NUMERICAL OPTIMISATION
ASSIGNMENT 6

MARTA BETCKE
KIKO RUL·LAN

Minimal Surface Cost Function

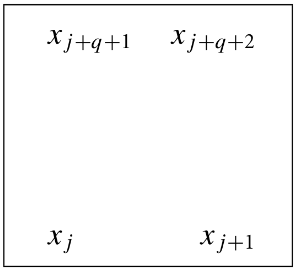
[Adapted from Exercise 7.7 from Nocedal-Wright]
The minimum surface problem is a classical application of the calculus of variations and can be found in many textbooks. We wish to find the surface of minimum area, defined on the unit square, that interpolates a prescribed continuous function on the boundary of the square. In the standard discretization of this problem, the unknowns are the values of the sought-after function $z(x, y)$ on a $q \times q$ rectangular mesh of points over the unit square.

More specifically, we divide each edge of the square into q intervals of equal length, yielding $(q+1)^2$ grid points. We label the grid points as:

$$x_{(i-1)(q+1)+1}, \dots, x_{i(q+1)} \quad \text{for } i = 1, 2, ..., q+1,$$

so that each value of i generates a line. With each point we associate a variable z_i that represents the height of the surface at this point. The values of the function are fixed by the boundary conditions at the $4q$ grid points on the boundary. The optimization problem is to determine the other $(q+1)^2 - 4q$ variables z_i so that the total surface area is minimized.

A typical subsquare in this partition looks as follows:



We denote this square by A_j and note that its area is $1/q^2$. The desired function is $z(x, y)$, and we wish to compute its surface over A_j. Calculus books show that the area of the surface is given by:

$$f_j(\boldsymbol{x}) = \int\int_{(x,y)\in A_j} \sqrt{1 + \left(\frac{\partial z}{\partial x}\right)^2 + \left(\frac{\partial z}{\partial y}\right)^2} \mathrm{d}x\mathrm{d}y$$

Approximating the derivatives by finite differences, f_j has the form

$$f_j(\boldsymbol{x}) = \frac{1}{q^2}\left[1 + \frac{q^2}{2}\left[(z(x_j) - z(x_{j+q+2}))^2 + (z(x_{j+1}) - z(x_{j+q+1}))^2\right]\right]^{\frac{1}{2}}. \tag{1}$$

The function F in `surfaceVector.m` implements the sum of all f_j over the unit square. The gradient and the Hessian of the cost function F are provided in `gradientSurface.m` and `hessianSurface.m`, respectively.

Exercise 1

Implement the Line Search Newton - Conjugate Gradient (LS-Newton-CG) algorithm (Algorithm 7.1 from Nocedal-Wright). More help is provided in `Cody Coursework`.
Submit your implementation via `Cody Coursework`. **[30pt]**

Exercise 2

You are given an adaptation of the trust region SR-1 function in `trustRegionLS.m` and the 2D subspace in `solverCM2dSubspaceExtLS.m`. Point out and explain the relevant modifications in the solver `solverCM2dSubspaceExtLS.m`.
Submit your solution via `TurnitIn`. **[20pt]**

Exercise 3

Compute and plot a minimal area surface function with given the boundary conditions:

- $\star$ Bottom edge: $B_B(t) = \sin(\pi t)$, $t \in [0, 1]$.
- $\star$ Left edge: $B_L(t) = \sin(\pi t + \pi)$, $t \in [0, 1]$.
- $\star$ Top edge: $B_T(t) = \sin(3\pi t)$, $t \in [0, 1]$.
- $\star$ Right edge: $B_R(t) = \sin(3\pi t + \pi)$, $t \in [0, 1]$.

(a) using the LS-Newton-CG algorithm,

(b) using the BFGS algorithm (provided),

(c) using the Trust Region-SR1 algorithm (provided in **Ex2**).

Discretization with $q = 14$ yields a problem of reasonable size. Provide any parameters and explanations that you consider relevant in the minimisation.
Submit your solution via `TurnitIn`. **[50pt]**

Remark. The submission to `TurnitIn` should not exceed 6 pages. Avoid submitting code unless explicitly asked for and focus on explaining your results.