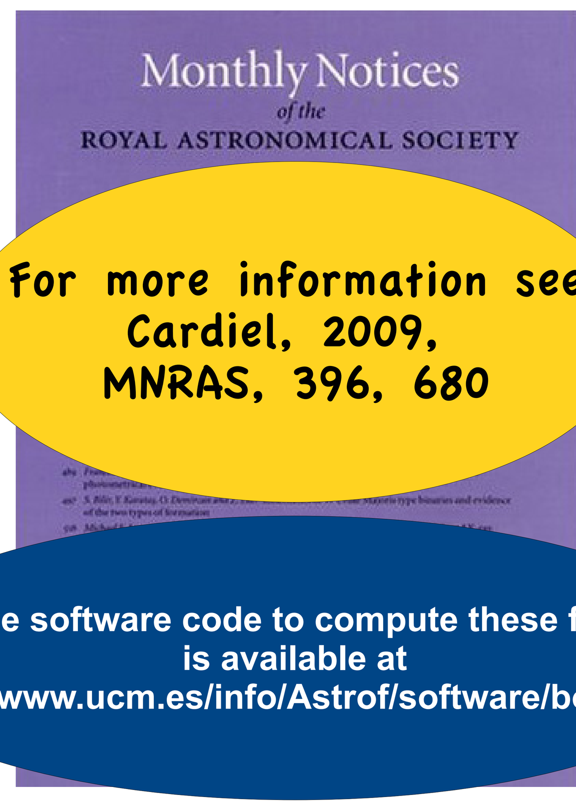
So, you do not need a fit through the data points, but to obtain the boundaries of your data, do you?

Nicolás Cardiel

Departamento de Astrofísica y CC. de la Atmósfera
Facultad de CC. Físicas, Universidad Complutense de Madrid
Avda. Complutense s/n, 28040-Madrid, Spain
e-mail: cardiel@fis.ucm.es



For more information see Cardiel, 2009, MNRAS, 396, 680

The software code to compute these fits is available at http://www.ucm.es/info/Astrof/software/boundfit

1) THE METHOD

In many astronomical problems one often needs to determine the upper and/or the lower boundary of a given data set. An automatic and fast approach consists in fitting the data using a **Modified Least Squares Method**, where the function to be minimized, χ^2, is defined to handle, asymmetrically, the data at both sides of the boundary. For example, in the case of a set of N points of coordinates (x_i, y_i), and considering uncertainties only in the Y axis (σ_i), that function can be written as

Eq. (1): $$\chi^2 = \sum_{i=1}^{N} \{ w_i \; | y(x_i) - y_i |^{\alpha} \},$$

boundary	w_i	condition
upper	$1/\sigma_i^{\beta}$	$y(x_i) \geq y_i$
	ξ/σ_i^{β}	$y(x_i) < y_i$
lower	ξ/σ_i^{β}	$y(x_i) \geq y_i$
	$1/\sigma_i^{\beta}$	$y(x_i) < y_i$

where α is an exponent (in normal Least Squares α=2), $y(x_i)$ is the fitted function evaluated at x_i, and w_i is an overall weighting factor that is responsible for introducing the asymmetry in the fit. This factor is computed as explained in the table: β is the exponent that determines whether the fit is error weighted or not (β=0 to ignore errors; typically β=α=2 for error weighted fits), and the asymmetry factor ξ must be greater than 1. This function can be easily minimized following a numerical strategy. The use of single polynomials can provide a good answer and it is computationally very simple. An example of this polynomial boundary fit is presented in Figure 1. However, when the data exhibit rapidly changing values, a single polynomial is not always able to reproduce the observed trend. A more powerful alternative in these cases consists in the use of **adaptive splines**, which exhibit a much larger flexibility.

