Convex Optimization for Machine Learning and Computer Vision

Lecture: T. Wu
Exercises: E. Laude, T. Möllenhoff
Summer Semester 2017

Computer Vision Group
Institut für Informatik
Technische Universität München

Weekly Exercises 1

Room: 02.09.023

Monday, 08.05.2017, 12:15-14:00

Submission deadline: Wednesday, 03.05.2017, 16:15, Room 02.09.023

Convex Sets (8 Points + 4 Bonus)

Exercise 1 (4 Points)**.** Show that a set is convex if and only if its intersection with any line is convex.

Solution. Let $X \subset \mathbb{R}^n$ be a set and let $L_x^y := \{y + \lambda x : \lambda \in \mathbb{R}\} \subset \mathbb{R}^n$ for some $x, y \in \mathbb{R}^n$ be a line.

"$\Rightarrow$": Let X be a convex. Clearly, L_x^y is a convex set and therefore (cf. lecture) $L_x^y \cap X$ is convex.

"$\Leftarrow$": Let $L_x^y \cap X$ be convex for all $x, y \in \mathbb{R}^n$. Let $x_1, x_2 \in X$ and let $\lambda \in [0, 1]$. Then

$$\lambda x_1 + (1 - \lambda)x_2 = x_2 + \lambda(x_1 - x_2) \in L_{x_1 - x_2}^{x_2}.$$

Clearly, $x_1, x_2 \in L_{x_1-x_2}^{x_2}$ and since $L_{x_1-x_2}^{x_2} \cap X$ is convex, $\lambda x_1 + (1 - \lambda)x_2 \in X$ which completes the proof.

Exercise 2 (4 Points)**.** Let $\mathcal{C}$ be a family of convex sets in $\mathbb{R}^n$, $C_1, C_2 \in \mathcal{C}$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $\lambda \in \mathbb{R}$. Prove convexity of the following sets:

- $\bigcap_{C \in \mathcal{C}} C$
- $P := \{x \in \mathbb{R}^n : Ax \leq b\}$
- $C_1 + C_2 := \{x + y : x \in C_1, y \in C_2\}$ (the Minkowski sum of C_1 and C_2)
- $\lambda C_1 := \{\lambda x : x \in C_1\}$ (the λ-dilatation of C_1).

Solution.

- Let $x_1, x_2 \in \bigcap_{C \in \mathcal{C}} C$. Then $x_1, x_2 \in C$ for all $C \in \mathcal{C}$. Since any C is convex, $\mu x_1 + (1-\mu)x_2 \in C$ for all $\mu \in [0, 1]$ and $C \in \mathcal{C}$ and therefore $\mu x_1 + (1-\mu)x_2 \in \bigcap_{C \in \mathcal{C}} C$.
- Let $x_1, x_2 \in P$, which means that $Ax_1 \leq b$ and $Ax_2 \leq b$. Let $\mu \in [0, 1]$. Then, $A(\mu x_1 + (1 - \mu)x_2) = \mu Ax_1 + (1 - \mu)Ax_2 \leq \mu b + (1 - \mu)b = b$. Therefore $\mu x_1 + (1 - \mu)x_2 \in P$.

- Let $x, y \in C_1 + C_2$. Then there exist $x_1, y_1 \in C_1, x_2, y_2 \in C_2$ so that $x = x_1 + x_2$ and $y = y_1 + y_2$. Let $\mu \in [0,1]$. Then, since C_1, C_2 convex $\mu x + (1-\mu)y = \mu x_1 + \mu x_2 + (1-\mu)y_1 + (1-\mu)y_2 = \underbrace{\mu x_1 + (1-\mu)y_1}_{\in C_1} + \underbrace{\mu x_2 + (1-\mu)y_2}_{\in C_2} \in C_1 + C_2$.

- Let $x, y \in C_1$ and $\mu \in [0,1]$. Then, since C_1 convex, $\mu\lambda x + (1-\mu)\lambda y = \lambda \underbrace{(\mu x + (1-\mu)y)}_{\in C_1} \in \lambda C_1$.

Definition (Convex Hull)**.** The convex hull $\operatorname{conv}(S)$ of a finite set of points $S \subset \mathbb{R}^n$ is defined as

$$\operatorname{conv}(S) := \left\{ \sum_{i=1}^{|S|} a_i x_i : x_i \in S, \sum_{i=1}^{|S|} a_i = 1, a_i \geq 0 \right\}$$

Exercise 3 (4 Points)**.** Prove the following statement: Let $n \in \mathbb{N}$ and let $A \subset \mathbb{R}^n$ contain $n+2$ elements: $|A| = n+2$. Then there exists a partition of A into two disjoint sets A_1, A_2

$$A = A_1 \dot{\cup} A_2,$$

(meaning that $A_1 \cap A_2 = \emptyset$) so that the convex hulls of A_1 and A_2 intersect:

$$\operatorname{conv}(A_1) \cap \operatorname{conv}(A_2) \neq \emptyset.$$

You may use the following hint. Don't forget to prove the hint!

