000

>> A=[56, 655; 0, 24.4]
A =
 56.0000 655.0000
   0  24.4000
```

```
B&A
ans =
   1  0
   0  1
```

```
A|~B
ans =
   1  1
   0  1
```
Matlab boolean logic examples

\[
\text{>> } B = [1.22312, 0; 34.343, 12]
\]

\[
B =
\begin{bmatrix}
1.2231 & 0 \\
34.3430 & 12.0000
\end{bmatrix}
\]

\[
\text{>> } A = [56, 655; 0, 24.4]
\]

\[
A =
\begin{bmatrix}
56.0000 & 655.0000 \\
0 & 24.4000
\end{bmatrix}
\]

\[
B \& A
\]

\[
\text{ans} =
\begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}
\]

\[
A | \sim B
\]

\[
\text{ans} =
\begin{bmatrix}
1 & 1 \\
0 & 1
\end{bmatrix}
\]

### Comparison operators

<table>
<thead>
<tr>
<th>Math</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>(=)</td>
<td>(==) double equal sign!</td>
</tr>
<tr>
<td>(\sim=)</td>
<td>(\sim)</td>
</tr>
<tr>
<td>(&lt;)</td>
<td>(&lt;)</td>
</tr>
<tr>
<td>(\leq)</td>
<td>(\leq)</td>
</tr>
<tr>
<td>(&gt;)</td>
<td>(&gt;)</td>
</tr>
<tr>
<td>(\geq)</td>
<td>(\geq)</td>
</tr>
</tbody>
</table>

\[
x = [1, 2, 3, 4, 5]
\]

\[
x =
\begin{bmatrix}
1 \\
2 \\
3 \\
4 \\
5
\end{bmatrix}
\]

\[
x \geq 3
\]

\[
\text{ans} =
\begin{bmatrix}
0 \\
0 \\
1 \\
1 \\
1
\end{bmatrix}
\]

% chose such 'x' where x\(\geq\)3

\[
x(x \geq 3)
\]

\[
\text{ans} =
\begin{bmatrix}
3 \\
4 \\
5
\end{bmatrix}
\]
Comparison operators

<table>
<thead>
<tr>
<th>Math</th>
<th>Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>$=$</td>
<td>$==$</td>
</tr>
<tr>
<td>$\neq$</td>
<td>$\sim$</td>
</tr>
<tr>
<td>$&lt;$</td>
<td>$&lt;$</td>
</tr>
<tr>
<td>$\leq$</td>
<td>$\leq$</td>
</tr>
<tr>
<td>$&gt;$</td>
<td>$&gt;$</td>
</tr>
<tr>
<td>$\geq$</td>
<td>$\geq$</td>
</tr>
</tbody>
</table>

$x = [1, 2, 3, 4, 5]$

$x \geq 3$

ans =

0 0 1 1 1

Comparison with matrices

$A = [1, 2; 3, 4]$

$B = [33, 11; 53, 42]$
Comparison with matrices

>> A=[1,2;3,4]  
A =  
1  2  
3  4  

>> B=[33,11;53,42]  
B =  
22  11  
53  42  

A>=2

ans =  
0  1  
1  1  

B(A>=2)

ans =  
53  
11  
42  

Chose such elements of B where elements of A ≥ 2

Notes
Comparison with matrices

\[
\begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix}
\]

\[
\begin{bmatrix}
33 & 11 \\
53 & 42
\end{bmatrix}
\]

\(A \geq 2\)

\[
\begin{array}{c}
\text{ans} = \\
0 & 1 \\
1 & 1
\end{array}
\]

\[A(A \geq 2)\]

\[
\begin{array}{c}
\text{ans} = \\
3 & 2 \\
4
\end{array}
\]

Chose such elements of \(B\) where elements of \(A \geq 2\)

\[
\begin{array}{c}
\text{ans} = \\
53 & 11 \\
42
\end{array}
\]

if-else-end statement

if \(\text{expression}\)
this part is executed only if \(\text{expression}\) is true
else
this part is executed only if \(\text{expression}\) is false
end

if hungry
buy some food
else
keep working
end

Eugeniy Mikhailov (W&M)
Practical Computing
Lecture 03 8 / 19

Eugeniy Mikhailov (W&M)
Practical Computing
Lecture 03 9 / 19
if expression
this part is executed only if expression is true
else
this part is executed only if expression is false
end

if hungry
buy some food
else
keep working
end

if \{x \geq 0\}
y = \sqrt{x};
else
error('cannot do');
end

Eugeniy Mikhailov (W&M)
Practical Computing
Lecture 03 9 / 19
Common mistake in the 'if' statement

if \{x = y\}
D = 4;
Z = 45;
C = 12;
else
D = 2;
end

the value of 'D' is always 4, except the case when y=0

Eugeniy Mikhailov (W&M)
Practical Computing
Lecture 03 9 / 19
Common mistake in the 'if' statement

if \{x = y\}
D = 4;
Z = 45;
C = 12;
else
D = 2;
end

the value of 'D' is always 4, except the case when y=0
someone used assignment operator (=) instead of comparison (==)
Short form of 'if-end' statement

if expression
this part is executed
only if expression is true
end

if won a million
  go party
end

if (deviation<=0)
  exit;
end

The 'while' statement

while expression
this part is executed
while expression is true
end
The 'while' statement

while expression
this part is executed
while expression is true
end

while hungry
keep eating
end

while loop is extremely useful but they are not guaranteed to finish. For a bit more complicated conditional statement and loop it is impossible to predict if the loop will finish.

Yet another common mistake is

i=1;
while (i<=10)
c=a+b;
z=c+4+5;
i=i+2;
end
The 'while' statement

