KU LEUVEN

FACULTY OF SCIENCE

Unsupervised sample selection for multivariate calibration for agrofood applications

Ricardo Andres Castañeda Rueda

Supervisor: Prof. Wouter Saeys
Affiliation *Affiliation K. U. Leuven*

Co-supervisor: *Valeria Fonseca Diaz*
Affiliation *Affiliation K. U. Leuven*

Thesis presented in
fulfillment of the requirements
for the degree of Master of Science
in Statistics and Data Science

Academic year 2021-2022

Acknowledgements

First and foremost, I would like to praise and thank God, who has been my greatest motivation and has provided me with the knowledge, and emotional stability to reach the end of this thesis project. Furthermore, I would like to thank my co-promotor Valeria Fonseca Díaz who guided and supported me with great enthusiasm during the whole development of the project, and whose constant encouragement uplifted me through the complex stages of the process. Apart from that, I take this opportunity to thank my promotor Prof. Wouter Saeys, who has led the complete research project. Finally, I also want to express all my gratitude to my parents Feisal castañeda and Lucy Stella Rueda, and my aunt Claudia Castañeda who were the people who supported me unconditionally during the whole process.

Abstract

Chemometrics and calibration models have become necessary and important tools for industries performing quality control on the chemical concentration of products. Indeed, this technique optimizes the process of predicting the concentration levels of chemical constituents which have been classically calculated with expensive methods. Calibration models are built based on a set of spectral measurements and reference analyses so that predictions on the chemical composition can be obtained in future samples. Because collecting reference values is costly, it is of interest to select a limited number of samples based on spectral measurements to reduce costs and effort significantly. Nonetheless, the quality of the selected samples to collect the reference analyses to build the models is crucial to generate satisfactory calibrations. Until now, this sample selection problem has been addressed with several methods that do not directly ensure obtaining a satisfactory calibration model since they are not designed based on the architecture of the model of interest. Hence, this study proposes a methodology that fits a multivariate calibration model structure aiming to find an optimal subset of samples which would render a calibration model of satisfactory prediction performance.

Contents

Chapter 1

Introduction

Throughout the years, spectroscopy has gained remarkable influence in the agroofood field as well as other types of industries such as the pharmaceutical, petrol, biomedical, environmental, mining, and recycling sectors [1]. This field studies the interaction between matter and light, and its most frequent uses might be related to product or process characterization, monitoring, and quality control inspections [2]. Moreover, spectroscopy is mainly known for this as a low-priced, non-invasive, fast, and non-product-damaging procedure that collects several spectral measurements (the X matrix) to reveal significant information [3, 4]. Nonetheless, the spectroscopy results are processed by multivariate calibration methods to build prediction models. This requires the measurements of reference values or analyses (the Y variable), whose collection is cumbersome and costly, and therefore it is of interest to select an optimal set of samples to acquire such reference analyses [1, 3].

First, multivariate calibration is a tool that uses the information obtained from the spectral measurements to determine or quantify relevant chemical compositions, micro components, and other features. Multivariate calibration transforms spectral measurements into quantitative chemical information by means of a prediction model [5]. $PLSR$ is the number one multivariate calibration methodology to predict chemical concentration levels. Some features that make this method attractive involve that it is an easy-to-implement method [6], produces models with great explanatory predictive power, the models are interpretable, and it is able to cope with multicollinearity problems [7].

Second, as mentioned above, the $PLSR$ models require reference analyses (Y) in order to build prediction models where the quality of the reference values is highly relevant to obtaining accurate predictions. However, given a large number of spectral measurements, it becomes cumbersome and costly to collect the corresponding amount of reference values due to the large expense of laboratory experiments [1]. Therefore, the use of unsupervised sample selection methods takes a crucial role in the process, because these methods select the most representative samples from a population X whose Y response is to be measured, leading to the reduction of reference analyses which decreases the costs and time [1]. The need to find techniques that select optimal sample sizes and sample selection strategies has led to the development of different approaches to solving this issue. The most popular techniques are the Kennard Stone (KS), Duplex (DUP), Puchwein $(PUCH)$, Clustering $(CLUST)$, and D-optimal design

of Experiments $(D - OPT)$ algorithms [1, 2, 8].

However, the state-of-the-art methods from the literature aim to find optimal samples by calculating different distance metrics, which are not directly connected to optimizing the $PLSR$ technique [1]. This results in uncertainty regarding the satisfactory performance of the resulting predictions from a model that was built based on the selected samples [1, 9]. Hence, this study will propose a method founded on the structure of $PLSR$, which identifies an optimal subset of samples. This proposed algorithm is hypothesized to select a subset of samples that would render a calibration model equivalent to the one rendered by the larger population. The proposed criterion detects a suitable subset of samples by comparing the covariance matrix between the full dataset $Cov(X_N)$ and the selected subset of samples $Cov(X_n)$. Indeed, if $Cov(X_N)$ is equivalent to $Cov(X_n)$, a calibration model that produces nearly the same results as a model using all the spectral measurements X_N becomes the hypothesis for unsupervised sample selection. The motivation for this method relies on the fact that $PLSR$, which is the workhorse model architecture that has proven effective for multivariate calibration, uses the X covariance matrix to be constructed. This element is the only part of the $PLSR$ model that can be manipulated when there is no previous information of reference values Y. Besides, it is hypothesized that the proposed method is competitive compared to the state-of-the-art methods, which depend on other similarity measures such as distance and projections without a direct connection to the architecture of the calibration models [1].

Chapter 2

State of the art

2.1 Chemometrics

According to the International Chemometrics Society (ICS), “Chemometrics is the science of relating measurements made on a chemical system or process to the state of the system via application of mathematical or statistical method” [10]. In other words, chemometrics tries to infer properties of interest from measured signals based on statistical techniques, and that is where the name comes from, as “chemo” refers to chemistry and “metrics” to mathematical or statistical measurements [11].

2.2 Spectroscopy

Spectroscopy is a field that focuses on the study, measurement, and interpretation of the electromagnetic spectrum produced by the interaction between matter and light or electromagnetic radiation. This process aims to separate electromagnetic radiation into a spectrum of wavelengths as the process a prism does when it divides light into a rainbow of colors [12]. Basically, the way it works is that emitted light from natural or artificial sources is absorbed by objects that reflect, transmit or scatter this light resulting in a change of color of the original light. The amount of light that could be absorbed depends directly on the structure of the object, namely on its atoms and molecules. Following this, the light that is not absorbed contains the molecular information regarding the object that interacts with the light. Please note that sometimes a part of the light might be transmitted without being altered [13].

Moreover, apart from light other types of electromagnetic radiation are also able to provide information regarding a certain biological system or constituent. In summary, the process involves a source of light or electromagnetic radiation, the object that will serve the purpose of transmitting, scattering, and absorbing light, and a wavelength-dependent detector to track the process [13]. There are different spectroscopy techniques and Near-Infrared spectroscopy (NIRS) is one of the most popular techniques as this method highlights some main features like the fact that it is low-cost, fast to perform, and non-invasive or destructive technique [3].

Following this, the spectrometers are the instruments required to perform this process, and they depict a series of frequencies, which represent the energy at a certain wavelength which may be reflected, transmitted or indeed emitted after absorption of light with a shorter wavelength (fluorescence). The technical name for the series of wavelengths is the spectrum, which can reveal important information concerning the atomic and molecular structure of components which absorb or emit electromagnetic radiation at specific wavelengths [13]. Also, the absorption spectrum is influenced by physico-chemical parameters such as changes in the temperature, or measurement conditions, which is a challenge at the moment to perform this task[11]

Finally, the previously mentioned electromagnetic radiation corresponds to a propagating wave of electrical energy through matter with orthogonal electrical and magnetic components oscillating with the same frequency in the direction of the propagation of the wave [14]. The electromagnetic radiation frequencies are absorbed in the infrared region of the electromagnetic spectrum, corresponding to wavelengths that vary between 800 and 10000 nm and are not part of the observable light, and In this case, the region of interest corresponds to the NIR region which ranges between 780nm and 2500 nm [15].

To summarize, this study utilized spectroscopy to obtain the measurements or data related to a specific constituent and to process that data through multivariate calibration techniques that will result in the desired predictions of the concentration levels related to constituents under evaluation.

2.3 Multivariate calibration

In chemometrics, multivariate calibration is a technique applied to find the function that associates the spectral information to the chemical information of interest [5]. More specifically, this technique is used to build models to predict the concentration levels of a chemical constituent. In general, the models use measurements of a given spectral range as the input or regressor variables aiming at exploiting the information from the wavelength ranges to obtain the relationship with the constituents [16].

Contrary to univariate calibration, multivariate calibration considers multiple spectral variables in order to build a calibration model, as in the univariate case, the model settings include only the most representative variable [17]. Hence, multivariate calibration brings the advantage of making predictions using multiple variables, which contain the important information from the element of study. This implies a significant increase in the accuracy of the predictions [17].

2.3.1 Partial least squares regression (PLSR)

There are different techniques to build multivariate calibration models, and selecting the most adequate methodology is a crucial step. $PLSR$ is the model architecture that has demonstrated to be effective for multivariate calibration [1]. This technique is

able to model the assumed linear relationship from spectral measurements to chemical constituents while coping with the strong multicollinearity that might exists among the wavelengths [7, 18]. $PLSR$ has been widely used in chemometrics applications proving to be successful at finding parsimonious and satisfactory models [1, 19].

$PLSR$ aims to project the predictors X onto a subspace of latent variables (LVs) T, and the most classical algorithm to calculate this subspace is called Non-linear Iterative Partial Least Squares ($NIPALS$) [20]. The goal is to use the correlated predictors to find uncorrelated LVs that could describe most of the covariability between X and Y. To discover the most optimal LVs, the model finds weight vectors W that maximize the covariance between the LVs ($T = XW$) and the response variable Y as shown in equations (2.1 and 2.2) [21].

$$w_i = argmax[Cov(Xw, Y)] \tag{2.1}$$

$$T = XW \tag{2.2}$$

Hence, $PLSR$ selects LVs or scores where the weights W aim to maximize the covariance between spectral measurements and reference analyses $argmax\ Cov(X, Y)$. The use of cross-validation is required to identify the optimal number of LVs to calculate the predictions, which mitigates the overfitting problem [22]. This optimal number of LVs is found where the root mean squared error from the cross-validation method (RMSECV) is minimum, and its R^2 or predictions accuracy is maximum [1]. Following this, the final $PLSR$ predicts the reference analyses for unknown samples, using the resulting model coefficients [22].

Regarding the latent variables, all scores need to be a set of orthogonal variables resulting in $t_i' t_j = 0$, where $(i < j)$, which can be decomposed as shown in equation 2.3. Therefore, the $NIPALS$ algorithm uses the covariance matrix of X as an unsupervised element of the algorithm to find orthogonal LVs that maximize the covariance between X and Y [1]. Consequently, this is the part of the process that is independent of the Y variable, and a motivation to propose a model that works based on the covariance matrix.

$$\begin{gathered} T = t_1, t_2, .., t_n \\ t_i^T t_j = w_i^T X^T X w_j \propto w_i^T cov(X) w_j = 0 \end{gathered} \tag{2.3}$$

Once the latent variables scores are calculated, $PLSR$ culminates with a least-squares regression from these variables to the Y variable as

$$\begin{aligned} y &= Tq + \epsilon \\ \hat{q} &= (T^T T)^{-1} T^T y \\ \hat{y} &= T\hat{q} \end{aligned} \tag{2.4}$$

where q acts as the regression coefficients from the LVs to Y. Note that in the equations above, the X and the Y elements are assumed to be centered [21, 23].

In summary, $PLSR$ is a multivariate calibration tool employed to predict the concentration levels of a chemical constituent, but this method requires spectral measurements X obtained through spectroscopy, and their reference analyses that correspond to the Y variable, where the Y response requires a different and costly measurement process. Therefore, multivariate selection techniques are required to choose an optimal subset of samples Z from the X matrix to measure their respective reference analyses Y and build a $PLSR$ model for future predictions in new samples. Consequently, the need for selection methods arises, and the next chapter explains some of the current state-of-the-art techniques.

2.4 Sample selection methods

There exist different unsupervised sample selection methods, which shall serve as the comparison point in this study. The most predominant and common methods found in the literature of multivariate calibration correspond to KS [24], DUP [25], $(PUCH)$ [26], $CLUST$ [27], and $(D-OPT)$ [1] algorithms.

2.4.1 The Kennard Stone (KS) algorithm

This approach aims to choose a sample that contains most of the variability from the whole population, where the algorithm selects a sample iteratively, with the farthest distance from the previous samples as the target. This method relies on a distance metric, such as the Euclidean or Mahalanobis distance [28].

Typically, the algorithm will start from the most central position from the full dataset to further continue with the iterations as mentioned above. This is a slight adjustment from the original procedure to ensure that extreme points are not selected as the initial point [1]. Hence, with this procedure, each selected point maximizes the distance with respect to the selected points, iterating this process until achieving the expected or required number of samples [29].

2.4.2 The Duplex (DUP) algorithm

This approach is similar to the KS algorithm, but in this case, the Duplex algorithm creates two identical groups to assign the samples with equivalent distance distributions. To start, some authors suggest the samples need to be standardized and orthonormalized (make a set of vectors orthogonal and normal) to calculate a distance metric in the next step, which is the Euclidean or Mahalanobis distance like the previous methods [25]. Next, a couple of samples with the largest distance are selected and allocated to the first class, and the same process is repeated to the second pair of observations with the largest distance, which are assigned to group number two. Further, the algorithm places the next single sample that is most distant to group one in that group, and in the same way, the next single sample, which is farthest from group 2 into this category. The sequence is constant until the desired sample size n is completed in one

of the groups, which is the group of samples to use in the construction of the calibration model [1].

2.4.3 The Puchwein (PUCH) algorithm

The Puchwein technique can be considered as a class of clustering method as it constructs clusters as the samples are being selected in an iterative way. Given a distance metric (Mahalanobis or Euclidean), this algorithm selects the first sample as the point with the largest distance in comparison to the center of the complete sample, and then a limiting distance is defined. Following this, the points within this limiting distance are not considered anymore. The process is repeated and a new point is selected, and its neighbors eliminated as before until discarding all possible points from the total sample. Finally, the selected points are those chosen at the beginning of each iteration [1].

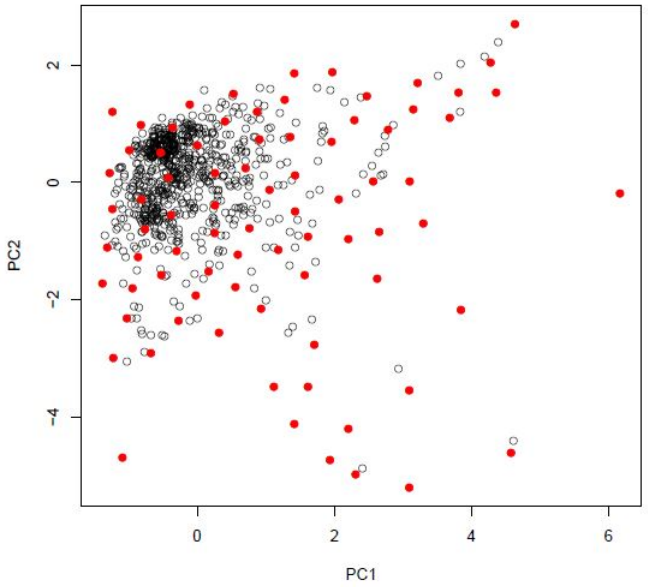

Figure 2.1: Representation of the Puchwein algorithm using an example with principal components. Image taken from Stenvens, A., and Ramirez-Lopez, L. (2013) [30]

2.4.4 Agglomerative Hierarchical Clustering

In the family of hierarchical clustering methods, there are two ways to approach the selection problem. There exist the agglomerative (bottom-up) and divisive (top-down) methodologies. The former starts from a single sample and merges the rest of the samples iteratively into subgroups based on their distance, while the latter decomposes the whole population into subgroups in an iterative fashion until each sample is a separate cluster [31]. This process uses distance measures known as the linkage criteria. Such criteria include the single linkage that combines two clusters based on the minimum distance related to pairs of samples between both groups. The complete

linkage uses the maximum distance between pairs of samples in the clusters. Finally, the average linkage, which calculates the distances between all pairs of objects in two different clusters, and then finds the average of the measured distances [32]. Here also, the most common distance metrics are Euclidean and Mahalanobis [33].

Therefore, the agglomerative clustering algorithm starts with several clusters corresponding to each sample, for which a distance matrix $D = d_{ik}$ between pairs of points of two different groups is calculated, where i is the i^{th} observation from the first group and k is the k^{th} observation from the second group. Then, the distance $d_{a,b}$ between the most similar pair of clusters a, b is calculated to further merge the two groups into one element ab. This process is repeated $N - 1$ times ($N =$ sample size) until all clusters are merged into one final group [34].

