SPECTRES: A FAST SPECTRAL RESAMPLING TOOL IN PYTHON

A. C. CARNALL - adamc@roe.ac.uk

Institute for Astronomy, University of Edinburgh, Royal Observatory, Edinburgh EH9 3HJ

ABSTRACT

I present a fast Python tool, SpectRes, for carrying out the resampling of spectral flux densities and their associated uncertainties onto different wavelength grids. The function works with any grid of wavelength values, including non-uniform sampling, and preserves the integrated flux. This may be of use for binning data to increase the signal to noise ratio, obtaining synthetic photometry, or resampling model spectra to match the sampling of observed data for spectral energy distribution fitting. The function can be downloaded from https://www.github.com/ACCarnall/SpectRes.

1 THE SPECTRES CODE

SpectRes is a Python function which efficiently resamples spectra onto an arbitrary wavelength grid using the method described in Sections 2, 3 and 4. To illustrate the speed of the code, a standardised file from Bruzual & Charlot (2003) containing 221 models each with 6917 wavelength points is resampled onto a uniform 5 Å grid between 1000 Å and 11000 Å, as might be typical for spectral fitting, in a time of ~ 40ms. An example result is shown in Figure 2. Resampling of a single model onto a uniform 20 Å grid between 3000 Å and 50000 Å, as might be typical for photometric fitting, takes ~ 10ms. This makes the code highly applicable to, for example, rapid calculation of photometric redshifts. Figure 3 shows the rebinning of a high redshift quasar spectrum from Carnall et al. (2015) in order to improve the signal to noise ratio.

The code is available for download from https://www.github.com/ACCarnall/SpectRes. The function takes three arguments, firstly spec_wavs, an array of wavelengths corresponding to the current sampling of the spectrum. Secondly spec_fluxes, a 1D or 2D array of flux values with the first axis running over wavelength and the second (if present) over different spectra to be resampled. The third argument, resampling, is the desired sampling. A keyword argument, spec_errs may also be passed of the same shape as spec_fluxes containing the uncertainty on each flux value. The function returns an array, resampled with first dimension the same length as resampling and second dimension (if present) the same length as the second dimension of spec_fluxes. If the keyword argument spec_errs is passed, a second array, resampled_errs of the same shape, containing the resampled error spectra is also returned.

2 PROBLEM

One dimensional astronomical spectra normally take the form of a set of wavelength values \( \lambda_i \) (i.e. 1, 2, ..., n) with associated flux per unit wavelength values \( f_{\lambda i} \), or alternatively frequency values \( v_i \) with associated flux per unit frequency values \( f_{v i} \), along with uncertainties \( \sigma_i \). For the remainder of this article we will assume a grid of wavelength values and associated flux per unit wavelength values, however the function presented should be equally applicable to either case.

The units of \( f_{\lambda i} \) are normally of the form energy per unit time per unit of collecting area per unit wavelength (e.g. erg/s/cm\(^2\)/Å). Each \( f_{\lambda i} \) value can be thought of as the average flux per unit wavelength value across the interval

\[
\left( \frac{\lambda_{i-1} + \lambda_i}{2}, \frac{\lambda_i + \lambda_{i+1}}{2} \right)
\]

(1)
We therefore define \( w_i \) as the width of each of these intervals in wavelength space. The subject of this article is to define another set of values \( f_{\lambda_j} \) and uncertainties \( \sigma_j \) with different associated wavelength values \( \lambda_j \) (\( j \in 1, 2, \ldots, m \)). The described configuration is shown graphically in Figure 1.

### 3 Calculation of Flux Values

The first step in the calculation of the new bin flux values \( f_{\lambda_j} \) is to obtain a matrix of \( P_{ij} \) values, which are the fraction of the wavelength range covered by each initial bin \( i \) which also falls under the final bin \( j \). A simple, yet computationally intensive method for doing this would be to calculate

\[
(P_{ij} = \begin{cases} 
(L_{i+1} - L_i) & \text{if } L_j > L_i \text{ and } L_{i+1} > L_{i+1} \\
(L_{j+1} - L_j) & \text{if } L_j < L_i \text{ and } L_{j+1} < L_{i+1} \\
(L_{j+1} - L_j) & \text{if } L_j > L_i \text{ and } L_{j+1} < L_{i+1} \\
(L_{i+1} - L_i) & \text{if } L_j < L_i \text{ and } L_{i+1} > L_{i+1} 
\end{cases})
\]

where \( L_{i,j} = (\lambda_{i-1,j-1} + \lambda_{i,j})/2 \) is the left hand edge of each bin, and any \( P_{ij} \) value of less than zero should subsequently be set to zero. The method implemented in SpectRes is much more highly optimised, but more convoluted, so a discussion is not presented here. Interested parties should refer to the code.

