# *LAB3: Dimensionality Reduction*

*Goal: to study the problem of dimensionality reduction through Principal Component Analysis (PCA) and feature selection through Orthogonal Matching Pursuit (OMP) for a synthetic dataset.*

This lab is divided into two parts depending of their level of complexity (**Beginner, Advanced**). Your goal is to complete entirely, at least, one of the two parts. Please note that a different notation can be used in the code as we used in the lectures.

## PART I: Beginner

### Warm up: Data Generation

Generate a 2-class dataset of D-dimensional points with N points for each class. Start with N = 100 and D = 30 and create a train and a test set.

1. The first two variables (dimensions) for each point will be generated by `MixGauss.py`, i.e., extracted from two Gaussian distributions, with centroids (1, 1) and (-1,-1) and `sigmas` = 0.7 (the first one with Y=1, the second with Y=-1). Adjust the output labels of the classes to be {1,-1} respectively, e.g. using

   `Ytr[Y2tr==0] = 1.`

   To visualise the data: `plt.scatter(Xtr[:,0], Xtr[:,1], 25, Ytr)`
   `plt.show()`
2. The remaining (D-2) variables will be generated by Gaussian noise with `sigma_noise` = 0.01, e.g.,

   `X2tr = normal(0, sigma_noise, size=(2*N, D-2))`
   `X2ts = normal(0, sigma_noise, size=(2*N, D-2))`
3. The final train and test data matrices will be composed as:

   `Xtr = np.concatenate((Xtr, X2tr, axis=1)`

### Principal Component Analysis

4. Compute the principal components of the training set, using the provided class `PCA.py`, i.e.

   `V, d, X_proj = PCA(Xtr, k)`

   Plot the first component of X_proj, the first 2 and the first 3 components. Reason on the meaning of the obtained plots and results. What is the effective dimensionality of this dataset?

5. Visualise/display the square root of the first k=10 eigenvalues, and the coefficients (eigenvectors) associated with the largest eigenvalue, i.e.,

```
plt.plot(range(10), np.sqrt(d))
plt.show()
plt.scatter(range(D), np.abs(V[:,i]))
plt.show()
```

   Same for the second and third largest.

6. Repeat the above steps with a dataset generated using different `sigma_noise` in {0, 0.01, 0.1, 0.5, 0.7, 1:0.2:2}. To what extent is data visualisation by PCA affected by noise?

## Variable Selection

7. Standardize the data matrix, so that each column has mean 0 and standard deviation 1. Use the statistics from the train set Xtr (mean and standard deviation) to standardize the corresponding test set Xts. Useful commands:

```
mean = np.mean(Xtr, axis=0)       % mean of each column
std = np.std(Xtr, axis=0, ddof=1) % standard deviation
Xtr -= mean        % remove mean of each column/dimension
Xtr = Xtr/std      % scale std in each dimension for data point
```

8. Use the orthogonal matching pursuit algorithm, implemented in the class `OMatchingPursuit.py`, using T repetitions, to obtain T-1 coefficients for a sparse approximation of the training set Xtr. Plot the resulting coefficients `w` using

```
plt.stem(range(D), w)
plt.xlim([-1, D])
plt.show()
```

   What is the output when setting T = 3 and what is the interpretation of the indices of these first active dimensions (coefficients)?

9. Check the predicted labels on the training (and test) set when approximating the output using `w`:

```
Ypred = np.sign(np.dot(Xts, w))
error.append(calcErr(Y2ts, Ypred))
```

How does the train and test error change with the number of iterations of the method? Plot the errors on the same plot for increasing T.

1. By applying cross-validation on the training set through the provided `holdoutCVOMP.py` find the optimum number of iterations in the range `intIter = 2:D`

(indicative values: `perc = 0.5, nrip = 30`).

```
it, Vm, Vs, Tm, Ts = holdoutCVOMP(Xtr, Y2tr, 0.5, 5, intIter)
```

Plot the training and validation error on the same plot. What is the behaviour of the training and the validation errors with respect to the number of iterations?

# PART II: Advanced

Compare the results of previous parts and evaluate the benefits of the two different methods for dimensionality reduction and feature selection when choosing N >> D, N ~= D and N << D.