Convex Optimization for Machine Learning and Computer Vision

Lecture: Dr. Tao Wu
Exercises: Zhenzhang Ye
Winter Semester 2019/20

Computer Vision Group
Institut für Informatik
Technische Universität München

Weekly Exercises 0

Room: 02.09.023

Wednesday, 23.10.2019, 12:15 - 14:00

Intro to Sparse Matrices in MATLAB (or Python)

For python users: for now, we need the `numpy`, `scipy`, `pillow`, `matplotlib` packages. You can also use Jupyter notebook for nice visualization but it is not mandatory.

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 $f \in \mathbb{R}^{m\times n}$ ($\mathrm{vec}(f) \in \mathbb{R}^{mn}$) respectively $f \in \mathbb{R}^{m\times n\times 3}$ ($\mathrm{vec}(f) \in \mathbb{R}^{3mn}$). 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 $\mathrm{vec}(f)$ of images $f \in \mathbb{R}^{m\times n}$ so that

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

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

Exercise 1 (0 Points)**.** 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:

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

and

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

Exercise 2 (0 Points)**.** 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 Python: Use e.g. `scipy.sparse.spdiags`; Use `pillow` to read images as grayvalue and take the data as numpy array; Use `matplotlib` to display your result.

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 3 (0 Points)**.** 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.

Exercise 4 (0 Points)**.** 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. (Python: check out `scipy.sparse.{kron, hstack, vstack}`)

Exercise 5 (0 Points)**.** 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?