JOURNAL OF CHEMOMETRICS
J. Chemometrics 2005; **19**: 317–328
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cem.935

Unimodal smoothing

Paul H. C. Eilers*
Department of Medical Statistics, Leiden University Medical Centre, P.O. Box 9604, 2300 RC, Leiden, The Netherlands

Received 3 August 2004; Revised 1 August 2005; Accepted 2 August 2005

In many applications it is expected from theory that a signal should be non-negative and unimodal, that is, show only one peak. If the data are noisy, standard smoothing algorithms will not always give the desired result: peaks may be rounded and negative lobes may occur in the tails. Positive unimodal fits can be obtained by modeling the logarithm of a curve and combining a standard roughness penalty with a specialized asymmetric penalty. The theoretical basis and implementation in Matlab are presented, as well as performance on real and simulated data. Copyright © 2006 John Wiley & Sons, Ltd.

KEYWORDS: asymmetric penalty; log-concave; monotone; P-splines

1. INTRODUCTION

Smoothing is an established component of the chemometric toolbox. It is used to reduce noise, to estimate trends, and to interpolate (missing) data. Sometimes one knows, from theory or experience, that the signal we are recovering from noise has certain properties, like showing a monotone decrease or having only one peak. This paper discusses how to exploit both properties, but it concentrates on the latter, unimodality.

Standard smoothing algorithms often do not deal well with this type of data, especially in noisy circumstances. Several undesirable artifacts can occur: peaks might be broadened or flattened too much, or negative undershoot may occur. Two elements can be combined to get a better smoother for peaks: (1) do not model the signal itself, but its logarithm (which is forced to be smooth), and (2) impose a specialized penalty on its second derivative (which leads to a concave logarithm).

The logarithm is modeled as a sum of scaled B-splines and penalties are applied on the coefficients, in the spirit of P-splines [1]. The B-splines allow evaluation of the fitted curve and derivatives at any desired resolution, which is an advantage for presentation and analysis.

The algorithm is fundamentally different from that of Bro and Sidiropoulos [2]. They interpret a peak as two tails, the left one increasing monotonically, the right one decreasing monotonically. They estimate the split point from the data and use positive monotone regression for each tail separately. Smoothness is not considered. A comparison will be shown in the section on applications.

Gemperline and Cash [3] present a penalty approach to shape constraints. Their proposal is to search for regions where a shape constraint is violated, and to introduce a (local) penalty to (gently) force the solution in the right direction. Smoothness is not considered and a peak is split into monotone left and right parts.

*Correspondence to: P. H. C. Eilers, Department of Medical Statistics, Leiden University Medical Centre, P.O. Box 9604, 2300 RC, Leiden, The Netherlands. E-mail: p.eilers@lumc.nl

The paper is structured as follows. Section 2 gives a short introduction to P-splines, and Section 3 introduces asymmetric penalties. Section 4 combines these components into a log-linear unimodal smoother. Applications are presented in Section 5. The last section contains a short discussion.

2. SMOOTHING WITH P-SPLINES

A B-spline is a curve constructed from (smoothly) joining polynomial segments, as shown in Figure 1. The positions on the horizontal axis where the segments come together are called the knots. Only equidistant knots will be considered here, but B-splines can be defined for an arbitrary grid of knots. Many details and algorithms can be found in the books by de Boor [4] and Dierckx [4]. A cubic (quadratic) B-spline consists of cubic (quadratic) polynomial segments.

When computing a set of B-splines, each shifted by one knot distance, one gets a basis of local functions that is well suited for smoothing of a scatterplot of points $(x_i, y_i), i = 1, \ldots, m$. If $b_{ij} = B_j(x_i), j = 1, \ldots, n$ indicates the value of the jth B-spline at x_i, and $\mathbf{B} = [b_{ij}]$, one minimizes

$$S = |\mathbf{y} - \mathbf{B}\boldsymbol{\alpha}|^2 \tag{1}$$

with the explicit solution

$$\hat{\boldsymbol{\alpha}} = (\mathbf{B}'\mathbf{B})^{-1}\mathbf{B}'\mathbf{y} \tag{2}$$

Given $\hat{\boldsymbol{\alpha}}$, the estimated point on the curve at any (new) x is $\Sigma_j B_j(x)\hat{\alpha}_j$ This way smoothing is reduced to linear regression. The number of B-splines is equal to their degree plus the number of segments between the left-most and right-most knot.

