Modelling and Programming IT & Health

Week 6

Deliverables

- Report 2: Section 2. Software for finding roots and extrema.

Documenting software

An important part of Report Assignment 2 is to document the software properly. This is done by creating an HTML document that, as much as possible, follows the good advise in the internet article "Guidelines for writing software documentation". If you provide proper *help text* for your MATLAB functions and scripts, there is a tool that you can use to create the HTML document. This tool is called `m2html` and it has been uploaded to the Exercise_Material folder at CampusNet. The following exercises are intended to get you started using `m2html`:

- Unpack `m2html.zip` (available at CampusNet file sharing) to a folder on your hard drive.
- Choose the `m2html` folder as your Current Directory in MATLAB.
- Right click in the Current Directory window and choose "Add to Path"→"Current Folder".
- Run the following command from the Command Window:
  `m2html('htmldir', '.\doc', 'recursive', 'on');`
  This creates HTML documentation for m2html in a subfolder called `doc`.
- Create a new folder for your Report 2 hand-in. Create a subfolder in this folder called `matlab` and move all the `m`-files that you have written so far for Report 2 to this subfolder.
- Go to your new folder for Report 2 and run the following command from the Command Window:
  `m2html('mfiles', '.\matlab', 'htmldir', '.\doc');`
  This creates the HTML document for Report 2 in a subfolder called `doc`.

Numerical differentiation

Section 2 of Report Assignment 2 is about finding the first maximum along a curve. If you are not sure about how to find a local maximum using differentiation, do the following exercises:

- **C**: Problem 13 of Section 5.3.4. (*Hint*: Closely related to Example 5 of Section 5.3.3 in **C**.)
- **C**: Problem 12 of Section 5.3.4. (*Hint*: Closely related to Example 6 of Section 5.3.3 in **C**.)

The curve in the report assignment is not given by a functional expression, but by a set of data points stored in two vectors, let us call them t and C such that the data points are (t_i, C_i), $i = 1, \ldots, n$. You still need differentiation to find the maximum of the curve, but, since you do not have a functional expression, the differentiation will have to be a numerical approximation. The following exercises help you implement numerical differentiation in MATLAB.

How do you find a numerical approximation of the derivative of a curve? Using a piecewise linear approximation, the derivative turns into a ratio of differences:

$$\frac{\mathrm{d}C}{\mathrm{d}t} \approx \frac{\Delta C}{\Delta t} = \left(\frac{C_2 - C_1}{t_2 - t_1}, \ldots, \frac{C_n - C_{n-1}}{t_n - t_{n-1}} \right) \quad .$$

The question is now what t-values these derivatives correspond to. A good choice of t-values is

$$t_{1:n-1} + \frac{1}{2}\Delta t = \left(t_1 + \frac{1}{2}(t_2 - t_1), \ldots, t_{n-1} + \frac{1}{2}(t_n - t_{n-1})\right) \ .$$

Using these t-values with the derivatives is referred to as using *central differences*.

As a summary, given two vectors t and C which describe a curve we can use central differences to obtain the following two vectors as a numerical approximation of the derivative of the curve:

$$t_{1:n-1} + \frac{1}{2}\Delta t \quad \text{and} \quad \frac{\Delta C}{\Delta t} \ .$$

- Write a MATLAB function called `deriv` which takes two vectors `t` and `C` as arguments and returns the two vectors `t_dt` and `dCdt` that approximate the derivative. (*Hint*: MATLAB has a command called `diff` which can help you find the difference vectors ΔC and Δt.)
- Write a MATLAB function called `maximum` which takes two vectors `t` and `C` as arguments and returns two values `t_max` and `C_max` which point out the first maximum along the curve. The algorithm to implement is the following:
  1. Find the data points (`t_dt`, `dCdt`) that provide a numerical approximation of the derivative of the curve. (*Hint*: Use `deriv`.)
  2. Find the index `i` where the derivative `dCdt(i)` turns from positive (at index `i-1`) to negative. (*Hint*: Use a `while`-loop.)
  3. Find `t_max` where the derivative is zero. (*Hint*: Use the equation of a straight line between the derivative points at indices `i-1` and `i`.)
  4. Find the index `j` where the time `t(j)` goes from below `t_max` (at index `j-1`) to above `t_max`. (*Hint*: Use a `while`-loop.)
  5. Find `C_max` where `t` equals `t_max`. (*Hint*: Use the equation of a straight line between the data points at indices `j-1` and `j`.)

To test your new functions, do the following:

- Plot the curve $M(t) = 1.006\, e^{-0.15\, t} - 1.011\, e^{-0.30\, t}$ from Figure 4.5 in **LA**.
  Use `t = linspace(0, 24, 13)` in MATLAB.
- Find the derivative of $M(t)$ analytically and plot it in the same figure.
- Find the maximum of $M(t)$ analytically and plot it in the same figure as a circle.
- Use `deriv` to find an approximate derivative and plot it in the same figure (to distinguish it from the analytical derivative, plot it as dots instead of a full-line curve).
- Use `maximum` to find the maximum point of the curve and plot it in the figure (in a different colour than the one you found analytically).
- Increase the number of data points in the `t`-vector (the third argument of the `linspace` command). Make sure that your numerical approximation approaches the accurate analytical results.

Curriculum

C	Sections 5.3–5.4. *Applications of Differentiation*.
LA	Sections 2.4–2.6. *Matrix Factorization*.
**	Internet article. *Guidelines for writing software documentation*.

JRF 2009