Chemometrics for Data Interpretation: Application of Principal Components Analysis (PCA) to Multivariate Spectroscopic Measurements

Chemometrics for Data Interpretation: Application of Principal Components Analysis (PCA) to Multivariate Spectroscopic Measurements

Leonardo Iannucci

Dipartimento di Scienza Applicata e Tecnologia, Politecnico di Torino

leonardo.iannucci@polito.it

Extracting relevant and useful information from measurements is an issue of paramount importance and it can be considered as complementary to the process of data acquisition. This is a crucial point especially in the field of chemical measurements, where data sets can consist of hundreds or even thousands of variables so their interpretation can require long time. Chemometrics try to tackle this issue by applying mathematical and statistical tools to data coming from chemical, biological or medical analyses. Among possible methods, Principal Components Analysis (PCA) has found wide application in the I&M field thanks to its ability to identify patterns in acquired measurements and classify data in different groups. Possible applications span from chemicals detection [1] to concentration estimation of compounds in a given system [2]. Actually many studies demonstrated the possibility to use PCA to process different kind of data [3], in some cases coupling PCA to other tools such as artificial neural networks to improve the processing performance [4].

Many books addressed PCA in an exhaustive and complete way, so readers can refer to them in order to obtain a thorough discussion of this topic [5] [6]; aim of this article is rather to provide only a simple dissertation for a beginner in this field to show how powerful the PCA is. Specifically, detailed indications for the analysis of spectroscopic measurements are provided, as this kind of data is often left out in many reviews concerning PCA. Moreover, tips and instructions are given to let the reader write his/her own code and implement such data processing using the Python programming language not only for spectroscopic data, but more in general for any kind of data.

PCA Basics

Modern instrumentations allow researchers to collect huge amount of data in an easy and often automated way. Obviously, this is true also for the field of chemistry, where the possibility of

performing fast and inexpensive analyses has generally moved the critical point in the design of experiments from phenomenon measurement to data interpretation.

When more than one quantity is measured for each sample, it is possible to define this data set as *multivariate,* i.e. each measurement is composed of many variables; this is the case for example when measuring length, width, and weight of different objects. It is possible to describe multivariate measurements as a matrix composed of *m* rows, representing *m* analysed samples, and *n* columns, indicating each of the analysed variables. In chemistry, common examples for multivariate data sets are spectroscopic measurements, i.e. those analyses in which the interaction between the sample and an electromagnetic radiation is studied. Actually, techniques like Infra-Red spectroscopy or UV-Visible spectroscopy probe the sample using not just a single wavelength, but rather using a range of wavelengths in order to identify and study different chemical bonds. So, when dealing with spectroscopic measurements it is possible to talk about multivariate measurements, and each of the analysed wavelengths is a variable composing the acquired data set.

In multivariate measurements there is real possibility to have correlation between different variables, so a great benefit can derive from removal of redundant information. Principal Components Analysis (PCA) is, in its simplest definition, a method to perform variables reduction in an acquired data set. The original data matrix is transposed into a new space having lower dimension but where the new variables constituting the model (named principal components) account for most of the variability contained in the original data set. In matrix notation, it is possible to describe this operation as follows:

$$X=T\cdot P^{T}+E \qquad (1)$$

where X is the original data matrix (having dimension *m x n*), P is the loading matrix that is the eigenvectors representing the new space (with dimension *n x k*, where k is the number of variables in the PCA model usually much lower than *n*), T is the score matrix, composed of the eigenvalues derived from X matrix decomposition and having dimension *m x k* and eventually E is the residual matrix, sometimes referred to as *error matrix,* which contains the variance burden not explained by the PCA model. The eigenvectors and eigenvalues introduced in equation (1) can be obtained from the diagonalization of the covariance matrix calculated for the original data matrix X. This procedure can be carried out using any programming language allowing matrix calculation; in this paper, a hands-on example is provided using the free Python language. Actually, in the last decades this programming language has gained increasing

consideration in the scientific world thanks to its easy syntax and to its numerous standard and external libraries. For example, PCA can be easily implemented using few lines of codes thanks to the open-source Scikit-learn library, which provides all necessary operations to perform needed calculations. Anyway, before moving to this step, first it is necessary to perform some preliminary processing to the acquired data matrix in order to obtain meaningful results from PCA. Actually, this chemometric technique is deeply influenced by measurement noise and by the signal magnitude, so these two factors should be minimized or made similar for all samples before computing the principal components of the PCA model. In the next section, these pre-processing operations will be described in detail, presenting the example of spectroscopic measurements.