The amount of smoothing is determined by the size of the B-spline basis and thus implicitly by the number of knots as illustrated by Figure 2. The larger the number of knots, the 'rougher' the curve. Still one can get a smooth result with many B-splines by forcing the coefficients to vary more smoothly, as illustrated in the lower panel of Figure 3. This

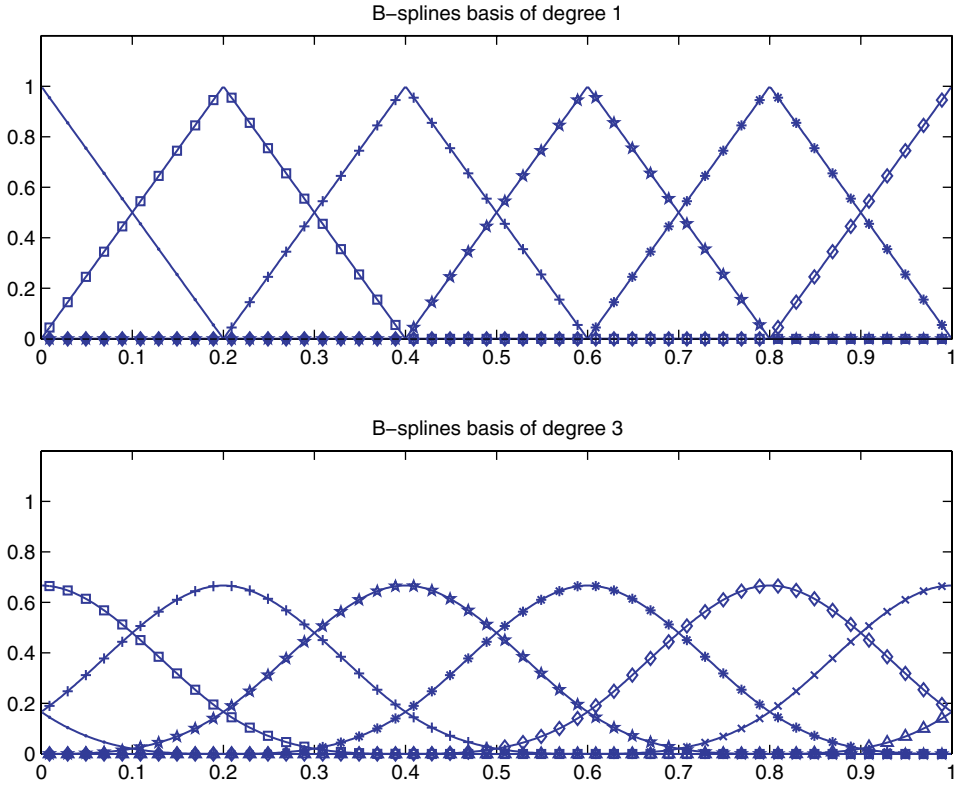

Figure 1. Linear (top) and cubic (bottom) B-spline bases, with knots at 0, 0.2, 0.4, 0.6, 0.8, and 1.0.