Lastly, after reaching one final cluster from the initial number of groups which equals the number of samples n, we proceed to the sample selection step. In this part, we use as a cutoff point the desired number of samples n and select n number of clusters based on the dendrogram, which is the graph that illustrates the clustering process. Afterwards, we choose the sample closest to the center of each selected cluster, and these are the final selected samples. Therefore, each cluster or group will provide the sample that best represents the average of the cluster [1].

2.4.5 D-Optimal Design of Experiments $(D - OPT)$

The D-OPT is an algorithm belonging to the optimal experimental designs where a linear model is assumed for the relationship between X and Y. That model is used to find the coefficients with maximum relative variance $argmax[var(\hat{\beta})]$ which becomes the criterion for sample selection. The models might include first or higher-order polynomials according to the specific case and the relationship between the predictors and the response variable [1]. In the context of the current thesis, $D - OPT$ serves as an unsupervised sample selection method that uses the Federov algorithm, which finds a submatrix, namely Z, that represents the selected rows or samples from the original matrix X that maximizes the variance criterion named above. The Federov algorithm exchanges samples between a starting subset Z and the not included samples X–Z and stops when there is no more improvement in the criterion, or when the indicated number of iterations is achieved [35].

Chapter 3

Objectives

The objectives of this work are:

- Develop an unsupervised sample selection algorithm based on maximizing the similarity between two covariance matrices.
- Propose specific metrics that can quantify the similarity between two covariance matrices.
- Design a sample selection algorithm that renders a selected set of samples by optimizing the proposed similarity metrics.
- Evaluate the performance of the proposed sample selection algorithm in comparison with state-of-the-art methods based on the prediction performance of calibration models using the selected samples.

With these objectives, the research hypothesis to study is:

- The proposed algorithm renders a set of samples to build a calibration model that has similar performance to the model built with the total set of samples.

Chapter 4

Unsupervised sample selection based on domain invariability (DIUSS)

4.1 Theoretical development of criterion

The proposed method aims to find a subset of samples Z that has almost the same covariance as X. The assumption is that if the covariance matrices are similar ($Cov(X) \approx Cov(Z)$), then the results from the $PLSR$ are expected to be alike to the case where the model contains all the spectral measurements and all reference values. The proposed method relies on two principles that help compare the covariance matrices, the Domain-Invariant Partial Least Squares $(di - PLS)$ and the eigendecomposition ED method.

4.2 Domain-Invariant Partial Least Squares $(di - PLS)$

Based on the objective of designing a criterion to compare two covariance matrices, the so-called $di - PLS$ method was taken as reference. Note that this method was not used for model building, it is here presented to illustrate the motivation for a criterion to compare two covariance matrices. The $di - PLS$ algorithm is a method that aims to build a $PLSR$ model with a constraint to align a source and target domain in the latent-variable space. This is done by adding a domain regularizer involving the covariance matrices of the domains in the regression solution [21]. Let $d1$ refer to a source domain and $d2$ refer to a target domain. In $di - PLS$, the part that concerns the covariance matrices is included by constraining the regression problem with the minimization of the difference in variance of the latent variables of the two domains, as shown in equation 4.1. Such difference in variances is handled in the $di - PLS$ method by the ED of the difference between the covariance matrices as shown in 4.2 [36].

$$
\begin{aligned}
w_{dj} &= argmin|X - yw^T|^2 + \lambda|var(t_{d1}) - var(t_{d2})| = \\
w_{dj} &= argmin|X - yw^T|^2 + \lambda|\frac{1}{n_{d1} - 1} w^T X_{d1}^T X_{d1} w - \frac{1}{n_{d2} - 1} w^T X_{d2}^T X_{d2} w|
\end{aligned}
\tag{4.1}
$$

$$
\begin{aligned}
|\frac{1}{n_{d1}-1}w^T X_{d1}^T X_{d1} w - \frac{1}{n_{d2}-1}w^T X_{d2}^T X_{d2} w| &= V^T diag(|s_1.....s_n|)V \\
|\frac{1}{n_{d1}-1}w^T X_{d1}^T X_{d1} w - \frac{1}{n_{d2}-1}w^T X_{d2}^T X_{d2} w| &= V^T S V
\end{aligned}
\tag{4.2}
$$

Therefore, this ED of the difference between the covariance matrices of the two domains was taken for the current thesis as the criterion to design an optimization algorithm for sample selection. For that purpose, $d1$ represents the entire population of spectral measurements X and $d2$ will correspond to the selected samples Z from X.

4.3 The DIUSS_MAX and DIUSS_SUM algortims

The proposed algorithm has been named Domain Invariant Unsupervised Sample Selection $(DIUSS)$, as its core is motivated by the criterion of the $di - PLS$ algorithm. The $DIUSS$ algorithm also uses the ED to decompose the matrix that represents the absolute value of the difference between the covariance of the original data $Cov(X)$ and the covariance of a subset of samples $Cov(Z)$ to use the eigenvalues from this decomposition to create metrics for comparison optimization [37].

4.3.1 Selecting subsamples Z and calculating the covariances

The first step consists in splitting the X matrix into two submatrices. The first matrix Z contains the desired sample size n to include in the calibration model, which are selected randomly from the X matrix, while the matrix $X - Z$ contains those samples that were not included in Z. The second step corresponds to calculating the covariance matrix for the original dataset $Cov(X)$ and the subset of samples $Cov(Z)$.

4.3.2 Covariance matrix difference and ED

The third stage of the process depends on the calculation of the difference between both covariance matrices ($D = Cov(X) - Cov(Z)$) to further apply ED to the absolute value of $|D|$ as shown in equation 4.3. The eigenvalues S from the decomposition are used to define a criterion that quantifies the difference of the covariance matrices numerically.

$$
D = |cov(X) - cov(Z)| = VSV^T \tag{4.3}
$$

4.3.3 Criteria to quantify the difference between two covariance matrices

Following this, to have a metric that measures the difference between the two covariance matrices, two criteria were considered: the first one consists in quantifying the

difference with the maximum eigenvalue of $|D|$ in absolute value ($\max(|S|)$). The second proposed metric quantifies the difference by the sum of all eigenvalues in absolute value ($\sum |S|$).

The eigenvalues S account for the variability in the $PC's$ of the matrix, because ED is the solution to PCA when using the covariance matrices in each case. This means that these two criteria aim to show whether the $COV(Z)$ explains the same variability as the $COV(X)$ [37]. The method under the first metric is labeled as "$DIUSS\_MAX$" and the method using the second metric is labeled as "$DIUSS\_SUM$". The optimization process with $DIUSS\_MAX$ or $DIUSS\_SUM$ aims at minimizing the proposed metric which is taken as an indication of how similar the covariance matrices are.

In this part, please note that $DIUSS\_MAX$ and $DIUSS\_SUM$ are two different selection methods because their way of calculating the differences between the covariance matrices is not the same, meaning that the samples each method selects shall be different. Nonetheless, both criteria are the subject of analysis for the current thesis and their capacity to provide similar covariance matrices is to be analyzed.

4.3.4 Finding an optimal subset of samples

The following step exchanges and keeps rows according to the aforementioned mathematical criteria, and it was based on the Fixed-size LS-SVM algorithm [38]. After the algorithm calculates the first difference D between the covariance matrices and obtains a value either using $DIUSS\_MAX$ or $DIUSS\_SUM$ the result is stored and it is referred to as the $initial\_criterion$. Following this, the algorithm exchanges a row from Z with a row from $X - Z$, and repeats the previous process, or in other words it computes the covariance matrices and their differences to obtain a new value from $DIUSS\_MAX$ or $DIUSS\_SUM$ which is known as the $new\_criterion$.

The decision to keep the exchanged sample is made based on the minimization of the criterion. Therefore, if the exchanged sample renders a lower metric value than the obtained with the previous one, the sample is kept, otherwise the Z matrix with the smallest criterion is kept, and Z is used to exchange a new row with $X - Z$. This algorithm continues for a high number of iterations for convergence. The final result will be the Z matrix that contains an optimal subset of samples which produce the minimum mathematical criterion at the end of the iterations for $DIUSS\_MAX$ or $DIUSS\_SUM$. The samples in Z are then to be used to build the calibration model. The step by step process is described in figure 4.1 as shown below.

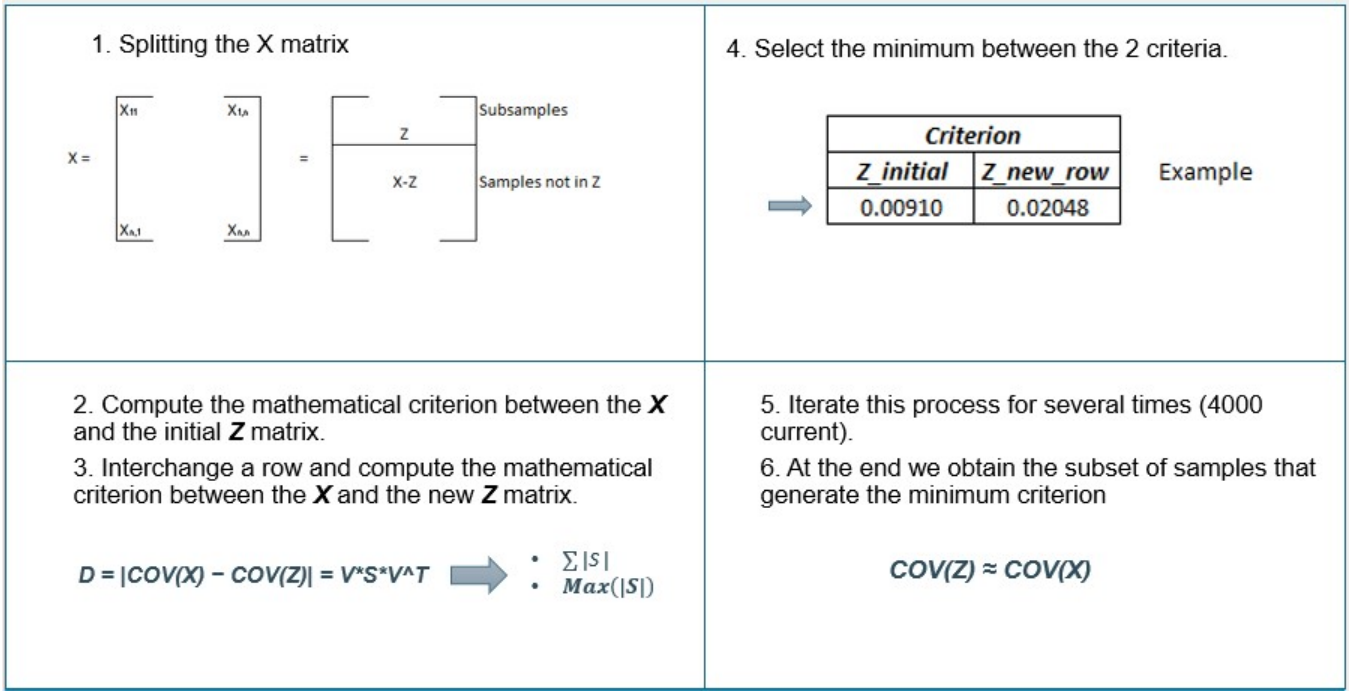

Figure 4.1: Steps to perform the $DIUSS$ algorithm

Chapter 5

Experimental

5.1 Data Description

This study evaluated four datasets, three of them from the agrofood industry and one with pharmaceutical samples. The pharmaceutical data is included to compare if the proposed techniques behave differently concerning non-agrofood data [1]. Hence, a variety of scenario cases were selected to observe the performance of the proposed method and compare how the most common state-of-the-art approaches react to the same datasets.

The four datasets employed to build calibration models and analyze the performance of the selection methods depicted different features and sizes as follows. The pig manure case contains near-infrared spectroscopy (NIR) measurements of pig manure samples with 141 wavelengths. The reference analyses are related to the dry matter concentration levels with spectral range $426nm$–$1686nm$ and resolution of $9nm$. The data were split into calibration and test with 256 and 164 samples, respectively. The milk case presented 316 samples that were measured in the first initial period which served as the calibration samples. There were 79 test samples obtained two weeks afterwards. Moreover, the spectral measurements show 256 wavelengths in the $900nm$–$1700nm$ range with a resolution of $3nm$. Each sample had its corresponding reference analysis of lactose concentration levels. The corn case study is a small data set with 56 and 24 observations for the calibration and test data sets, respectively. This case study contained 700 NIR measurements. Also, four different response variables were included in different $PLSR$ calibration models, whose concentration levels corresponded to moisture, oil, protein, and starch; the spectral measurements are in the $1100 - 2500nm$ with a resolution of 2nm. Finally, the samples regarding the pharmaceutical tablets contain 100 and 55 samples for the calibration and test sets, respectively. This NIR data contained 650 wavelengths and the reference analyses that represented the concentration levels of the assay in the samples. The spectral measurements are in the range $600 - 1900nm$ with a resolution of $2nm$. Except for the milk dataset, the other datasets were split into calibration and test sets randomly with 60% of the samples in the calibration set.

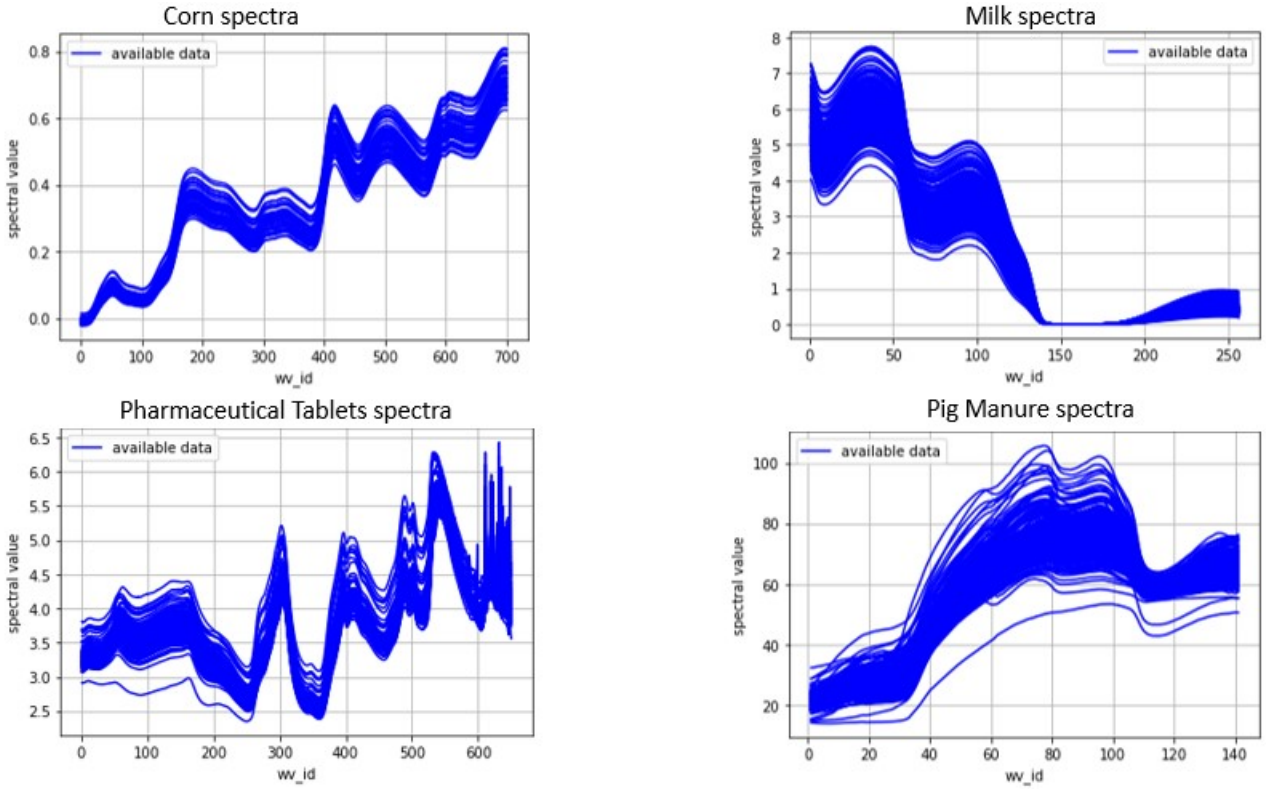

Figure 5.1: Specra for all cases

5.2 Preprocessing

In the current study, all cases were mean-centered as part of the processing stage. In fact, to build the calibration models and compute the covariance matrices to utilize in the proposed methods $DIUSS\_MAX$ and $DIUSS\_SUM$, the data required to be mean-centered. No additional preprocessing techniques were applied motivated on previous studies that have reported that further preprocessing for unsupervised sample selection may lead to poor performance or unwanted effects in the final calibration models [1, 39].

