Manual of PCA PyFitIt module
allowing the spectral decomposition of an experimental XANES data mixture

17 May 2019

1. Input data format and commands to import and plot the experimental XANES data

The experimental input data file must consist of a .dat or .txt file formatted in the following way: the first column must contain the energy values while the following columns must be filled by the related absorption coefficients acquired during a time-dependent variation of a physical/chemical variable (e.g. time, temperature, PH … ect). It is worth noting that to every XANES spectrum must correspond the same energy column. This fact implies that the number of points, characterising each XANES spectrum, must be equals to the number of points sited in the energy column. The PCA module allows the user to perform the data normalization (see Section 3.1), however it is recommended to use as a input file a set of XANES spectra already normalized for their absorption jump. This process can be realised by the external XAS data analysis software such us ATHENA [1]. An example of a correctly formatted input file is reported in Figure 1.

	Energy Column	Spectrum n° 1	Spectrum n° 2	...	Spectrum n° N
1	11550.014	0.044375376	0.044665908		0.043951884
2	11550.498	0.047152748	0.047322184		0.045778714
3	11551.012	0.050540038	0.049210272		0.049330954
4	11551.495	0.052695746	0.052424903		0.052580219
5	11552.011	0.056235559	0.056349347		0.055628296
6	11552.493	0.060017208	0.059241466		0.059056265
7	11553.009	0.063950903	0.062870636		0.062904212
8	11553.492	0.070000736	0.067010532		0.068621283
9	11554.006	0.073728324	0.073597701		0.073322351
10	11554.49	0.080664818	0.079153692		0.077604864
11	11555.006	0.086491272	0.084773389		0.083879207
12	11555.489	0.092940394	0.091129278		0.091251624
13	11556.004	0.10139366	0.099563056		0.10001089
14	11556.487	0.11105769	0.1093356		0.10805654
15	11557.003	0.12020907	0.12024933		0.1174523

Figure 1: Example of a proper formatted input dataset. The energy columns must be followed by the related spectra located in the columns of the data file.

The data file name be passed as argument to the function **`openFile`** as: **`dataFileBrowser = openFile('file_name.dat (or.txt)')`**. The following cell, containing the function **`plot_data(energy,data)`** allows to visualise the experimental data before the statistical analysis and the related spectral decomposition. In the same cell, user can modify the data energy range (selecting the regions where the data variation is more significant) adding and executing the following code lines:

`e_Min`= minimum value chosen ; **`e_Max`**= maximum value chosen

```
e_Min_near=min(energy, key=lambda x:abs(x-e_Min)); e_Max_near=min(energy, key=lambda x:abs(x-e_Max))
```

```
p_Min=list(energy).index(e_Min_near);p_Max=list(energy).index(e_Max_near)
```

```
energy=energy[p_Min:p_Max]; data=data[p_Min:p:Max]
```

2. Statistical Analysis of the Experimental Dataset

2.1. Principal Component Analysis (PCA) on the experimental XANES dataset

The **`calcSVD(data)`** performs the Singular Value Decomposition (SVD) of the XANES data matrix. Starting from this procedure, the eigenvalues (`l`) of the covariance matrix associated to the original dataset and the data projections over its eigenvectors (**`principal_components`**) are retrieved. This two output are at the basis of the statistical and empirical results provided by the following functions (see Sections 2.2 and 2.3).

All the following pictures refer to the PtH.dat file.

2.2. Calculations and graphical representation of the statistical parameters used to identify the exact number of principal components (PCs)

Function **`MalinowskyParameters(data, l)`** returns the statistical parameters: IND, IE and the percentage of significance (F) associated to each PC together with the principal dimensions **`pc`** (or component number). These values can be saved in a .txt file inside the sub-folder *results* in the *PCA* folder by the command **`saveToFile('results/statistic.dat', statistic)`**. The same command is also used to save the **`l`**−values (i.e. the variance associated to each PC). Each of this statistic parameters can be plotted *vs* the number of the principal dimensions. This purpose is realised by the command **`plotTestStatistic(statistic, pc, l)`**. The graphical output of this function is reported in Figure 2.