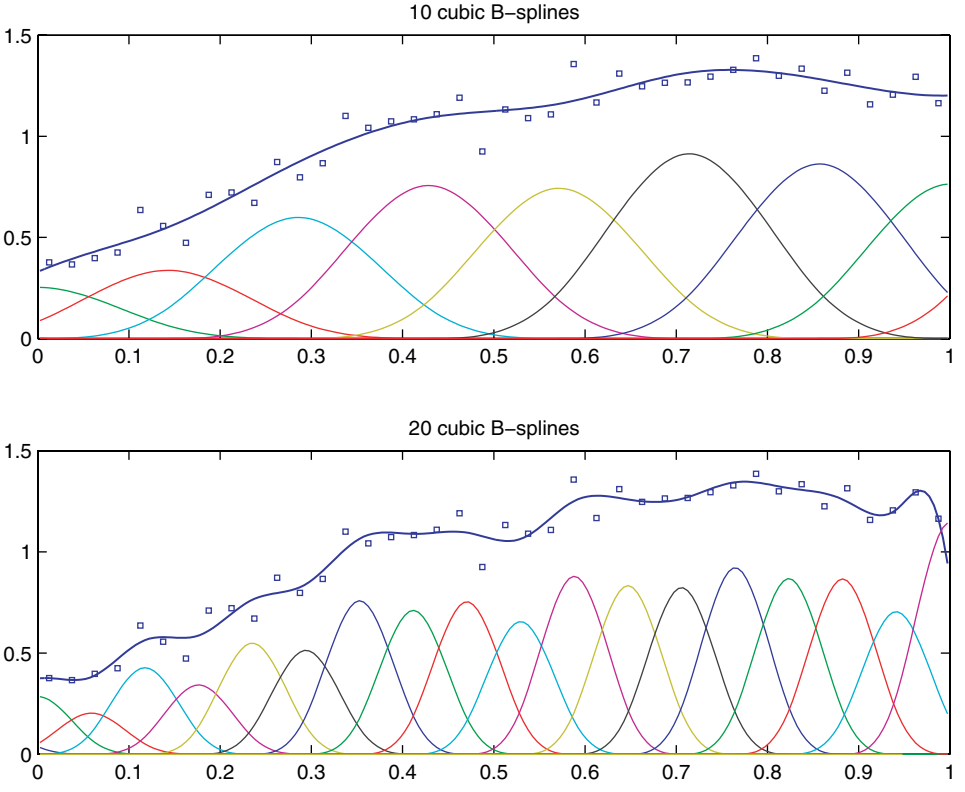

Figure 2. Fitting a smooth curve to a scatterplot using equally-spaced cubic B-splines. Top: using 10 basis functions; bottom: using 20 basis functions, giving a less smooth result. The individual scaled B-splines are shown as well as their sum (the thick line).

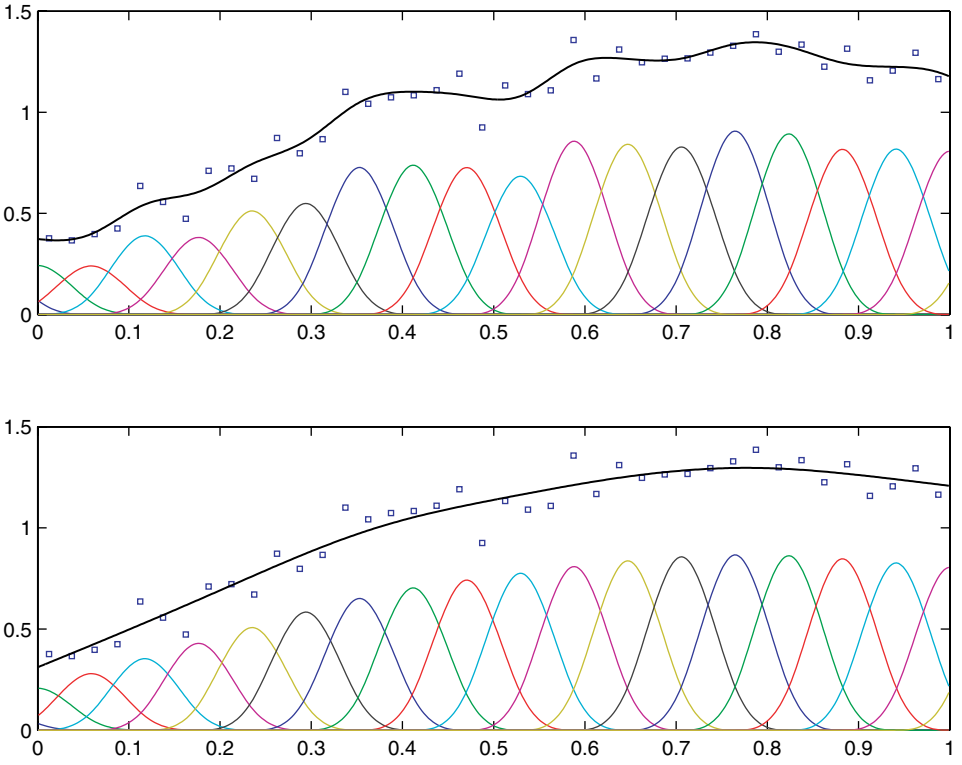

Figure 3. Fitting a smooth curve to a scatterplot using a large set of penalized cubic B-splines. Top: light penalty; bottom: strong penalty, giving a smoother result. The individual scaled B-splines are shown as well as their sum (the thick line).