5.3 Methodology

5.3.1 Factors for unsupervised sample selection

To perform this experiment there are some important factors to consider. First, the sample size or the number of samples to select through the sample selection methods to further build the calibration models. Second, the selection methods as they pick those optimal samples to build the calibration models. Third, the input dimensionality that refers to the number of principal components to employ in the sample selection process in the case of selection selection methods [1].

5.3.1.1 Input dimensionality

This study is motivated by results that assessed the performance of different calibration models using the same milk and pig manure datasets that the current experiment also evaluates[1]. Therefore, based on the reported results, the input dimensionality for all case studies and all methods was set to 20 principal components as this number is considered sufficiently large given that generally more than 99% of the spectral variability is contained in the first 10 principal components [1].

5.3.1.2 Sample Size

All scenario cases present different sample sizes based on the suggestion by Fonseca Diaz [1] which recommended a sample size that exceeds the number of LVs (model complexity) by a factor of 12, and consequently, the subset of samples to select through the sample selection methods are the following: In the milk case the recommended complexity level is $LV = 16$, meaning that the optimal sample size is at least 123 $(n = 16 * 12 + 1)$. For the pig manure case, the recommended model complexity is $LV = 11$ meaning that the sample size should be at least 133 $(n = 11 * 12 + 1)$ [1]. The corn case is small dataset that contains only 56 samples in the calibration set, and 24 in the test set. Because of this small population size, the optimal sample size was obtained splitting the calibration set into 66% and 33% as a rule of thumb, meaning that the sample size is $n = 37$. Finally, in the pharmaceutical case a study related to meloxicam assay, reported that the optimal model complexity or LVs to use with $PLSR$ is 4, and hence the calculated sample size to select samples in the current study is 49 $(n = 4 * 12 + 1)$ [40]. In addition, an extra analysis related to the effect of the sample size in model performance was included. This was done with the milk dataset building $PLSR$ models for the prediction of lactose with sample sizes of $60, 90, 120$ and 193.

5.3.1.3 Selection methods

This study includes the most reported unsupervised sample selection methods to be compared with the proposed method ($DIUSS\_MAX/DIUSS\_SUM$). The state-of-the-art methods are Kennard Stone (KS), Duplex (DUP), Puchwein $(PUCH)$, complete linkage hierarchical clustering (CL), and D-optimal designs based on the Federov algorithm $(D-OPT)$ [1]. Please see the section 2.4 to find a wider explanation about each methodology.

5.3.2 Evaluation of the comparison of covariance matrices

The aim of this stage of the experiment is to evaluate whether the selected subset of samples Z generate principal components (PC_S) with almost the same variability as those produced using the X sample set. To analyze the resulting similarity, the covariance matrices of the original set and the subset of samples S_X, S_Z is calculated to further apply $ED = V * \lambda * V^T$ which allows the comparison of both matrices through the eigenvalues S and the eigenvectors V. Following this, the ratio between the two sets of eigenvalues λ_X/λ_Z is computed to compare whether the variability produced

using the selected samples Z is under or over estimated, as the eigenvalues represent the variability that each PC explains [35], and thereby, the ratio shows how much the covariance matrices differ. Besides, the determinant of the eigenvectors illustrates if the subsamples Z contain the same PCA loadings structure as the set of samples in X. Hence the closer the determinant value to 1 and the closer the similarity between the eigenvectors [1].

5.3.3 Model building

All the methods were further evaluated based on the resulting calibration model. $PLSR$ models were built using the selected samples by each method. For each case study, the performance was on the test set $(RMSEP)$ and (R^2P). The curves of these metrics were calculated for each value of LVs from 1 to 20.

5.3.4 Computational tools

This research was executed using Python $version3.8$, and everything was programmed with the in-house codes. The D-OPT algorithm was applied by means of the function $optfederov$ from the R-package $AlgDesign$ [41]. The proposed algorithm DIUSS took from $3min$ to $10min$ to run on a $64 - bit$ Intel(R) Core (TM) $i5 - 8250U8th$ generation with 4 cores and $16GB$ of RAM.

Chapter 6

Results

6.1 Corn

The corn case illustrates the equivalence in the covariance matrices Cov_X and Cov_Z, the equivalence in the variability related to the principal components PC_S, and the predictions performance for different reference analyses Y.

6.1.1 Unsupervised sample selection and covariance equivalence evaluation

The unsupervised sample selection for the corn dataset was done by selecting 37 samples. The evaluation of the equivalence between the covariance matrices is shown in table 6.1 and figure 6.1. In particular, the selected samples by both criteria of $DIUSS$ are illustrated in the tables 9.1 and 9.2. The eigenvectors determinant illustrates whether a group of selected samples produces the same eigenvectors in Z as in X which are indicative of the structure of the spectra. The results illustrated in the table 6.1 indicate that the $D-OPT$, $PUCH$, and the KS algorithms are the methods that produce more similar eigenvectors between the two matrices, as the determinant is closer to 1.

Selection method	determinant eigenvectors
ks˙sample	0.696
duplex˙sample	0.26
puch˙sample	0.69
clus˙sample	0.183
D-OPT	0.811
DIUSS_MAX	0.38
DIUSS_SUM	0.247
all˙samples	1

Table 6.1: Evaluating covariance equivalence between PC_X and PC_Z

In the case of the eigenvalues ratio the results are more equivalent but still have differences as shown in figure 6.1. For instance, the DUP has some significant decreases in eigenvalues ratio from $6\ PC's$ onwards but especially in the $6th$ and the $16th\ PC's$. The $DIUSS$ methods behave well until $10\ PC's$, and after that point, both methods but essentially $DIUS\_SUM$ underestimate the variance. This means that those components with underestimation explain smaller variance of what they should explain compared to S_X. Also, most methods show high peaks in the 2nd component, meaning they overestimate the variance, but both $DIUSS$, and DUP tend to have a closer equivalence regarding overestimation. Overall, all models present notorious differences among them regarding the variability of the eigenvalues, but most show ratios closer to the values produced with the original data X. The cases that differ the most are DUP and $DIUSS\_SUM$ for some specific components.

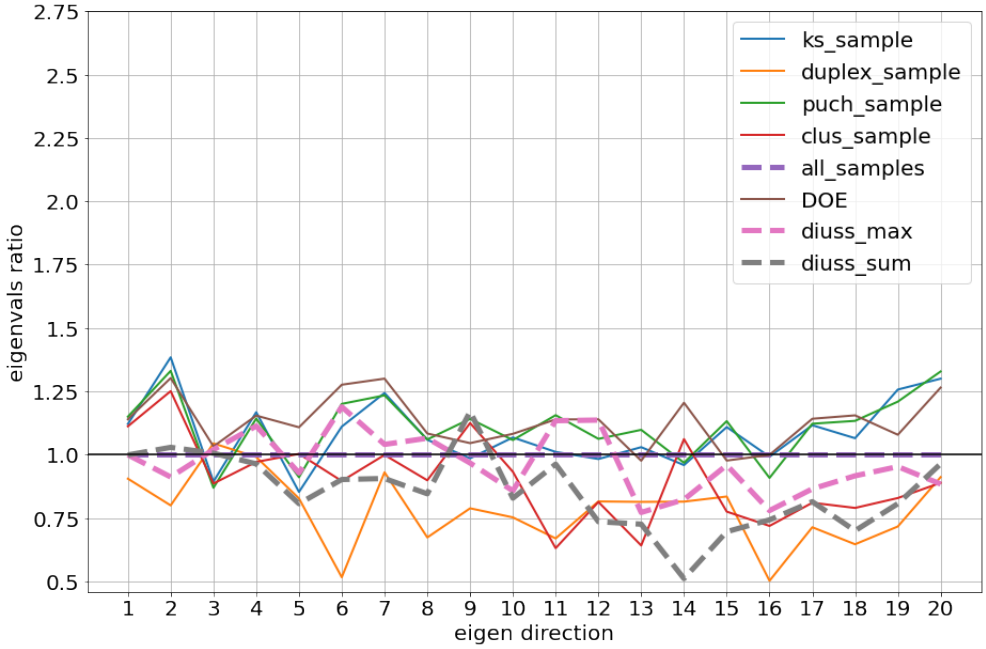

Figure 6.1: Eigenvalues ratio results for all selection methods

6.1.2 PLSR prediction models

6.1.2.1 PLSR predictions for Moisture

In this part, the model is built using the 37 samples that the unsupervised selection methods picked, and the results are in figure 6.2. First, the $RMSEP$ plot depicts that all models have similar performance using around $10LV$ as the cases of KS and $PUCH$ that yield an error of around 0.125, which is almost the same performance that the model with the original data X using $10LV$, and the same as a model that uses the data obtained with $D-OPT$ that had $10LV$, so these methods had the best results.

Furthermore, the rest of the models also illustrated an accurate performance as most of them reached a minimum $RMSEP$ of nearly 0.150. For instance, the $DIUSS\_MAX$ and $DIUSS\_SUM$ methods require the least number of LV (7 and 8) to score errors slightly lower than 0.15, but DUP has virtually identical results. Finally, $CLUST$ behaved very well, because it showed a low error of around 0.14 with only $9LV$.

Regarding the R^2P, the $DIUSS\_SUM$ and $DIUSS\_MAX$ algorithm presented an optimal prediction performance, as they reached a value of almost 0.9 using $7LV$, which was about the same results as the those produced using the original data ($R^2P = 0.94$) with $10LV$. The KS, $DOPT$, and $PUCH$ approaches, showed the best performance as they attained an R^2P close to 0.95 using $10LV$. The $CLUST$ algorithm had a remarkable performance as well, as the R^2P was almost 0.9 like the $DIUSS$ methods, but it needs more LVs (10) to reach this performance.

To summarize, all models performed well for the predictions, the differences among them do not seem significant, and all methods need around $10LV$ to reach their best performance.

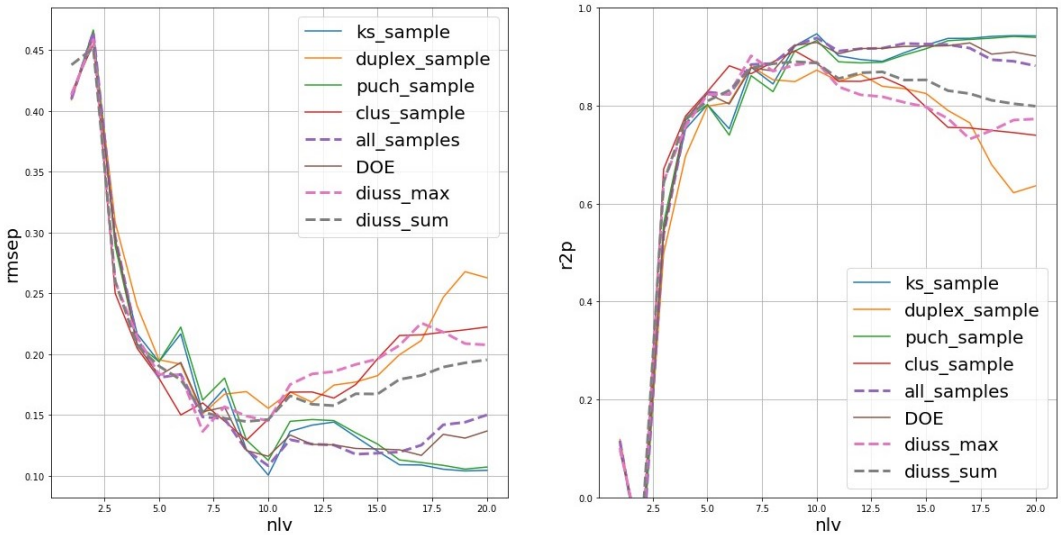

Figure 6.2: Moisture PLSR predictions results: ($RMSEP$) and (R^2P)

6.1.2.2 PLSR predictions for Oil

In the case of the reference analyses related to oil, the prediction model shows some satisfactory results, because the $RMSEP$ yields good outcomes for all models based on each selection method. All methods have a minimum error ranging between 0.06 to 0.09, showing small differences among them. For example, the DUP, $DIUSS\_SUM$, and $DIUSS\_MAX$ obtained errors of nearly 0.085 with $7LV$, $6LV$ and $7LV$, respectively. The rest of the methods achieved smaller errors as they were all under 0.8 using from $7LV$ to $8LV$.

Concerning the R^2P, each algorithm showed a prediction accuracy of more than 70%. The KS and $CLUST$ approaches indicate a peak of 0.8 using $7LV$, which is the same that all data can reach using $8LV$. The $PUCH$ and $D-OPT$ are the methods that follow with results around 0.75 using $8LV$, and finally, both $DIUSS$ and the DUP methods scored 0.7 with $8LV$, but $DIUSS\_SUM$ needs $7LV$.

Therefore, the predictions indicate that all methods performed almost identically, as there were small differences, but none of them made a relevant impact on the results. Thereby, the proposed algorithms $DIUSS\_MAX/DIUSS\_SUM$ depicted a competitive performance, with similar performance as the other methods.

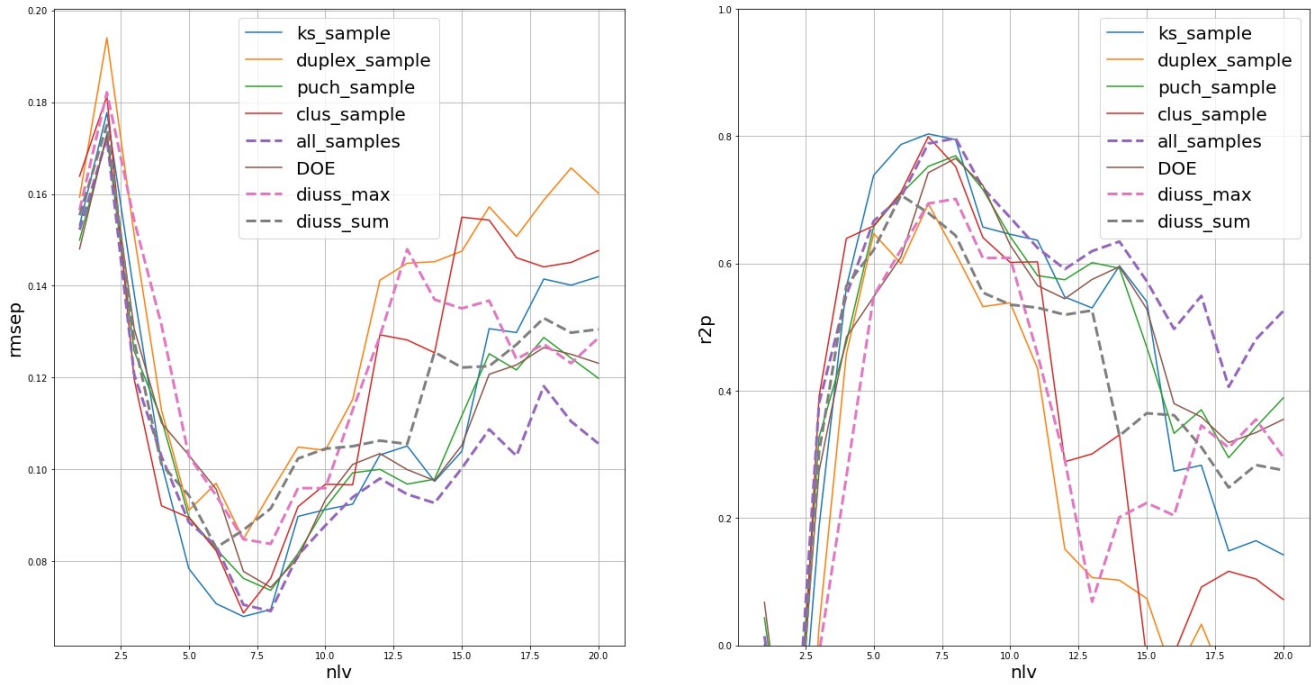

Figure 6.3: Oil PLSR predictions results: ($RMSEP$) and (R^2P)

6.1.2.3 PLSR predictions for Protein

In the case of prediction of protein content, figure 6.4 confirms that the model fitted the data accurately, as the $RMSEP$ for all models reached as low values as the model

with all the samples, and the R^2P obtained notoriously high scores. In summary, the KS algorithm shows the best error reaches values of 0.10 when the model uses $10LV$, which is almost the same results obtained using all samples. Then, the $PUCH$ and $D-OPT$ algorithms reached error values of 0.12 with $10LV$ and $9LV$, respectively. Finally, $DIUSS\_MAX$, $DIUSS\_SUM$, and DUP showed errors of nearly 0.13 using $7LV$.

