School of Mathematics and Statistics
MAST30028 Numerical Methods & Scientific Computing
Week 5

Drag and drop the folder `Week5` from `L: \MAST30028` to `C:\...\MATLAB` and include it in the path. Now MATLAB knows how to find the files in `Week5`.

1 Error propagation

These relate to material in Lecture 8.

Exercise Set 1

a. A recurrence relation

Prove (it's easy!) that the integral

$$I_n = \int_0^1 \frac{x^n}{x+2}\,dx$$

satisfies the 2-term recurrence relation (or difference equation)

$$I_n = 1/n - 2I_{n-1}$$

Hint: rewrite $x/(x+2)$

Prove that $I_0 = \log(3/2)$ and that $0 < I_n < 1/n$.

This algorithm is implemented in `BadRecurrence.m` in order to compute I_{100}. Run it.

Explain the magnitude of the error you find.

Explain how you could run the recurrence *backwards*, starting from an estimate for I_{200}. This algorithm is implemented in `GoodRecurrence.m`. **Run it. What do you find?**

b. A numerical derivative.

A reasonable approximation for $f'(x)$ might be expected to be $\frac{f(x+h)-f(x)}{h}$, at least as h gets small (recall how the derivative is defined) .

This algorithm is implemented in `ForwardDifference.m` where $f = \exp$ and $x = 1$.

Run it. What do you see?

Explain why the error first falls as h is reduced, then rises.

c. I would feel quite justified in asking an exam question like the following: only attempt after you have completed Exercise 1a.

Suppose we want to compute the integrals

$$I_n = \int_0^1 x^n e^{x-1}\, dx \quad n = 0, 1 \ldots$$

(a) Show that integration by parts produces the recurrence formula

$$I_n = 1 - nI_{n-1}$$

and that

$$0 < I_n < \frac{1}{n+1}$$

(b) Starting from $I_0 = 1 - e^{-1}$, use the recurrence to find I_{25}. Notice that I_n must be positive for all n.

(c) Explain what is going wrong.

(d) Instead run the recurrence backwards from the approximate value $I_{40} \approx 1/41$ and compare with part b. Use `integral` with an absolute tolerance of 10^{-10} to check your answer.

2 Root-finding

These relate to the material in Lecture 9.

Exercise Set 2

Fixed point iteration

a. Run the function `FixedPoint` used in lectures; understand what it does and what behaviour each of the 5 fixed point functions $g_1(x)$–$g_5(x)$ exhibit.

b. Run the function `FixedPointErrors` used in lectures, understand what it does and what behaviour each of the fixed point functions $g_3(x)$–$g_5(x)$ exhibit.

c. Fixed point iteration can be visualized by *cobweb diagrams*. Examine the function `cobweb` and understand what it does. Test it by calling `cobweb(@cos,0,1,0.5,50)`.

d. By adapting the codes provided or otherwise, investigate the fixed points of the following functions, with the specified initial values x_0 at least.

- $\cos(x)$ Use $x_0 = 1, 3, 6$
- $\exp(\exp(-x))$ Use $x_0 = 2$
- $x - \log_e(x) + \exp(-x)$ Use $x_0 = 2$
- $x + \log_e(x) - \exp(-x)$ Use $x_0 = 2$

The last 3 are related in what way?

Exercise Set 3

Bisection

The function M-file `Bisection.m` uses the bisection algorithm to find roots of a nonlinear function. Examine the code. It has several useful features you may want to adopt in your MATLAB programming:

- since vectors and arrays are so fundamental in MATLAB it's easy to use vector input arguments, such as `int` and `tol`.
- similarly it's easy to output vector/array variables by using the output argument list in the function declaration. These output arguments are optional, so you can call `Bisection` with 1, 2 or 3 (or 0!) output arguments. Try it.
- the comment lines immediately following the function declaration form the help for your M-file. Try `help Bisection`. The help includes an example of usage, as well as a description of input and output variables.
- using a debug switch allows a clean way to output useful information for debugging purposes and switch it off for production runs.
- an output error flag, here `ierr`, indicating whether the function terminated normally or in some other way is good practice.
- default values for input arguments can be specified using the `nargin` command, which counts the number of input arguments.
- the function can be forced to return immediately on an error by using the `return` command.
- all the output variables must be given values inside the function.
- since root-finding requires a function as input, we pass a function as an argument using a function handle.

What kind of convergence criterion does it use?

To see an example of usage, run `driverBisect`.

Preparing Assignment 1

Submit assignments as pdf documents and zip files. The results as tables or plots should be included in the document, together with discussion. Relevant M-files which produced the results must be be included as in the body of the document and as separate files. Zip files only include relevant M-files.

Please answer Questions 3 and 4 on different pages to Questions 1 and 2.

Use plots freely since "a picture is worth 1000 words".

Since a typical M-file to answer a question might be less than 50 lines, there is no need for voluminous documentation. Hopefully your code should be mostly self-explanatory but use comments to explain key steps or subtleties. Use understandable variable names etc.

You will need to submit your assignment using the LMS, just as you will all assessment in this subject.