Lab 3

Matlab Fundamentals; Part III

Flow control and Functions

Marine Modelling January 21, 2019

Katja Fennel
Oceanography
Dalhousie University

Loops in MatLab: the FOR construct

Try

```
>> for i = 1:5, disp(i), end
```

The `disp` statement is repeated five times, showing the value of i.

Basic syntax

```
for index=iStart:[increment:]iEnd
  statements
end
```

Note: It is good programming style to indent the statements inside a for loop.

Also possible in a single line:

```
for index=iStart:[increment:]iEnd, statements, end
```


Example

Calculate:

$$\sum_{n=1}^{1000} n$$

Example

Calculate:

$$\sum_{n=1}^{1000} n$$

This can be calculated as follows:

```
s = 0;
for n=1:1000
  s = s + n;
end
```


More examples:

Factorials: Display factorials 1! to 10!

```
n = 10;
fact = 1;

for k=1:n
  fact = k*fact;
  disp( [k fact])
end
```


More examples:

Calculate a function: $f(x) = x * x$

```
x = [0:10];
for i=1:length(x)
    f(i) = x(i)*x(i);
end
plot(x,f,'o:')
xlabel('x')
ylabel('f(x) = x*x')
title('Quadratic function')
```


Functions

Functions are special forms of scripts that have input and output variables.

Basic syntax:

```
function result = function_name(var1[, var2,..])
% descriptive text
statements that calculate the value of result ...
using input vars
```

e.g. `fact_function.m` and `quad.m` as follows:

```
function f = quad(x)
% Calculate square: f(x) = x*x
f = x*x;
```


Practice:

1) Write a script that calculates the sum :

$$1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \frac{1}{5} \cdots - \frac{1}{999}$$

(Result: 0.6936)

2) Write a function that calculates the mean for an input vector.

Solution:

for 1)

```
sign = -1;
s = 0;
for n=1:999
  sign = -sign;
  s = s + sign/n;
end
```


Solution:

for 2)

```
function mn = my_mean(x)
% calculates the mean
mn = 0;
for i=1:length(x)
    mn = mn + x(i);
end
mn = mn/length(x);
```


Next: Look at roundoff-error script

Consider the equations (n is positive integer)

$$\phi^{n+1} = \phi^{n-1} - \phi^n$$

For $n = 1$: $\phi^2 = 1 - \phi$ (quadratic equation)
Solutions:

$$\phi_{1,2} = \frac{-1 \pm \sqrt{1+4}}{2}$$

We are only interested in the positive solution here:

$$\phi_1 = \frac{1}{2}(\sqrt{5} - 1) \approx 0.6180$$

(ϕ_1 is an irrational number, it has infinitely many digits)

Roundoff-error

ϕ^n can be calculated in two ways:

1) Simply by taking the n-th power of ϕ_1 : ϕ^n

2) Iteratively:
   If you know $\phi^0 = 1$ and $\phi^1 = \phi$
   you can calculate $\phi^2 = \phi^0 - \phi^1 = 1 - \phi$
   and then $\phi^3 = \phi^1 - \phi^2$
   and so forth ...
   for $\phi^n = \phi^{n-2} - \phi^{n-1}$

Roundoff-error

ϕ^n can be calculated in two ways:

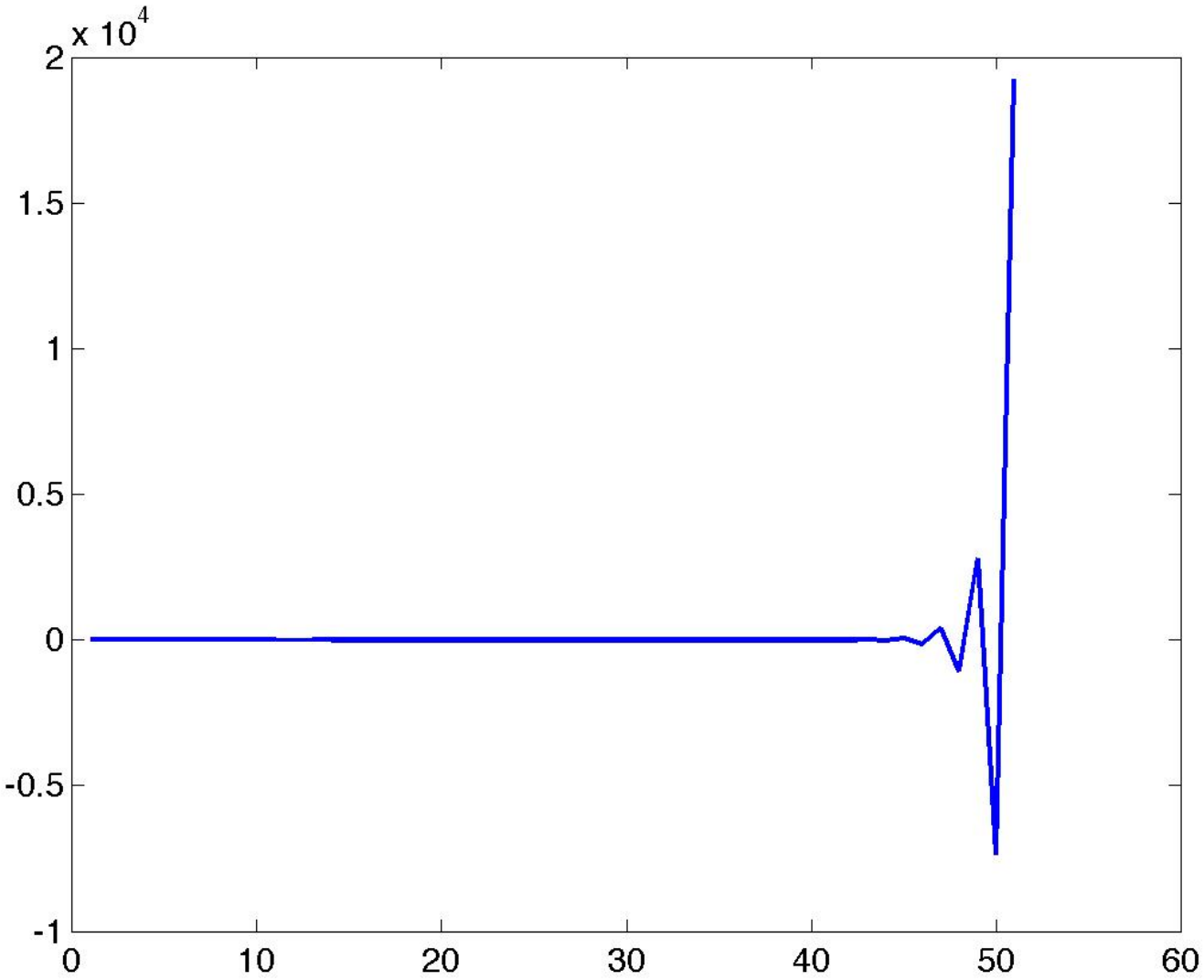
1) Simply by taking the n-th power of ϕ_1 : ϕ^n

2) Iteratively:
   If you know $\phi^0 = 1$ and $\phi^1 = \phi$
   you can calculate $\phi^2 = \phi^0 - \phi^1 = 1 - \phi$
   and then $\phi^3 = \phi^1 - \phi^2$
   and so forth ...
   for $\phi^n = \phi^{n-2} - \phi^{n-1}$

Note that we are only subtracting numbers when calculating iteratively; no powers involved!

The script `roundoff_error.m` calculates the powers of ϕ in these two ways and the relative error.

Plot the relative error!