Regarding the R^2, all models show a similar trend or behaviour. For instance, $D-OPT$, KS, $PUCH$ and $all\_samples$ models, indicate a prediction accuracy of 0.92 approximately, when using $10LV$. Also, $CLUST$ with $9LV$ $DIUSS\_MAX$, $DIUSS\_SUM$, and DUP with $7LV$ illustrate R^2P values of almost 0.83 to 0.87, which indicate quite accurate predictions, and consequently, all methods had a good performances in this case. Moreover, it can be seen that this case seems to be more similar or relate more to the Moisture case than the Oil experiment. In summary, the $DIUSS\_MAX$ and $DIUSS\_SUM$ are the models that reached the best performance with fewer LVs (7), and explain the variability of the response variable competitively. Hence, that is why, in the protein case, these two $DIUSS$ methodologies could be considered adequate options.

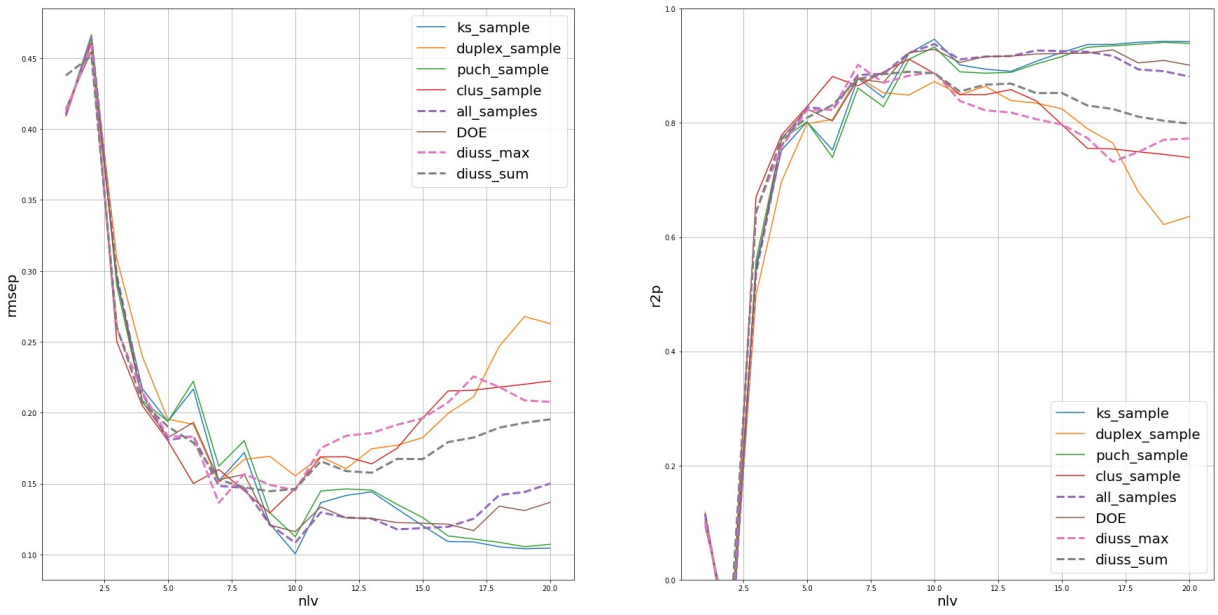

Figure 6.4: Protein PLSR predictions results: ($RMSEP$) and (R^2P)

6.1.2.4 PLSR predictions for Starch

The predictions for starch illustrated in figure 6.5 showed similar results for the models, but they had three different patterns. First, $D-OPT$, $PUCH$, and $allsamples$ have $RMSEP$ that oscillate between 0.15 and 0.20 with high R^2P that are around 0.83 when the model uses $8LV$. Second, the KS, $DIUSS\_MAX$, and $CLUST$ depicted $RMSEP$

of around 0.25 using $8LV$, but $11LV$ in the case of $DIUSS\_MAX$. Also, all these methods had an R^2P of approximately 0.82. Finally, $DIUSS\_SUM$ and DUP presented the lowest performance as their errors ranged between 0.30 and 0.35 with $8LV$. However, their R^2 behaved well because their values were 0.81 for $DIUSS\_MAX$ and 0.78 for DUP.

In summary, all models are capable of making accurate predictions due to the high results obtained for the R^2, but in terms of the error ($RMSEP$), $D-OPT$ and $PUCH$ are the models that seem to be the best fit for the data.

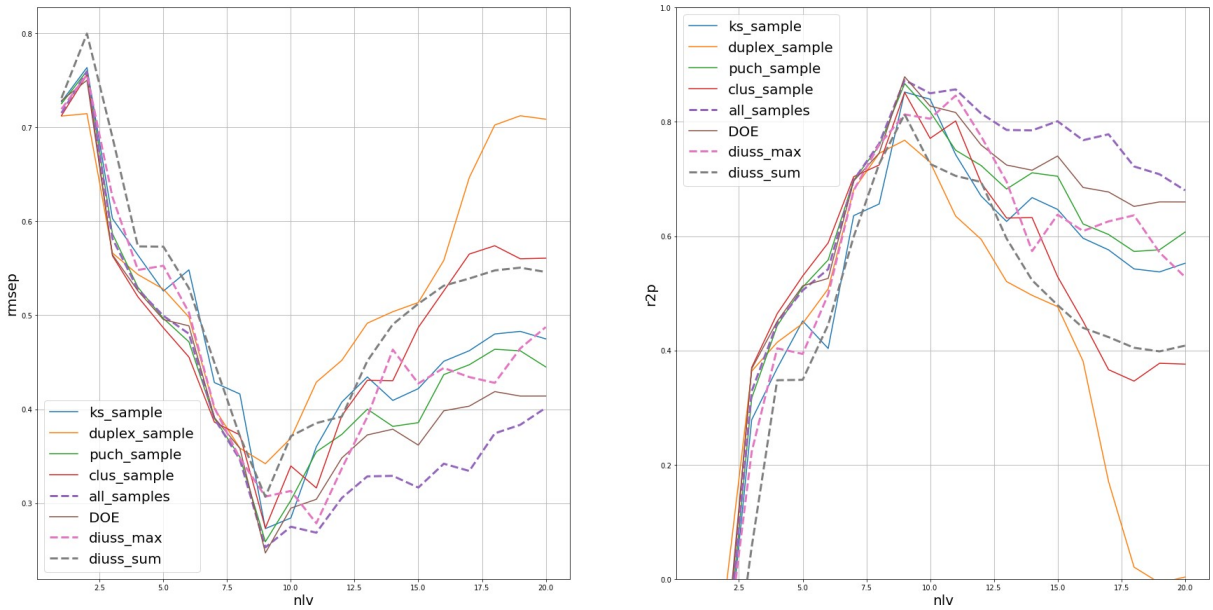

Figure 6.5: Starch PLSR predictions results: ($RMSEP$) and (R^2P)

6.2 Milk

The results of the milk case are presented to show the performance of the different selection methods as well as an analysis of the impact of the sample size on the prediction models. Therefore, the process includes 60, 90, 120, and 193 samples.

In the case of $n = 60$, the results in figure 6.6 reveal that all models have small prediction errors $RMSEP$ with results ranging between 0.075 and 0.093. However, the $D - OPT$, DUP, and $DIUSS\_SUM$ had the lowest errors, whose values were 0.075, 0.078, and 0.08, when the models employed $15LV$, $18LV$ and $13LV$, respectively. The $DIUSS\_SUM$ method also reached an error of 0.078 using $18LV$, but it seems more efficient to increase the error to 0.08, but only with $13LV$. The $DIUSS\_MAX$ and KS, were the next best models with errors of 0.083, and 0.085 using $20LV$, and $CLUST$ and $PUCH$ had errors of around 0.093 with $18LV$ and $20LV$, respectively. In terms of the R^2, the models present the same order or trend, meaning that $D - OPT$, DUP, and $DIUSS\_SUM$ remain the best models with values of around 0.78, 0.77 and 0.76 using $15LV$, $18LV$ and $19LV$, sequentially. The next models are also $DIUSS\_MAX$ and KS scoring 0.7 with $19LV$ in both cases, and the lowest prediction accuracy is presented by $CLUST$ and $PUCH$ with an R^2 of nearly 0.68 when the model contains $18LV$ and $20LV$.

In the case of 90 samples from 6.7, there is an interesting trend that indicates that the models reach their lowest $RMSEP$ using from $18LV$ to $20LV$. Most of the models have almost the same error which is about 0.065, but the cases of $DIUSS\_MAX$ and DUP scored almost 0.09. In terms of the R^2 the results show a small increase as the majority of the models have a prediction accuracy of nearly 0.84, but $DIUSS\_MAX$ and DUP showed R^2P of almost 0.7. Please note, the $RMSEP$ remain constant compared to the model with 60 samples, but the maximum R^2P with $n = 60$ was 0.78 meaning that the accuracy in the predictions increased.

For the third case shown in figure 6.8, the behavior of the models is the same but the $RMSEP$ shows a small decrease. The best models scored errors of around 0.06 to 0.07 using 90 samples, and with 120 the trend is the same, but the errors range between 0.05 and 0.06 which is closer to the results that the full data set X computes. The $DIUSS\_MAX$ and DUP algorithms showed here a higher error but it is also lower than the previous results as it is nearly 0.075, while before it was about 0.09. Likewise, the R^2P have the same trend but the accuracy of the predictions is closer to the one obtained with the original data X for all models, but the $DIUSS\_MAX$ and DUP algorithms are still under 0.8.

Finally, when the $PLSR$ uses 193 samples as shown in figure 6.9 the trends remain constant, but in this case, all models have a $RMSEP$ lower than 0.05 using from $18LV$ to $20LV$, and the accuracy of the predictions is around 0.09.

Hence, the results illustrate that the sample size does not influence the trend of the results directly, but it affects the error and accuracy of the predictions. Indeed, the trends remain constant for all methods, but as the the sample size increases, the accuracy in the predictions increases as well. However, the selection methods showed high performance as they produced low errors and significantly good predictions with 18%,28%,38%, and 61% of the original data $X_n = 316$ as shown in table 6.2.

PLSRP Performance Using Multiple Sample Sizes

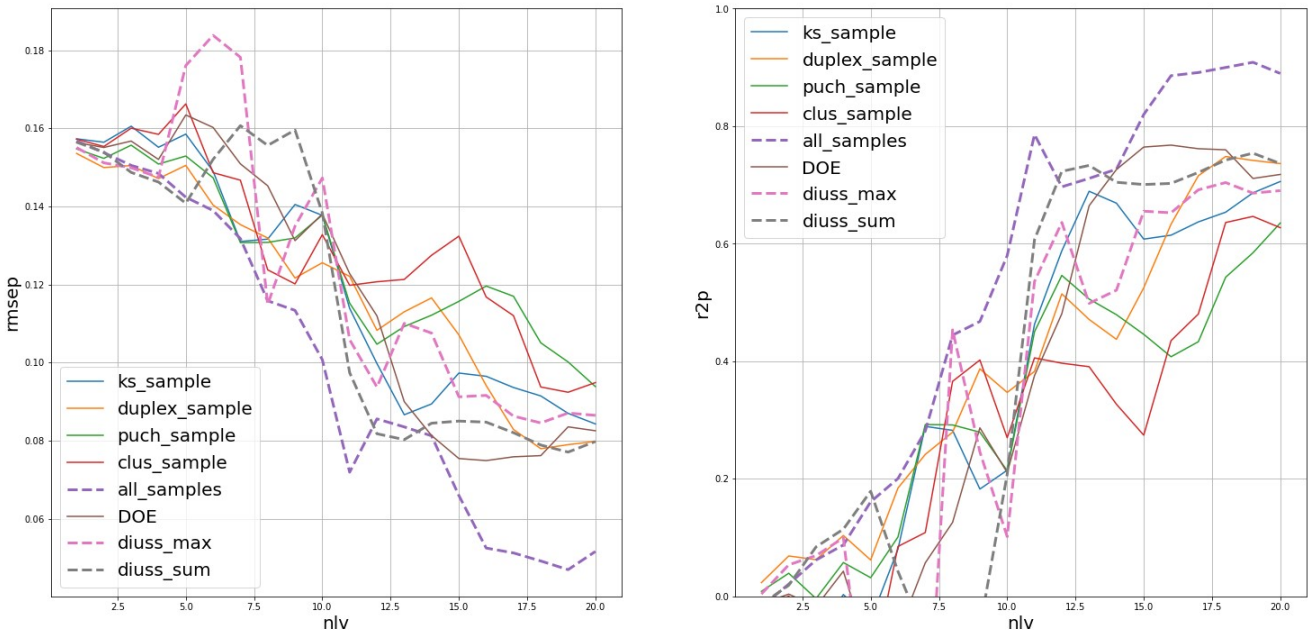

Figure 6.6: Lactose PLSR predictions results $n = 60$: ($RMSEP$) and R-squared for prediction (R^2P)

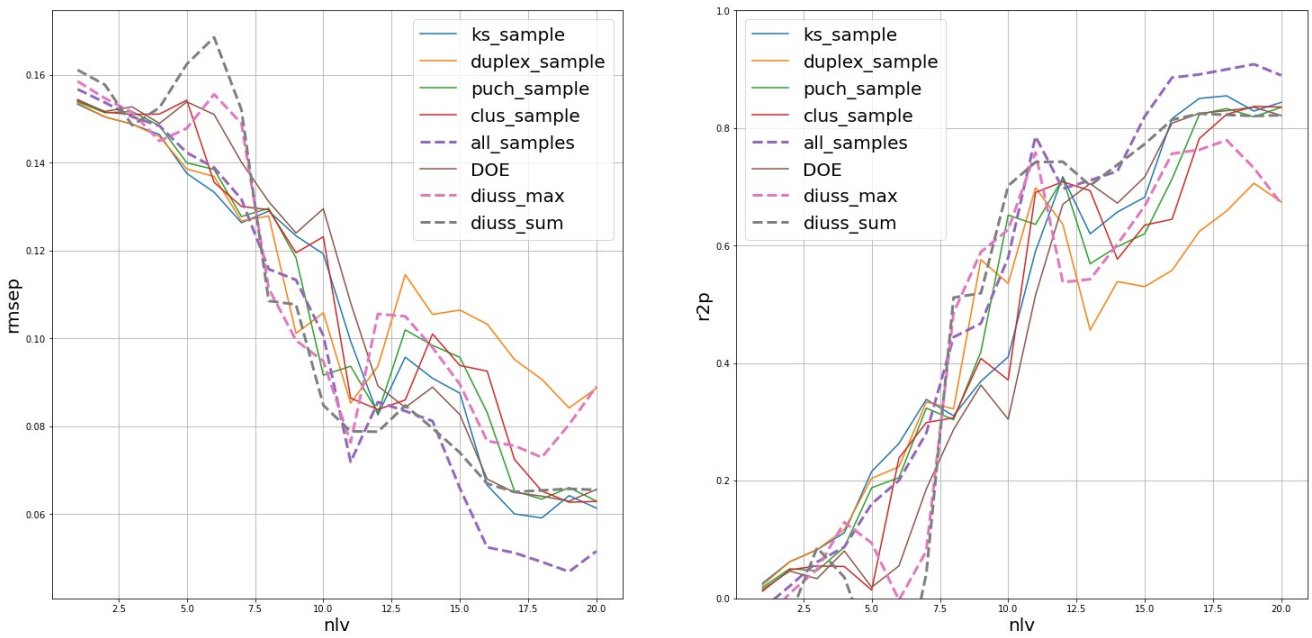

Figure 6.7: Lactose PLSR predictions results $n = 90$: ($RMSEP$) and R-squared for prediction (R^2P)

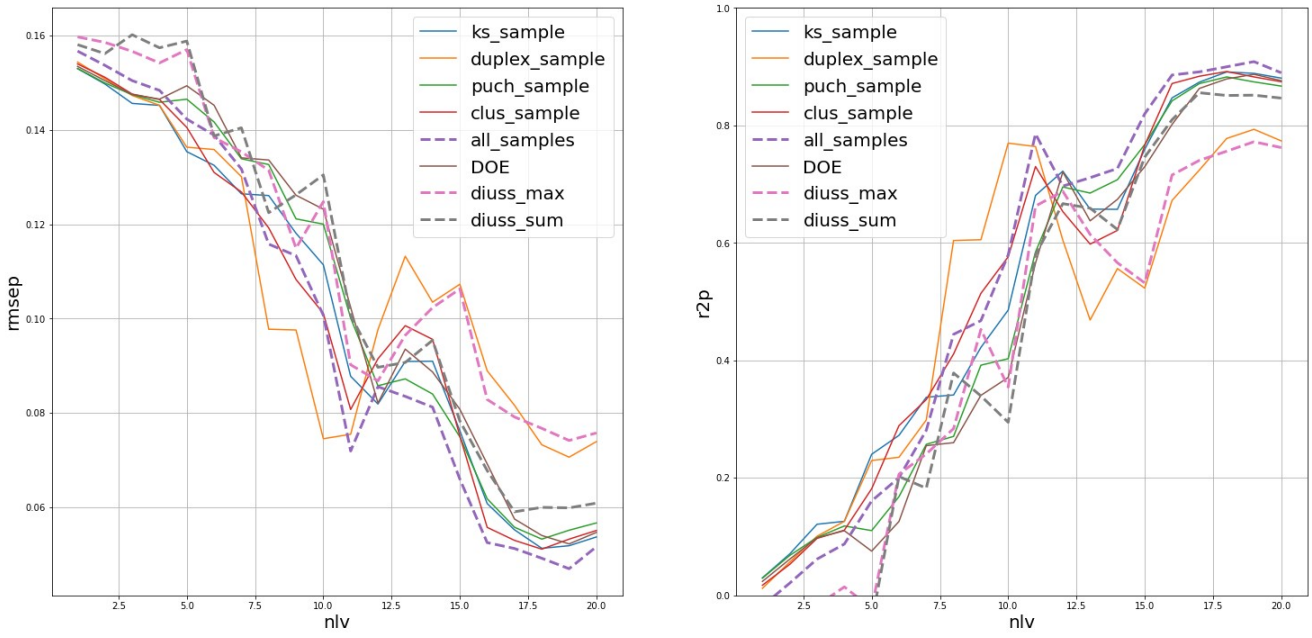

Figure 6.8: Lactose PLSR predictions results $n = 120$: ($RMSEP$) and R-squared for prediction (R^2P)

