Implementation in MATLAB of the Partial Least Squares algorithm for classification

Case study: fault detection and diagnosis on steel plates

Lorenzo Ferrari Lorenzo Leoni

Department of Engineering and Applied Sciences, University of Bergamo

May 12, 2023

Outline

1. Introduction
2. Description of the PLS algorithm
3. Fault detection and isolation on steel plates
4. Conclusion

Introduction to the PLS technique

Partial least squares (**PLS**), as known as **projection to latent structures**, is a dimensionality reduction technique for maximizing the **covariance** between the predictor (independent) matrix $X \in \mathbb{R}^{n \times m}$ and the predicted (dependent) matrix $Y \in \mathbb{R}^{n \times p}$ for each component of the reduced space $\mathbb{R}^{\alpha}$ with $\alpha \leq m$, where:

- $n =$ number of observations;
- $m =$ number of covariates (input variables);
- $p =$ number of dependent variables (output variables);
- $\alpha =$ dimension of the reduced space in which X is projected.

Popular application of PLS

This technique is often used in **fault detection** and **isolation**. With PLS is possible to treat both regression and classification problems. The matrix X always contains the process variables (e.g. diameter and thickness of a gasket), while the matrix Y only (quantitative) quality variables (e.g. its mechanical seal) in the regression case, whereas in pattern classification the predicted variables are dummy variables (1 or 0) such as:

$$Y = \begin{bmatrix} \boldsymbol{y}_1 & \boldsymbol{y}_2 & \boldsymbol{y}_3 \end{bmatrix} = \begin{bmatrix} 1 & \dots & 1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \dots & 0 & 1 & \dots & 1 & 0 & \dots & 0 \\ 0 & \dots & 0 & 0 & \dots & 0 & 1 & \dots & 1 \end{bmatrix}^\top \quad p = 3$$

where each column of Y corresponds to a *fault class*. The first n_j elements of column j are filled with a 1, which indicates that the first n_j rows of X are data from fault j. In classification case PLS is named **discriminant**.

NIPALS algorithm

The most popular algorithm used in PLS to compute the model parameters is known as **non-iterative partial least squares** (**NIPALS**). There are two versions of this technique:

- **PLS1**: each of the p predicted variables is modeled separately, resulting in one model for each class;
- **PLS2**: all predicted variables are modeled simultaneously.

The first algorithm is more accurate than the other, however it requires more computational time than PLS2 to find the α eigenvectors into which project the m covariates.

Data structures

Before starting with the description of the algorithm, we recall that:

- the matrix $X \in \mathbb{R}^{n \times m}$ is decomposed into a **score matrix** $T \in \mathbb{R}^{n \times \alpha}$ and a **loading matrix** $P \in \mathbb{R}^{m \times \alpha}$ such that $X = \hat{X} + E = T \cdot P^\top + E$, where $E \in \mathbb{R}^{n \times m}$ is the (true) **residual** matrix for X;
- the matrix $Y \in \mathbb{R}^{n \times p}$ is decomposed into a **score matrix** $U \in \mathbb{R}^{n \times \alpha}$ and a **loading matrix** $Q \in \mathbb{R}^{p \times \alpha}$ such that $Y = \hat{Y} + \widetilde{F} = U \cdot Q^\top + \widetilde{F}$, where $\widetilde{F} \in \mathbb{R}^{n \times p}$ is the (true) **residual matrix** for Y.
- the matrix $B \in \mathbb{R}^{\alpha \times \alpha}$ is the **diagonal regression matrix** such that $\hat{U} = T \cdot B$.

Therefore:

$$Y = \hat{U} \cdot Q^\top + F = T \cdot B \cdot Q^\top + F$$

where F is the **prediction error matrix**; B is selected such that the induced 2-norm of F is minimized.

MATLAB code

The following MATLAB code implements the PLS2 algorithm:

```
E = X; % residual matrix for X
F = Y; % residual matrix for Y
[~, idx] = max(sum(Y.*Y));
% search of the j-th eigenvector
for j = 1:alpha
    u = F(:, idx);
    tOld = 0;
    for i = 1:maxIter
        w = (E'*u)/norm(E'*u); % support vector
        t = E*w; % j-th column of the score matrix for X
        q = (F'*t)/norm(F'*t); % j-th column of the...
            % loading matrix for Y
        u = F*q; % j-th column of the score matrix for Y
```

```
    if abs(tOld - t) < exitTol
        break;
    else
        tOld = t;
    end
end
p = (E'*t)/(t'*t); % j-th column of the...
    % loading matrix of X
% scaling
t = t*norm(p);
w = w*norm(p);
p = p/norm(p);
% calculation of b and the error matrices
b = (u'*t)/(t'*t); % j-th column of the...
    % coefficient regression matrix
E = E - t*p';  % update of the residuals for matrix X
F = F - b*t*q'; % update of the residuals for matrix Y
```

```
    % calculation of W, P, T and B2
    W(:, j) = w;
    P(:, j) = p;
    T(:, j) = t;
    B2 = W*(P'*W)^-1*(T'*T)^-1*T'*Y;
end
Y_hat = X*B2; % computation of predictions
```

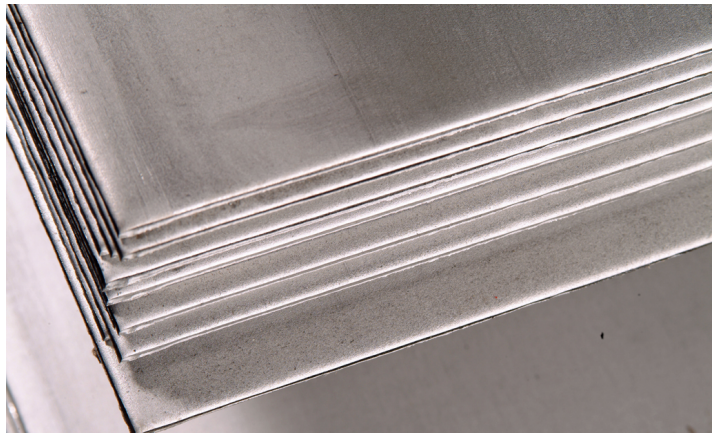
For each row of `Y_hat` the fault class is chosen by assigning 1 to the column whose value si greater than that of the others, 0 otherwise.
Moreover, to increase the performances of PLS it is necessary **normalize** both X and Y before running the algorithm.