Next important concept: Decisions (IF statement)

The matlab function `rand` creates a random number between 0 and 1.
Try:

```
>> r = rand;
>> if r > 0.5 disp('r is greater than 0.5'), end
```

Check the value of `r`! Matlab should display the message only when `r` is greater than 0.5.

Next important concept: Decisions (IF statement)

The matlab function `rand` creates a random number between 0 and 1.
Try:

```
>> r = rand;
>> if r > 0.5 disp('r is greater than 0.5'), end
```

Check the value of `r`! Matlab should display the message only when `r` is greater than 0.5.

A related but different exercise:
Try:

```
>> 2 > 0
```

and

```
>> -1 > 0
```


Next important concept: Decisions (IF statement)

The matlab function `rand` creates a random number between 0 and 1.
Try:

```
>> r = rand;
>> if r > 0.5 disp('r is greater than 0.5'), end
```

Check the value of `r`! Matlab should display the message only when `r` is greater than 0.5.

A related but different exercise:
Try:

```
>> 2 > 0
```

and

```
>> -1 > 0
```

These are called logical expressions. Matlab assigns value 1 if true and 0 if false.

IF statement

Most basic syntax for the `if` statement:

```
if condition statement, end
```

where *condition* is a logical expression using a relational operator (`<`, `<=`, `==`, `~=`, `>`, `>=`)

If the condition is true the statement is executed, if false nothing happens.

IF statement

Most basic syntax for the `if` statement:

```
if condition statement, end
```

where *condition* is a logical expression using a relational operator (`<`, `<=`, `==`, `~=`, `>`, `>=`)

If the condition is true the statement is executed, if false nothing happens.

See if you can get the following relational statements right (then test):

```
>> x = 3 > 2
>> x = 2 > 3
>> x = -4 <= -3
>> x = 1 < 1
>> x = 3 == 3
>> x = 0 < 0.5 < 1
```


Next: IF ... ELSE

Basic syntax:

```
if condition
  statementsA
else
  statementsB
end
```

If true *statementsA* will be executed, otherwise *statementsB*.

Next: IF ... ELSE

Basic syntax:

```
if condition
  statementsA
else
  statementsB
end
```

If true *statementsA* will be executed, otherwise *statementsB*.

Example:

```
if x < 0 disp('negative'), else disp('not negative'), end
```


ELSEIF

You can even use `elseif`!

Syntax:

```
if condition1
  A
elseif condition2
  B
else
  C
end
```

Generalize the previous example so that Matlab displays whether x is negative, zero or positive.

Note: You can also nest if statements.

Back to our data: lobo_data_subset.txt

Plot histograms for nitrate, salinity and chlorophyll.
Try calculating mean, median and variance (matlab functions `mean`, `median` and `var`).

Back to our data: lobo_data_subset.txt

Plot histograms for nitrate, salinity and chlorophyll.
Try calculating mean, median and variance (matlab functions `mean`, `median` and `var`).

Note: NaNs don't cause trouble during plotting; Matlab just ignores them. The functions however will give NaN if applied on data containing NaN.

Back to our data: lobo_data_subset.txt

Plot histograms for nitrate, salinity and chlorophyll.
Try calculating mean, median and variance (matlab functions `mean`, `median` and `var`).

Note: NaNs don't cause trouble during plotting; Matlab just ignores them. The functions however will give NaN if applied on data containing NaN.

In this case you can remove them as follows:

```
>> x = data(:,1);
>> bad = isnan(x);
>> x(bad) = [];
```


Back to our data: lobo_data_subset.txt

Plot histograms for nitrate, salinity and chlorophyll.
Try calculating mean, median and variance (matlab functions `mean`, `median` and `var`).

Note: NaNs don't cause trouble during plotting; Matlab just ignores them. The functions however will give NaN if applied on data containing NaN.

In this case you can remove them as follows:

```
>> x = data(:,1);
>> bad = isnan(x);
>> x(bad) = [];
```

Alternative: use functions `nanmean`, `nanmedian` and `nanvar`; they ignore NaNs