```plaintext
while expression
  this part is executed
  while expression is true
  end
end
```

The 'while' loop is extremely useful but they are not guaranteed to finish. For a bit more complicated conditional statement and loop it is impossible to predict if the loop will finish.

Yet another common mistake is

```plaintext
i=1;
while (i<=10)
  c=a+b;
end
```

not updating the term leading to fulfillment of the while condition

The 'for' statement

```plaintext
for variable = expression
do something
end
```

In this case variable is assigned consequently with columns of the expression, and then statements inside of the loop are executed

```plaintext
sum=0;
x=[1,3,5,6]
for v=x
  sum=sum+v;
end
>> sum
sum =
15
```

for loops are guaranteed to complete after predictable number of iterations (the amount of columns in expression).
Example

\[ S = \sum_{i=1}^{100} i = 1 + 2 + 3 + 4 + \cdots + 99 + 100 \]

S=0; i=1;
while (i<=100)
    S=S+i;
    i=i+1;
end

S=0; i=1;
for i=1:100
    S=S+i;
end

Example

\[ S = \sum_{k=1}^{100} a_k \]

While \( k \leq 100 \) and \( a_k \geq 10^{-5} \), where \( a_k = k^{-8} \).
Example

\[ S = \sum_{k=1}^{\infty} a_k \]

While \( k \leq 100 \) and \( a_k \geq 10^{-5} \), where \( a_k = k^{-k} \).

\[
S=0; k=1;
while( (k<=100) & (k^-k >= 1e-5) )
 S=S+k^-k;
 k=k+1;
end
\
>> S
S =
 1.2913

Notes
Same example with ‘for’ loop and use of matrix ops

\[ S = \sum_{k=1}^k a_k \]

While \( k \leq 100 \) and \( a_k \geq 10^{-5} \), where \( a_k = k^{-k} \).

```matlab
S=0;
for k=1:100
    a_k=k^-k;
    if (a_k < 1e-5)
        break;
    end
    S=S+a_k;
end
```

兴趣率相关例子

假设银行给你50%的利率（称为‘x’），并且你放了一美元。

- 一年的支付金额:
  \[ M_1 = 1 \times (1 + x) = 1 \times (1 + .5) = 1.5 \]
Interest rate related example

Suppose bank gave you 50% interest rate (let's call it 'x'), and you put one dollar in. How much would you get at the end of the year?

- one payment at the end of the year
  \[ M_1 = 1 \times (1 + x) = 1 \times (1 + 0.5) = 1.5 \]

- interest payment every half a year
  \[ M_2 = 1 \times (1 + x/2) \times (1 + x/2) = 1 \times (1 + 0.25)^2 = 1.5625 \]

- interest payment every month
  \[ M_{12} = 1 \times (1 + x/12)^{12} = 1.6341 \]

Now let's find how your money growth \((M_N)\) depends on the number of payments per year

\[
x = 0.5; \\
N_{\text{max}} = 100; \\
N = 1:N_{\text{max}}; \\
\text{M} = \emptyset; \\
\text{for } i = N \\
\phantom{\text{M}(i)} = (1 + x/i)^i; \\
\text{end} \\
\text{plot}(N, \text{M}, '-'); \\
\text{xlabel('N, number of payments per year');} \\
\text{ylabel('Money grows');} \\
\text{title('Money grows vs number of payments per year');} \]

Of course we do not need computer to show that \(M_{\infty} = e^x = 1.6487\) but we need it to calculate something like \(M_{1001} - M_{1000} = 2.0572 \times 10^{-7}\)