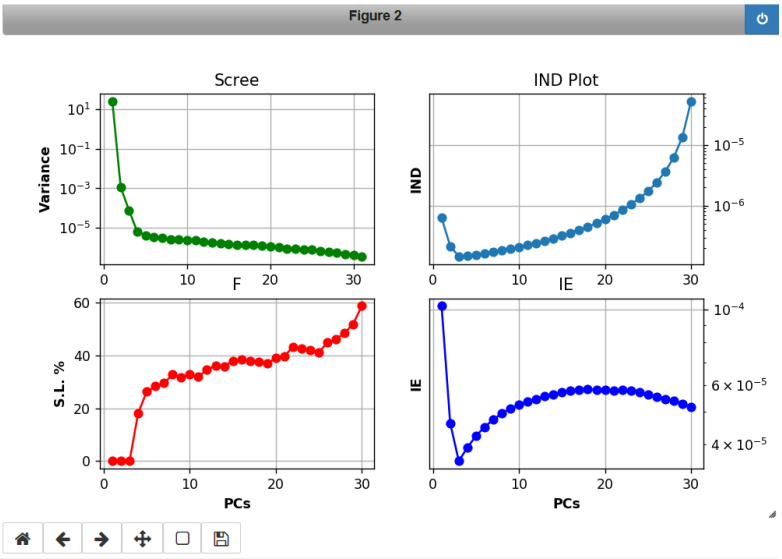

Figure 2: Graphical representation of the statistical parameters used to identify the correct (signal-related) number of PCs. All four test show a number of signal-related PCs equal to three.

A detailed description about the usage and the interpretation of each of these parameters can be found in books [2, 3] and in Section 2.2 of the reference text for PyFitIt [5]. In the following, only a brief and practical description is given for each of them.

- **Scree Plot:** In the scree plot each variance value contained in the **`l`** array (i.e. the eigenvalues of the data covariance matrix) is plotted *vs* its related PC. Because of noise, non-signal related components will be characterized by similar values of variance. For this reason, these PCs will be localized around a common flat line. This implies that the correct number of signal related PCs must be chosen as the elbow point of the graph.
- **IE** and **IND:** The imbedded error (IE) function and the Factor indicator function (IND) are two an empirical function developed to identify those PCs related to noise without relying upon an estimate of the error associated with the data matrix. For both of them the correct number of physical/chemical PCs must be identified in their minimum value. It is worth to mention that the IND function usually seems to be more sensitive than IE to identify the true dimensionality of the dataset [2-4].
- **F-Test:** The decision of what components correspond to the noise and what are the signal-related PCs can be made also on the basis of the Fisher test of the variance associated j^{th} PC and the summed variances associated with noise components. The j^{th} PC is accepted as a signal-related component is if the percentage of significance level (%SL) for the F-test is lower than some test level, generally 5%.

Finally, function **`recommendPCnumber(statistic)`** returns, see Figure 3, the minimum value of the IND function (more precise then IE, as told before) and the number of PCs having a %SL located under a test level fixed to 5% (generally used).

```
Min IND value:  1.46739406113589e-07
Number of PCs suggested by IND-factor:  3 PC

Highest SL(%) < 5%:  0.009520856046000024
Number of PCs suggested by F-Test:  3 PC
```

Figure 3: Output of the **`recommendPCnumber`** function.

2.3. Qualitative estimation of the exact number of PCs

The qualitative analysis of the PCs extracted by SVD can help to identify the correct number of components related to physical variation of signals. It is worth noting that the number of point characterising each PC is equal to the number of points of the energy column. This fact implies that each PC can be plotted *vs* the column energy and interpreted as an *abstract spectrum* without any chemical/physical meaning. The function **`plotPCAcomponents(energy, principal_components)`** allows to plot each abstract component and test if or not it takes account for the noise contribution.