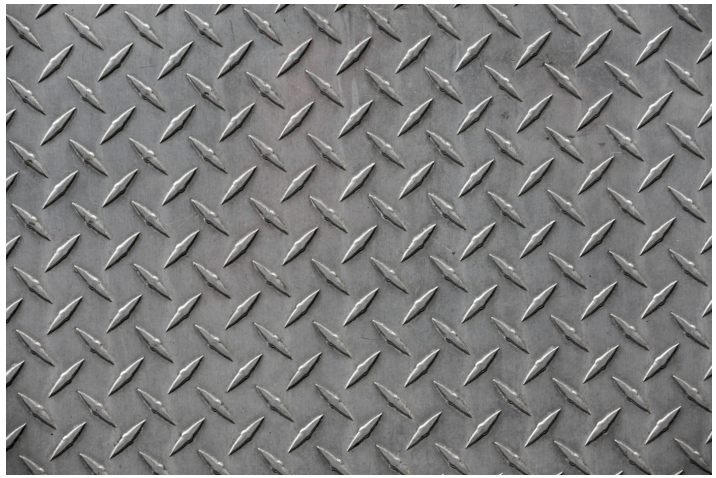
Dataset description

The steel plates faults dataset comes from research by Semeion, Research Center of Sciences of Communication. The aim of the research was to correctly classify the type of surface defects in stainless steel plates. Some information about the dataset:

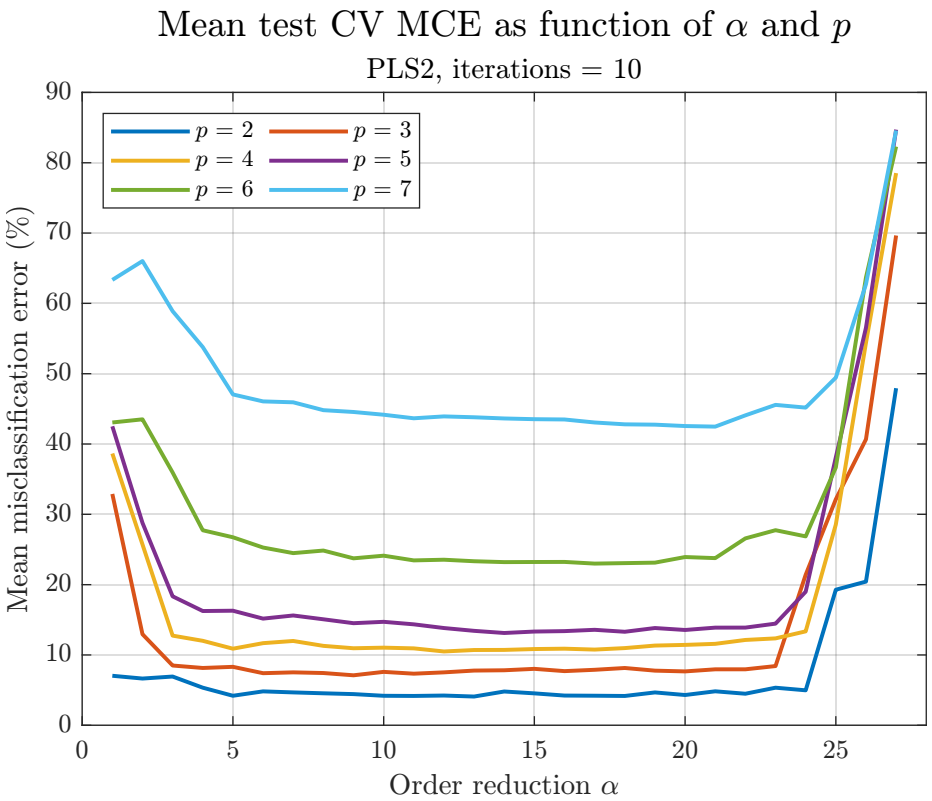
- number of fault classes: $6 + 1$ (no faults);
- number of attributes: 27;
- number of instances: 1941;
- absence of missing values.

