Exercise: Gradients, Hessians, and Level Sets of Convex Quadratic Functions

Introduction to Optimization lecture
at Ecole Centrale Paris / ESSEC Business School

Dimo Brockhoff
firstname.lastname@inria.fr

November 28, 2016

Abstract

In the lecture, we have seen the concepts of gradients, Hessian matrices, and level sets. Here, we will apply these concepts to specific convex quadratic functions of the form $f(x) = \mathbf{x}^T A \mathbf{x}$.

1 Optima and Gradients of Specific Convex Quadratic Functions

Specifically, we will have a closer look at the following two sets of functions that are both parametrized by the scalar value $\alpha \in \mathbb{R}$:

$$f_\alpha(\mathbf{x}) = \alpha \sum_{i=1}^{n} \mathbf{x}_i^2 \ \ , \alpha > 0$$

$$f_\alpha^{\text{elli}}(\mathbf{x}) = \frac{1}{2} \sum_{i=1}^{n} 10^{\alpha(\frac{i-1}{n-1})} \mathbf{x}_i^2$$

1. What are the optima of f_α and f_α^{elli}?
2. Compute the gradient of the above functions.
3. Compute the Hessian for the above functions.

4. Implement objective functions `falpha` and `felli` that take as input a vector $\mathbf{x}$ of $\mathbb{R}^n$ and a scalar $\alpha \in \mathbb{R}$ and output $f_\alpha(\mathbf{x})$ and $f_\alpha^{\text{elli}}(\mathbf{x})$ respectively.

5. Likewise, implement functions `gradientFalpha` and `gradientFelli` that take as input $\mathbf{x} \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$ and output the gradient of the above functions in $\mathbf{x} \in \mathbb{R}^n$.

6. Last, implement functions `hessianFalpha` and `hessianFelli` that take as input $\mathbf{x} \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$ and output the Hessian matrix of the above functions in $\mathbf{x} \in \mathbb{R}^n$.

2 Plotting Level Sets

To get a better idea of how the two above parameterized objective functions look like, we will now investigate their level sets with python.

7. Plot a few level sets of your own choice for $f_\alpha(\mathbf{x})$ and $\alpha = 1/2$ in dimension 2. Useful python commands for plotting are contained in the `pyplot` module of `matplotlib` which you can load as

   ```
   from matplotlib import pyplot as plt
   ```

   and which provides a MATLAB-like interface. Probably the best way to plot level sets is via the `contour` command which you can evoke as

   ```
   plt.contour(X,Y,Z)
   ```

   and in which `X` and `Y` can be created for example by

   ```
   delta = 0.025
   x = np.arange(-2.0, 2.0, delta)
   y = np.arange(-2.0, 2.0, delta)
   X, Y = np.meshgrid(x, y)
   ```

   and where `Z` is a two-dimensional numpy array with the f-values to display (i.e. `Z[i][j]` should contain the function value to be displayed in the point `(X[i][j],Y[i][j])`). Remember that adding a '`?`' after a python command(, module, class, function, ...) in ipython displays the corresponding help.

7. Do the same for f_α^{elli} and a few values of α, e.g., $\alpha \in \{1, 3, 6\}$. In order to better compare the level sets for varying α, it might be interesting to look at the `subplot` command (see `plt.subplot?`). What do you observe?

3 Optional

- What happens if $\alpha < 1$ for the level sets of f_α^{elli}, in particular, what if $\alpha < 0$?
- Think about good strategies to find the optimum from an arbitrary starting point if the algorithm is allowed to take into account (only) the function values. What can you do if you know the gradient? What is possible when you also know the Hessian?