arPLS-vignette

Introduction

In the domain of (Raman) Spectroscopy it is a common problem and nuisance that the measured signal is a combination of "chemical information", a (background) baseline and random noise (Zhang, Chen, and Liang (2010)). The so called baseline signal appears in the measurement as a smooth curve over the measured spectrum and it is important to estimate it and correct for it to allow e.g. for the correct integration of the part of the measured signal that truly belongs to chemical characteristics of the sample of interest.

A sample of a raw Raman spectrum (for the mineral Abelsonite) is shown in the below picture. The data was taken from Lafuente et al. (2016)

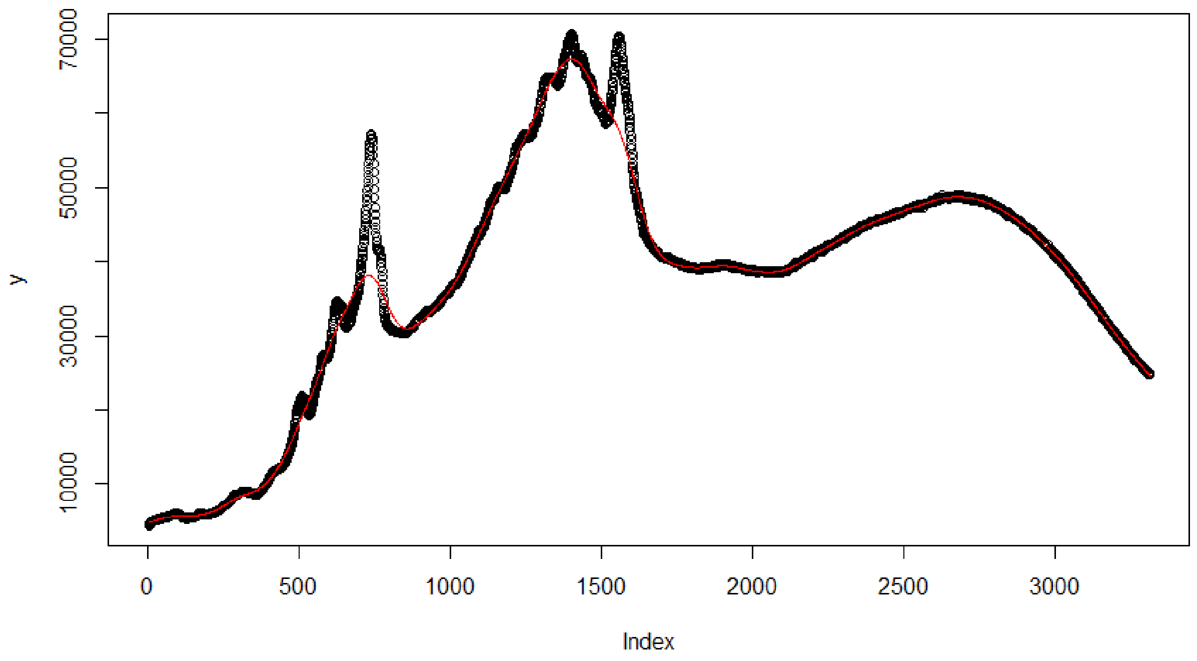


Figure 1: Raw raman spectrum of Abelsonite

The package at hand implements a semi-automatic baseline correction using asymmetrically reweighted penalized least squares smoothing as described in Baek et al. (2015). This algorithm estimates a baseline by trying to solve a multi criteria optimisation problem. It tries to find a line that satisfies (to various degrees the following criteria):

- closely follow the original measurements
- don't have much curvature ("wigglyness")
- be located below the original measurements

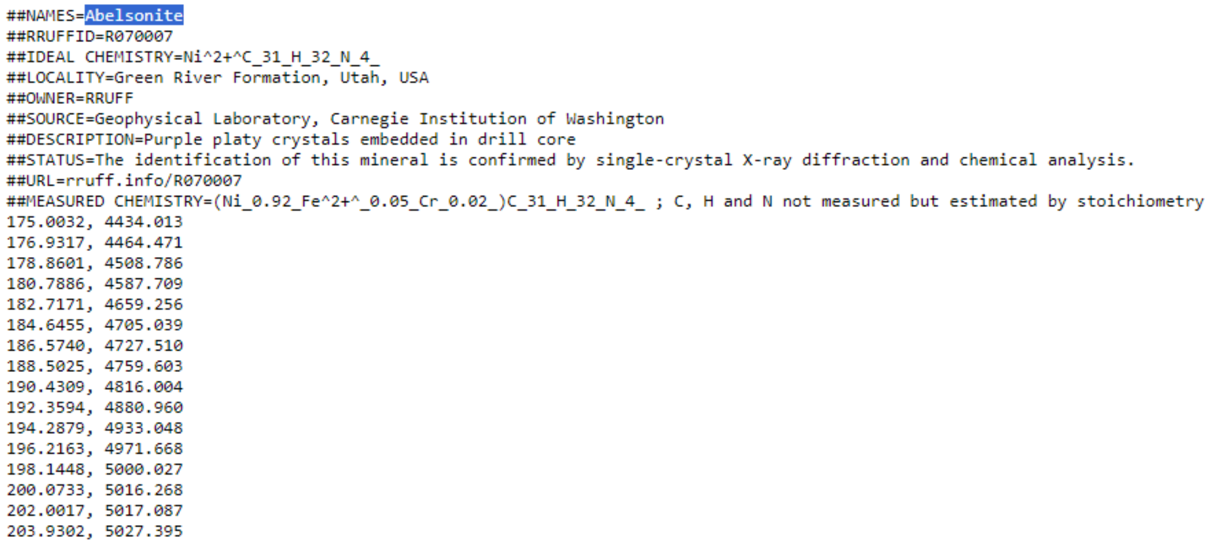
To balance 1. and 2. a user provided smoothness parameter λ is taken into account. To satisfy 3., the algorithms uses a list of weights that put the estimation emphasis (error minimisation) of any given iteration on areas where the baseline is over or close to the original signal. While areas with a large difference between baseline and original measurement are assumed to be due to a valid signal peak in that area. The algorithm will assign low weights for the optimisation efforts to those areas as it would be counter productive to try and force the baseline to mirror any actual signal peak, given that the whole point of a baseline estimation is to tell the background noise apart from the valid signal.