is exactly the purpose of an additional penalty, weighted by a positive regularization parameter λ, that we attach to (1)

$$S^* = |\mathbf{y} - \mathbf{B}\boldsymbol{\alpha}|^2 + \lambda|\mathbf{D}_d\boldsymbol{\alpha}|^2 \tag{3}$$

The matrix $\mathbf{D}$ constructs dth order differences of $\boldsymbol{\alpha}$

$$\mathbf{D}_d\boldsymbol{\alpha} = \Delta^d\boldsymbol{\alpha} \tag{4}$$

The first difference of $\boldsymbol{\alpha}$, $\Delta^1\boldsymbol{\alpha}$ is the vector with elements $\alpha_{j+1} - \alpha_j$, for $j = 1, \ldots, n-1$. By repeating this computation on $\Delta\boldsymbol{\alpha}$, one arrives at higher differences like $\Delta^2\boldsymbol{\alpha}$ and $\Delta^3\boldsymbol{\alpha}$. The $(n-1) \times n$ matrix $\mathbf{D}_1$ is sparse, with $d_{j,j} = -1$ and $d_{j,j+1} = 1$ and all other elements zero. Examples of $\mathbf{D}_1$ and $\mathbf{D}_2$ of small dimension look like

$$\mathbf{D}_1 = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}; \quad \mathbf{D}_2 = \begin{bmatrix} 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \end{bmatrix}$$

Actually, the number of equally spaced knots does not matter much, provided that enough are chosen to insure more flexibility than needed: the penalty gives continuous control for further smoothing. The solution of Equation (3) is

$$\hat{\boldsymbol{\alpha}} = (\mathbf{B}'\mathbf{B} + \lambda\mathbf{D}'_d\mathbf{D}_d)^{-1}\mathbf{B}'\mathbf{y} \tag{5}$$

P-splines were proposed by Eilers and Marx [1] as a simplification of the discrete penalty approach of O'Sullivan [5].

B-splines are easily computed, as can be seen in the Matlab code presented in the Appendix. If the number of observations is moderate and they are equally spaced, like in uniformly sampled time series or spectra, and no interpolation is needed, one can even take the identity matrix as basis matrix. In that case P-splines are exactly equivalent to the Whittaker smoother [6].

3. ASYMMETRIC PENALTIES

The previous section presented difference penalties to force smooth behavior of an estimated signal. In addition, using asymmetric penalties, one can impose a variety of other useful properties, like a monotone or concave shape.

Consider the following goal function

$$Q = \sum_i (y_i - \sum_j b_{ij}\alpha_j)^2 + \lambda \sum_j (\Delta^3\alpha_j)^2 + \kappa \sum_j v_j(\Delta\alpha_j)^2 \tag{6}$$

The first two terms are the same as those for P-spline smoothing with a third order difference penalty. The third term looks very similar to the standard P-spline penalty, but there is a twist: the weights $\mathbf{v}$ depend on the sign of $\Delta\boldsymbol{\alpha}$

$$v_j = 0, \quad \text{if} \quad \Delta\alpha_j > 0 \tag{7}$$

$$v_j = 1, \quad \text{if} \quad \Delta\alpha_j \leq 0 \tag{8}$$

This means that the second penalty only has the effect where differences of $\boldsymbol{\alpha}$ are negative. The parameter κ generally is quite large, say 10^6 or larger, meaning that where the penalty is in effect, it really has a very strong influence.