Pre-processing

Pre-processing operations can be divided in four steps: 1) interval selection, 2) baseline removal, 3) smoothing and 4) normalization.

Step 1: Not all measurements carry useful information, so the first step is usually to identify the meaningful data. When dealing with spectroscopic measurements, often not the whole spectrum is worth attention, but only some ranges where characteristic peaks of the analyzed compounds are present. Because of this, generally the best choice is to limit the processed dataset to this part of the spectrum and to linearly interpolate it to generate a set number of data points. This way it is possible to avoid building a model influenced by not relevant spectral regions and all samples will be constituted by the same abscissa coordinates.

Step 2: The baseline of the acquired spectrum is removed. Actually, it is not uncommon to have spectra characterized by an offset or by a sloping line due to instrument drift or side phenomena occurring when performing the measurement. One of the possibilities is to compute the baseline by means of symmetric least square smoothing, as described in [7]. This operation can be easily performed in Python taking advantage of the Scipy library, that allows the user to perform matrix operations and solve matrix expressions [8]. After computing the baseline, this background can be subtracted from the original spectrum.

Step 3: The signal-to-noise ratio (SNR) can be improved by applying a smoothing filter to the original data. One of the most popular filter is the so-called Savitzky-Golay filter, which is based on local least-squares polynomial approximation [9]. It can be applied to the analyzed dataset using the *savgol_filter* function from *Scipy* library. Input parameters for this function are the window length and the polynomial order for the fitting curve; they should be carefully

chosen in order to avoid any over-smoothing, that would lead to loss of information in the final data, as one of the side effects of any smoothing filter is to reduce peak height, with possible elimination of small peaks or shoulders. Because of this, input parameters for Savitzky-Golay filter should be tailored to each specific application, but a general rule of thumb often found in many publications is to use a window length of 15 points and a 2nd order polynomial.

Step 4: The normalization can be carried out by means of the Standard Normal Variate Transformation, i.e. using the following expression for each measurement point:

$$y_{SNV} = \frac{y - \acute{y}}{std} \quad (2)$$

where, y_{SNV} is the variable value after transformation, y is the original variable, $\acute{y}$ is the mean value in the original spectrum and *std* is the standard deviation [10]. In this way, all spectra composing the dataset will be mean-centered and scaled to unit variance.

The effect of all pre-processing operations can be observed in Figure 1. The reported spectra are acquired analyzing copper sulphate and copper hydroxychloride crystals (respectively, on the left and on the right) using Raman spectroscopy. This spectroscopic technique probes the sample under investigation using a laser radiation having wavelength either in the visible or in the infra-red range. Vibrational modes at molecular level can cause inelastic scattering in the analyzed material, which would result in the emission of photons with different energies (Raman scattering). This signal is finally collected by a detector in order to compose the Raman spectrum of the sample. The great advantage of this technique is the possibility to identify specific compounds, because each bond in the compound generates its characteristic peaks, but at the same time spectra interpretation is often not straightforward, due to possible presence of a large number of peaks in the acquired spectrum. In the two examples shown in Figure 1, only wavelengths between 200 cm^{-1} and 1200 cm^{-1} are selected (step 1), as this range is the most important for inorganic compounds. Actually, even if additional peaks can be found in the region between 3000 cm^{-1} and 3500 cm^{-1} (see spectrum on the right in Figure 1) associated to the OH bond stretching, the first part of the spectrum is generally considered as the 'fingerprint region' for identification of minerals. After selection of the interval of interest, the large background signal due to the fluorescence emission is removed (step 2), Savitzky-Golay filter is applied to improve the signal-to-noise ratio (step 3) and Standard Normal Variate Transformation is applied to normalize the spectra to unit variance (step 4). In this example, SNR (computed as the ratio between the highest peak and the baseline noise) improved of about

7% in both spectra. Results of all pre-processing are shown at figure bottom, where the most important peaks are labelled indicating the bond they can be related to.