Usage

In terms of usage of the package, the end user will need to provide an input vector of a measured spectrum, a smoothness parameter λ to control the allowed curvieness of the baseline estimate and a stopping criterion in terms of maximum number of iterations and/or percentage changes of the internal weight parameters from one iteration to the next.

Input Data

The main function of the package is baseline_estimation() and it requires as input data a vector of positive integers or floats representing the measurements across a range of wavelengths or so called wavenumbers[1]. A sample of such data is e.g. the raw spectrum measured for Abelsonite[2] that can be obtained from the RRUFF database (Lafuente et al. (2016)). The first couple of rows of this spectrum are depicted in Figure below.



```
##NAMES=Abelsonite
##RRUFFID=R070007
##IDEAL CHEMISTRY=Ni^2+^C_31_H_32_N_4_
##LOCALITY=Green River Formation, Utah, USA
##OWNER=RRUFF
##SOURCE=Geophysical Laboratory, Carnegie Institution of Washington
##DESCRIPTION=Purple platy crystals embedded in drill core
##STATUS=The identification of this mineral is confirmed by single-crystal X-ray diffraction and chemical analysis.
##URL=rruff.info/R070007
##MEASURED CHEMISTRY=(Ni_0.92_Fe^2+^_0.05_Cr_0.02_)C_31_H_32_N_4_ ; C, H and N not measured but estimated by stoichiometry
175.0032, 4434.013
176.9317, 4464.471
178.8601, 4508.786
180.7886, 4587.709
182.7171, 4659.256
184.6455, 4705.039
186.5740, 4727.510
188.5025, 4759.603
190.4309, 4816.004
192.3594, 4880.960
194.2879, 4933.048
196.2163, 4971.668
198.1448, 5000.027
200.0733, 5016.268
202.0017, 5017.087
203.9302, 5027.395
```

It is left to the user to only feed the column with the measurements to the baseline estimation function. The algorithm performs reasonably fast on common consumer laptops for a couple of thousand rows, but slows down considerably for much larger spectra (of e.g. 10k rows). Further input data required from the user are the parameter values for lambda (the smoothness parameter), ratio (one of the stopping criteria based on conversion of the results) and maximum number of iterations (a fall back stopping criterion in case the algorithm doesn't convert). For all of these parameters the estimation function will provide default values taken from p. 253 of Baek et al. (2015).

The function has a verbose mode that will render the current estimate of the baseline and print some diagnostic information to the console during the estimation process. This might be particularly useful for large spectra that can take a long time. The console output can serve as some kind of progress indication.

[1] Raman spectra are often recorded across socalled wavenumbers with a unit of measurement that is the inverse wavelength expressed in cm, so 1/cm

[2] https://rruff.info/repository/sample_child_record_raman_full/by_minerals/Abelsonite___R070007___Broad_Scan___532___0___unoriented___Raman_Data_RAW___13756.txt

As part of the package the spectrum of Abelsonite from Lafuente et al. (2016) is provided and can be loaded as shown below.

```
library(baselineARPLss)
data("Abelsonite")
head(Abelsonite,10)
```

```
##    wavenumber. measurement
## 1    175.0032,    4434.013
## 2    176.9317,    4464.471
## 3    178.8601,    4508.786
## 4    180.7886,    4587.709
## 5    182.7171,    4659.256
## 6    184.6455,    4705.039
## 7    186.5740,    4727.510
## 8    188.5025,    4759.603
## 9    190.4309,    4816.004
## 10   192.3594,    4880.960
```

Function call

With such input data at hand one of the most basic calls of the baseline_estimation function would be as shown below