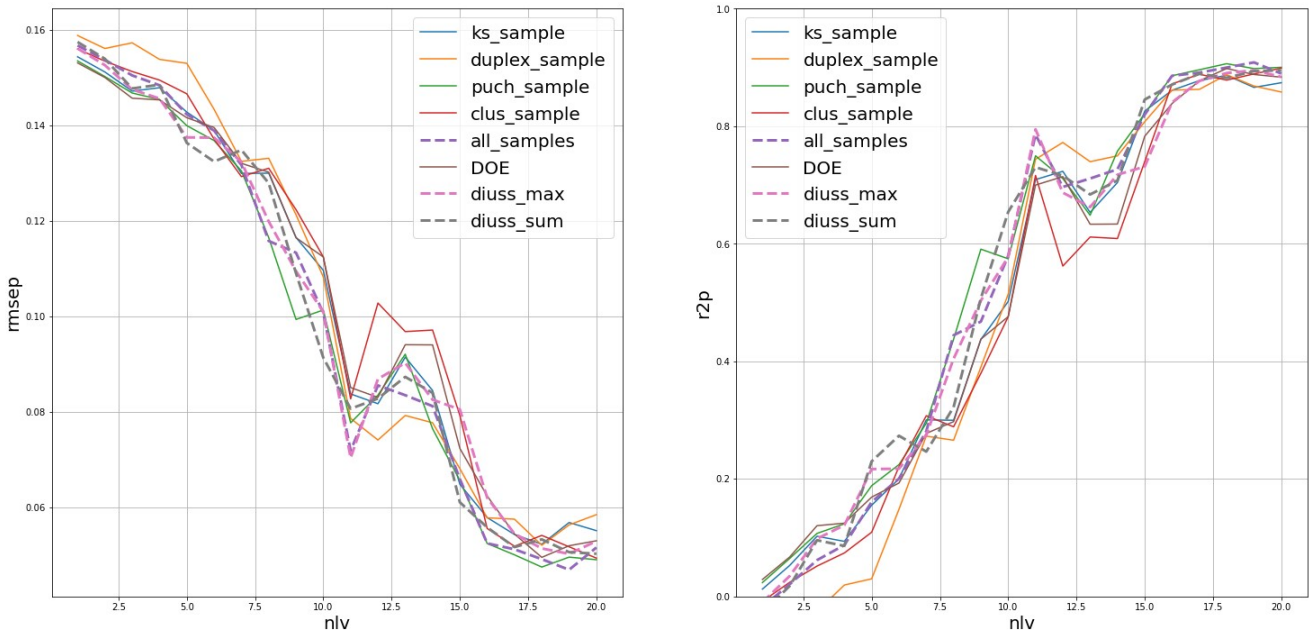

Figure 6.9: Lactose PLSR predictions results $n = 193$: ($RMSEP$) and R-squared for prediction (R^2P)

Sample size	Used Data	Approx best $RMSEP$	Aprox best R^2
$n = 60$	18%	0.08	0.75
$n = 90$	28%	0.07	0.80
$n = 120$	38%	0.05	0.85
$n = 193$	61%	0.04	0.90

Table 6.2: Changes in the $RMSEP$ and R^2P according to the sample size ($X_N = 316$)

6.3 Pig Manure

The pig manure data which contains a total of 420 samples, from which $n\_cal = 133$ and $n\_test = 164$, saving 123 samples from reference analysis collection. Besides the evaluation of model performance, this case was used to analyze the selected samples in the space of the principal components to illustrate if the samples differ between the selection methods in their variability patterns.

The results from the prediction model in figure 6.10 show some interesting aspects as follows: In both cases $RMSEP$ and R^2, the results tend to be constant from $7LV$ to $20LV$, and the predictions seem to be accurate. All methods performed almost identically, and like the original data X, meaning they do not differ notoriously. Regarding the $RMSEP$, the values might reach to 10, which in this particular case suggest a low $RMSEP$. Following this, some researches suggest that an R^2 higher than 0.75 indicates the model predicts well [42], and thereby, the $RMSEP$ in this case is accurate using from $7\ LVs$ onwards as their R^2P is approximately 0.9.

In the part related to the $PC's$ plots, there is an analysis of the main trends regarding all selection methods based on the first two dimensions of the principal components as illustrated in figures (6.11, and 6.12) where the coloured dots represent the selected samples and the black points represent the samples that were not included in the calibration model. Hence, the first pattern in the graphs is that KS, $PUCH$, $CLUST$, and D_OPT selected all extreme values or those located in the limits of the 1st and 2nd principal components (PC) space, which account to more than 90% of the spectral variability. Following this, there is a second pattern, which shows that DUP, $DIUSS\_MAX$, and $DIUSS\_SUM$ selected more samples from the total area of the PC space. Moreover, these three algorithms did not select all extreme values located on all axis of the plot like the other approaches. Hence, some methods follow one behavior, and the other methods have a different trend in common. However, all methods seem to select samples aiming to include elements from all areas of the PC space or plot. Hence, the fact that the models do not target an specific area of plot may lead to the observed similarities in the predictions.

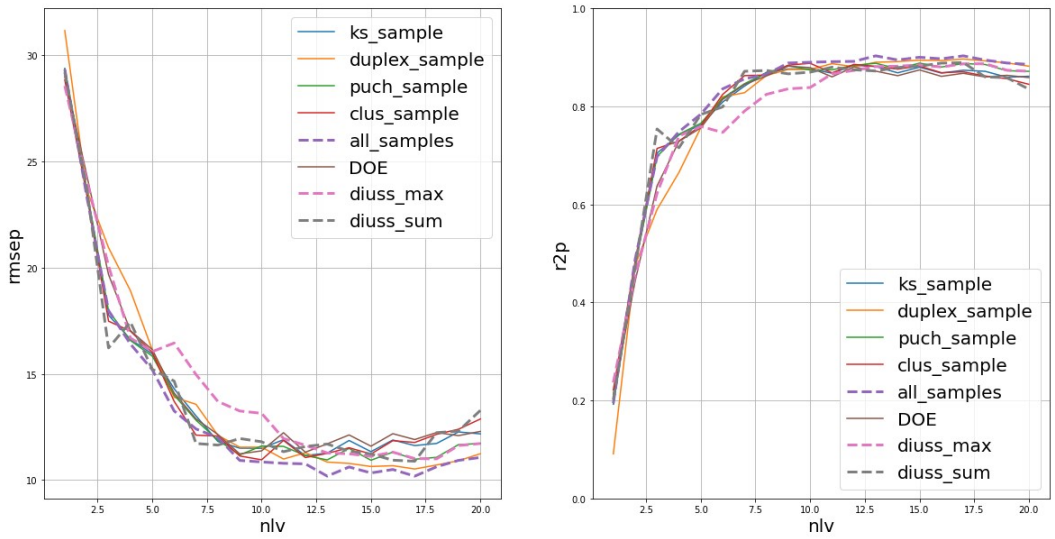

Figure 6.10: Dray Matter PLSR predictions results ($RMSEP$) and R-squared for prediction (R^2P)

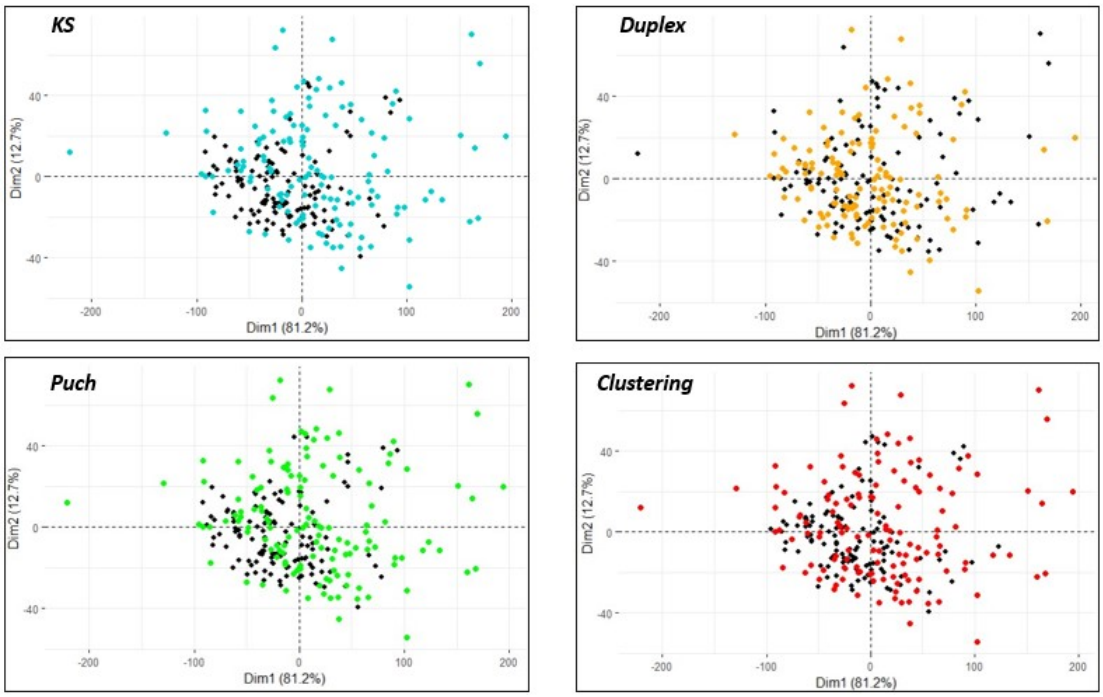

Figure 6.11: Selected samples in the space of PCA - Part 1

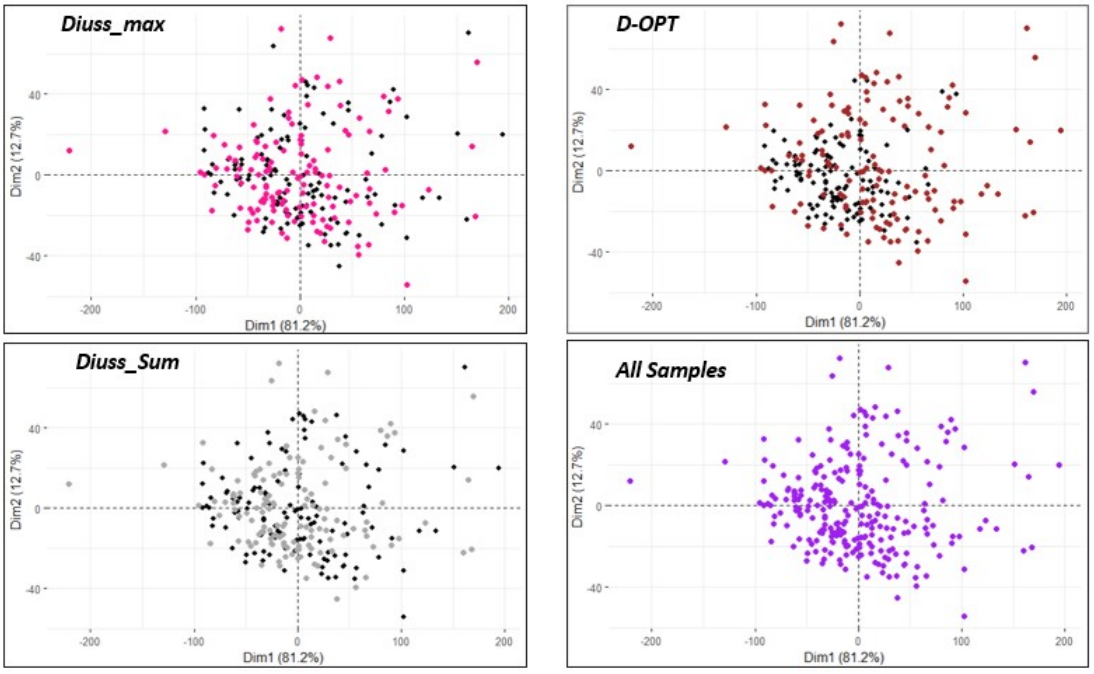

Figure 6.12: Selected samples in the space of PCA - Part 2

6.4 Pharmaceutical Tablets

To conclude, the last results aim to illustrate a case where the data do not correspond to the agroofood sector to check whether the selection methods have completely different behaviors or if they present the same trends than in previous cases.

6.4.1 Prediction Results for Assay reference analyses

The final results depicted by figure 6.13 indicate that all models had a great performance because the $RMSEP$ are not higher than 0.018, and the R^2 results are not lower than 0.97 using the minimum number of LVs. This data did not affect the performance of the models, but instead, it produced the most competitive results among all study cases.

Regarding the proposed methods, $DIUSS\_MAX$ appears to be the model with the worst performance around the optimal number of latent variables shown by the other methods. In the case of $DIUSS\_SUM$, the model showed better performance than $DIUSS\_MAX$, being of similar performance as the other methods, except for DUP. In general, all models had more than satisfactory performance, and $DIUSS\_MAX$ is the second-best option among them. In this particular application DUP showed the best performance being equivalent to the model with the entire population of samples.

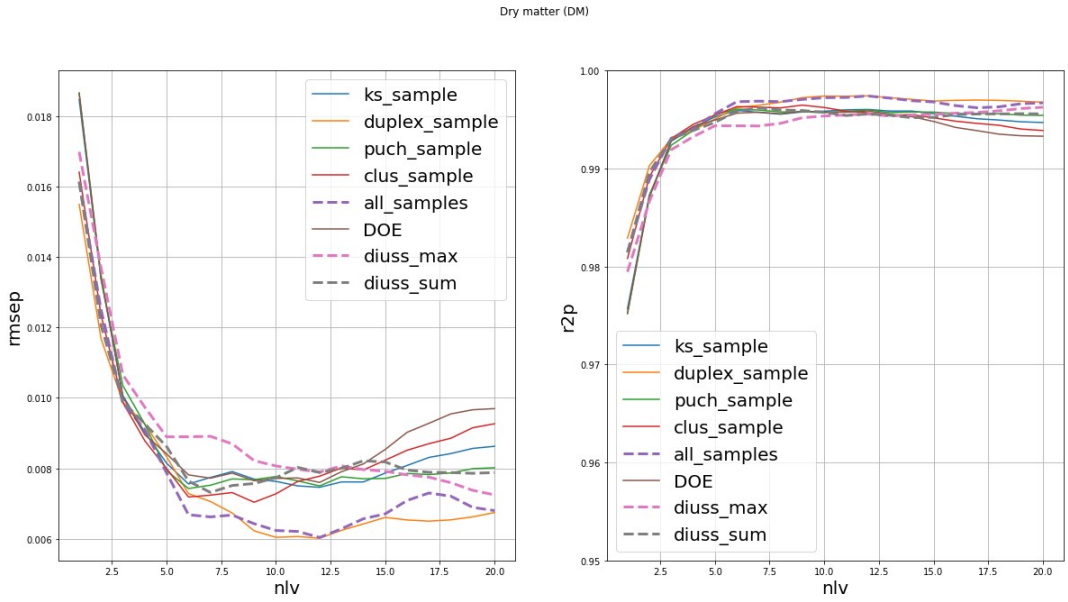

Figure 6.13: Assay PLSR predictions results ($RMSEP$) and R-squared for prediction (R^2P)

6.4.2 Eigenvectors and Eigenvalues equivalence evaluation in the Pharmaceutical case

The pharmaceutical case also served the purpose of evaluating whether the covariance between the original matrix X and the submatrix Z have significant differences for a case that is not related to agrofood applications, as illustrated in table 6.3 and figure 6.14.