To find the minimizer $\hat{\boldsymbol{\alpha}}$ of Q in Equation (6), one repeatedly solves the system

$$(\mathbf{B}'\mathbf{B} + \lambda\mathbf{D}'_3\mathbf{D}_3 + \kappa\mathbf{D}'_1\mathbf{V}\mathbf{D}_1)\hat{\boldsymbol{\alpha}} = \mathbf{B}'\mathbf{y} \tag{9}$$

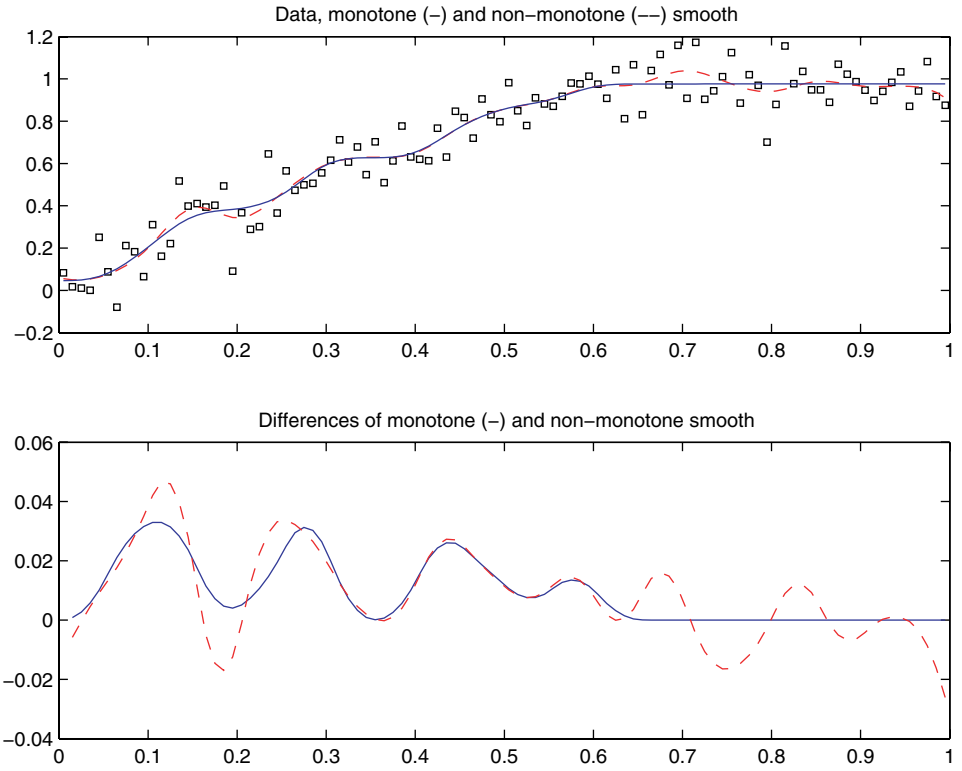

Figure 4. Smoothing of simulated data. The broken line is the result of cubic P-splines with 10 segments on the domain from 0 to 1, a second order difference penalty and $\lambda = 0.01$. The solid line is obtained with an asymmetric first order difference penalty ($\kappa = 10^6$, $\boldsymbol{\delta} = 0$). Top: data and smooth curves. Bottom: difference of the fitted curves.

where $\mathbf{V} = \mathrm{diag}(\mathbf{v})$, which is updated after a new $\hat{\boldsymbol{\alpha}}$ has been computed, according to Equation (8). An easy check for convergence is to look for any changes in $\mathbf{v}$.

Figure 4 shows an example for simulated data. With a small value of λ a wiggly curve is obtained, which is far from monotone. Adding the asymmetric penalty, with $\kappa = 10^6$, gives essentially the desired result. Five iterations were needed. The minimum of $\Delta\hat{\boldsymbol{\alpha}}$ is -3×10^{-7} which is below zero.