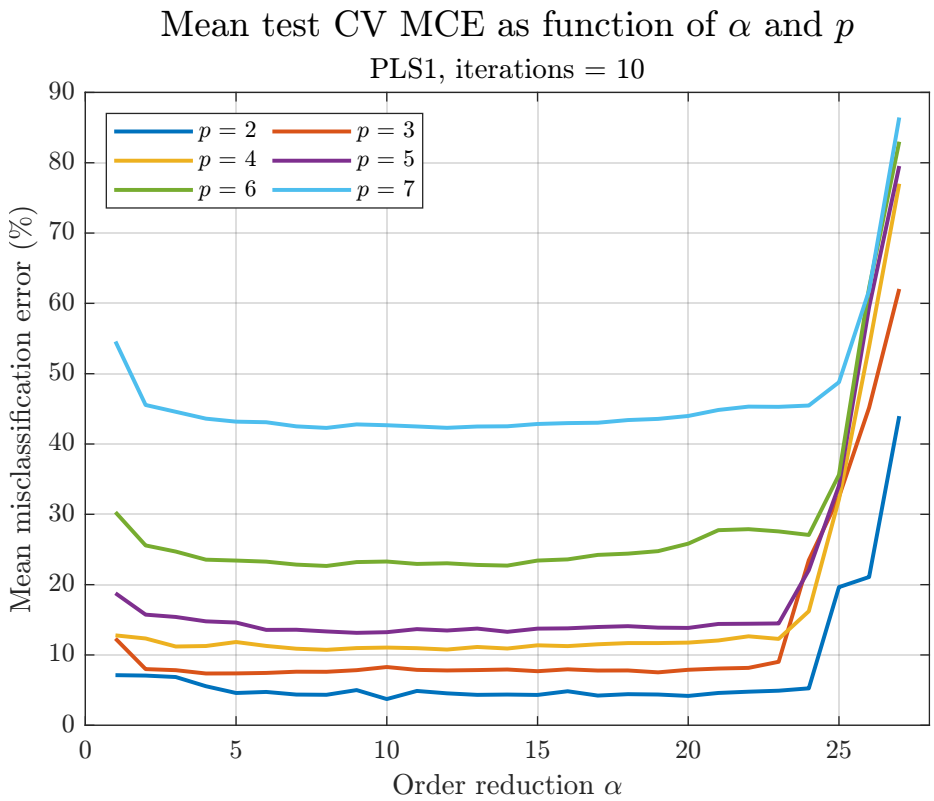
Unfortunately, no further details on the covariates are available.





Preliminary analysis



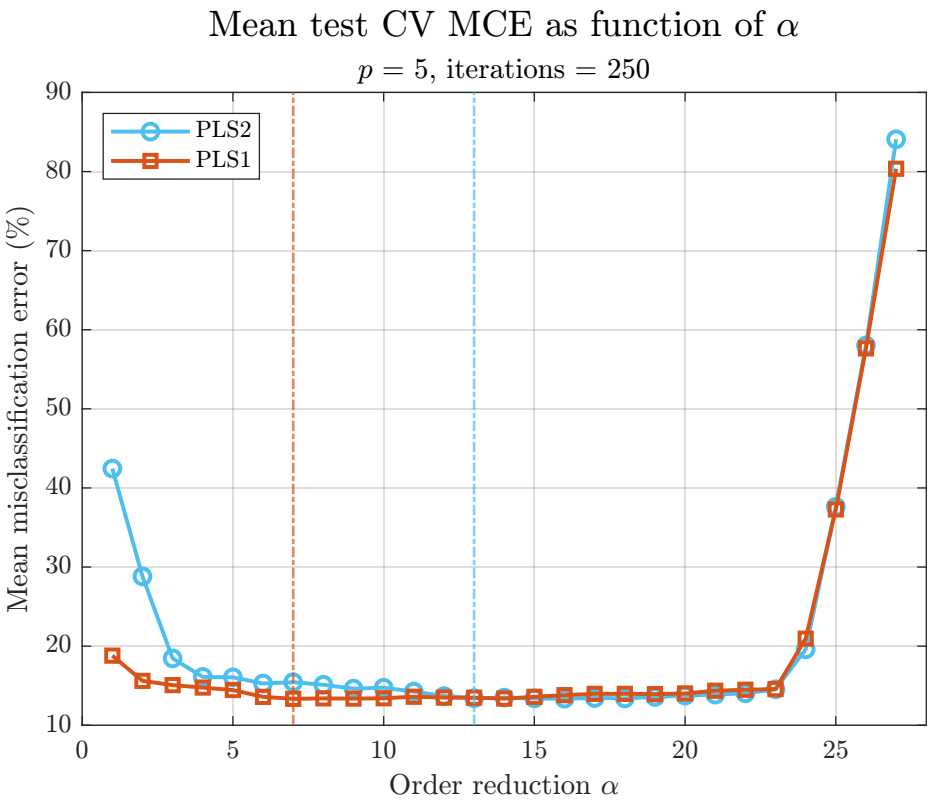


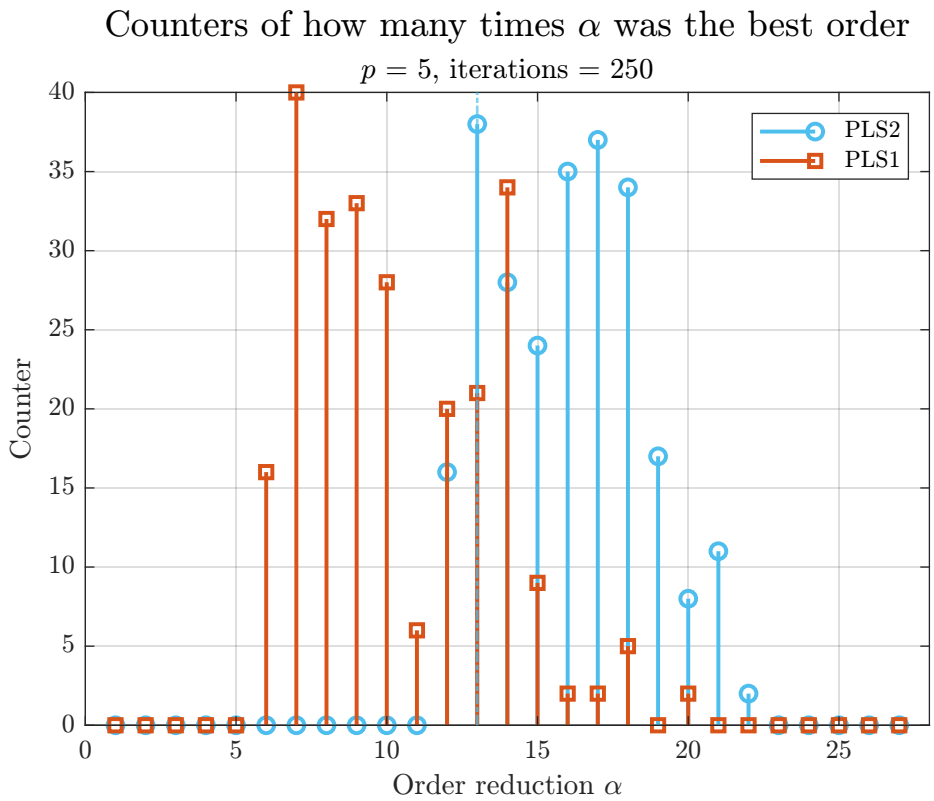
As the number of fault classes p taken into consideration increases, the mean test CV MCE also increases. Both algorithms show a similar behaviour in case of overfitting, while PLS1 underfits less than PLS2.

