Variational Methods for Computer Vision
Lecture: M. Möller
Exercises: J. Geiping
Winter Semester 17/18

Visual Scene Analysis
Institute for Computer Science
University of Siegen

Weekly Exercises 2

Room: H-C 6336/37

Friday, 03.11.2017, 14:15-15:45

Submission deadline: Monday, 30.10.2016, 8:15 in Room H-C 6336

Programmming: Email your solution to `jonas.geiping@uni-siegen.de`

You have two weeks to finish this exercise, so don't be alarmed ¯_(ツ)_/¯ .

Theory

Exercise 1. (4 Points) Let $K \in R^{n \times m}$ be a matrix and $f \in \mathbb{R}^n$ a vector. Show that the quadratic loss function

$$E(u) = \frac{1}{2}||Ku - f||^2$$

is convex.
Hint: You may want to show as an intermediate result, that the composition of a matrix and a convex function is also convex.

Exercise 2. (4 Points) Convexity is a fundamental tool in optimization, due to the following theorem:
Let $E : \mathbb{R}^n \to \mathbb{R} \in C^1(\mathbb{R}^n)$ be a convex function. Then $\nabla E(\hat{u}) = 0$ implies that $\hat{u}$ is a global minimizer of E.
A global minimizer is a vector $\hat{u} \in R^n$ so that $E(\hat{u}) \leq E(u) \ \forall u \in R^n$. Your task is to write a short proof of this theorem.

Exercise 3. (2 Points) Append your previous result and show that if the function E is strictly convex, then $\hat{u}$ is a unique minimizer ($\hat{u}$ is a unique minimizer of E if $E(\hat{u}) < E(u) \ \forall u \in \mathbb{R}^n$).

Exercise 4. (4 Points) We want to approximate a function $f : [0, 1] \to \mathbb{R}$ by a polynomial $p(x) = c_0 + c_1 x^1 + \cdots + c_n x^n$ of degree n or less. For this we define the objective function

$$E(c) = \int_0^1 \left(f(x) - p_c(x)\right)^2 \, dx.$$

Show that the coefficients $c = (c_0, \ldots, ...c_n)$ that minimize the function E are given as solutions to a linear system $Ac = b$. Give explicit formulas for A and b.

Programming

For our programming exercise we will construct a small Matlab framework for energy minimization.

Exercise 5. (8 Points) Write a matlab class "energy" that codifies an energy functional and its gradient, so that we can minimize it conveniently. Your class should have the following features:

- An object can be constructed by giving a function handle to an energy and a function handle to a gradient
- The operator "+"(`plus`) is overloaded, so that two energy objects can be added.
- The operator ".*" (`times`) is overloaded, so that the energy can be multiplied by a constant scalar
- The operator "*" (`mtimes`) is overloaded so that an energy can be composed with a matrix (Given an energy $G(x)$ and a matrix A, one should be able to write `G*A` to denote the energy $G(Ax)$.
- Implement methods that evaluate the energy and the gradient of the energy at a given point.
- Implement a "solve" method that uses gradient descent with backtracking as defined in the lecture to find a local minimizer of E, if it exists.

Test your implementation with the minimization problem

$$\min_{x \in \mathbb{R}^n} x^T Q x - b^T x$$

for some symmetric, positive definite matrix Q and a vector $b \in \mathbb{R}^n$.