With respect to the eigenvectors the methods are divided into two trends because KS, $PUCH$, and $D-OPT$ depicted an equivalence between the eigenvectors determinants higher than 0.3, being $D-OPT$ the method with the highest equivalence, and $PUCH$ the one with the minimum equivalence. Regarding the second trend, both $DIUSS$ methods, the DUP and the $CLUST$ algorithms have eigenvectors equivalence values smaller than 0.01. This indicates that the eigenvectors from the original matrix X and the subset matrix Z were not equivalent.

Furthermore, in the case of the eigenvalues, the plot illustrates the same trend, but in this case, the group that showed small equivalence concerning the eigenvectors shows a higher equivalence for the ratio of the eigenvalues. Also, this plot shows that both $DIUSS$ algorithms and the DUP algorithm present significantly similar behavior. The rest of the methods show great levels of overestimation in the variability, especially after $7LVs$. In other words, the group of methods that had better equivalence related to the eigenvectors does not present a good equivalence in terms of the eigenvalues.

Selection method	determinant eigenvectors
ks˙sample	0.442
duplex˙sample	0.018
puch˙sample	0.354
clus˙sample	0.005
D-OPT	0.628
DIUSS_MAX	0.002
DIUSS_SUM	0.0
all˙samples	1

Table 6.3: Evaluating covariance equivalence between PC_X and PC_Z in the pharmaceutical case

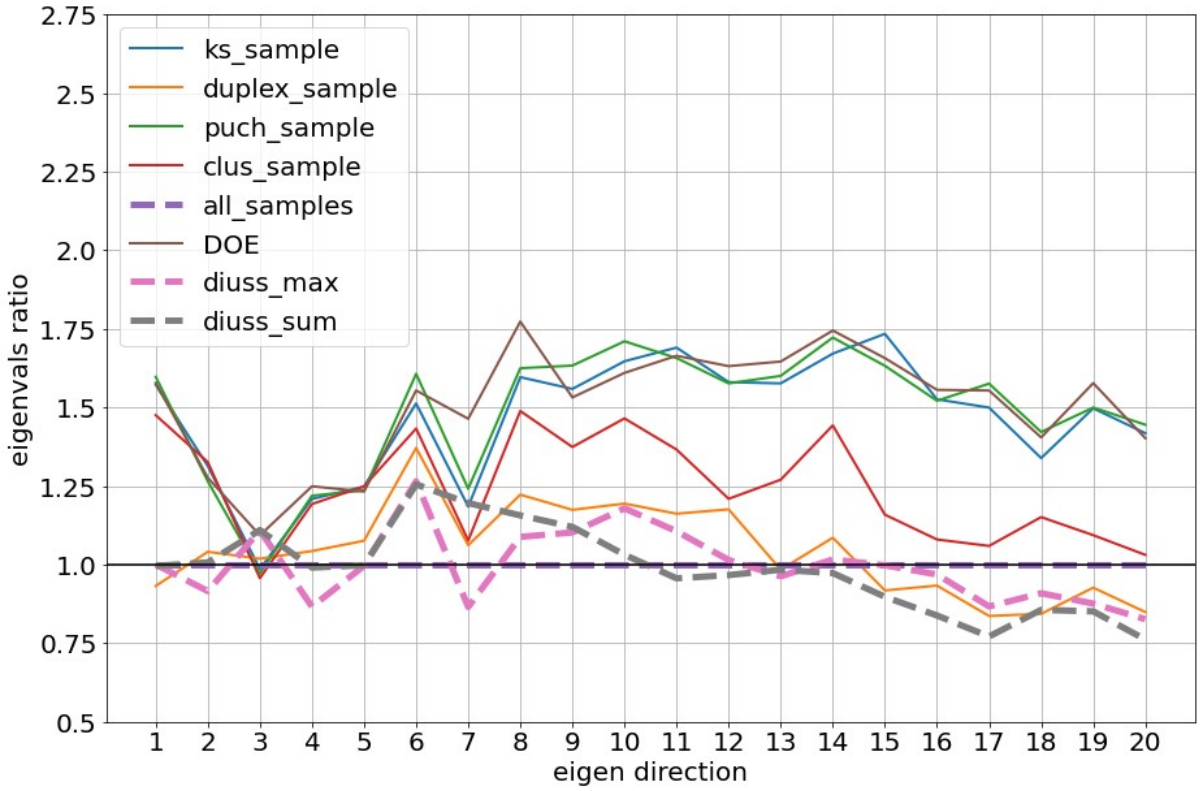

Figure 6.14: Eigenvalues ratio for the pharmaceutical case

Chapter 7

Discussion

7.1 Analysis to the research hypothesis

The initial hypothesis stated that an optimal subset of samples Z able to produce a covariance matrix almost identical to the covariance matrix produced by the complete set of samples X is expected to behave like the original dataset in the calibration model. Hence, the error ($RMSEP$) and the accuracy of the predictions (R^2P) should be as accurate as of the results from the complete X matrix.

Regarding this hypothesis, the results illustrated two remarkable patterns, which are directly related to the chosen sample size n. In the cases of corn and the first two cases of milk, the sample sizes were significantly small, and the results showed pronounced fluctuations. Consequently, the results concerning both $DIUSS\_MAX$ and $DIUSS\_SUM$ as well as the rest of the methods did not behave exactly like the original data. This was an indication that the proposed method is great challenged by small sample sizes. Second, when the sample sizes were established based on the threshold with respect to the model complexity ($n = LVs * 12 + 1$) [1], all results were constant and closer to the results from X [1].

7.2 Eigenvalues and Eigenvectors equivalence

Regarding the structure (Eigenvectors) and the variability (Eigenvalues) of the samples selected using the $DIUSS$ algorithms, there were some clear trends depicted by the results from the corn and the pharmaceutical cases. First and foremost, figures 6.1 and especially 6.14 illustrated that those samples selected by $DIUSS$ aimed to replicate the variability produced by the original data X, which was the objective of the method. In the pharmaceutical case, the ratio of the eigenvalues depicted several similarities, and just some factors had some overestimation or underestimation. In the corn case, the variability was also close to the variability obtained with original data X, but it was slightly better for the $DIUSS\_MAX$ method. Overall, it was clear that the equivalence in variance was more controlled throughout the PC space to be similar to the total population, which is not the case for the other selection methods.

Secondly, the results also depicted that both $DIUSS$ algorithms do not render a similar structure of the covariance matrix, based on the determinant from the eigenvectors

which did not show equivalence as illustrated by tables 6.1 and 6.3. Hence, this aspect can be part of future work, because the $DIUSS$ algorithm results could present improvements if both, the eigenvectors and the eigenvalues are similar to those results produced by the original data X.

7.3 Observed trends

The results showed some similarities between the DUP algorithm and the $DIUSS$ methods, especially with the $DIUSS\_MAX$ in terms of the predictions. For instance, in the case of the eigenvectors and eigenvalues equivalence, the results produced by both $DIUSS$ algorithms and the DUP method are highly similar for the corn and the pharmaceutical cases. Besides, in the Pig Manure case, figure 6.12 illustrates some similarities as these three methods proportionally select extreme and central values, while KS, $PUCH$, $D-OPT$ and $CLUST$ select all the outliers in the dataset. Finally, in the corn case for all reference analyses Y, the predictions and errors from DUP algorithm behaved similarly to those from the $DIUSS$ methods. In the Moisture, Protein, and Oil cases the DUP results related mostly to $DIUSS\_MAX$, and in the Starch case, it behaved almost like the $DIUSS\_SUM$.

Because DUP selects the samples by creating two groups that aim to select samples with equivalent distance distribution (see chapter 2), the process seems to produce similar results to the approach that $DIUSS$ achieves, which also splits the data into two groups, but it aims to reduce the differences between the covariance matrices. In other words, the selection process between these methods seems to be somehow related, and thereby, the variance, the structure and some predictions are alike.

7.4 Performance of the models

First, the results illustrated that the predictions of the models were not more associated to the eigenvectors determinants and the eigenvalues ratio equivalence than to the sample size. For instance, the eigenvectors determinant is very low for some methods like $CLUST$, $DIUSS\_SUM$, $DIUSS\_MAX$ and DUP, and there is low equivalence in the variability for KS, $PUCH$, $D-OPT$, and $CLUST$, but all models showed a good performance for the predictions when the sample size was large enough, and the proposed methods $DIUSS$ appeared to be highly competitive in most cases. Nonetheless, this illustrates some methods aim to have higher equivalence values for the eigenvectors and other models focus on the equivalence in the eigenvalues.

Third, the overall performance for both $DIUSS$ algorithms was similar, but in most cases $DIUSS\_SUM$ presented smaller errors, higher prediction accuracy, and results closer to those presented by the original data X than $DIUSS\_MAX$. To be more precise, in the Corn case for Moisture and Oil, the $DIUSS\_SUM$ method presented slightly better results, than $DIUSS\_MAX$. In the Starch and Protein cases both methods performed alike, and in general the two methods showed good results.

In the Milk case, the $DIUSS\_SUM$ surpassed the $DIUSS\_MAX$ algorithm and showed an accurate performance for some small sample sizes. In that, $DIUSS\_SUM$ showed potential to achieve an optimal performance using only 18% and 28% of the data, which corresponds to sample sizes of 60 and 90, respectively. Furthermore, all models reached accurate results when the sample size was closer to the selected threshold based on the model complexity [1]. Therefore, it seems that the sample size highly influenced the prediction accuracy regardless of the model, but $DIUSS\_SUM$ handled better the selection with small sizes than $DIUSS\_MAX$.

Finally, in the Pig Manure and the pharmaceutical cases, all methods had high performance and no significant difference or aspect was shown in the results. However, in both cases, $DIUSS\_MAX$ showed less stability than $DIUSS\_SUM$ which had constant results throughout all factors.

In general, the $DIUSS$ methods but especially the $DIUSS\_SUM$ showed constant results, as in most of the cases both algorithms produced optimal results. Also, the $DIUSS\_SUM$ was among the best three models, most of the times. Hence, the $DIUSS\_SUM$ depicted high stability while, some of the state-of-the-art methods were influenced by the specific case study. For example, in the case of Milk, the $Clustering$ and the $Puch$ methods depicted a poor performance with small sample sizes.

Chapter 8

Conclusions

- The $DIUSS$ algorithms do select samples that contain a variability closer to the variability produced by the original data X. However, the covariance structure or the eigenvectors is not similar as it does not show any equivalence when comparing the eigenvectors determinant.

- Concerning the presented hypothesis about having samples that replicate almost the same variability from the original data X, the results showed that the sample size is a critical factor to determine this. When the sample size was small the models produced approximately good results compared to those resulting from the original data X, and also optimal or accurate predictions. However, the proposed methods did not replicate the same model predictions and errors that would be obtained with the original data. This could be because the eigenvalues equivalence showed that the variability was not identical, probably due to lack of samples to evaluate, but also because there was a pronounced difference between the eigenvectors from the selected samples Z and the original data X.

  Nonetheless, when the sample size was optimal like in the pig manure and the pharmaceutical cases, the results illustrated that the selected samples nearly replicated the trends from the original data, as this specific scenario showed that the $DIUSS$ algorithms, as well as the rest of the methods, had a greater similarity with the results from X, and especially regarding the R^2P.

- As mentioned in the previous comment, the sample size seems to be the factor with the highest impact in the predictions, and not the selection method. In fact, all methods had optimal predictions and errors when the sample size was determined as suggested by Fonseca Díaz ($n = LVs * 12 + 1$) [1]. In addition, if the sample size is adequate, both $DIUSS$ methods but particularly $DIUSS\_SUM$, are likely to behave closer to the original dataset behavior, if the sample size is small $DIUSS\_SUM$ worked better.

- There was a significant difference between the two proposed algorithms, meaning that one of them has more advantages even when both methods had accurate predictions and low errors. Therefore, the $DIUSS\_SUM$ algorithm presented better results in most cases, especially when the sample size was small, and its

results showed more of a constant behavior than the results from $DIUSS\_MAX$. This can be related to the fact that $DIUSS\_SUM$ considers the variance from all eigenvalues obtained from the covariance matrix of a subset of samples Z under evaluation to determine if a sample is included or excluded, while $DIUSS\_MAX$ selects samples based only on the maximum eigenvalue from the covariance matrix of a subset of samples Z under evaluation.

- The proposed algorithm, particularly $DIUSS\_SUM$, seems to handle the influence of the sample size, as in the milk case when the data had 60 and 90 samples, this method depicted optimal results. Hence, the proposed selection algorithms could be significantly useful when there is a budget constraint or the number of samples to select requires to be considerably small because the rest of the methods, apart from $D - OPT$ appear to vary their behavior notoriously, according to the size of the experiment.
- Both $DIUSS$ methods seem to select more central samples than outliers, which is the opposite than the cases of the KS and $PUCH$ algorithms.

Chapter 9

Appendix

9.1 Python Code

```
1
2  #-------------------------------------------------------#
3  # Propposed method for unsupervised sample selection  #
4  # by: Ricardo Andres Casta eda Rueda                   #
5  # student id: r0731529                                 #
6  #-------------------------------------------------------#
7
8  #====================#
9  # Required packages  #
10 #====================#
11
12 import pandas as pd
13 import numpy as np
14 import random as rnd
15 import itertools
16 import numpy as geek
17 import matplotlib.pyplot as plt
18
19
20 class domain_invariant_unsupervised_sample_selection(object):
21
22 # ---------------------------- Z matrix with subset of samples from X
      --------------------------------
23     def sub_matrix(x_matrix, num_samples):
24             '''
25 This function selects rows or samples from a population matrix (X)
      randomly. The goal is to obtain a submatrix matrix (Z) that contains
      a subset of samples to use in a calibration model. Besides, this
      function yields another matrix (Y) that contains the samples that
      were not included in the submatrix matrix (Z) and a tow vectors (
      Z_samples, Y_samples) that indicate the samples index contained in
       the resulted matrices.
26
27         --- Input ---
28
29         x_matrix: Population matrix that includes all samples to
      evaluate as possible candidates to use in the calibration model.
```

```
        num_samples: The number of samples the user wants to include in
    the calibration model. This variable indicates how many samples are
    to be included in the submatrix (Z).

        --- Output ---
        Z: An array that contains some samples from the population
    matrix (X). The length of this matrix (Z) depends on the variable
    num_samples, and it's expected to be smaller than (X), but it can be
    set depending on the user's needs.
        Y: An array that contains the samples from the population matrix
     (X), that where not included in the submatrix (Z).
        Z_samples: A one-dimensional array that shows the index of the
    samples contained in the submatrix (Z). Please note that this index
    shows the position that a sample has in the population matrix (X),
    and this way, it is possible to identify which samples were included
    in the submatrix (Z).
         Y_samples: A one-dimensional array that shows the index of the
    samples not contained in the submatrix (Z). Then it identifies which
    samples were not included in the submatrix (Z).

        '''

            Z_samples = np.zeros(num_samples).astype(np.int32)
            Z= np.zeros((num_samples, len(x_matrix[1])))
            Y= x_matrix.copy()
            Num_rows = num_samples
            i = 0
            l= len(x_matrix)
            index = np.array(range(0,l))
            while i < Num_rows:
                r = rnd.randint(0,l-1)
                if r not in Z_samples:
                    Z_samples[i] = r
                    row = np.copy(x_matrix[r,:])
                    Z[i] =  row
                    i += 1

            Y_samples = np.delete(index.copy(), Z_samples, axis = 0)
            Y = np.delete(x_matrix.copy(), Z_samples, axis = 0)

            result=[Z,Y,Z_samples, Y_samples]

            print("Z= element [0]", "dimetion:", result[0].shape,"numpy.
    ndarray")
            print("Y= element [1]", "dimetion:", result[1].shape,"numpy.
    ndarray")
            print("Zsamples= element [2]", "length:",len(result[2]),"
    numpy.ndarray")
            print("Ysamples= element [3]", "length:",len(result[3]),"
    numpy.ndarray")
            return result

 #------
    def diuss_max(X,Z,Y,Z_samples,Y_samples,iterations):
                    '''
This function starts finding a criterion based on the differences of the
```

```
    covariance matrices from the original data X and a subset of samples
    from X, namely Z. The algorithm exchanges a row from Z_initial with
    one of the rows that was not included to further obtain a new subset
    of samples Z_new. The criteria are compared and the matrix with the
    minimum criterion is set as the Z_initial from which another row will
    be exchanged and the process is to be repited for a certain number
    of iterations acording to the user selection. At the end the
    algorithm exanges as many rows as the selected number of iterations
    and retains the subset of samples that produced the minimum criterion
    .

        --- Input ---

        x_matrix: Population matrix that includes all samples to
    evaluate as possible candidates to use in the calibration model.

        Z: An array that contains some samples from the population
    matrix (X). The length of this matrix (Z) depends on the variable
    num_samples, and it's expected to be smaller than (X), but it can be
    set depending on the user's needs.

         Y: An array that contains the samples from the population
    matrix (X), that where not included in the submatrix (Z).

        Z_samples: A one-dimensional array that shows the index of the
    samples contained in the submatrix (Z). Please note that this index
    shows the position that a sample has in the population matrix (X),
    and this way, it is possible to identify which samples were included
    in the submatrix (Z).

        Y_samples: A one-dimensional array that shows the index of the
    samples not contained in the submatrix (Z). Then it identifies which
    samples were not included in the submatrix (Z).

       Iterations: The number of times the algorithm is repeated (4000
    recomended).

        --- Output ---
        Z_prev: An array with the final selected subsamples

        Y_prev: An array that contains the samples from the population
    matrix (X), that where not included in the submatrix (Z).

        Z_samples_prev: A one-dimensional array that shows the index of
    the samples contained in the submatrix (Z). Please note that this
    index shows the position that a sample has in the population matrix (
    X), and this way, it is possible to identify which samples were
    included in the submatrix (Z).

         Y_samples_prev: A one-dimensional array that shows the index of
     the samples not contained in the submatrix (Z). Then it identifies
    which samples were not included in the submatrix (Z).

         all_crit: A one-dimensional array that shows the sequence of
    criteria that were calculated