Methodology

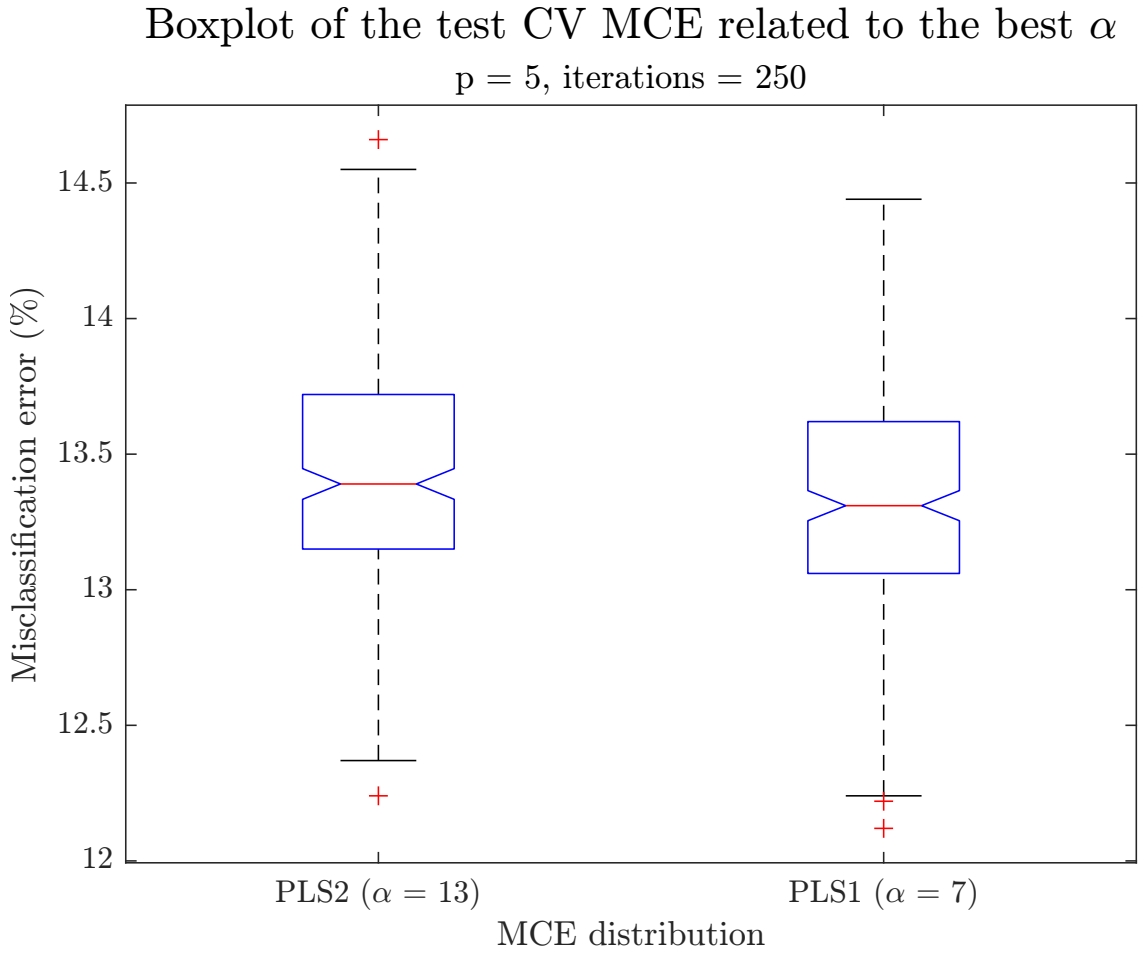
To reduce the dependency on a specific dataset randomization during the 10-folds cross-validation, we decided to iterate this procedure n times (10 in preliminary analysis). We used the CV technique to make inference on the model parameters, especially to choose the best model reduction α.
As we can see from graphs in the previous frame, with $p = 5$ we obtain a model that is a good compromise between complexity and cross-validation performances, therefore we decided to make our in-depth analysis with this model.

