Math 661 Optimization (Bueler) 29 August, 2016

Assignment #1

Due Friday, 9 September 2016, at the start of class

Please read chapter 1 and section 2.1, pages 1–20, in the textbook.[1] Then do the following exercises, which are based on the notes *Five example optimization problems*, handed out in class and available online at

bueler.github.io/M661F16/fiveexamples.pdf

Problem P1. Solve `calcone`.

In particular, describe in well-written english, a strategy (algorithm) for solving this and similar one-variable optimization problems on bounded, closed intervals. Your strategy will necessarily be iterative. Your goal is demonstrable 6-digit accuracy. Discuss any issues about the general performance/success of your strategy, emphasizing how it might fail on other problems of this type.[2] Implement your strategy as a MATLAB/OCTAVE[3] algorithm using elementary programming structures such as variables, arrays, `for` loops, `if` conditionals, and such.

Do *not* use black boxes, such as MATLAB/OCTAVE commands `fzero`, `fsolve`, `fminsearch`, or `fminbnd`, for this exercise. You may use MATLAB/OCTAVE to visualize the function—that is recommended!—but your algorithm should *not* be based on human interaction with a figure window.[4]

Problem P2. Solve `fit`.

Follow the same rules as above: Describe a strategy (algorithm) for solving this and similar problems, with the goal of demonstrable 6 digit accuracy. Discuss any issues about the general performance/success of your strategy. Implement your strategy as a MATLAB/OCTAVE algorithm using elementary programming. Also, plot the solution curve on the same graph as the data.

Noting this kind of problem is standard in the statistics curriculum, please *avoid* describing/using a recipe which either requires copying formulas from books or which involves steps you do not understand. Instead, start from scratch and explain what to do.

[1] J. Nocedal & S. Wright, *Numerical Optimization*, 2nd ed., Springer 2006

[2] Essentially every numerical black box can be made to fail by careful input (i.e. problem) design. Professionals know that what they build can break, and how to break it.

[3] You may use other languages such as PYTHON, but I will only provide examples and solutions in MATLAB/OCTAVE.

[4] Reasons for this prohibition include that higher-dimensional problems are un-visualizable and that programs need to run autonomously to be useful.

Problem P3. **(a)** Solve `salmon`.

In fact this problem is embarassingly simple to solve, so start by writing a few clear sentences justifying the solution. Then visualize, in 3D and probably with pencil and paper, the set of feasible solutions; mark the solution as well. Also use a straightforward substitution to eliminate the equality constraint, and then re-visualize the feasible set and solution in 2D.

(b) Put `salmon` into form (1.1) by introducing "slack" variables. (The textbook describes slack variables on pages 356–357.) As a hint, note that the problem is five dimensional when put in standard form (1.1).

Problem P4. Complete the following classification table of all five example problems. Except for the last column, use a check (✓) if the property applies, leave blank if it does not apply, and write a dash (–) for "not applicable". In the last column give an integer for the dimension, or ∞, or a dash.

name	discrete	constrained	linear	quadratic	dimension
`calcone`					
`fit`					
`salmon`					
`tsp`					
`beam`					