# Algorithm description

In this section I described algorithm to extract fragments which are necessary for implementation because of matrix operations are essentially faster than for loops both in Matlab and python.

## Data

We have -dimensional dataset with labelled data: for each input vector we have label . We have also unlabelled dataset with elements . Let us denote labels as

## Problem

We want to find -dimensional subspace such that for projection of points of dataset the distance between projections of points of different classes will be greater than distance between projections of points of one class.

We also want to have projection of subset as close as possible to projection of subset .

## Model

According to (6) and (7) in [1] for weights matrix ( are numbers of observations) we need to find the greatest eigenvalues of matrix with elements

|  |  |
| --- | --- |
|  | (1) |

where is one data point.

In formula (1) the second summand can be rewritten as product of three matrix

Check of myself.

The first summand can be rewritten as sum of product of elements of three vectors , where vector is sum of column elements of matrix :

## Additional calculations

For our algorithm we also need to find for each element of dataset nearest neighbours from dataset . Let us denote this set and element of this set as .

Let us denote normalised distance (squared distance?) between point and as

Weight of nearest neighbour can be constant or normalised distance. Really we can consider arbitrary set of weights .

## Weights for DAPCA

We have two datasets and . We have unique different labels in dataset . For each element of dataset we have set of NN in dataset . Now we can define weights matrix:

1. If then (attraction of projections of points with the same label in labelled dataset).
2. If then (repulsion of projections of points with different labels).
3. If then (repulsion of projections of points of unlabelled dataset).
4. If then (attraction between projection of unlabelled point and nearest labelled neighbour).
5. If then (no action!).

## Scheme with normalised weights

We have two datasets and with and objects correspondingly. We have unique different labels in dataset . Number of cases with label is . For each element of dataset we have set of NN in dataset . Now we can define weights matrix:

1. If then (attraction of projections of points with the same label in labelled dataset).
2. If then (repulsion of projections of points with different labels).
3. If then (repulsion of projections of points of unlabelled dataset).
4. If then (attraction between projection of unlabelled point and nearest labelled neighbour).
5. If then (no action!).

## Parameters of algorithm

is user defined. Mandatory. We do not define any default value for it.

is optional. Default value is 1.

is optional. Default value is 0.5. I am not sure.

is optional. Default value will be defined later.

is optional. Default value will be defined later.

is optional. Default value is 1 for all pairs of labels.

is optional. Default function is constant 1.

# Bibliography

1. Gorban, A.N., Grechuk, B., Mirkes, E.M., Stasenko, S.V. and Tyukin, I.Y., 2021. High-dimensional separability for one-and few-shot learning. *arXiv preprint arXiv:2106.15416*.