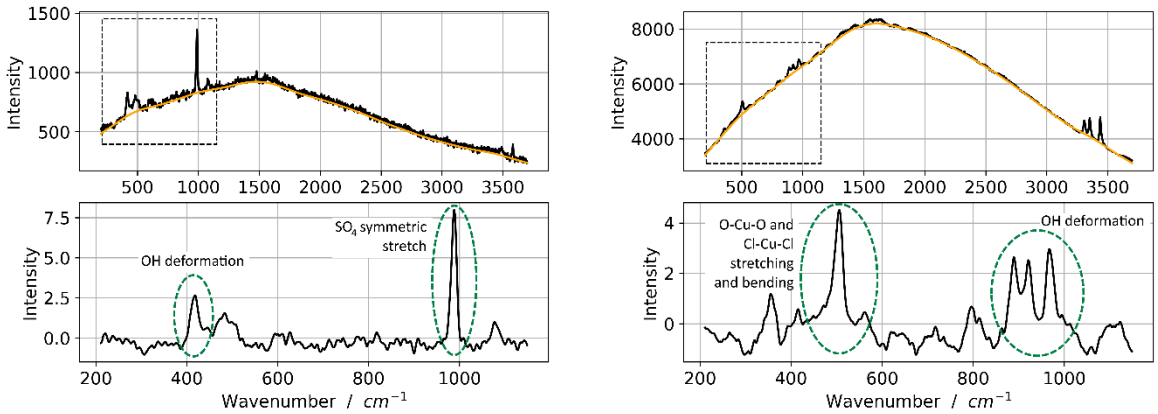

Figure 1 On top: spectra acquired using Raman spectroscopy on copper sulphate (antlerite $Cu_3(SO_4)(OH)_4$, - on the left) and on copper hydroxychloride (clinoatacamite $Cu_2(OH)_3Cl$ -, on the right). At the bottom: the same spectra after applying the four-step pre-processing.

Principal Components computation

After the pre-processing, it is possible to build the PCA model and compute the principal components. As mentioned above, this can be readily done using the Scikit-learn class *sklearn.decomposition.PCA* [11], that as input parameter allows the user to choose the number of components to build the model.

In order to show the capabilities of the PCA for spectra processing, hereinafter a practical example is provided. Fourteen Raman spectra of different copper sulphates (namely antlerite, brochantite, chalcanthite, langite and orthoserpierite) are processed to show the possibility of easily discriminating among spectra that, at a first glance, could look very similar. All spectra have been extracted from the RRuff database, an open access library that provides a unique tool for researchers interested in mineralogy and materials science [12], so the reader can download them to carry out the processing independently and then use this code to interpret his/her own data.

In Python environment, let the initial fourteen spectra be in an array-like object (named *all_data)*, in which the spectra are stored after performing the pre-processing as previously described. It is possible to apply dimensionality reduction of this dataset, that has initial dimension of 14 x 1000, as follows:

from sklearn.decomposition import PCA as sklearnPCA

k=3

model_pca = sklearnPCA(n_components= k)

model_pca.fit(all_data)

eigenvalues = model_pca.transform(all_data)

First the number of components for the model is chosen, here *k*; then the model is fitted with the original dataset (*all_data*) and eventually the original data (*all_data)* is projected on the principal components just computed. This way it is possible to calculate the final *eigenvalues* (with dimensions *m x k*). The choice of the components' number in the first instruction does not affect the subsequent computation and the eigenvectors will not be modified by the number of considered components. However, it is a critical point because it affects the total variance explained by the model: using a low number of components could create a model that is not able to correctly fit the acquired data, leading to not reliable results.

It is possible to start analyzing the obtained result computing the per cent cumulative explained variance for the considered components:

*cev = np.cumsum(model_pca.explained_variance_ratio_)*100*

As hinted previously, choosing the correct number of components for the model is not trivial. First of all, it is necessary to build a model that captures a sufficiently high fraction of the initial variance, otherwise it would be not representative of the acquired data set; a common advice is to reach at least 70% or even 80% of the initial variance. Another important rule of thumb is to add components only if they have an explained variance of at least 4% or 5%, to avoid fitting noise instead of spectra features.