In-depth analysis with $p = 5$



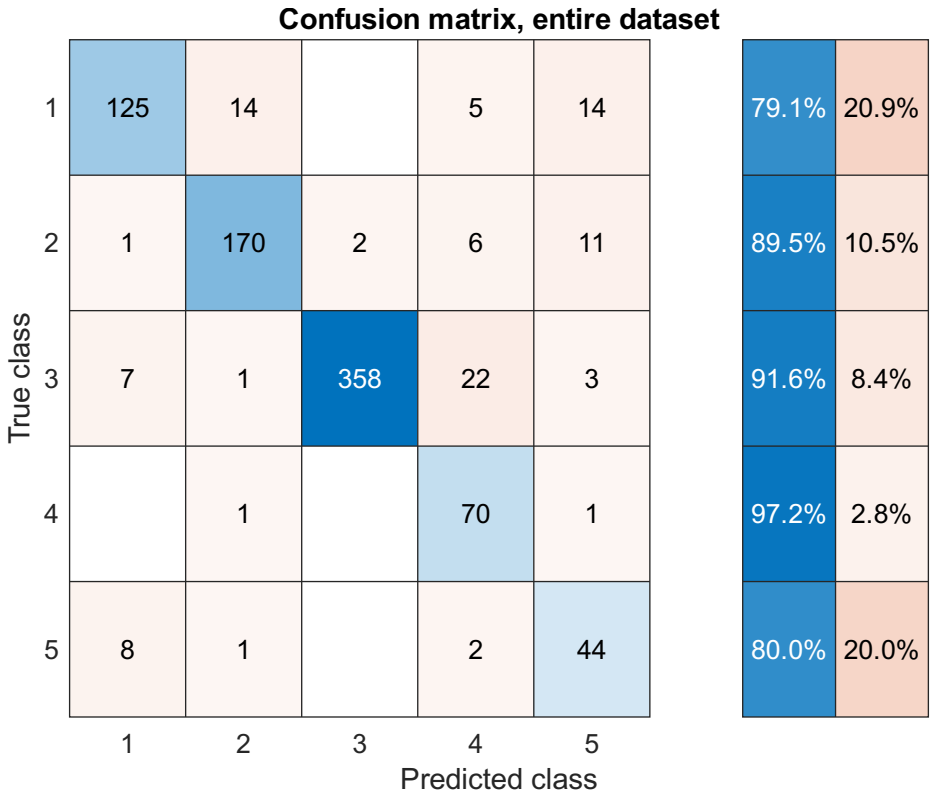


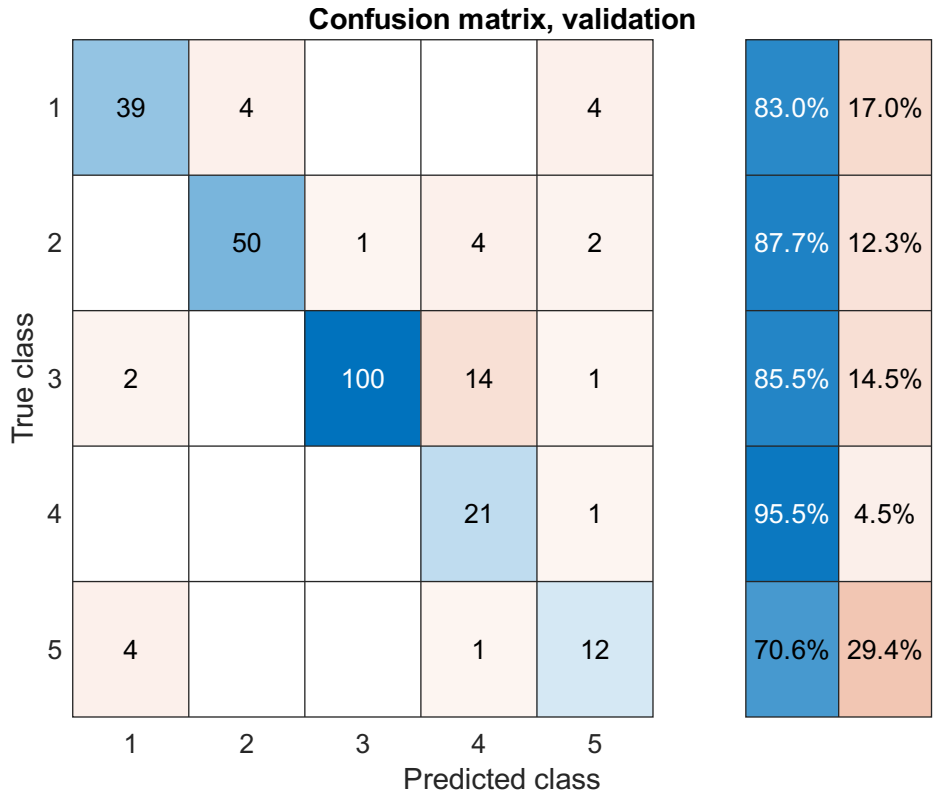
In the right figure we can note that PLS2 tends to advise more complex models than PLS1. This behaviour is confirmed by the left graph: PLS2 goes underfit faster than the other algorithm.



Although PLS1 performs slightly better on average than PLS2, we decided to use the latter one for the following analyses because it is the model with the shortest estimation time.

