CS3414. Homework Assignment - No tool is perfect.

The first objective of this assignment is to become aware that even well-written, commercial, tried-and-true software can produce wrong answers without any warning signs. The second objective is to learn how to recover once you understand what is going on. The moral of the homework is "Never use a numerical method until you understand how it works. No black boxes".

1 What to submit

Individual work: each student submits his or her own work. A single PDF file, 12 pt font. Each relevant part of the homework must be accompanies by plots. You can use Mathematica or any other software for the plotting.

2 Definitions

READ THE DEFINITION VERY CAREFULLY. SOLUTIONS THAT DO NOT COMPLY WITH THE DEFINITION (solutions to a wrong problem) WILL LOSE LOTS OF POINTS.

Let's call function $f(x)$ *nice* if it is defined everywhere on [-1, 1], $-10 < f(x) < 10$, the function is infinitely differentiable on [-1, 1], and has a *single* minimum on (-1, 1) (trivial case of extrema at the ends are excluded). For example, $f(x) = x^2$ is a nice function, but $sin(1/x^2)$ or x^3 are not. Suppose you use a numerical procedure to find the minimum x^{approx} of a nice function. We call such numerical solution *right* if $|x^{approx} - x^{exact}| < 10^{-3}$, x^{exact} being the exact answer. Otherwise, the solution is called *wrong*. Note that our definition is very generous: generically, one expects the correct solution to be within $\sqrt{\epsilon}$ of the exact, that is within $\sim 10^{-7}$.

2.1 Part I. Explore. 10 pts

Use *Mathematica* to explore the straightforward Newton's method for finding local minimum. Nice functions have only one minimum by definition, so not to worry. Use FindMinimum[]; let Mathematica select all input parameters automatically, except the method "Newton", which you specify explicitly (see examples on the class site, you may use any of them as a template). Explore a nice function $f(x) = ax^2 + bx^4$, consider limiting cases such as $a = 0$, $b = 0$, and some intermediates. Present convergence graphs (use FindMinimumPlot[]). Make your conclusions.

2.2 Part II. Break. 20 pts

Now that you understand well how Newton's method works, show it! **Come up with a nice function (use the definition above) that breaks Newton's method** that is Mathematica, with default settings, gives a wrong solution (see above defs.) without so much as a peep - no warnings or errors. Present convergence graphs (use FindMinimumPlot[]). Explain the failure.

2.3 Part III. Fix. 10 pts

Reason which method from your C&K textbook may mitigate the problem. Give an intuitive explanation for why the alternative to Newton's works; in a few sentences, give at least one pro and one con for your alternative.

2.4 Part IV. Fix again. 10 pts

See if you can harness Mathematica's unique functionality and options to make it find the right solution, using the same Newton's. In fact, you may be able to get to it within $\sqrt{\epsilon}$ (Find what machine epsilon is for your machine. Use code on the class site). Explain why the solution, while useful in research, is not very useful in situations when you need to quickly find lots of minima as part of a larger code written in a standard language such as C.