In the described example, the trend for cumulative explained variance is shown in Figure 2. In this case, the best choice would be probably to use the first three components in order to avoid any over-fitting and also allow a prompt visual representation of the data. As can be seen from the plot, the first three components already account for almost 90% of the initial variance, so the fourth and subsequent, capturing individually less than 5% of the variance, can be omitted. At this point it is possible to visualize the eigenvectors that, dealing with spectroscopic data, show the main features present in the spectra:

*eigenvectors = model_pca.components_*

It is possible to use the loadings in order to check which features (i.e. peaks) the explained variance is representing and thus reach a more clear results interpretation. Actually, the original spectrum can be expressed as a linear combination of eigenvectors and eigenvalues, according to the following expression, equivalent to (1):

$$X = t_1 p_1^T + t_2 p_2^T + \ldots + t_k p_k^T + E \qquad (3)$$

As it is not possible to have negative peaks in Raman spectroscopy, features in negative domain will correspond to negative eigenvalues or they will be compensated by another component so as to obtain positive values in the linear combination (3).

As an example, in Figure 2 it is possible to see that the first two components are characterized by a sharp peak before 1000 cm^{-1}, which is characteristic of sulphates, slightly shifted depending on the specific mineral. Immediately close to it and also in other regions of the spectrum there are minimum points, that thus can be explained looking at the corresponding eigenvalues.

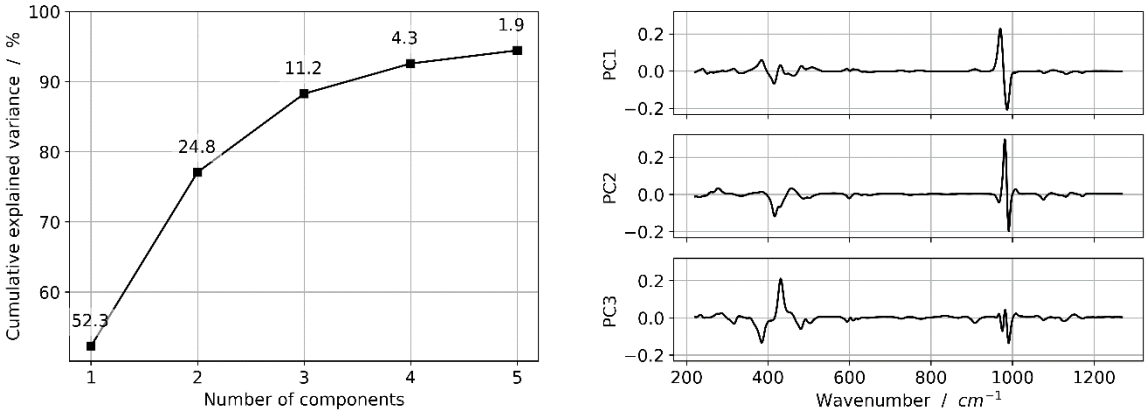

Figure 2 On left: cumulative explained variance as a function of the components number in the PCA model. Each marker is labelled with the per cent variance captured by the corresponding PC. On the right: loadings associated to the first three components in the PCA model.

The effect of the addition of an increasing number of components can be observed in Figure 3. The figure shows how the model progressively approximates the acquired spectrum thanks to the use of a higher number of components, until reaching a satisfactory result with 3 PC. Specifically, the first two components fit the main features in the high-wavenumber region, while the third one is able to fit the peaks in the part of the spectrum below 600 cm^{-1}.

