ME46060 “Engineering Optimization – Concepts and Applications” Exercise 3

Subject: “Valve spring”

Objectives

- Understand the basic concept of line searches in more dimensional problems.
- Learn how to use Matlab line search algorithm `fminbnd` and how to set optimization `options`.
- Investigate the influence of convergence parameters.

Course material

- Section 4.5 and 4.6 of Chapter 4, Section 6.1 and 6.2 of Chapter 7 of Papalambros and Wilde [1]

Introduction

This exercise considers an **unconstrained** optimization problem of the valve spring involving two design objectives. On the one hand we want to come as close as possible to a specified value for the spring stiffness, and on the other hand we want to reach a specified value for the lowest eigenfrequency of the spring. Mathematically the problem is defined as follows:

$$\text{Minimize } f(\mathbf{x}) = \text{abs}((k-ktarget)/ktarget) + w*\text{abs}((freq1-frtarget)/frtarget) \tag{1}$$

Subject to the set constraint:

$$0.02 \le x(1) \le 0.04 \text{ (m)}$$
$$0.002 \le x(2) \le 0.005 \text{ (m)}$$

where:

- k = the axial spring stiffness
- $ktarget$ = target value for the spring stiffness (= 10000 N/m).
- $freq1$ = lowest eigenfrequency
- $frtarget$ = target value for lowest eigenfrequency (= 300 Hz).
- w = weighing factor between the two objective function components.

Row vector **x** represents the design variables, being the coil diameter D (= x(1)) and the wire diameter d (= x(2)) respectively. The other parameters are given in the file `springparams1.m`.

The problem can be visualized for w = 1.0 by running the given program `springw3ex1` (we call the resulting plot “Figure 1”). It can immediately be seen that the solution of problem (1) is at the intersection of the contours k = 10000 N/m and $freq1$ = 300 Hz (identify these contours in the plot). However, the main goal of these exercises is to learn about (one-dimensional) line searches within the more-dimensional problem (1). The line search is defined as:

$$\text{Minimize} \quad f(alpha)$$
$$\text{subject to} \quad \mathbf{x} = \mathbf{x_q} + alpha*\mathbf{S_q}, \qquad 0 \le alpha \le alpha_{max} \tag{2}$$

where:

- $\mathbf{x_q}$ = row vector representing the initial point of the line search.
- $\mathbf{S_q}$ = row vector representing the line search direction.
- $alpha$ = scalar representing the step size along the search direction.
- $alpha_{max}$ = upper bound of alpha such that the design point **x** does not violate the set constraint in (1).

The function *f*(*alpha*) is computed by the (given) file springobjw3.m. In this function *alpha* is the (one-dimensional) design "vector"; *xq, sq, ktarget, frtarget* and *w* are extra input parameters.
Note: springobjw3.m uses the given files springanalysis1.m and springparams1.m.

Exercise 3.1: Line searches using Figure 1.

Consider in Figure 1 point $\mathbf{x_q}$ = [0.022 0.004] as initial point, and consider in point $\mathbf{x_q}$ three search directions $\mathbf{S_{q1}}$ = [0.002 0.0], $\mathbf{S_{q2}}$ = [0.0 -0.0005] and $\mathbf{S_{q3}}$ = [0.002 -0.0005] respectively. Estimate from the figure the point (*D* and *d*) where *f*(*alpha*) is minimal. Estimate the optimal value of *alpha* too.

Exercise 3.2: Line searches using the line search algorithm fminbnd

Write a file springw3ex2.m in which the algorithm fminbnd from the Matlab Optimization Toolbox is used to perform the line search within problem (1) for w = 1.0. Use the function springobjw3 to compute the objective function of problem (2). Note that the necessary extra input parameters of springobjw3 can be passed to it through the input of fminbnd;see "Help fminbnd", in particular:

> If your function FUN is parameterized, you can use anonymous functions to capture the problem-dependent parameters. Suppose you want to minimize the objective given in the function `myfun`, which is parameterized by its second argument `c`.
> Here `myfun` is an M-file function such as:
>
> ```
> function f = myfun(x,c)
> f = (x - c)^2;
> ```
>
> **To optimize for a specific value of** `c`, first assign the value to `c`. Then create a one-argument anonymous function `@(x)` that captures that value of `c` and calls `myfun` with two arguments. Finally, pass this anonymous function to `FMINBND:`
>
> ```
> c = 1.5; % define parameter first
> x = fminbnd(@(x) myfun(x,c),0,1)
> ```

Use a similar approach to pass the necessary parameters to springobjw3 (instead of myfun).

Your program should perform the following steps:

- Define the extra parameters as needed by springobjw3.
- Call fminbnd using appropriate input and output parameters (see the documentation of fminbnd). Use default values for "options".
- Apply to the resulting (optimal) value of *alpha* to compute coil diameter *D* and the wire diameter *d*.

Run your program to perform the line searches $\mathbf{S_{q1}}$, $\mathbf{S_{q2}}$ and $\mathbf{S_{q3}}$ as given above (one at a time). Compare the results with your estimations in Exercise 3.1.

Type in the Matlab command window:

> optimset('fminbnd')

to see which default options were used by fminbnd.

Next modify your program in the sense that, before calling fminbnd, you set different options using the Matlab function optimset (see "Help optimset"). Set 'Display' to 'iter', and change tolX to 1.0e-8.
Rerun the program for search direction $\mathbf{S_{q2}}$ and interpret the results. Investigate the influence of tolX on the number of function evaluations. Regard also the influence on the resulting, *alpha, D, d,* objective function value, and the procedure used in the successive iteration steps (tip: change to "format long" to see differences).

References

[1] Papalambros, P.Y. and Wilde, D.J., *Principles of optimal design: modeling and computation*, 3rd edition, Cambridge University Press, 2017.