Aalto-yliopisto

MATLAB II

Exercise for Lecture 3

This exercise involves the third lecture of the minicourse Matlab continuation, concerning mostly with handling and interpreting data. Once you have solved the problems, please send **published pdf** and your **source code** to *juha.kuortti@aalto.fi*.
The deadline for the return of the exercise is 15.3.2018.

1. Let and z_0 be a complex number. We define the following recursion:

$$z_n = z_{n-1}^2 + z_0$$

   This is a dynamical system known as a quadratic map. Given different choices of the initial value z_0 the recursion leads to a sequence of complex numbers $z_1, z_2, \ldots$ known as the orbit of z_0. This dynamical system is highly *chaotic*, meaning that depending on the z_0, a huge number of different orbit patterns are possible.

   Most choices of z_0 tend towards infinity (i.e. $|z_n| \to \infty$ as $n \to \infty$). For some z_0, however, the orbit is stable, meaning that it goes into periodic loop; and finally there are some orbits, that seem to do neither, dancing around the complex space apparently at random.

   In this assignment, your task is to you write a MATLAB script that visualizes the filled-in variation of a Mandelbrot set, which is the set of all z_0 with orbits which do not tend towards infinity.

   a) Let's set-up the exercise by defining the area in which we work: create a 1000×1000 complex matrix C that meshes the area of the complex plane limited by

```
rlim = [-0.748766713922161, -0.748766707771757];
ilim = [ 0.123640844894862,  0.123640851045266];
```

   b) Write a function called `mandelIter` that does the iteration for any given z_0 until $|z_n| \geq 2$. The function should return the number of iterations required for $|z_n| \geq 2$. Also assume that if $|z_n| < 2$ after $n \geq 500$, the orbit is stable - in that case, return 500.

   c) Use `arrayfun` to apply the function you just wrote to every element of the matrix C: use `tic` and `toc` to measure the time.

   d) Write a loop that iterates over the elements of the matrix C and computes the `mandelIter` for every element. Again, measure performance. Here the looping order and preallocation are very important.

   e) Visualize your iteration by `imagesc(log(M+1))`, where M is the result of applying `mandelIter` to every element of C

   f) (Optional) Implement the iteration without using a function, but vectorisation and logical indexing. You should see a significant bump in speed.