Hint: Let $x_1, \dots, x_{n+2} \in \mathbb{R}^n$. Then the set $\{x_1 - x_{n+2}, \dots, x_{n+1} - x_{n+2}\}$ is linearly dependent and there exist multipliers $a_1, \dots, a_{n+2}$, not all of which are zero, so that

$$\sum_{i=1}^{n+2} a_i x_i = 0, \quad \sum_{i=1}^{n+2} a_i = 0.$$

The desired partition is formed via all points corresponding with $a_i \geq 0$ and all points with $a_i < 0$.

Solution. Let $A := \{x_1, x_2, \dots, x_{n+2}\} \subset \mathbb{R}^n$. Since $n+1$ vectors in $\mathbb{R}^n$ are always linearly dependent there exist scalars $a_1, \dots, a_{n+1}$, not all of which are zero so that

$$\sum_{i=1}^{n+1} a_i(x_i - x_{n+2}) = \sum_{i=1}^{n+1} a_i x_i + \underbrace{\left(-\sum_{i=1}^{n+1} a_i \right)}_{=:a_{n+2}} x_{n+2} = 0.$$

Then, by construction $\sum_{i=1}^{n+2} a_i = 0$. Define $A_1 := \{x_i : a_i > 0\}$ and $A_2 := \{x_j : a_j \leq 0\}$. Clearly, $A = A_1 \dot{\cup} A_2$ forms a partition and A_1, A_2 are both nonempty. Suppose A_2 was empty. Then $a_i > 0$ for all $1 \leq i \leq n+2$. But $a_{n+2} := -\sum_{i=1}^{n+1} a_i < 0$ contradicts this assumption (The same holds for A_1). We have that

$$0 = \sum_{\{i : a_i < 0\}} a_i x_i + \sum_{\{j : a_j \geq 0\}} a_j x_j \iff \sum_{\{i : a_i < 0\}} \underbrace{-a_i}_{\geq 0} x_i = \sum_{\{j : a_j \geq 0\}} a_j x_j,$$

and on the other hand

$$0 = \sum_{\{i:a_i<0\}} a_i + \sum_{\{j:a_j\geq 0\}} a_j \iff \sum_{\{i:a_i<0\}} -a_i = \sum_{\{j:a_j\geq 0\}} a_j =: w > 0.$$

Altogether this yields

$$\underbrace{\sum_{\{i:a_i<0\}} \frac{-a_i}{w} x_i}_{\in \mathrm{conv}(A_1)} = \underbrace{\sum_{\{j:a_j\geq 0\}} \frac{a_j}{w} x_j}_{\in \mathrm{conv}(A_2)},$$

which completes the proof. The theorem is called Radon's Theorem.

Intro to Sparse Matrices in MATLAB (5 Points)

Throughout the course we will work in the finite dimensional setting, i.e. we discretely represent gray value images $f : \Omega \to \mathbb{R}$ or color images $f : \Omega \to \mathbb{R}^3$ as (vectorized) matrices $f \in \mathbb{R}^{m\times n}$ ($\text{vec}(f) \in \mathbb{R}^{mn}$) respectively $f \in \mathbb{R}^{m\times n\times 3}$ ($\text{vec}(f) \in \mathbb{R}^{3mn}$). To discretely express functionals like the total variation for smooth f

$$TV(f) := \int_\Omega \|\nabla f(x)\| \,\mathrm{d}x$$

you will therefore need a discrete gradient operator

$$\nabla := \begin{pmatrix} D_x \\ D_y \end{pmatrix}$$

for vectorized representations $\text{vec}(f)$ of images $f \in \mathbb{R}^{m\times n}$ so that

$$TV(f) = \|\nabla \text{vec}(f)\|_{2,1} = \sum_{i=1}^{nm} \sqrt{(D_x \cdot \text{vec}(f))_i^2 + (D_y \cdot \text{vec}(f))_i^2}.$$

The aim of this exercise is to derive the gradient operator and learn how to implement it with MATLAB.