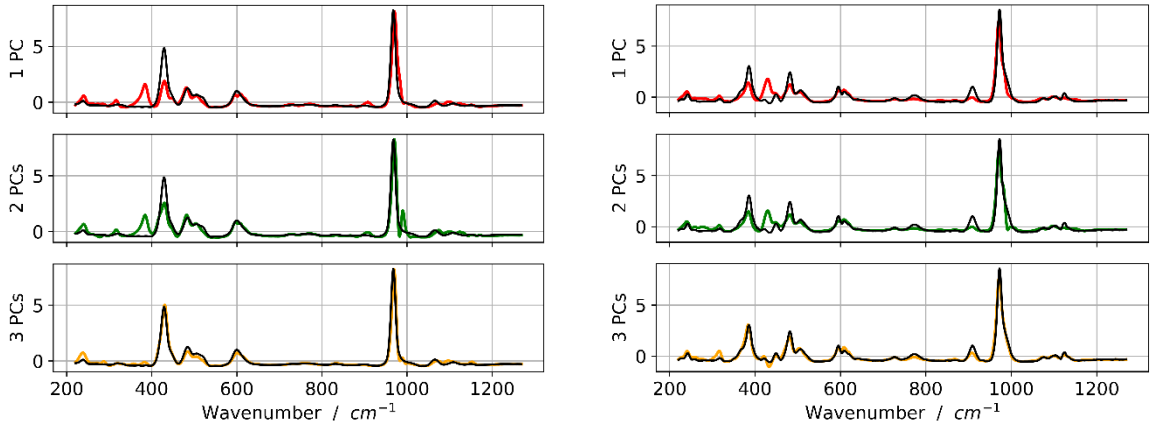

Figure 3 Reconstruction of two spectra (R060090 on the left and R100199 on the right) using an increasing number of PCs. Original spectrum is represented in black, colored lines are used for the model; it is possible to see how the addition of PCs improves the goodness of fit. Samples names stand for the spectra Rruff ID.

After the number of components has been chosen, it is possible to move to the main outcome of the PCA model, that is the clustering of the different samples in the so-called 'score' plot. Score charts, also named 'biplots', show the eigenvalues of the different samples (two at the time), so spectra that appear close in these plots are similar to each other. In the presented example (shown in Figure 4), it is possible to see that the measurements group in five clusters, corresponding to the five analyzed copper sulphates (in order to make it more evident, different compounds were labelled with different colors). As can be seen, langite and brochantite have similar positive PC1 values, but are discriminated by PC2, which is positive only for the latter. Both have a peak at about 970 cm^{-1}, modelled by PC1, while langite has an additional sharp peak at about 430 cm^{-1}, captured by a minimum in PC2 loading, which is not present in brochantite. The three other compounds, having negative PC1 values, are characterized by a peak at about 986 cm^{-1}, modelled by the PC1 minimum in that region. The discrimination among chalcantite, antlerite and orthoserpierite is then obtained thanks to PC2 and PC3, that are able to model the differences in the spectral region below 600 cm^{-1}. Antlerite and orthoserpierite have similar spectra, as highlighted by the similar PC1 and PC3 eigenvalues, but can be differentiated thanks to PC2.

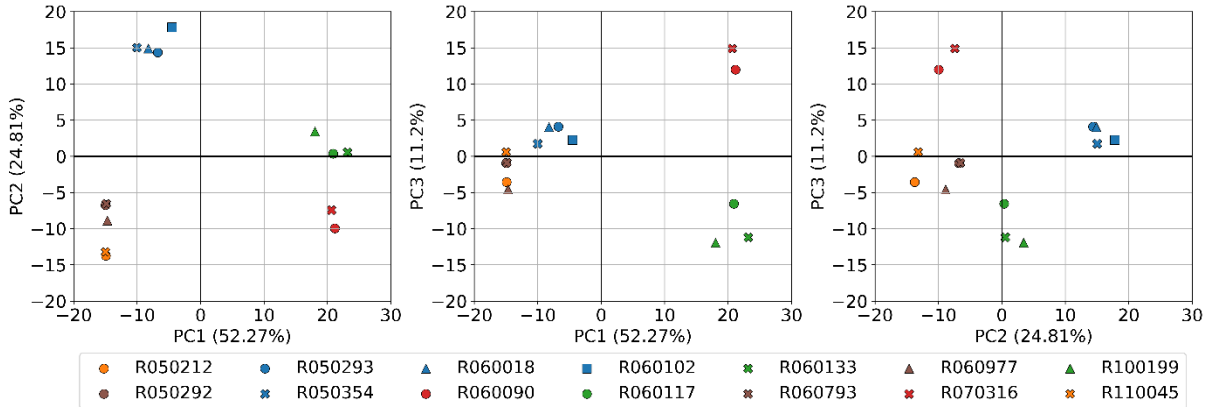

Figure 4 Score plots of the first three components (PC1-PC2, PC1-PC3 and PC2-PC3). Percent variance captured by each PC is reported in parenthesis along each axis. Samples names stand for the spectra Rruff ID.

This simple example demonstrated how it would be possible to easily discriminate different compounds after acquiring several measurements on unknown samples. As a matter of fact, PCA alone would not be able to identify the species, as this would require additional techniques such as Partial Least Square Discriminant Analysis (PLS-DA) [5][6], but it can greatly speed up the analysis of large number of measurements providing an effective and unsupervised method to recognize differences in acquired spectra.