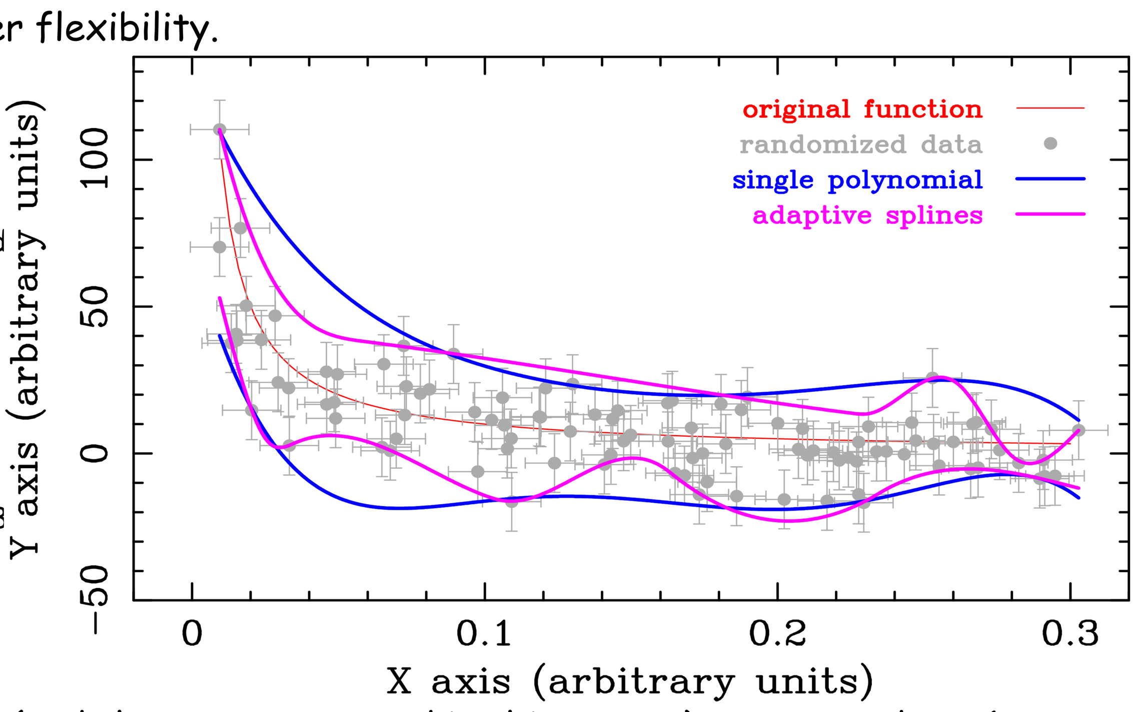


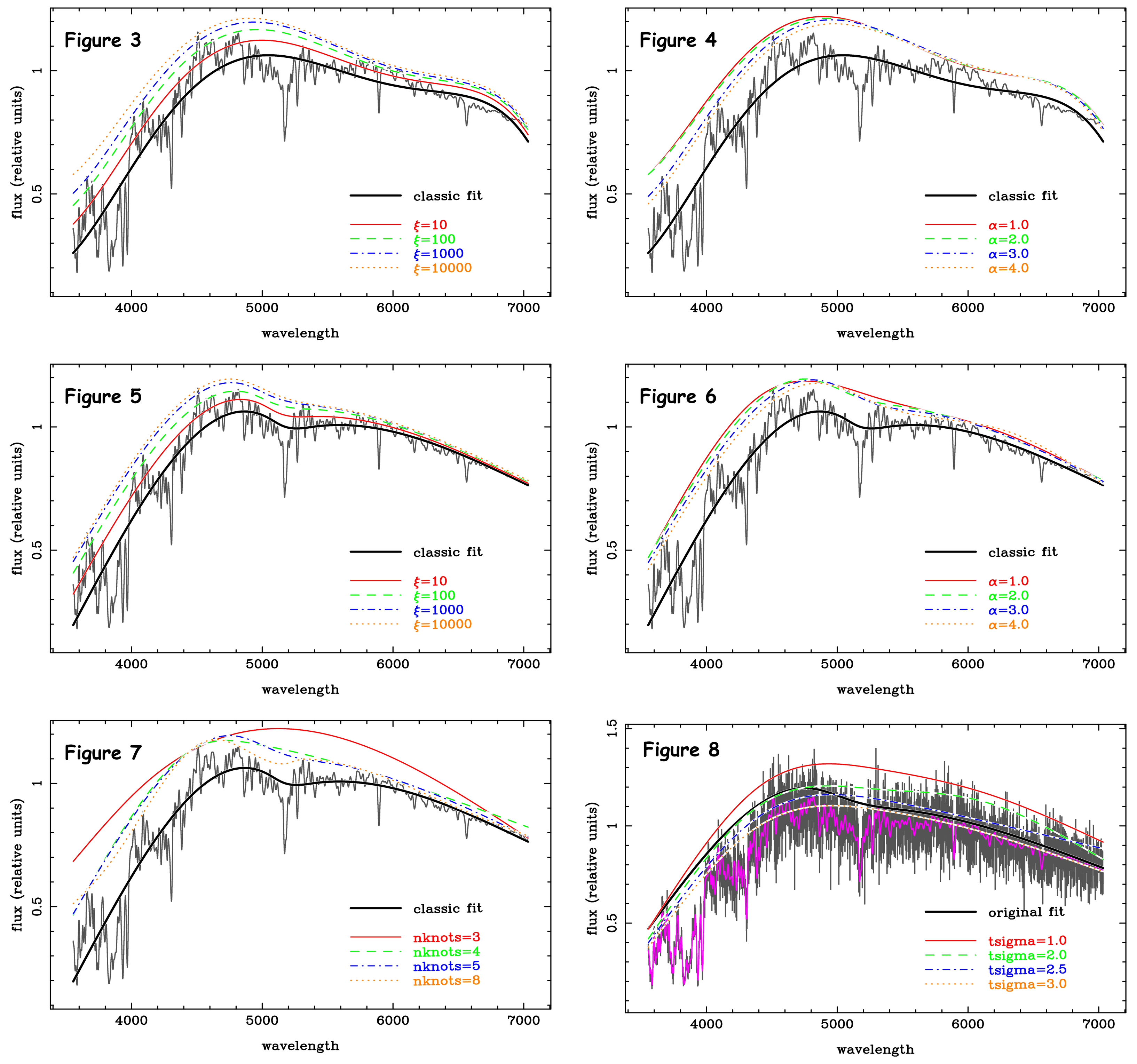
Figure 1: comparison between different strategies to determine data boundaries. The grey data correspond to 100 points randomly drawn from the function y=1/x (red line), assuming Δx=0.01 and Δy=10. The boundaries have been determined using simple polynomials (blue lines; upper boundary 5[th] degree; lower boundary 8[th] degree) and adaptive splines (magenta; upper boundary: 8 knots; lower boundary: 10 knots). As expected, splines are more flexible and provide tighter boundaries than polynomials.