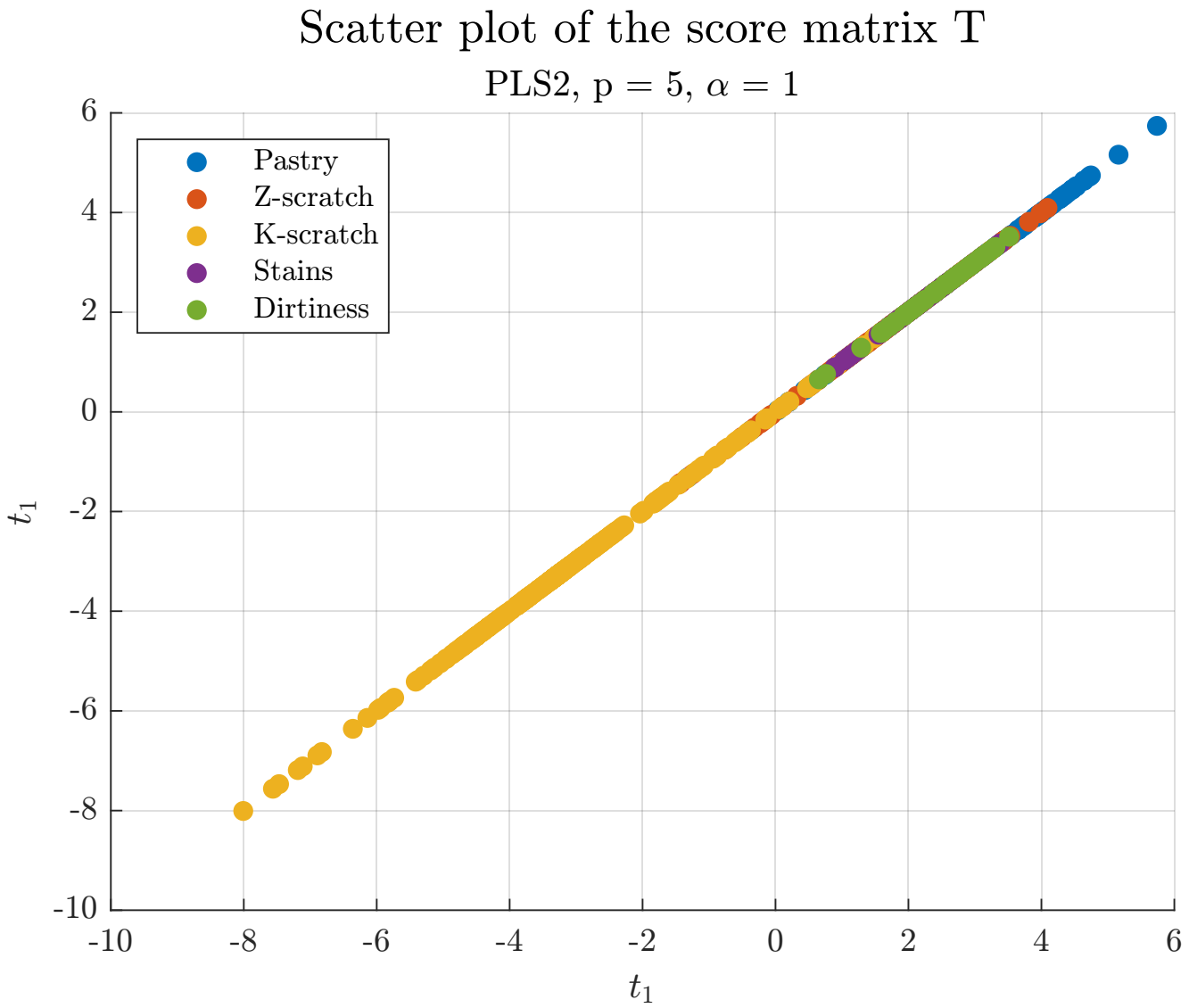
Validation 70/30 of the PLS2 model with $\alpha = 13$

