
LAB 2: Spectral filters & multi-class classification

- This laboratory is about binary and multi-class classification and model selection on synthetic as well as real data, focusing on the role and on the properties of spectral filters.
- The aim of the lab is to play with the libraries and 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 entirely complete at least one of the three parts.

Setup instructions:

PC Login (LABS)

If you don't have a laptop with MATLAB, use a PC in SW1 and SW2, DIBRIS

- Username: regml
- Password: RegML2016

Running MATLAB

1. Double-click on Windows 7 icon on the desktop
2. Run MATLAB from Start → All Programs → MATLAB → R2013a → Matlab executable
3. Download the file `regml2016_lab2.zip` from the syllabus on the course website (<http://lcs1.mit.edu/courses/regml/regml2016/#syllabus>), extract it and add all the sub-folders to the MATLAB path. This file includes all the code you need!

Important notice: All files on desktop PCs will be erased at the end of each lab session. If you want to keep your files, you will have to save them to other locations (e.g. personal cloud storage services).

PART I: Beginner

Overture: Warm up

Run the file `gui_filter.m` and a GUI will start. Check out the various components.

- With the data simulation option generate a dataset of type *spiral* (press the button `load data` to generate it).
- 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 at the parameter selection part and the various options of KCV. To select 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 at the content of directory `./spectral_reg_toolbox`. There you will find, among the others, the code for commands `learn` (used for training), `patt_rec` (used for testing), `kcv` (used for model selection on the training set).

For more information 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 kinds.

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 called τ instead of λ . For the *NU-Method* (`nu.m`) and the *Landweber-Method* (`land.m`), the regularization parameter τ 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. Consider three algorithms, namely *RLS*, *Truncated SVD* and *NU-Method*. Observe how the training and test errors change 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, by modifying the *fixed value* field.
- The amount of wrong labels in the generated data grows. For instance, *wrong labels ratio* = `[0.01, 0.02, 0.05, 0.1]`.

Run training and testing 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 training set and plot the training and test errors. What do you observe as $n \rightarrow \infty$? How do 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 regularization and on the correct choice of `sigma`.

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

- Try for some pairs `(sigma, tau)` and compare the obtained `training_error` and `test_error`.
- Fix `tau` and observe the effects of changing `sigma`.
- Fix `sigma` and observe the effects of changing `tau`.

- Do you notice (and if so, when) any overfitting/oversmoothing effects?
- 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 One vs. All classification for all the combinations of the digits 3, 8 and 0.

Once the classifiers have been 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/regml2016> with password `regml2016` by the end of the challenge session.

The results will be presented during the next class. The score is based on the accuracy of the classifier on a completely independently sampled test set.

Deadline: 6:00 PM.