Although the last example (and the rest presented in this poster) corresponds to 1-dimensional (1D) boundaries in 2D diagrams, the method can be applied to higher dimensions (e.g. the determination of a surface as boundary for data in a 3D-parameter space), once an appropriate metric (distance) is defined in the multidimensional space.

Full details about this work have appeared in Cardiel (2009, MNRAS, 396, 680).

2) AN APPLICATION TO REAL DATA

A common problem when handling spectroscopic data is the determination of a "reasonable" fit to the spectra continuum. Most of the times people are happy enough just by fitting a polynomial to the general trend of the spectra, masking disturbing spectroscopic features such as important emission lines or deep absorption characteristics. A good alternative is to obtain the upper boundary fit, either by using polynomials or adaptive splines, as explained before. In the next plots some examples of these fits to the continuum of the K0V star HD003651 are shown, in which the effects of modifying different relevant parameters (e.g., the asymmetry factor ξ, the exponent α, or the number of knots) are examined. For simplicity, the examples are not error weighted (i.e., β=0 has been assumed). Figures 3 and 4 correspond to single polynomials, whereas Figures 5, 6, 7 and 8 represent the fits to adaptive splines.



3) ADAPTIVE SPLINES

Splines are commonly employed for interpolation and modeling of arbitrary functions. Many times they are preferred to simple polynomials due to their flexibility. A spline is a piecewise polynomial function that is locally very simple, typically third-order polynomials (the so called cubic splines). These local polynomials are forced to pass through a prefixed number of points, which we will refer as knots. The coefficients of these polynomials are easily computed by imposing in addition that the first and second derivatives match at the knots (two additional conditions are required; normally they are provided by assuming that the second derivatives at the two endpoints to be zero, leading to what are normally called "natural splines"). The computation of splines is widely described in the literature (see e.g. Gerald C.F., Wheatley P.O., 1989, in Applied Numerical Analysis, 4[th] edition).

The final result of a fit to splines will strongly depend on both, the number and the precise location of the knots. In order to provide more flexibility in the fit, Cardiel N., (1999, PhD Thesis, Universidad Complutense de Madrid) explored the possibility of setting the location of the knots as free parameters, and determine the optimal coordinates of these knots that improve the fit to the data. The solution to the problem can be derived numerically using any minimization algorithm. Here we have used DOWNHILL (Nelder J.A., Mead R., 1965, Computer Journal 7, 308), which only requires to evaluate the function to be minimized (and not the derivatives), provided an initial guess to the solution is available. An implementation of the method can be found in Press et al. 1989 (Numerical Recipes in FORTRAN: The Art of Scientific Computing, 2[nd] edition).