One should realize that some differences have to be negative to make the penalty work. For most practical applications this will be sufficiently close to monotone. However, if one has very strict ideas about monotonicity, the goal function can be modified to

$$Q = \sum_i (y_i - \sum_j b_{ij}\alpha_j)^2 + \lambda \sum_j (\Delta^3 \alpha_j)^2 + \kappa \sum_j v_j (\Delta\alpha_j - \delta_j)^2 \tag{10}$$

The weights are now computed as

$$v_j = 0, \quad \text{if} \quad \Delta\alpha_j > \delta_j \tag{11}$$

$$v_j = 1, \quad \text{if} \quad \Delta\alpha_j \leq \delta_j \tag{12}$$

Here $\boldsymbol{\delta}$ is a vector of small positive numbers. The iteration equation now becomes

$$(\mathbf{B}'\mathbf{B} + \lambda \mathbf{D}_3'\mathbf{D}_3 + \kappa \mathbf{D}_1'\mathbf{V}\mathbf{D})\hat{\boldsymbol{\alpha}} = \mathbf{B}'\mathbf{y} + \kappa \mathbf{D}_1'\mathbf{V}\boldsymbol{\delta} \tag{13}$$

One should play with $\boldsymbol{\delta}$ to find the value that makes $\Delta\boldsymbol{\alpha}$ positive everywhere, but as small as desired. An interesting consequence of $\boldsymbol{\delta}$ being a vector is that one can selectively impose monotone behavior. In regions where monotonicity is not necessary, one simply sets the elements of $\boldsymbol{\delta}$ to a large negative value.

In the presentation above monotone increasing behavior was desired. To get a monotone decrease, only a simple reversal of how the signs of $\Delta\boldsymbol{\alpha}$ determine the asymmetric weights is needed, and the elements of $\boldsymbol{\delta}$ should be set to small negative numbers.

When the trend in the data is already almost monotone, just smoothing will be enough to get a monotone estimated signal. The asymmetric penalty then is superfluous, but it can also do no harm. On the other hand it can force monotonicity on patently non-monotone data. An exaggerated example is shown in Figure 5. The data have a clear peak and smoothing without the asymmetric penalty brings this out. But once this is included a monotone smooth curve is obtained, even if it has to miss the data by far.

Consider now concave smoothing. The principle is the same, only the order of the asymmetric penalty changes. The goal function becomes

$$Q = \sum_i (y_i - \sum_j b_{ij}\alpha_j)^2 + \lambda \sum_j (\Delta^3 \alpha_j)^2 + \kappa \sum_j v_j (\Delta^2\alpha_j - \delta_j)^2 \tag{14}$$