```

```
96          samples_binary: An array with the final selected subsamples but
    in binary form.
97
98          '''
99          #Mathematical criterion
100
101
102         def criterion(x_matrix, z_matrix):
103         '''
104 This function compares the differences between 2 covariance matrices
    using singular value decomposition.
105
106         --- Input ---
107
108         x_matrix: Population matrix that includes all samples to
    evaluate as possible candidates to use in the calibration model.
109
110
111         Z: An array that contains some samples from the population
    matrix (X). The length of this matrix (Z) depends on the variable
    num_samples, and it's expected to be smaller than (X), but it can be
    set depending on the user's needs.
112
113         --- Output ---
114         crt: criterion related to maximum eigenvalue
115
116         '''
117             cov_X = np.cov(x_matrix.T,ddof=0)
118             cov_Z = np.cov(z_matrix.T,ddof=0)
119
120             D = (cov_X-cov_Z)
121             svd_D = np.linalg.svd(D)
122             crt = max(abs(svd_D[1]))
123             return crt
124
125
126
127         # Vector of samples in Binary form
128         def binary (x_matrix, z_samples):
129
130             index = np.array(range(0,len(x_matrix)))
131             samples = z_samples
132             vector = np.zeros(len(index))
133             for i in range(len(samples)):
134                 for j in range(len(index)):
135
136                     if samples[i] == index[j]:
137                         vector[j] = 1
138             return(vector)
139
140         #Matrices
141         Z_new = np.zeros(( len(Z), len(Z[1])))
142         Y_new = np.zeros(( len(Y), len(Y[1])))
143         Z_prev = Z.copy()
144         Y_prev = Y.copy()
145
```

```
        #List of samples contained in the matrices
        Z_Samples_new = np.zeros((0,len(Z))).astype(np.int32) #**
        Y_Samples_new = np.zeros((0,len(Y))).astype(np.int32) #**
        Z_Samples_prev = Z_samples.copy() #**
        Y_Samples_prev = Y_samples.copy() #**

        # Number of samples or rows in the matrices
        rows_X = len(X)
        rows_Y = len(Y)
        rows_Z = len(Z)
        # Cov matrices and mathematical criterion
        crit = criterion(x_matrix=X, z_matrix=Z)
        all_crit = np.zeros(iterations+1)
        all_crit[0] = crit

        for i in range(iterations):
            # Random rows
            r1 = rnd.randint(0,rows_Z-1)
            r2 = rnd.randint(0,rows_Y-1)
            # Swaping variables in Z
            Z_new = np.delete(Z_prev.copy(), r1, axis = 0)
            Z_new = np.insert(Z_new, r1, Y_prev[r2], axis =0)

            ## Swaping variables in Y
            Y_new = np.delete(Y_prev.copy(), r2, axis = 0)
            Y_new = np.insert(Y_new, r2, Z_prev[r1], axis =0)

            #Step 2: Criterion
            New_crit = criterion(x_matrix=X, z_matrix=Z_new)

            #Step 3: Selecting Best matrix
            if New_crit < crit:
                crit = New_crit
                Z_prev = Z_new.copy()
                Y_prev = Y_new.copy()

                # List of samples in the matrices
                Z_Samples_new = np.delete(Z_Samples_prev.copy(), r1,
    axis = 0)
                Z_Samples_new = np.insert(Z_Samples_new, r1,
    Y_Samples_prev[r2], axis =0)
                Y_Samples_new = np.delete(Y_Samples_prev.copy(), r2,
    axis = 0)
                Y_Samples_new = np.insert(Y_Samples_new, r2,
    Z_Samples_prev[r1], axis =0)
                Z_Samples_prev = Z_Samples_new.copy()
                Y_Samples_prev = Y_Samples_new.copy()

                # Vector containing all calculated criterions
            all_crit[i+1] = New_crit

        samples_binary = binary(x_matrix=X, z_samples=Z_Samples_prev)
        samples_binary = samples_binary.astype(np.int32)
        result = [Z_prev, Z_Samples_prev, Y_prev, Y_Samples_prev,
    all_crit, samples_binary]
        return result
```

```
197 #--------------
198     def diuss_sum(X,Z,Y,Z_samples,Y_samples,iterations):
199                 '''
200 This function starts finding a criterion based on the differences of the
    covariance matrices from the original data X and a subset of samples
    from X, namely Z. The algorithm exchanges a row from Z_initial with
    one of the rows that was not included to further obtain a new subset
    of samples Z_new. The criteria are compared and the matrix with the
    minimum criterion is set as the Z_initial from which another row will
    be exchanged and the process is to be repited for a certain number
    of iterations acording to the user selection. At the end the
    algorithm exanges as many rows as the selected number of iterations
    and retains the subset of samples that produced the minimum criterion
    .
201
202         --- Input ---
203
204         x_matrix: Population matrix that includes all samples to
    evaluate as possible candidates to use in the calibration model.
205
206         Z: An array that contains some samples from the population
    matrix (X). The length of this matrix (Z) depends on the variable
    num_samples, and it's expected to be smaller than (X), but it can be
    set depending on the user's needs.
207
208          Y: An array that contains the samples from the population
    matrix (X), that where not included in the submatrix (Z).
209
210         Z_samples: A one-dimensional array that shows the index of the
    samples contained in the submatrix (Z). Please note that this index
    shows the position that a sample has in the population matrix (X),
    and this way, it is possible to identify which samples were included
    in the submatrix (Z).
211
212         Y_samples: A one-dimensional array that shows the index of the
    samples not contained in the submatrix (Z). Then it identifies which
    samples were not included in the submatrix (Z).
213
214        Iterations: The number of times the algorithm is repeated (4000
    recomended).
215
216         --- Output ---
217         Z_prev: An array with the final selected subsamples
218
219         Y_prev: An array that contains the samples from the population
    matrix (X), that where not included in the submatrix (Z).
220
221         Z_samples_prev: A one-dimensional array that shows the index of
    the samples contained in the submatrix (Z). Please note that this
    index shows the position that a sample has in the population matrix (
    X), and this way, it is possible to identify which samples were
    included in the submatrix (Z).
222
223          Y_samples_prev: A one-dimensional array that shows the index of
     the samples not contained in the submatrix (Z). Then it identifies
    which samples were not included in the submatrix (Z).
```

```
224
225         all_crit: A one-dimensional array that shows the sequence of
    criteria that were calculated
226
227         samples_binary: An array with the final selected subsamples but
     in binary form.
228
229         '''
230         #Mathematical criterion
231         def criterion(x_matrix, z_matrix):
232 '''
233 This function compares the differences between 2 covariance matrices
    using singular value decomposition.
234
235         --- Input ---
236
237         x_matrix: Population matrix that includes all samples to
    evaluate as possible candidates to use in the calibration model.
238
239
240         Z: An array that contains some samples from the population
    matrix (X). The length of this matrix (Z) depends on the variable
    num_samples, and it's expected to be smaller than (X), but it can be
    set depending on the user's needs.
241
242         --- Output ---
243         crt: criterion related to the sum of eigenvalues
244
245         '''
246             cov_X = np.cov(x_matrix.T,ddof=0)
247             cov_Z = np.cov(z_matrix.T,ddof=0)
248
249             D = (cov_X-cov_Z)
250             svd_D = np.linalg.svd(D)
251             crt2 = sum(abs(svd_D[1]))
252             return crt2
253         # Vector of samples in Binary form
254         def binary (x_matrix, z_samples):
255
256             index = np.array(range(0,len(x_matrix)))
257             samples = z_samples
258             vector = np.zeros(len(index))
259             for i in range(len(samples)):
260                 for j in range(len(index)):
261
262                     if samples[i] == index[j]:
263                         vector[j] = 1
264             return(vector)
265
266         #Matrices
267         Z_new = np.zeros(( len(Z), len(Z[1])))
268         Y_new = np.zeros(( len(Y), len(Y[1])))
269         Z_prev = Z.copy()
270         Y_prev = Y.copy()
271
272         #List of samples contained in the matrices
```

```
        Z_Samples_new = np.zeros((0,len(Z))).astype(np.int32) #**
        Y_Samples_new = np.zeros((0,len(Y))).astype(np.int32) #**
        Z_Samples_prev = Z_samples.copy() #**
        Y_Samples_prev = Y_samples.copy() #**

        # Number of samples or rows in the matrices
        rows_X = len(X)
        rows_Y = len(Y)
        rows_Z = len(Z)
        # Cov matrices and mathematical criterion
        crit = criterion(x_matrix=X, z_matrix=Z)
        all_crit = np.zeros(iterations+1)
        all_crit[0] = crit

        for i in range(iterations):
            # Random rows
            r1 = rnd.randint(0,rows_Z-1)
            r2 = rnd.randint(0,rows_Y-1)
            # Swaping variables in Z
            Z_new = np.delete(Z_prev.copy(), r1, axis = 0)
            Z_new = np.insert(Z_new, r1, Y_prev[r2], axis =0)

            ## Swaping variables in Y
            Y_new = np.delete(Y_prev.copy(), r2, axis = 0)
            Y_new = np.insert(Y_new, r2, Z_prev[r1], axis =0)

            #Step 2: Criterion
            New_crit = criterion(x_matrix=X, z_matrix=Z_new)

            #Step 3: Selecting Best matrix
            if New_crit < crit:
                crit = New_crit
                Z_prev = Z_new.copy()
                Y_prev = Y_new.copy()

                # List of samples in the matrices
                Z_Samples_new = np.delete(Z_Samples_prev.copy(), r1,
  axis = 0)
                Z_Samples_new = np.insert(Z_Samples_new, r1,
  Y_Samples_prev[r2], axis =0)
                Y_Samples_new = np.delete(Y_Samples_prev.copy(), r2,
  axis = 0)
                Y_Samples_new = np.insert(Y_Samples_new, r2,
  Z_Samples_prev[r1], axis =0)
                Z_Samples_prev = Z_Samples_new.copy()
                Y_Samples_prev = Y_Samples_new.copy()

                # Vector containing all calculated criterions
            all_crit[i+1] = New_crit

        samples_binary = binary(x_matrix=X, z_samples=Z_Samples_prev)
        samples_binary = samples_binary.astype(np.int32)
        result = [Z_prev, Z_Samples_prev, Y_prev, Y_Samples_prev,
  all_crit, samples_binary]
        return result
#--------------------------------- Count of ones for vectors of
```

```
   selected samples --------------
    def count_of_ones(array):
                            '''
Counts the number of ones from a binary array

        --- Input ---

        array: A one-dimensional array with the selected samples in
   binary form

        --- Output ---
        a: count related to the number of 1 conteined in the binary
   array.
        '''
        a=0
        for i in range(len(array)):
            if  array[i] == 1:
                a = a +1
        return(a)
#-----------------
# Function to plot a vector that contain the recorde result from the
   criterion obtained at each iteration
    def crit_behavior(all_crit):

                                    '''
The function plots the behavior that all criteria showed

        --- Input ---

        all_crit: A one-dimensional array that shows the sequence of
   criteria that were calculated

        --- Output ---
        plt.show(): plot related to the binary array.
        '''
        a = all_crit
        index=np.array(range(0,len(a)))
        plt.plot(index,a)
        plt.ylabel('All sorted criteria')
        plt.show()
        return(plt.show())
```

Listing 9.1: Python example

9.2 Tables

Tables related to selected samples using the $DIUSS$ algorithm

Selected Samples List		
Corn Cargill	Pig Manure	Pharma Tablets
0 1 3 4 5 6 7 8 9 11 12 13 14 15 16 17 18 20 23 24 27 28 32 33 36 37 39 41 43 44 45 47 48 49 51 54 55	2 3 4 5 9 15 16 18 20 21 23 25 29 30 31 33 34 36 43 45 47 48 49 50 52 55 57 58 61 62 64 68 72 74 75 76 79 80 81 82 83 84 85 89 91 93 94 95 99 100 101 102 103 104 109 110 111 113 115 116 117 119 120 122 123 125 126 127 128 130 138 139 140 143 147 148 151 152 154 157 158 160 161 163 164 166 168 169 172 175 176 177 180 182 183 184 188 191 192 195 198 203 204 205 206 207 208 212 213 214 216 220 221 222 223 224 226 228 231 233 234 235 236 237 238 241 242 243 246 247 248 253 255	1 2 7 10 11 12 14 15 16 17 19 20 23 26 27 29 30 31 35 36 37 39 40 41 44 48 51 53 56 58 59 62 65 67 72 73 75 77 78 79 80 81 84 87 89 91 93 96 98

Table 9.1: Subset of samples Z that were selected using the $diuss\_max$ algorithm.

Selected Samples List		
Corn Cargill	Pig Manure	Pharma Tablets
0 1 3 5 6 7 8 11 12 14 15 16 17 18 19 20 22 23 24 25 27 30 31 32 33 37 38 39 40 41 42 44 45 46 53 54 55	1 3 4 5 7 9 11 14 17 18 20 21 23 27 28 30 35 36 39 42 44 45 47 48 50 51 52 57 60 61 70 72 74 75 78 79 80 82 84 87 89 90 92 93 94 96 97 98 99 100 101 103 104 107 109 110 111 112 113 115 117 119 120 123 125 126 127 128 130 132 134 135 138 140 143 144 147 148 154 157 160 163 166 167 168 169 170 172 174 175 176 177 179 181 182 184 185 186 187 188 189 190 191 192 195 197 198 204 206 207 208 211 212 213 214 216 217 218 221 222 224 231 233 237 238 240 241 242 245 248 252 253 255	0 1 2 3 5 7 14 16 17 18 19 20 21 26 29 32 34 36 37 38 39 40 48 53 55 57 58 59 60 64 65 67 70 72 73 75 77 79 80 81 82 84 85 89 90 92 96 98 99

Table 9.2: Subset of samples Z that were selected using the $diuss\_sum$ lgorithm.

Selected Samples List			
Milk 60	Milk 90	Milk 120	Milk 193
3 12 26 27 36 40 47	7 14 15 18 19 21 31	1 5 10 14 22 30 32 34	4 5 6 8 9 10 11 13
49 64 68 78 90 97 98	33 37 40 42 48 49 53	36 37 38 42 43 47 54	14 17 18 19 21 24 25
100 106 108 112 115	56 57 58 59 62 70 72	55 66 68 69 72 73 74	26 28 29 30 31 33 34
120 122 127 128 129	74 75 78 82 84 87 89	76 77 78 81 83 85 88	36 39 40 43 48 50 51
136 146 151 154 158	94 95 97 98 99 100	89 94 95 96 101 103	52 53 54 57 59 60 61
163 166 167 170 173	101 102 109 114 118	105 108 110 117 120	62 63 66 71 72 73 74
178 179 186 194 197	121 124 129 136 140	121 124 127 129 131	77 79 80 82 83 85 86
215 217 223 228 232	146 147 151 152 160	140 142 144 149 150	87 90 91 92 93 94 95
233 238 242 245 246	169 170 172 176 179	151 158 160 161 162	96 97 98 100 102 104
250 261 262 268 269	180 184 186 192 198	163 165 166 169 171	105 106 107 108 109
270 274 278 289 312	203 205 212 213 217	174 179 181 185 186	110 111 114 115 116
314	218 222 225 226 227	190 198 199 201 202	118 121 122 123 125
	229 230 233 240 250	203 204 205 207 210	126 127 128 129 130
	255 262 263 266 268	213 214 215 218 220	131 132 134 135 136
	271 272 274 275 277	227 229 231 234 237	140 143 144 146 151
	279 288 306 307 308	241 245 248 250 251	153 154 155 156 157
	310	253 254 255 256 258	158 159 160 163 165
		261 264 265 269 270	167 168 169 172 173
		272 276 278 280 281	174 176 177 178 179
		284 285 286 287 288	182 183 184 185 186
		289 290 293 294 296	188 189 190 191 192
		301 303 306 308 314	193 196 200 202 203
			205 207 208 210 211
			212 213 218 219 220
			222 223 225 226 227
			228 231 234 235 237
			243 246 247 248 249
			250 251 253 254 255
			256 257 258 259 264
			265 266 269 271 272
			273 274 277 278 281
			283 286 287 290 291
			292 294 295 296 298
			299 300 303 304 307
			308 309 310 311 315

Table 9.3: Subset of samples Z that were selected using the $diuss\_max$ algorithm for the Milk case.