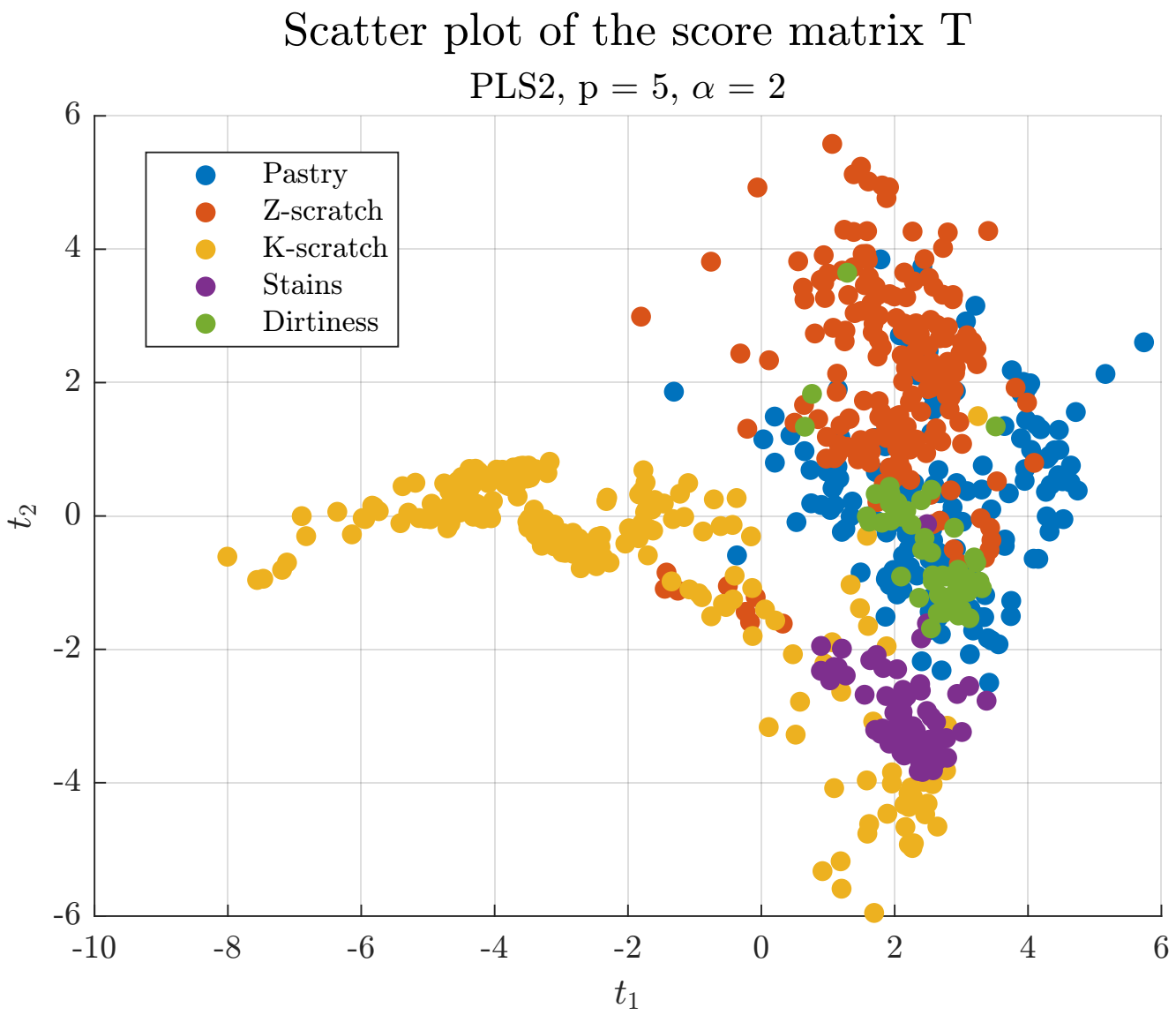


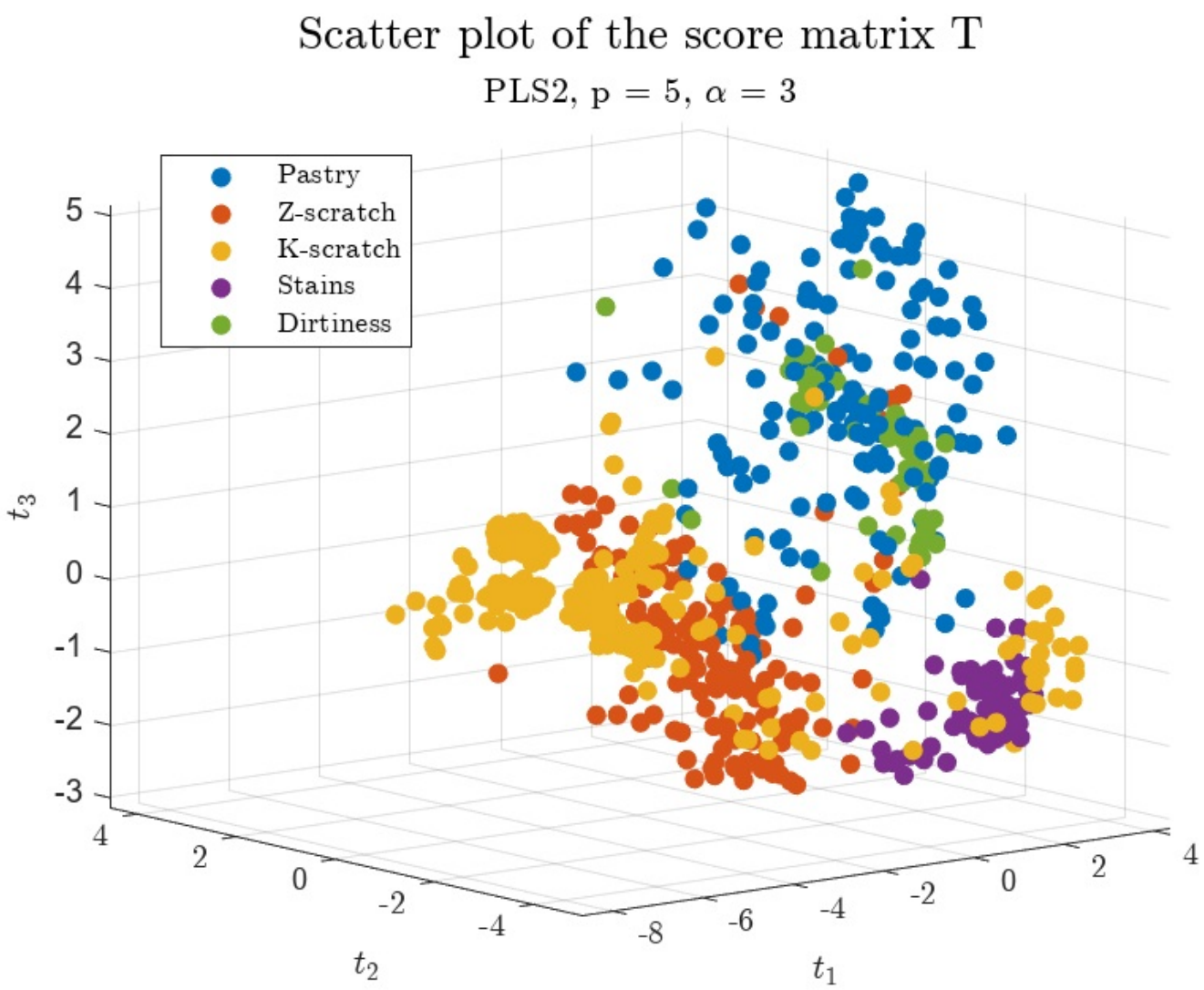


We can see that in validation there is a natural decrease in the predictive capabilities of the model. In particular, the fault classes on which PLS2 struggles the most are classes 1 and 5.