Outliers detection and model optimization

Obtaining a satisfactory and clear clustering is not always straightforward when dealing with PCA. This could be because analyzed samples do not show relevant differences to be classified in different groups, but sometimes this could be due to a not correct model construction. PCA is deeply influenced by outliers and noisy measurements, so they should be removed and analyzed separately to avoid spoiling the model. It is possible to identify the presence of outliers thanks to the calculation of two quantities: leverage and root mean square deviation (RMSD) [13]. The former is defined as the diagonals of the "hat matrix" (H):

$$H = T(T^T T)^{-1} T^T \qquad (4)$$

where T is the score matrix, while the latter can be computed as follows:

$$RMSD = \sqrt{\frac{\sum_{i=1}^{J} (X - TP^T)^2}{J}} \qquad (5)$$

where is *J* the number of points for each measurement (1000 in the example here presented), *X* is the original data matrix, *T* is the score matrix and *P* is the loading matrix. Leverage quantifies the influence of a single sample on the model construction; measurements characterized by high leverage values should be discarded from the processing because they tend to bias the model. A common choice often found in literature is to set the threshold value equal to 3 times the average leverage value, but sometimes a more restrictive view sets this limit to 3 times the median value [14]; in this way, all measurements will give a similar contribution to model construction. On the other hand, root mean square deviation quantifies the difference between the original spectrum and the same spectrum after inverse transformation from PC space to the original variables. So, in this case the principal components number has a great influence on the calculated RMSD because it affects the residual variance in each spectrum. In Figure 5 it is possible to see the plot of leverage and RMSD for each of the analyzed samples. All measurements fall in the range below three times the median leverage value, indicating the absence of outliers. Then, looking at the RMSD, it is possible to see that almost all samples are characterized by a value below 0.30, demonstrating that the model is able to correctly fit them.

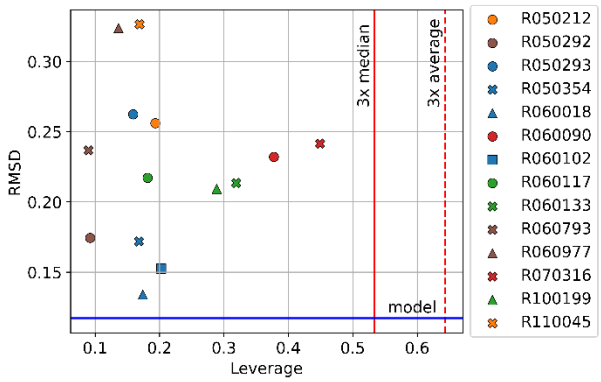

Figure 5 Plot showing leverage and root mean square deviation (RMSD) for each of the analyzed samples in the three-component PCA model. The blue horizontal line represents the model residual variance, while the two red vertical lines indicate three times the median and average leverage value. Samples names stand for the spectra Rruff ID.

In order to further investigate the goodness of the model, it is possible to plot the spectrum after inverse transformation superimposed to the original one. In this way the user can directly control which regions of the spectra are correctly fitted and which not. This is a crucial point because, even if the model is not able to fit the whole spectrum, it is essential that it correctly fits the most important features, otherwise it should be discarded. In Figure 6 one spectrum is

shown for each of the identified groups. As can be seen, the model is able to correctly fit all the main peaks and discrepancies can be observed only in limited parts of the spectrum.

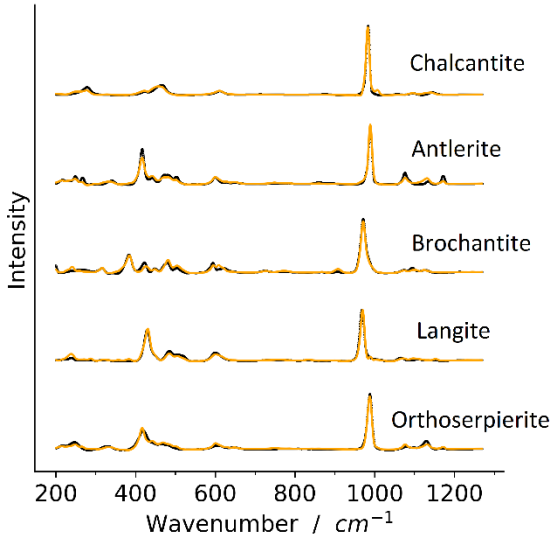

Figure 6 Goodness of fit can be shown superimposing the measured spectra (here as black dots) and the result coming from the three-component PCA model (here as yellow line). In this plot, one representative spectrum for each of the identified groups is displayed, from top: chalcantite, antlerite, brochantite, langite and orthoserpierite.