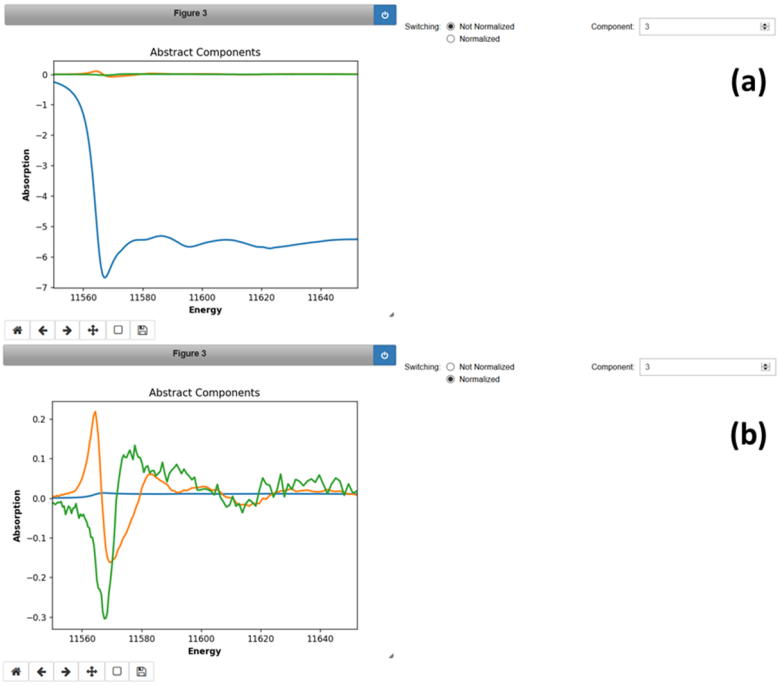

Figure 4: Abstract Components plot. (a) Not normalized components. (b) Normalized components.

Figure 4(a,b) shows the output of this function. The *Component* arrows allow to increment or reduce the number of components visualised while the *Switching* radio button enables to visualise them in a normalised form (each abstract spectrum is divided by the area) magnifying their noisy features (see for an example)

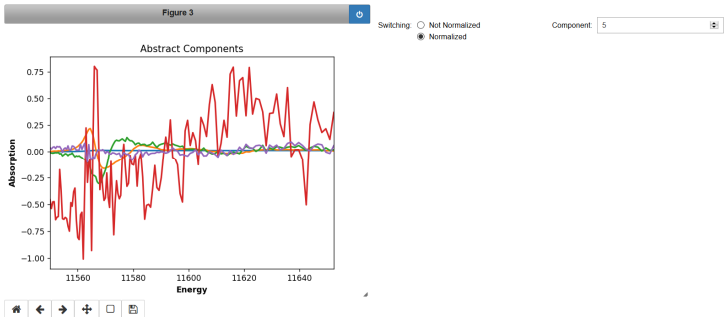

Figure 5: Plot of the first five normalized abstract components applying the normalization condition. While for the third abstract component some physical signal features are present in the range between 11560-11580 eV (see Figure 4(a,b)), the forth and the fifth components are completely dominated by noise and they can not be considered physical.

Finally, the **`PCAcomparison(energy,data)`** function can be used to identify the proper number of PCs. It creates the plot showed in Figure 6(a,b).

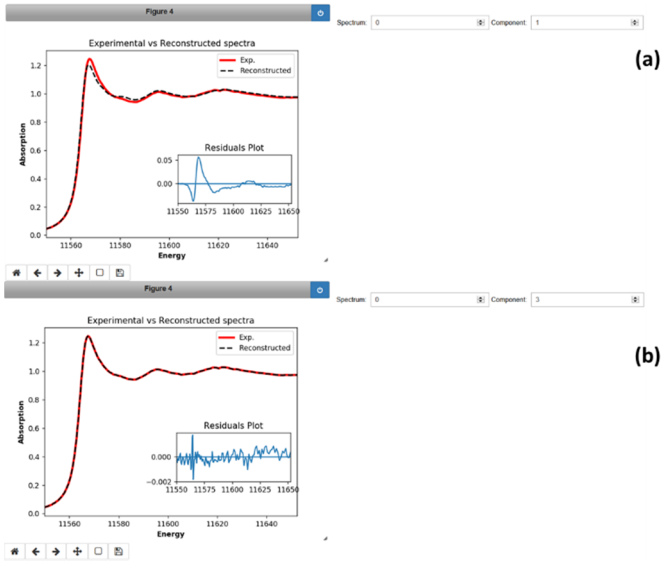

Figure 6: (a) Reconstruction of the first spectrum of the dataset using 1 PC. How it is possible to see the residual plot shows traces of signal features between 11555 and 11580 eV. (b) Reconstruction of the same spectrum using 3 PCs. In this case the residual plot is characterized only by noise features.