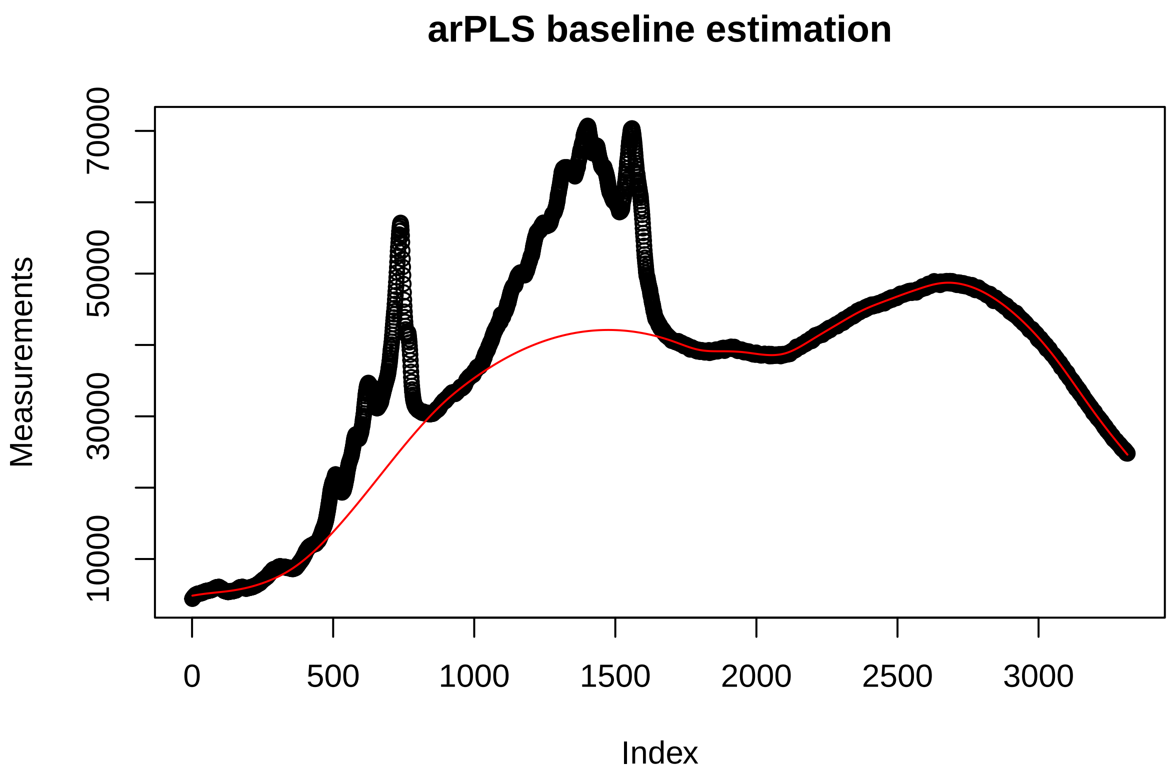
```
result<-baseline_estimation(Abelsonite$measurement,cpp=FALSE)
```

The package will use an S3 object of the class arPLSresult to store the results of the baseline estimation. The data structure behind this object is a named list with input data, result and parameter values.
The rational is that similarly to the result object of a lm-regression is would be important or at least useful to have contextual information beside the end result. This additional information includes the parameter values with which the algorithm was run:

- raw spectrum that was used as input data
- smoothness parameter lambda that the user provided
- ratio parameter provided by the user as one stopping criterium
- maximum iteration parameter provided by the user as an additional stopping criterium
- estimated baseline after the algorithm stopped
- last iteration executed when the algorithm stopped
- last weight vector ration at the time when the algorithm stopped

The last two pieces of information allow to determine if the algorithm stopped because it converged or because it ran out of iteration. Given that the result is a custom data structure there is a summarize and plot function for this class. The custom plot function will plot the original raw spectrum with the estimated baseline overlaid. The usage is shown below.

```
plot(result)
```



The custom summary function will print the control parameter with which the algorithm was called as well as the reason why it stopped (e.g. conversion vs. maximum iterations). The call of this function is demonstrated below.

```
summary(result)
```

```
## [1] "The lambda parmeter value used was:  1e+06"
## [1] "The ratio parameter value used was:  1e-06"
## [1] "The max_iter parameter value used was:  50"
## [1] "The alogrithm stopped after the following number or iterations:  50"
## [1] "The last weight ratio value was:  0.000153623596796175"
## [1] "It appears that the algorithm stopped because the maximum number of iterations was reached"
```

Baek, Sung-June, Aaron Park, Young-Jin Ahn, and Jaebum Choo. 2015. "Baseline Correction Using Asymmetrically Reweighted Penalized Least Squares Smoothing." *Analyst* 140: 250–57.

Lafuente, Barbara, R. T. Downs, H. Yang, and N. Stone. 2016. "1. The Power of Databases: The RRUFF Project." In *Highlights in Mineralogical Crystallography*, edited by Thomas Armbruster and Rosa Micaela Danisi, 1–30. Berlin, München, Boston: De Gruyter (O).

Zhang, Zhi-Min, Shan Chen, and Yi-Zeng Liang. 2010. "Baseline Correction Using Adaptive Iteratively Reweighted Penalized Least Squares." *Analyst* 135: 1138–46.