Assignment #9

Due Monday, 3 December 2018, at the start of class

Please read textbook sections 12.4, 12.5, 13.1, 13.2, 13.3, 13.4, 13.5, 14.1, 14.2, 14.3.

DO THE FOLLOWING §13.2 EXERCISE FROM PAGE 459:

- Exercise 2.1 *Ignore "ϵ" in Algorithm 13.1; stop when you have computed x_2. Display x_1, x_2 to 8 digit accuracy. Show that the computed vectors p_0, p_1 are conjugate and that r_0, r_1 are orthogonal.*

DO THE FOLLOWING §14.2 EXERCISE FROM PAGE 489:

- Exercise 2.1 *The relevant definition of "stationary point" is in this section! Find it!*

Problem P16. **(a)** Write a code

```
function [xk, xklist] = sdfdbt(x0,f,tol,fdgrad)
```

which does steepest-descent using a finite-differenced gradient (section 12.4) and, of course, back-tracking. Base your code on the steepest-descent-with-back-tracking code already written:

bueler.github.io/M661F18/matlab/sdbt.m

The inputs `x0` and `tol` have the same meaning as in `sdbt.m`. The input `fdgrad` is (initially) *ignored*; see part **(c)**. The input `f` is allowed to be a function which returns *only* the objective function value:

```
function z = f(x)
```

Regarding the finite-differencing, assume that the gradient is approximated using the formula on page 426,

$$[\nabla f(x)]_j \approx \frac{f(x + he_j) - f(x)}{h}$$

and where $\{e_j\}_{j=1,\dots,n}$ is the standard basis of $\mathbb{R}^n$. Choose h following the advice in section 12.4, assuming that the typical values of $f(x)$ and $f''(x)$ are of order one (i.e. neither very large nor small).

(b) Compare `sdbt.m` and `sdfdbt.m` on the problem

$$f(x) = 5x_1^2 + \frac{1}{2}x_2^2$$

using the initial iterate $x^{(0)} = (1, 1)^\top$. Compare results, addressing both accuracy and number of iterations, for $\texttt{tol} = 10^{-2}, 10^{-4}, 10^{-6}, 10^{-8}$. (*Explain what is going on!*) Also compare number of scalar function evaluations for $\texttt{tol} = 10^{-2}$.

(c) Now make the input `fdgrad` into an optional boolean flag which is, by default, set to `true`.[1] If `fdgrad==true` then the program runs as in part **(a)** but if `fdgrad==false` then it runs the same way as `sdbt.m`.

Now suppose $f(x)$ is defined by the following erroneous user-supplied code:

ediswrong.m

```
function [f, df] = ediswrong(x)
% EDISWRONG  A quadratic function in 2D with a wrong gradient.

if length(x) ~= 2,  error('x must be length 2 vector'),  end
f = 5 * x(1)^2 + 0.5 * x(2)^2;
df = [5 * x(1);
      x(2)];
```

Advise this user on how to use `sdfdbt.m` to detect that the supplied $\nabla f(x)$ code is wrong.

Problem P17. Explain why each iterate of `sdfdbt.m` in **P16** can be regarded as evaluating the objective functional at the $n+1$ vertices of a simplex in $\mathbb{R}^n$. Based on the wiki page

en.wikipedia.org/wiki/Nelder-Mead_method

give two or three sentences which compare and contrast `sdfdbt.m` and the Nelder-Mead algorithm. A sketch can be part of your comparison also.

Problem P18. (*It is a short proof. At least get it right, even if you have to look it up!*)

Assume $A \in \mathbb{R}^{n\times n}$ is invertible and that $u, v \in \mathbb{R}^n$ are nonzero vectors so that $v^\top A^{-1}u \neq 1$. Prove that if $\alpha = 1/(1 - v^\top A^{-1}u)$ then

$$\left(A - uv^\top\right)^{-1} = A^{-1} + \alpha(A^{-1}u)(v^\top A^{-1}).$$

Problem P19. **(a)** Implement Algorithm 13.3 on pages 466–467 as a function which does nonlinear conjugate-gradient with back-tracking:

`function [xk, xklist] = ncpbt(x0,f,tol)`

The inputs `x0` and `tol` have the same meaning as in `sdbt.m`. As with earlier codes `sdbt.m` and `sr1bt.m`,[2] the user-supplied function must return both objective value and gradient: `[fxk, dfxk] = f(xk)`.

(b) I have already posted a code which computes the Rosenbrock function:

bueler.github.io/M661F18/matlab/rosenbrock.m

Compare performance (iterations) of `sdbt.m`, `sr1bt.m`, and `ncpbt.m` on this example when $x_0 = (0,0)^\top$. Use `tol` $= 10^{-2}, 10^{-4}, 10^{-6}$ and make a comparison table.

---

[1]In Python it is easiest to make `fdgrad`, and perhaps also `tol`, into keyword arguments.

[2]Feel free to use my online versions. If you use your own, make sure they are debugged!