Score matrix T with $\alpha = 1$, 2 and 3




Scatter plot of the score matrix T
PLS2, p = 5, α = 2
Pastry
Z-scratch
K-scratch
Stains
Dirtiness
6
4
2
0
-2
-4
-6
-10
-8
-6
-4
-2
0
2
4
6
t_2
t_1

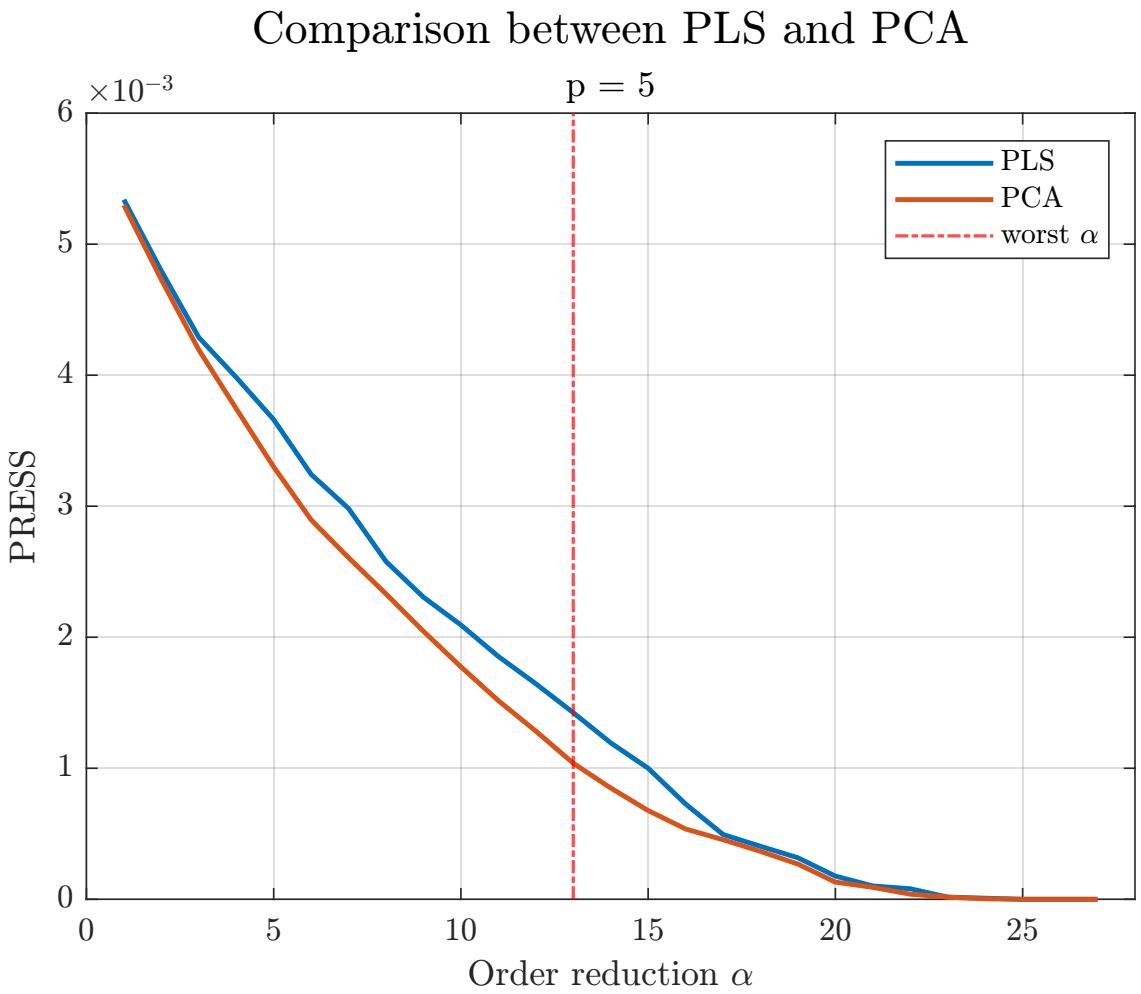

Scatter plot of the score matrix T
PLS2, p = 5, α = 3
Pastry
Z-scratch
K-scratch
Stains
Dirtiness
5
4
3
2
1
0
-1
-2
-3
t_3
4
2
0
-2
-4
t_2
-8
-6
-4
-2
0
2
4
t_1

α	**Pastry**	**Z-scratch**	**K-scratch**	**Stains**	**Dirtiness**	**Average**
1	1.27%	100%	11.76%	100%	100%	42.15%
2	77.85%	8.95%	12.53%	1.39%	100%	28.29%
3	15.82%	10%	14.58%	1.39%	100%	18.13%

As the order reduction α increases, the number of fault classes that are confused with the remaining ones decreases. In particular, we can note that the error rate related to the fault class Stains decreases substantially in the passage from $\alpha = 1$ to $\alpha = 2$; indeed, from $\mathbb{R}^2$ its subspace predominates over the subspaces of the other classes. Instead, the Dirtiness class continues to be hidden from the others for any order reduction.

Comparison between PLS and PCA

	PLS	**PCA**
Type of technique	Supervised	Unsupervised
Goal	Regression and classification	Feature reduction for clustering
Aim of the maximization	Covariance between X and Y	Variance of X
Eigenvectors orthogonality	Not	Yes
Type of decomposition	NIPALS (iterative approach)	SVD



For the central orders PCA is slightly more accurate than PLS in the reconstruction of the matrix X starting from the reduced domain. For the extreme orders, instead, the two techniques are similar.

Bibliography

Book

Fault Detection and Diagnosis in Industrial Systems
L. H. Chiang, E. L. Russel and R. D. Braatz

Dataset

Faulty Steel Plates
https://www.kaggle.com/datasets/uciml/faulty-steel-plates

Repository GitHub

https://github.com/LorenzoF6/PLS_Algorithm_Implementation.git