Exercises
Computational Intelligence Lab
SS 2017

Machine Learning Institute
Dept. of Computer Science, ETH Zürich
Prof. Dr. Thomas Hofmann
Web http://cil.inf.ethz.ch/

Series 11, May 22-23, 2016
(Sparse Coding and Dictionary Learning)

Problem 1 (Sparse Coding for Image Processing):

In Exercise 9, you have seen two baseline methods for the task of Road Extraction from Satellite Images (one based on a linear classifier, and one based on a convolutional neural network). Both methods predict a label {`road=1`, `background=0`} for each patch (recall that each image was partitioned into disjoint 16x16 pixel regions, called patches). Solution python code for both variants is available here:

github.com/dalab/lecture_cil_public/tree/master/exercises/ex9

In this exercise here, we are going to take these patch-wise predictions, and use sparse-coding to improve them. Your task is as follows:

1. Download the training data from the competition website below, as well as the python-code for generating the predictions for each patch from the github repository above.

   inclass.kaggle.com/c/cil-road-segmentation.

2. From the given ground-truth images, generate a dictionary of square super-patches of size 5×5 patches (i.e. $5 \cdot 16\text{px} \times 5 \cdot 16\text{px} = 80\text{px} \times 80\text{px}$) each. Each such super-patch forms one column of your dictionary matrix $\mathbf{U}$.

3. Now we use the predictions generated from the CNN (or linear classifier) from Exercise 9. This gives one prediction per patch, that is 5×5 predictions per super-patch. We assign the same prediction to all pixels within the same patch. Using these predictions for one super-patch as your noisy target vector $\mathbf{x} \in \mathbb{R}^{80^2}$, use sparse coding to recover sparse weights $\mathbf{z}$ s.t.

$$\mathbf{x} \approx \mathbf{U}\mathbf{z}$$

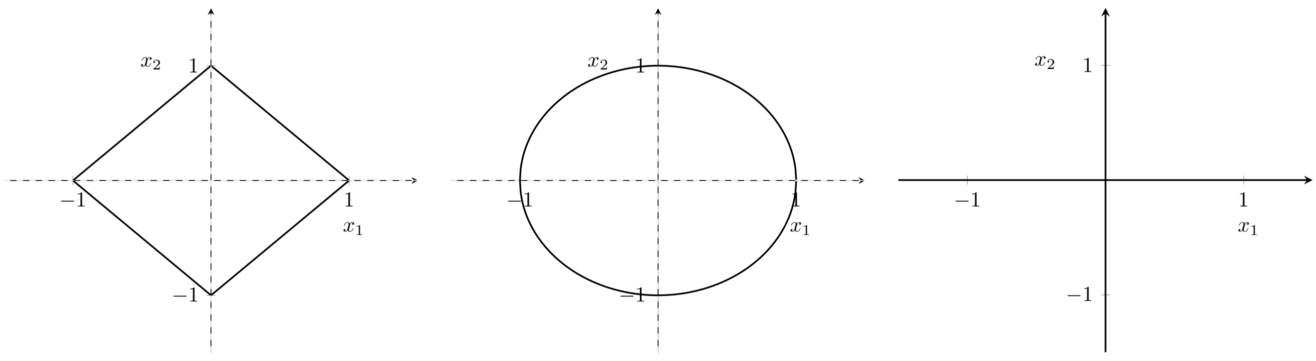
   in terms of the actual ground truth dictionary $\mathbf{U}$. Recall that each column of $\mathbf{U}$ corresponds to actual pixels. Given a raw prediction $\mathbf{x}$ and decoded $\mathbf{z}$, visualize the corresponding cleaned version $\mathbf{U}\mathbf{z}$ to inspect if your process works.

   As your sparse-coding algorithm, you can either your own implementation of matching pursuit, or use an existing method such as e.g. `sparse_encode` from `scikit-learn`.

4. The above method serves as a post-processing of raw patch-based predictions to improve the output for the actual road-segmentation task. Upload your improved predictions to `kaggle`, to see how it improves the score.
   Can you improve the dictionary to increase your score on the unknown test set? Can you improve the score on the test set further, by generating an even larger dictionary, by also using e.g. many translated versions of some ground-truth labels, for step 2?

Problem 2 (Compressed Sensing):

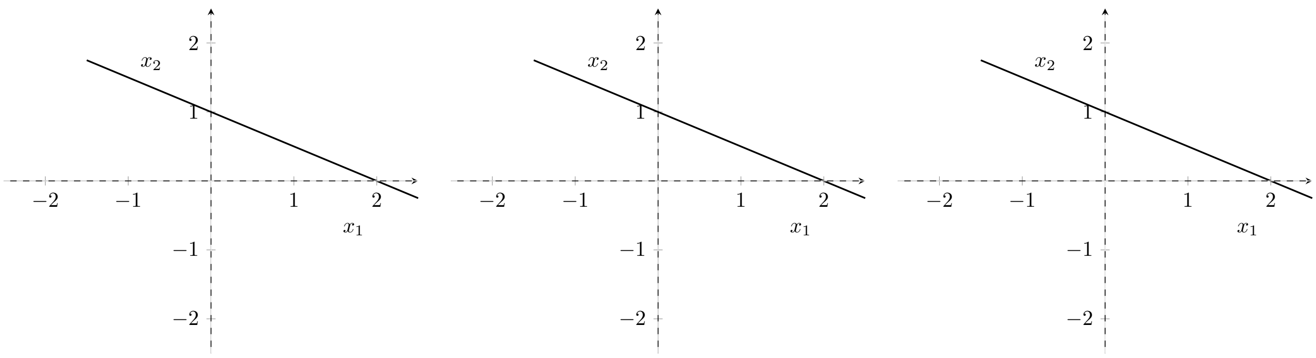
a. Map each of the three equations $\|\mathbf{x}\|_2 = 1$, $\|\mathbf{x}\|_1 = 1$, and $\|\mathbf{x}\|_0 = 1$ to a plot among a., b., or c. on the following figure. Note that $\mathbf{x}$ is s 2D vector with coordinates x_1 and x_2 (i.e. $\mathbf{x} = \Big[x_1, x_2\Big]$).



a. .... b. .... c. ....

b. Show the solution of each optimization problem on plots a., b., and c. of the following figure.

$\min \|\mathbf{x}\|_2$	$\min \|\mathbf{x}\|_1$	$\min \|\mathbf{x}\|_0$
Subject to $\frac{1}{2}x_1 + x_2 = 1$	Subject to $\frac{1}{2}x_1 + x_2 = 1$	Subject to $\frac{1}{2}x_1 + x_2 = 1$



a. b. c.

c. We can formulate the above three optimization problem as

$$\min \|\mathbf{x}\|_p$$
$$\text{subject to } \frac{1}{2}x_1 + x_2 = 1,$$

where $p \in \{0, 1, 2\}$. Mark the right sentence using your previous answers.

[ ] Solutions of the constrained problems have intersection for $p = 1$ and $p = 0$.

[ ] Solutions of the constrained problems have intersection for $p = 2$ and $p = 0$.

Problem 3 (Compressed Sensing):

Please find the iPython notebook Compressed_sensing.ipynb from

github.com/dalab/lecture_cil_public/tree/master/exercises/ex11

answer the question in this file.