Selected Samples List			
Milk 60	Milk 90	Milk 120	Milk 193
3 4 5 34 39 42 47 64	2 7 13 14 15 16 17 18	1 2 4 6 7 10 12 20	0 8 9 10 11 13 14 15
71 90 98 99 106 109	19 21 24 30 32 36 40	22 24 32 36 38 44 49	16 18 19 21 22 23 24
112 114 117 118 120	48 50 53 54 56 57 58	51 54 55 64 65 67 71	25 26 28 29 30 34 36
127 128 129 136 138	59 62 69 72 74 78 79	73 74 77 85 87 89 96	37 40 41 42 43 44 47
140 145 152 155 156	82 84 85 86 87 95 97	101 104 108 118 119	48 50 51 52 57 59 61
162 163 167 170 171	98 102 106 114 118	120 124 126 127 129	62 63 65 66 67 70 71
173 186 194 199 203	119 121 124 129 133	130 131 132 135 142	72 73 75 77 78 79 80
205 220 242 245 246	136 138 143 147 155	143 146 149 150 152	82 83 85 87 90 91 93
249 259 261 264 265	160 161 169 170 179	154 159 160 162 163	95 96 98 104 106 108
270 274 278 284 285	180 184 186 190 191	166 169 174 175 176	110 112 113 114 116
291 297 303 310 311	198 202 203 205 206	177 185 186 188 189	121 123 126 127 128
314	213 215 217 218 227	198 201 202 203 204	129 130 131 132 133
	230 239 240 255 267	208 210 215 216 217	134 135 140 141 143
	268 271 272 273 275	218 220 224 227 231	144 146 148 149 150
	277 280 281 288 293	236 237 243 245 246	151 153 154 155 156
	297 302 306 307	247 248 249 250 251	157 158 160 161 162
		253 254 256 257 258	163 164 165 166 167
		261 262 265 270 272	168 169 171 172 174
		277 280 281 284 285	178 179 180 182 184
		286 287 290 292 293	185 188 190 192 195
		296 299 303 304 306	196 197 199 200 202
		308 310 311 312 313	203 205 206 207 208
		314	210 211 212 213 216
			219 220 222 223 224
			225 226 228 229 231
			234 235 236 237 238
			239 245 246 250 251
			253 254 255 256 257
			259 261 263 264 265
			266 267 270 271 272
			273 274 277 278 280
			281 283 284 287 288
			291 292 294 295 296
			298 299 300 303 304
			305 307 309 311 315

Table 9.4: Subset of samples Z that were selected using the $diuss\_sum$ algorithm for the Milk case.

Bibliography

[1] Valeria Fonseca Diaz; Bart De Ketelaere; Ben Aernouts; Wouter Saeys. *" Cost-efficient unsupervised sample selection for multivariate calibration"*. Chemometrics and Intelligent Laboratory Systems Volume 215, 15 August 2021, 104352. URL: https://www.sciencedirect.com/science/article/pii/S0169743921001209.

[2] Jessie Au; Kara N Youngentob; William J Foley; Ben D Moore; and Tom Fearn. *"Sample selection, calibration and validation of models developed from a large dataset of near infrared spectra of tree leaves."*. Journal of Near Infrared Spectroscopy 2020, Vol. 28(4) 186–203 ! The Author(s) 2020 Article reuse guidelines: sagepub.com/journals-permissions. DOI: 10.1177/0967033520902536. URL: https://journals.sagepub.com/doi/pdf/10.1177/0967033520902536.

[3] Alessandra Biancolillo and Federico Marini. *"Chemometric Methods for Spectroscopy-Based Pharmaceutical Analysis"*. AFront. Chem., 21 November 2018. DOI: DOI:10.13140/RG.2.2.29741.59360/1. URL: https://doi.org/10.3389/fchem.2018.00576.

[4] Kawamura S.; Kawasaki M.; Nakatsuji H.r. *"Near-infrared spectroscopic sensing system for online monitoring of milk quality during milking."*. Sens. & Instrumen. Food Qual. 1, 37–43 (2007). URL: https://doi.org/10.1007/s11694-006-9001-x.

[5] Harald Martens. *"Multivariate Calibration"*. In book: Encyclopedia of Statistical Sciences. DOI: 10.1002/0471667196.ess1105. URL: https://www.researchgate.net/publication/229680027_Multivariate_Calibration.

[6] Valeria Fonseca Diaz; Bart De Ketelaere; Ben Aernouts; Wouter Saey. *"Robustness control in bilinear modeling based on maximum correntropy"*. Willey Journal of Chemometrics. DOI: DOI:10.1002/cem.3215.

[7] Sijmen tie Jong. *"SIMPLS: an alternative approach squares regression to partial least"*. Unilever Research Laboratorium Vlaardingen, Olivier van Noortlaan 120, 3133 AT Vlaardingen (Netherlands) - (Received 1 June 1992; accepted 21 September 1992). URL: https://doi.org/10.1016/0169-7439(93)85002-X.

[8] Joan Ferre and F. Xavier Rius. *"Selection of the Best Calibration Sample Subset for Multivariate Regression"*. Departament de Quı´mica, Universitat Rovira i Virgili, Pl. Imperial Tarraco, 1, 43005-Tarragona, Spain. DOI: 10.1021/ac950482a. URL: https://pubs.acs.org/doi/pdf/10.1021/ac950482a.

[9] Heronides Adonias Dantas Filhoa; Roberto Kawakami Harrop Galvao; Mario Cesar Ugulino Araujo; Edvan Cirino da Silva; Teresa Cristina Bezerra Saldanha; Gledson Emidio Jose; Celio Pasquini; Ivo Milton Raimundo Jr.; Jarbas Jose Rodrigues Rohwedder. *"A strategy for selecting calibration samples for multivariate modelling"*. Chemometrics and Intelligent Laboratory Systems 72 (2004) 83 – 91. URL: https://www.sciencedirect.com/science/article/pii/S0169743904000681.

[10] D. Brynn Hibbert; Pentti Minkkinen; N.M. Faber; Barry M.Wise. *"IUPAC project: A glossary of concepts and terms in chemometrics"*. Analytica Chimica Acta Volume 642, Issues 1–2, 29 May 2009, Pages 3-5. DOI: https://doi.org/10.1016/j.aca.2009.02.020. URL: https://www.sciencedirect.com/science/article/pii/S0003267009002694.

[11] Alejandro C. Olivieri. *"Introduction to Multivariate Calibration A Practical Approach"*. Universidad Nacional de Rosario Instituto de Química Rosario - CONICET Rosario, Argentina. DOI: ISBN978-3-319-97096-7ISBN978-3-319-97097-4(eBook). URL: https://doi.org/10.1007/978-3-319-97097-4.

[12] Glenn T. Seaborg; James T. Ramaey; Gerald F. Tape; Wilfrid E. Johnson; Francesco Costagliola. *"Spectroscopy "*. U.S. Atomic Energy Commision/Division of technical information/ One of a series on understanding the atom.

[13] OPTICS and PHOTONICS SERIES (OP-TEC). *"Basics of Spectroscopy"*. Published and distributed by OP-TEC University of Central Florida http://www.optec.org. DOI: ISBN1-57837-501-0.

[14] Vladimir Sinik; Dr Zeljko V. Despotovic. *"19.Basics˙of˙electromagnetic˙radiation"*. October 2019 Conference: IX International Conference Industrial Engineering and Environmental Protection 2019 (IIZS 2019)At: Zrenjanin, Serbia. URL: https://www.researchgate.net/publication/336316251_BASICS_OF_ELECTROMAGNETIC_RADIATION.

[15] Donald L. Pavia; Gary M. Lampman; George S. Kriz. *"Introduction to Spectroscopy 3th edition - A guide for students of chemistry"*. Deparment of chemestry Western Washington University Bellingham Washington.

[16] M.Forina S.; Lanteri M. Casale. *"MULTIVARIATE CALIBRATION"*. Journal of Chromatography A (2007). DOI: doi:10.1016/j.chroma.2007.03.082. URL: https://www.researchgate.net/publication/6402697_Multivariate_calibration.

[17] Rasmus Bro. *"Multivariate calibration What is in chemometrics for the analytical chemist?"*. Department of Dairy and Food Science, The Royal Veterinary and Agricultural University, Rolighedsvej 30, DK-1958 Frederiksberg, Denmark Received 20 March 2003; received in revised form 27 May 2003; accepted 27 May 2003. DOI: AnalyticaChimicaActa500(2003)185194. URL: https://www.sciencedirect.com/science/article/pii/S0003267003006810.

[18] S. Wold; A. Ruhe; H. Wold; and W. J. Dunn III. *"The Collinearity Problem in Linear Regression. The Partial Least Squares (PLS) Approach to Generalized Inverses"*. DOI: 10.1137-0905052. URL: https://doi.org/10.1137/0905052.

[19] Ling Lin; Qirui Zhang; Mei Zhou; Sijia Xu; and Gang Li. *"Calibration set selection method based on the "M+N" theory: application to non-invasive measurement by dynamic spectrum"*. DOI: 10.1039/c6ra19272f. URL: https://pubs.rsc.org/en/content/articlepdf/2016/ra/c6ra19272f.

[20] Sijmen de Jong. *"SIMPLS˙an˙alternative˙approach˙to˙partial˙least˙squares˙regression"*. Chemometrics and intelligent Laboratory Systems. DOI: 18(1993)-251-263. URL: https://www.sciencedirect.com/science/article/pii/016974399385002X?via%5C%3Dihub.

[21] Ramin Nikzad-Langerodi; Werner Zellinger; Edwin Lughofer; and Susanne Saminger-Platz. *"Domain-Invariant Partial-Least-Squares Regression*. Department of Knowledge-Based Mathematical Systems, Johannes Kepler University, 4040 Linz, Austria. URL: https://pubs.acs.org/doi/pdf/10.1021/acs.analchem.8b00498.

[22] David Gonzalez Gomez; Mercedes Lozano; Ana Maria Fernandez-Leon; María Fernanda Fernández León. *"PLS Calibration to Resolve Overlapping Peaks of Lutein and Zeaxanthin in Vegetable Samples by LC"*. January 2012Czech Journal of Food Sciences 30(4):358-363. DOI: 10.17221/121/2011-CJFS. URL: https://www.researchgate.net/publication/268213938_PLS_Calibration_to_Resolve_Overlapping_Peaks_of_Lutein_and_Zeaxanthin_in_Vegetable_Samples_by_LC.

[23] Titin Agustin Nengsih; Frédéric Bertrand;Maumy Myriam; Nicolas Meyer. *"Determining the number of components in PLS regression on incomplete data set"*. November 2019 Statistical Applications in Genetics and Molecular Biology 18(6). DOI: DOI:10.1515/sagmb-2018-0059. URL: https://www.researchgate.net/publication/337101310_Determining_the_number_of_components_in_PLS_regression_on_incomplete_data_set.

[24] R.W. Kennard and L.A. Stone. *"Computer Aided Design of Experiments"*. Technometrics Vol. 11, No. 1 (Feb., 1969), pp. 137-148 (12 pages). URL: https://www.jstor.org/stable/1266770.

[25] Ronald D. Snee. *"Validation of Regression Models: Methods and Examples"*. D Engineering Department; November 1977, Technometrics 19(4):415-428. DOI: 10.1080/00401706.1977.10489581.. URL: https://www.researchgate.net/publication/239799520_Validation_of_Regression_Models_Methods_and_Examples.

[26] Gerd Puchwein. *"Selection of calibration samples for near-infrared spectrometry by factor analysis of spectra"*. Anal. Chem. 1988 60 6 569–573 Publication Date:March 15 1988. DOI: 44. URL: https://doi.org/10.1021/ac00157a015.

[27] Tomas Isaksson and Tormod Naes. *"Selection of Samples for Calibration in Near-Infrared Spectroscopy. Part II: Selection Based on Spectral Measurements"*. First Published August 1, 1990 Research Article. URL: https://doi.org/10.1366/0003702904086533.

[28] Saptoro; Agus; Tadé; Moses O.; and Vuthaluru Hari (2012). *"A Modified Kennard-Stone Algorithm for Optimal Division of Data for Developing Artificial Neural Network Models"*. Chemical Product and Process Modeling: Vol. 7: Iss. 1, Article 13. DOI: doi:10.1515/1934-2659.1645. URL: https://www.researchgate.net/publication/270267853_A_Modified_Kennard-Stone_Algorithm_for_Optimal_Division_of_Data_for_Developing_Artificial_Neural_Network_Models.

[29] Neal B. Gallagher; Donal O'Sullivan. *"Selection of Representative Learning and Test Sets Using the Onion Method"*. 196 Hyacinth Road Manson, WA 98831. URL: https://eigenvector.com/wp-content/uploads/2020/01/Onion_SampleSelection.pdf.

[30] Antoine Stevens and Leonardo Ramirez Lopez. *"An introduction to the prospectr package"*. Georges Lemaître Centre for Earth and Climate Research, Earth and Life Institute, UCLouvain, Place Pasteur 3, 1348, August 14, 2013. URL: https://www.researchgate.net/publication/255941339_An_introduction_to_the_prospectr_package.

[31] Zahra Nazari; Masooma Nazari and Dongshik Kang. *"Bottom-Up Hierarchical Clustering Algorithm With Intersection Points"*. International Journal of Innovative Computing; 2019 ISSN 1349-4198 Volume 15, Number 1. URL: https://www.researchgate.net/publication/330385637_A_Bottom-up_Hierarchical_Clustering_Algorithm_with_Intersection_Points.

[32] Wolfgang Hardle; Leopold Simar. *"Applied multivariate statistical Analysis"*. Version: 29th April 2003.

[33] P.Baby K.Sasirekha. *"Agglomerative Hierarchical Clustering Algorithm- A Review"*. International Journal of Scientific and Research Publications, Volume 3. URL: https://www.semanticscholar.org/paper/Agglomerative-Hierarchical-Clustering-Algorithm-A-Sasirekha-Baby/bd9fe11a960001cb845ea74a75cf8c10b8c34615.

[34] Richard A. Johnson; Dearn W. Wichern. *"Applied multivariate statistical Analysis 6th"*. ed. ISBN 0-13-187715-1.

[35] Jolliffe IT; Cadima J. 2016. *"Principal component analysis: a review and recent developments"*. .Phil. Trans. R. Soc. A 374:20150202. URL: https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202.

[36] Ramin Nikzad-Langerodi; Werner Zellinger; Susanne Saminger Platz; Bernhard A. Moser. *"Domain adaptation for regression under Beer–Lambert's law"*. Contents lists available at ScienceDirect; Knowledge-Based Systems. URL: https://www.sciencedirect.com/science/article/pii/S0950705120305761.

[37] Michael E. Wall; Andreas Rechtsteiner; Luis M. Rocha. *" Singular value decomposition and principal component analysis In A Practical Approach to Microarray Data Analysis"*. University of California, 2002.

[38] Johan A. K. Suykens; Jony Van Gestel; Jos De Brabanter; Bart De Moor and Joos Vandewalle. *"Least Squares Support Vector Machines"*. World Scientific Publishing Co. Pte. Ltd. 5 TohTuck Link; Singapore 596224; USA office: 27 Warren Street, Suite 401-402, Hackensack; NJ 07601; UK office: 57 Shelton Street; Covent Garden; London WC2H 9HE; ISBN =9812381511.

[39] M. Schoot; C. Kapper; G.H. van Kollenburg; G.J. Postma; G. van Kessel; L.M. Buydens; J.J. Jansen. *" Investigating the need for preprocessing of near-infrared spectroscopic data as a function of sample size"*. April, Chemometr. Intell. Lab. Syst. 204 (2020), 104105, URL: https://www.sciencedirect.com/science/article/pii/S016974392030201X.

[40] Loan Tomuta; Rares Lovanov; Ede Bodoki; Sorin Leucuta. *"Quantification of meloxicam and excipients on intact tablets by near infrared spectrometry and chemometry"*. URL: https://www.researchgate.net/publication/215494503.

[41] R. Wheeler. *"optFederov.AlgDesign. The R Project for Statistical Computing, 2019."*. URL: https://cran.r-project.org/web/packages/AlgDesign/AlgDesign.pdf.

[42] Alex B Mcbratney; Budiman Minasny. *"Why you don't need to use RPD"*. URL: https://www.researchgate.net/publication/285634867_Why_you_don't_need_to_use_RPD.