User can select a determined spectrum in the dataset and then, by means of command *Component,* he can reconstruct the spectrum increasing progressively the number of components. The inset in the main plot shows the residual plot. The correct number of PCs must correspond to the minimum number of components able to report in the residual plot only noise features, as represented in Figure 6(b).

3. Spectral decomposition of the experimental dataset

3.1. Data pre-treatment

In order to perform a correct decomposition of the input dataset in a reduced set of *pure* independent spectra, each experimental spectrum $\boldsymbol{\mu}_i$ must be normalised using the following scale factor:

$$\sigma_i = \sqrt{\frac{1}{(E_{\mathrm{ma}}^{\mathrm{x}} - E_{\mathrm{mi}}^{\mathrm{n}})} \int_{E_{\mathrm{mi}}^{\mathrm{n}}}^{E_{\mathrm{ma}}^{\mathrm{x}}} dE[\boldsymbol{\mu}_i(E)]^2}$$

Where $E_{\mathrm{mi}}^{\mathrm{n}}$ and $E_{\mathrm{ma}}^{\mathrm{x}}$ are the minimum and maximum values of the energy range where spectra are defined. This step is realised by the command **`normalization(energy,data)`** and it is mandatory. It can be realised at the beginning of the analysis (i.e. before the statistical analysis, moving the command in the first cells before the **`plot_data`** function) or just before the *Target Transformation* procedure. It is worth

noting that higher will be the number of points characterizing each experimental spectrum, more accurate will be the normalization result. For this reason, before this step, user can interpolate the experimental spectra with a finer sampling interval changing the **`step`** parameter in the function: **`interpolation(energy,data,step=0.05)`**. The new energy values can be saved through the command: **`np.savetxt('Energy.dat',energy)`**. Clearly, as for the normalization function, also this command can be moved in the upper cells.

3.2. The Target Transformation Function

The **`targetTransformationPCA`** function allow to retrieve, from the experimental dataset, a set of pure spectra and their related concentration profiles having a well defined physical/chemical meaning. This technique foresees the usage of a transformation matrix whose elements can be directly modified by user moving some sliders, see Figure 7 . Once that the number of PCs has been identified, two working configuration are available.

- **Case: 1:** The "pure" spectral profiles are not normalized. Herein, user can choose between four options:
  i. No Constraints: No constraints are fixed.
  ii. 1st spectrum fixed: The first column of the transformation matrix is defined in order to fix as a *pure spectrum* the first experimental spectrum in the dataset.
  iii. Last Spectrum fixed: The last column of the transformation matrix is defined in order to fix as a *pure spectrum* the last experimental spectrum in the dataset.
  iv. 1st and Last spectrum fixed: The firs and the last columns of the transformation matrix are defined in order to fix as a *pure spectra* the first and the last experimental spectrum in the dataset.
- **Case 2:** The Normalization is imposed as a constraint. For a detailed description of this kind of constrain, user can refer to the Supporting information of [5]. As for Case 1, user can require that the first or the last (or both) experimental spectrum/a in the input dataset could be considered as a *pure spectrum/a*. The radio buttons in this case assume the following name:
  i. Normalization: only the normalization constraint is fixed.
  ii. Norm. and 1st spectrum: normalization of the pure spectral profiles is fixed together with the request that the first spectrum is considered *pure*.
  iii. Norm. and last spectrum: normalization of the pure spectral profiles is fixed together with the request that the last spectra is considered *pure*.
  iv. Norm., 1st and last spectrum: normalization of the pure spectral profiles is fixed together with the request that the first and last spectra are considered *pure*.

Clearly if user wants to fix as a *pure spectrum* any other spectrum in the dataset, he can put it at the beginning or at the end of the data matrix and execute the function.