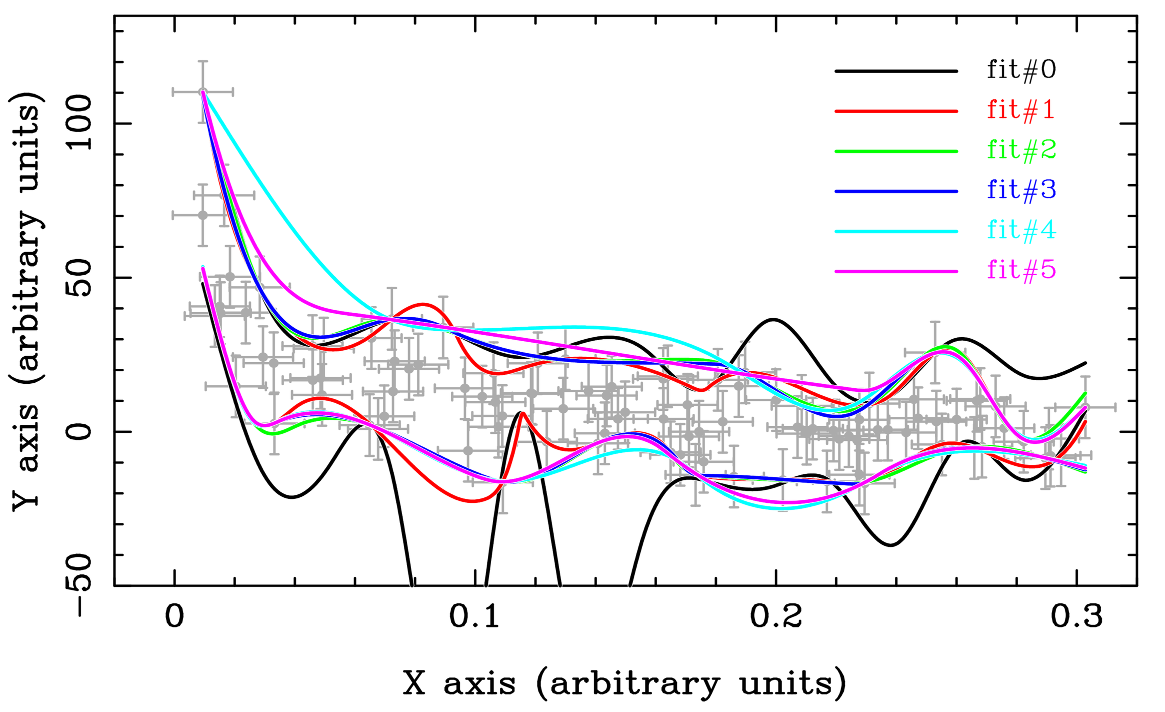


Figure 2: examples of boundary fitting to adaptive splines. The grey points correspond to the same data displayed in Figure 1.

Fit#0: initial fit using NKNOT=15 after step#2 (see Modus Operandi in column at the right).

Fit#1: after step#3, NITER=10.

Fit#2: after step#4, merging colliding knots (4 and 3 in the upper and lower boundaries).

Fit#3: after step#3, NITER=10.

Fit#4: after step#4, merging colliding knots (3 and 2 in the upper and lower boundaries).

Fit#5: after step#3, NITER=10.

Fit to Adaptive Splines: *Modus Operandi*

Step#1: Fix NKNOT, the initial number of knots to be employed. Using a high number provides more flexibility, although the number of parameters to be determined (the knot coordinates) logically scales with this number.

Step #2: Obtain an initial solution. For this purpose it is sufficient, for example, to start by dividing the full X range to be fitted by (NKNOT-1). This leads to a regular distribution of equidistant knots. The initial fit is then derived by minimizing the function given in Eq. (1), leaving as free parameters the Y coordinates of all the knots simultaneously.

Step#3: Refine the fit. Once some initial spline coefficients have been determined, the fit is refined by setting as free parameters the location of all the "inner" knots, both in the X and Y directions. The "outer" knots (the first and last knots) are only allowed to be refined in the Y-axis direction). The simultaneous minimization of both X and Y coordinates of all the knots at once will imply finding the minimum of multidimensional function with too many variables. This is normally something very difficult, with no guarantee of a fast convergence. In this work a different strategy has been adopted. The problem reveals to be treatable just by solving for the optimized coordinates of every single knot separately. In practice, an iteration has been defined as the process of refining the location of all the NKNOT knots (one at a time), where the order in which a given knot is optimized is determined randomly. Thus, at the end of every iteration all the knots have been refined once. An extra penalization has been introduced in the function to be minimized with the idea of avoiding that a knot exchange its order in the list of knots. This refinement typically implies that, if NKNOT is large, several knots end up colliding and having the same X-coordinate. The whole process can be repeated by indicating a given number of iterations, NITER.

Step#4: Optimize the number of knots. If, as the result of given number of iterations, several knots have "collided" and exhibit the same X-coordinate, this is an indication that NKNOT was probably too large. In this case, those colliding knots can be merged and the effective number of knots be accordingly reduced. With the new NKNOT, Step#3 is repeated again. If, on the contrary, the knots being used do not collide, it is interesting to check whether a higher NKNOT can be employed.

The process ends after a "satisfactory" fit is found at the end of Step#3. By "satisfactory" one can accept a fit that does not change by increasing NITER, and in which there are no "colliding" knots.