Once the \( P_{ij} \) values have been obtained, the values of the \( f_{\lambda_j} \) can be obtained by performing the weighted sum

\[
f_{\lambda_j} = \frac{\sum_{i=1}^{n}(P_{ij}w_i f_{\lambda_i})}{\sum_{i=1}^{n}(P_{ij}w_i)} = \sum_{i=1}^{n} c_{ij} f_{\lambda_i}.
\]

### 4 Calculation of Associated Uncertainties

The variances on the \( f_{\lambda_j} \) values calculated using Equation 3 can be obtained using standard uncertainty propagation formulae. The variance \( \sigma_j^2 \) on each \( f_{\lambda_j} \) is given, assuming all \( f_{\lambda_i} \) are independent, by

\[
\sigma_j^2 = \frac{\sum_{i=1}^{n} p_{ij}^2 w_i^2 \sigma_i^2}{(\sum_{i=1}^{n} P_{ij}w_i)^2} = \sum_{i=1}^{n} c_{ij}^2 \sigma_i^2.
\]
Figure 2: Spectral resampling of a model of a 1 Gyr old burst of star formation from Bruzual & Charlot (2003) with SpectRes. The original model is shown in blue, the red model has been resampled onto a uniform 5 Å grid and the green model onto a uniform 10 Å grid. Vertical offsets have been applied for clarity.

It is important to recognise that in any case where adjacent flux values \( f_{\lambda_i} \) have contributions from the same \( \lambda_i \), those flux values will be covariant with each other to some extent. To see this, consider the case in which one starts with a uniformly spaced set of \( \lambda_i \), and a uniform error spectrum \( \sigma_i = \sigma \), and offsets each bin by half of its width so that \( \lambda_j = (\lambda_{i+1} + \lambda_i)/2 \). After resampling we now have the same number of points in our spectrum, but each uncertainty \( \sigma_j = \sigma/\sqrt{2} \). Clearly the new flux values must be covariant, otherwise it would be possible to continue to reduce the errors by this process without impacting on the data quality. We now derive the covariance matrix for the new bins.

5 THE COVARIANCE MATRIX

The covariance matrix is constructed of elements \( \sigma_{jb} \), between the new flux values \( f_{\lambda_j} = \sum_{i=1}^{n} c_{ij} f_{\lambda_i} \) and \( f_{\lambda_b} = \sum_{a=1}^{n} c_{ab} f_{\lambda_a} \), where \( (j, b \in 1, 2, ..., m) \) and \( (i, a \in 1, 2, ..., n) \).

To obtain the \( \sigma_{jb} \) values we first express the flux values in the old and new bins as Gaussian random variables \( X \) and \( Y \) respectively, e.g. \( Y_j \sim N(f_{\lambda_j}, \sigma^2_j) \). We can express the random variables representing the new bins in terms of the old bins using the standard result from probability theory that

\[
Y_j = \sum_{i=1}^{n} c_{ij} X_i \sim N(f_{\lambda_j}, \sigma^2_j) = N\left( \sum_{i=1}^{n} c_{ij} f_{\lambda_i}, \sum_{i=1}^{n} c_{ij}^2 \sigma^2_i \right),
\]

which can be seen to be in agreement with our Equations 3 and 4. Now the covariance between the two random variables, \( Y_j \) and \( Y_b \) is defined as

\[
\text{cov}(Y_j, Y_b) = \sigma^2_{jb} = E\left[ (Y_j - E[Y_j])(Y_b - E[Y_b]) \right]
\]

where \( E[X] \) is the expectation value of \( X \). This can be expressed using Equation 5 as

\[
\sigma^2_{jb} = E\left[ \left( \sum_{i=1}^{n} c_{ij} X_i - E\left[ \sum_{i=1}^{n} c_{ij} f_{\lambda_i} \right] \right) \left( \sum_{a=1}^{n} c_{ab} X_a - E\left[ \sum_{a=1}^{n} c_{ab} f_{\lambda_a} \right] \right) \right].
\]
The distribution of \( (X - E[X_i]) \) is a Gaussian with mean of zero and variance of \( \sigma^2_i \), i.e. \( N(0, \sigma^2_i) \), meaning that Equation 7 can be written as

\[
\sigma_{jb}^2 = E\left[ \left( \sum_{i=1}^{n} c_{ij} N(0, \sigma^2_i) \right) \left( \sum_{a=1}^{m} c_{ab} N(0, \sigma^2_a) \right) \right].
\] (8)

Multiplying out the brackets and taking the expectation value will yield zero for all \( i \neq a \), as \( E[N(0, \sigma^2) N(0, \sigma^2)] = 0 \). We can therefore simplify Equation 8 to

\[
\sigma_{jb}^2 = E\left[ \sum_{i=1}^{n} c_{ij} c_{ib} N(0, \sigma^2_i)^2 \right] = \sum_{i=1}^{n} c_{ij} c_{ib} E[N(0, \sigma^2_i)^2].
\] (9)

Finally we can use another standard result, that the variance, \( \text{Var}[X] = E[X^2] - E[X]^2 \) to obtain

\[
\sigma_{jb}^2 = \sum_{i=1}^{n} c_{ij} c_{ib} \sigma^2_i,
\] (10)

which agrees with the result of Equation 4 in the case where \( j = b \) as expected. In this way the covariance matrix can be calculated, however where possible it is advisable to resample the model rather than observed spectrum in order to avoid dealing with covariances when performing spectral energy distribution fitting.

**ACKNOWLEDGEMENTS**

I would like to thank Raphael Errani, David Homan, Joseph Kennedy, Fran Lane, Andy Lawrence, Ross McLure and Tom Shanks for helpful discussions. The LaTeX template used was downloaded from http://www.LaTeXTemplates.com.

**REFERENCES**

Bruzual G., Charlot S., 2003, mnras, 344, 1000
Carnall A. C., et al., 2015, mnras, 451, L16