Keeping account of all this variety of constraints, the number of sliders that can be accessible by users are given by the following equation:

$$N_{\text{sliders}} = \begin{cases} \mathrm{N}^2 - \mathrm{N}: \text{ Normalization imposed;} \\ \mathrm{N}^2 - 2\mathrm{N} + 1: \text{ Normalization imposed and first/ last spectrum fixed;} \\ \mathrm{N}^2 - 3\mathrm{N} + 2: \text{ Normalization imposed and both first and last spectrum fixed;} \end{cases}$$

The **`targetTransformationPCA`** functions reads the following inputs: **`(energy,data,sign=-1,min_val=-5,max_val=5,step_val=0.05)`**. The min_val and max_val provide the range of variation of each slider, while step_val the related step of variation. Finally the **`sign=-1`** prevents the reversal of the retreived *pure components* due to the SVD algorithm in presence of the normalisation constraint. In case those spectra are affected by this problem, user must change the **`sign`** value from -1 to +1.

Finally, it is worth to mention that this technique foresees the usage of a transformation square matrix. When the function is called for the first time, the output is a blank plot near a null matrix of dimension N^2. When user starts to move the matrix elements by sliders or impose some constraints plots appear. However, it is worth noting that when transformation matrix is a singular matrix the Penrose pseudoinverse approach is applied allowing an approximate solution of the problem. Any time that a singular matrix is encountered the matrix rank is always lower than the number of PCs chosen.

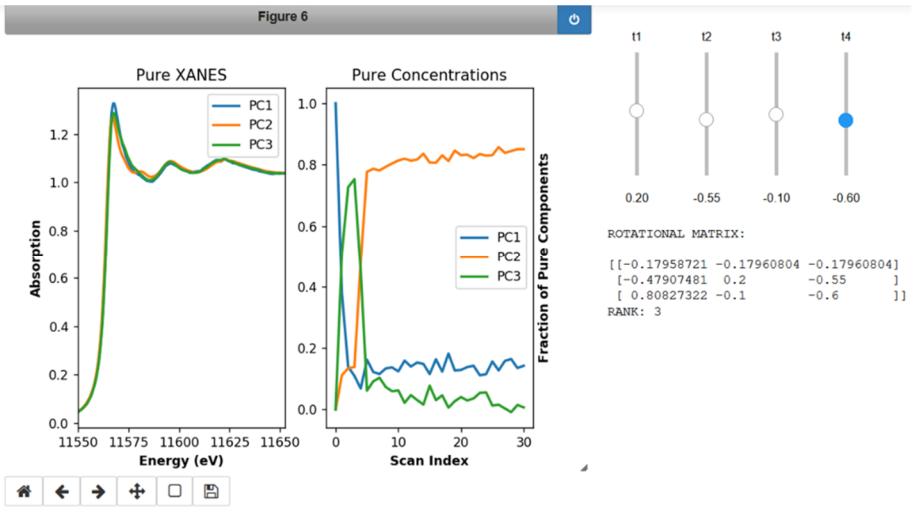

Figure 7: Pure Spectral and Concentration profiles obtained modifying by sliders the transformation matrix showed on the right.

4. Saving Data and Images

The pure spectra and their concentration values are saved, in the default folder *results* in *PCA*, by means of the following commands:

- **`saveToFile('results/params.txt', pcaResult.params)`**: saves the elements of the transformation matrix.
- **`saveToFile('results/Pure Spectra.txt', pcaResult.pureSpectra)`**: saves the pure spectral profiles.
- **`saveToFile('results/Pure Concentrations.txt', pcaResult.pureConcentrations):`** saves the pure concentration profiles.
- **`pcaResult.fig.savefig('results/image.png', dpi=200)`**: saves the plot produced by the **`targetTransformationPCA`** function.

Bibliography:

[1] B. Ravel, M. Newville, ATHENA, ARTEMIS, HEPHAESTUS: data analysis for X-ray absorption spectroscopy using IFEFFIT, J. Synchrot. Radiat., 12 (2005) 537-541.
[2] R.G. Brereton, Chemometrics: data analysis for the laboratory and chemical plant, John Wiley & Sons, 2003.
[3] E.R. Malinowski, Factor analysis in chemistry, Wiley, 2002.
[4] E.R. Malinowski, Determination of the number of factors and the experimental error in a data matrix, Anal. Chem., 49 (1977) 612-617.
[5] A. Martini et at., PyFitit: the software for quantitative analysis of XANES spectra using machine-learning algorithms, Submitted in Computer Physics Communications.