Conclusions

This paper presented the most important operations to perform Principal Components Analysis. As discussed, this chemometric technique provides a powerful tool for unsupervised features extraction from large data sets and it can be effectively used to discriminate between different groups in acquired measurements, facilitating results interpretation. It can represent a considerable help for researchers dealing with different kind of measurements, and specifically for chemists, in order to extract relevant information and reduce data sets dimension. Moreover, as the processing effort is not particularly high (time required for PCA model construction is less than 200 ms using an average computer), this technique can also be used to implement real-time applications [15].

References

[1] E. Garcia-Breijo, R. M. Peris, C. O. Pinatti, M. A. Fillol, J. I. Civera and R. B. Prats, Low-Cost Electronic Tongue System and Its Application to Explosive Detection, IEEE Transactions on Instrumentation and Measurement, 62 (2013), pp. 424-431.

[2] C. E. Teixeira, L. E. Borges da Silva, G. F.C. Veloso, et al., An ultrasound-based water-cut meter for heavy fuel oil, Measurement, 148 (2019), pp. 1-9.

[3] H. Lizhi, K. Toyoda, I. Ihara, Discrimination of olive oil adulterated with vegetable oils using dielectric spectroscopy, Journal of Food Engineering, 96 (2010), pp. 167-171

[4] L. Xu, C. Tan, X. Li, Y. Cheng and X. Li, Fuel-Type Identification Using Joint Probability Density Arbiter and Soft-Computing Techniques, IEEE Transactions on Instrumentation and Measurement, 61 (2012), pp. 286-296.

[5] J. N. Miller, J. C. Miller, Statistics and Chemometrics for Analytical Chemistry, Pearson, Harlow, 2010, ISBN 978-0-273-73042-2.

[6] P. Gemperline, Practical Guide to Chemometrics, CRC Press, Boca Raton (FL, USA), 2006, ISBN 1-57444-783-1.

[7] P. H. C. Eilers, A perfect smoother, Analytical Chemistry, 75 (2003), pp. 3631-3636.

[8] P. Virtanen et al., SciPy 1.0: fundamental algorithms for scientific computing in Python, Nature Methods, 17, (2020), pp. 261–272.

[9] A. Savitzky, M. J. E. Golay, Smoothing and Differentiation of Data by Simplified Least Squares Procedures, Analytical Chemistry, 36 (1964), pp. 1627-1639.

[10] R. J. Barnes, M. S. Dhanoa, S. J. Lister, Standard Normal Variate Transformation and De-Trending of Near-Infrared Diffuse Reflectance Spectra, Applied Spectroscopy, 43 (1989), pp. 772-777.

[11] F. Pedregosa et al., Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research, 12 (2011), pp. 2825-2830

[12] B. Lafuente, R.T Downs, H. Yang , N. Stone, The power of databases: the RRUFF project, In: Highlights in Mineralogical Crystallography, T. Armbruster and R. M. Danisi, eds. Berlin, Germany, W. De Gruyter, (2015) pp. 1-30.

[13] K. Kumar, Principal Component Analysis: Most Favourite Tool in Chemometrics, Resonance, 22, (2017) pp. 747-759.

[14] A. F. Mejia, M. B. Nebel, A. Eloyan, B. Caffo, M. A. Lindquist, PCA leverage: outlier detection for high-dimensional functional magnetic resonance imaging data, Biostatistics, 18 (2017) pp. 521-536.

[15] B. D. de Senneville, A. El Hamidi, C. Moonen, A Direct PCA-Based Approach for Real-Time Description of Physiological Organ Deformations, IEEE Transactions on Medical Imaging, 34 (2015), pp. 974-982.

Author bio

Leonardo Iannucci is currently a research fellow at Politecnico di Torino in the Department of Applied Science and Technology. He got the M.S. degree in Materials Engineering in 2016 and then in 2019 he received the PhD in Metrology cum laude from Politecnico di Torino. His main research fields are corrosion science, electrochemical measurements and materials characterization.