Exercise 4 (1 Point)**.** Let $f \in \mathbb{R}^{m\times n}$ be a discrete grayvalue image. Your task is to find matrices $\tilde{D}_x$ and $\tilde{D}_y$ for computing the forward differences f_x, f_y in x and y-direction of the image f with Neumann boundary conditions so that:

$$f_x = f \cdot \tilde{D}_x := \begin{pmatrix} f_{12} - f_{11} & f_{13} - f_{12} & \dots & f_{1n} - f_{1(n-1)} & 0 \\ f_{22} - f_{21} & \dots & & & 0 \\ \vdots & & & \vdots & 0 \\ f_{m2} - f_{m1} & \dots & & f_{mn} - f_{m(n-1)} & 0 \end{pmatrix} \tag{1}$$

and

$$f_y = \tilde{D}_y \cdot f = \begin{pmatrix} f_{21} - f_{11} & f_{22} - f_{12} & \dots & & f_{2n} - f_{1n} \\ f_{31} - f_{21} & \dots & & & f_{3n} - f_{2n} \\ \vdots & & & & \vdots \\ f_{m1} - f_{(m-1)1} & \dots & & & f_{mn} - f_{(m-1)n} \\ 0 & \dots & & 0 & 0 \end{pmatrix}. \tag{2}$$

Solution. The corresponding operators $\tilde{D}_x$ and $\tilde{D}_y$ are given as follows:

$$\tilde{D}_x = \begin{pmatrix} -1 & 0 & \dots & 0 & 0 \\ 1 & -1 & \dots & 0 & 0 \\ 0 & 1 & \ddots & 0 & 0 \\ 0 & 0 & \ddots & -1 & 0 \\ 0 & \dots & & 1 & 0 \end{pmatrix} \qquad \tilde{D}_y = \begin{pmatrix} -1 & 1 & & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ 0 & 0 & \ddots & \ddots & 0 & 0 \\ 0 & 0 & & -1 & 1 & 0 \\ 0 & 0 & \dots & & -1 & 1 \\ 0 & 0 & \dots & & 0 & 0 \end{pmatrix} \tag{3}$$

Exercise 5 (1 Point)**.** Implement the derivative operators from the previous exercise using MATLABs `spdiags` command. Load the image from the file `Vegetation-028.jpg` using the command `imread` and convert it to a grayvalue image using the command `rgb2gray`. Finally apply the operators to the image and display your results using `imshow`.

For our algorithms it is more convenient to represent an image f as a vector $\text{vec}(f) \in \mathbb{R}^{mn}$, that means that the columns of f are stacked one over the other.

Exercise 6 (1 Point)**.** Derive a gradient operator

$$\nabla = \begin{pmatrix} D_x \\ D_y \end{pmatrix}$$

for vectorized images so that

$$D_x \cdot \text{vec}(f) = \text{vec}(f_x) \qquad D_y \cdot \text{vec}(f) = \text{vec}(f_y)$$

You can use that it holds that for matrices A, X, B

$$AXB = C \iff (B^\top \otimes A)\text{vec}(X) = \text{vec}(C)$$

where $\otimes$ denote the Kronecker (MATLAB: `kron`) product.

Experimentally verify that the results of Ex. 2 and Ex. 3 are equal by reshaping them to the same size using MATLABs `reshape` or the `:` operator, and showing that the norm of the difference of both results is zero.

Solution. We have $f_x = f \cdot \tilde{D}_x = I \cdot f \cdot \tilde{D}_x$, where I is the identity matrix. If we set $A := I$, $X := f$, $B := \tilde{D}_x$ and $C := f_x$ we obtain using the formula,

$$D_x = \tilde{D}_x^\top \otimes I. \tag{4}$$

We have $f_y = \tilde{D}_y \cdot f = \tilde{D}_y \cdot f \cdot I$. We set $A := \tilde{D}_x$, $X := f$, $B := I$ and $C := f_y$ and obtain using the formula:

$$D_y = I \otimes \tilde{D}_y. \tag{5}$$

Exercise 7 (1 Point)**.** Assemble an operator ∇_c for computing the gradient (or more precisely the Jacobian) of a color image $f \in \mathbb{R}^{n \times m \times 3}$ using MATLABs `cat` and `kron` commands.

Solution.

$$\nabla_c := \begin{pmatrix} D_x & 0 & 0 \\ 0 & D_x & 0 \\ 0 & 0 & D_x \\ D_y & 0 & 0 \\ 0 & D_y & 0 \\ 0 & 0 & D_y \end{pmatrix} = \begin{pmatrix} I \otimes D_x \\ I \otimes D_y \end{pmatrix} \tag{6}$$

Exercise 8 (1 Point)**.** Compute the color total variation given as

$$TV(f) = \|\nabla_c \text{vec}(f)\|_{F,1} = \sum_{i=1}^{nm} \left\| \begin{pmatrix} (D_x \cdot \text{vec}(f_r))_i & (D_x \cdot \text{vec}(f_g))_i & (D_x \cdot \text{vec}(f_b))_i \\ (D_y \cdot \text{vec}(f_r))_i & (D_y \cdot \text{vec}(f_g))_i & (D_y \cdot \text{vec}(f_b))_i \end{pmatrix} \right\|_F$$

of the two images `Vegetation-028.jpg` and `Vegetation-043.jpg` and compare the values. What do you observe? Why?