

---

## LAB 2: Spectral filters and multi-class classification

---

- This lab is about binary and multiclass classification and model selection on synthetic as well as real data, focusing on the role and on the properties of the spectral filters.
- The aim of the lab is to play with the libraries and to get a practical grasp of what we have discussed in class.
- Follow the instructions below.

**Goal:**

This lab is divided in three parts depending of their level of complexity (**Beginner**, **Intermediate**, **Advanced**). Your goal is to complete entirely, at least, one of the tree parts.

Download the file `regml2014_lab2.zip`, extract it and add all the sub-folders to the MATLAB path. This file includes all the code you need!

### PART I: Beginner

#### Overture: Warm up

Run the file `gui_filter.m` and a GUI will start. Have a look to the various components.

- With the data simulation option generate a dataset of type *spiral* (press the button `load data` to generate).
- Observe the generated data (the buttons `plot training` and `plot test` will allow you to toggle between training and test set).
- Choose the *Truncated SVD* filter and the *Gaussian* kernel (be sure to check the *autosigma* checkbox as *on*).
- Have a look to the parameter selection part and the various options of KCV. To choose the regularization parameter you can either choose KCV or set a fixed value.
- Press the button `run` to perform training and classification.

## Interlude: The Geek Part

Back on the matlab shell, have a look to the content of directory `./spectral_reg_toolbox`. There you will find, among the others, the code for command `learn` (used for training), `patt_rec` (used for testing), `kcv` (used for model selection on the training set).

For more informations about the parameters and the usage of those scripts, type:

```
1 >> help learn
2 >> help patt_rec
3 >> help kcv
```

Finally, you may want to have a look at the content of directory `./dataset_scripts` and in particular to file `create_dataset.m` that will allow you to generate synthetic data of different types.

### NOTE:

Remember that in the MATLAB code the regularization parameter for the *Tikhonov* (`r1s.m`), *T-SVD* (`tsvd.m`) and *cut-off* (`cutoff.m`) is  $\tau$  instead of  $\lambda$ . For the *NU-Method* (`nu.m`) and the *Landweber-Method* (`land.m`), the regularization parameter  $\tau$  is the number of iterations, that is, roughly speaking  $\tau \sim \frac{1}{\lambda}$ .

## Allegro con brio: Analysis

Carry on the following experiments either using the GUI, when it is possible, or writing appropriate scripts.

i) Generate data of *Spiral* type. Considering three algorithms, namely *RLS*, *Truncated SVD* and *NU-Method*. Observe how the training and test error changes as:

- We change (increase or decrease) the cardinality of the training set. For instance, `[10,100,1000]` as long as MATLAB supports you!
- We change (increase or decrease) the amount of regularization for each algorithm. For instance, *fixed value*.
- The amount of wrong labels on the generated data grows. For instance, `[0.01,0.02,0.05,0.1]`.

Run training and test for various choices of the suggested parameters.

ii) Leaving all the other parameters fixed, use the KCV option to select the optimal model and see how it relates to the previous plot. Choose an appropriate range for the regularization parameters, and plot the training error and the test error for each regularization parameter.

iii) Leaving all the other parameters fixed choose an appropriate range `[n_min:n_step:n_max]` of the number of points in the test set and plot the training and test error. What do you observe as  $n$  goes to infinity? How the different regularization parameters affect the learning process? Which are the main differences in terms of regularization between the methods?

## PART II: Intermediate

### Crescendo: Advanced Analysis

Carry on the following experiments either using the GUI or the command line interface. In this part you have to focus more on the effects of the regularization and on the correct choice of `sigma`.

iv) Use *Gaussian* kernel and perform parameter tuning. This time together with the regularization parameter `tau`, you'll have to choose and appropriate kernel parameter `sigma`.

- Try for some `(sigma, tau)` and compare the obtained `training_error` and `test_error`.
- Fix `tau` and observe the effect of changing `sigma`.
- Fix `sigma` and observe the effect of changing `tau`.
- Do you notice (and if so, when) any overfitting/oversmoothing effect?

v) Compare *RLS* with *NU-Method* on a kernel of your choice

- Tune the parameters with KCV.
- Compare the time needed to obtain a solution.
- Compare the training and test errors.

## PART III: Advanced

## Finale: The Challenge

The challenge consists in a learning task using a real dataset, namely *USPS*. This dataset contains a number of handwritten digits images. The problem is to train *the best classifiers* that are able to discriminate between digits `3`, `8` and `0`.

Have a look at the script `demo_lab2.m`. This script contains a code snippet to perform a multi-class classification task using the previously presented MATLAB scripts (see **Interlude**).

You should understand what the scripts are supposed to do, and train the classifiers in order to perform a One vs. All classification for all the combination of the digits 3, 8 and 0.

Once the classifiers are trained, the model must be exported in a matrix file by means of the `save_challenge_2.m` script (to see how to use it please try the command `help save_challenge_2`).

**Submission:** You should drop your results in a MATLAB matrix file named `name-surname.mat` to the link: <http://www.dropitto.me/regml> with password `regml2014` by the end of the challenge session. The results are presented during the next class and the first five will be awarded by an **awesome gift**. The score is based on the accuracy of the classifier on a completely independently sampled test set.

**Deadline:** 6:00 PM.