The weights $\mathbf{v}$ are computed as

$$v_j = 0, \quad \text{if} \quad \Delta^2\alpha_j < \delta_j \tag{15}$$

$$v_j = 1, \quad \text{if} \quad \Delta^2\alpha_j \leq \delta_j \tag{16}$$

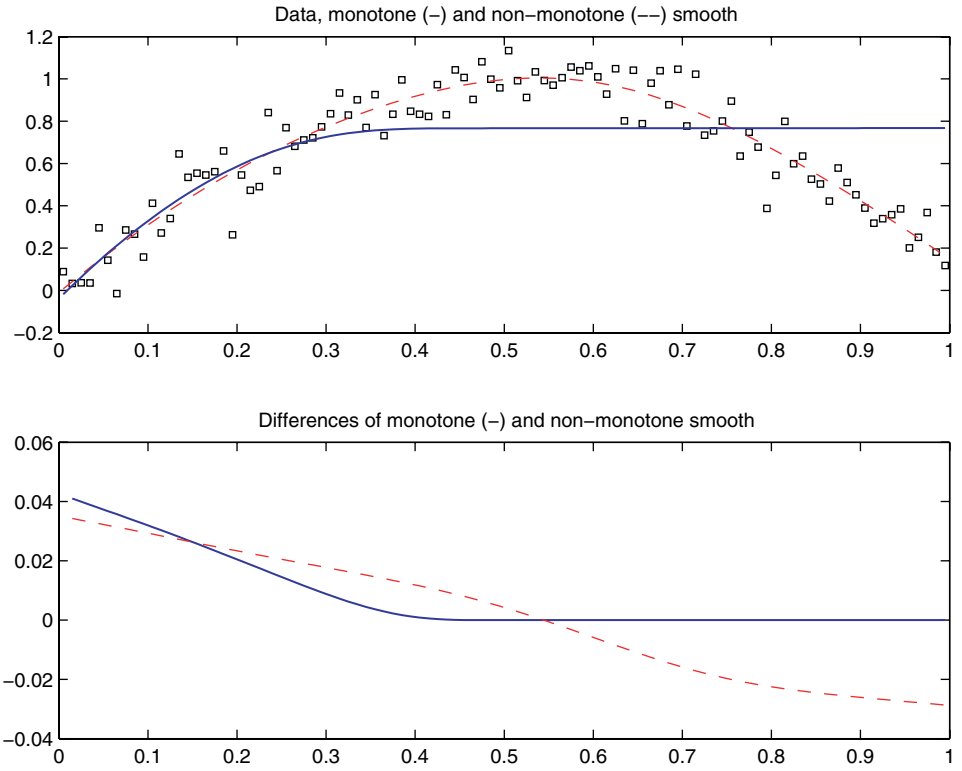

Figure 5. Smoothing of simulated data to show the power of the asymmetric penalty. The broken line is the result of cubic P-splines with 10 segments on the domain from 0 to 1, a second order difference penalty, and $\lambda = 100$. The solid line is obtained with an asymmetric first order difference penalty ($\kappa = 10^6$, $\boldsymbol{\delta} = 0$). Top: data and smooth curves. Bottom: difference of the fitted curves.

As before, κ is a large number, while $\boldsymbol{\delta}$ is a vector of small negative numbers or zeros. One iterates between solving

$$(\mathbf{B}'\mathbf{B} + \lambda \mathbf{D}_3'\mathbf{D}_3 + \kappa \mathbf{D}_2'\mathbf{V}\mathbf{D}_2)\boldsymbol{\alpha} = \mathbf{B}'\mathbf{y} + \kappa \mathbf{D}_2'\mathbf{V}\boldsymbol{\delta} \qquad (17)$$

and updating $\mathbf{v}$, starting with a vector of zeros for $\mathbf{v}$. Figure 6 shows an example with simulated data. The maximum of $\Delta^2\boldsymbol{\alpha}$ is 1.5×10^{-7}.

Unfortunately, concave smoothing does not work well with a peak that is surrounded by more or less flat regions, as Figure 7 shows. To maintain concave behavior, the estimated signal dives down the baseline. It also sacrifices curvature on the flanks of the peak in trying to fit data as good as possible while being concave. The next section presents log-concave smoothing as a remedy.

4. LOG-CONCAVE SMOOTHING

The inspiration for log-concave smoothing comes from the well-known Gaussian function, $y = \exp(-(x-\theta)^2/(2\sigma^2))$, which is unimodal. The function itself is not concave, but its logarithm is. Reasoning in the other direction: modeling the logarithm of a signal as concave, one expects to get a useful unimodal smoother. As a bonus the signal will also be positive, which is often desirable.

Let the signal model be

$$\mu_i = \exp\left(\sum_j b_{ij}\alpha_j\right) \qquad (18)$$

with $\mathbf{B} = [b_{ij}]$ a B-spline basis as before. Consider minimizing (without a penalty) $S = \Sigma_i(y_i - \mu_i)^2$, and linearizing μ_i in the neighborhood of an approximation $\tilde{\mu}_i$ as follows:

$$\mu_i \approx \tilde{\mu}_i + \sum_j \frac{\partial \tilde{\mu}_i}{\partial \alpha_j}\Delta\alpha_j = \tilde{\mu}_i + \sum_j \tilde{\mu}_i b_{ij}\Delta\alpha_j \qquad (19)$$

Then one can write

$$S \approx \sum_i (y_i - \tilde{\mu}_i - \sum_j \tilde{\mu}_i b_{ij}\Delta\alpha_j)^2 \qquad (20)$$

which is equivalent to linear regression of $\mathbf{y} - \tilde{\boldsymbol{\mu}}$ on $\tilde{\mathbf{M}}\mathbf{B}$, with $\mathbf{M} = \mathrm{diag}(\tilde{\boldsymbol{\mu}})$, to find $\Delta\boldsymbol{\alpha}$. With $\tilde{\boldsymbol{\alpha}}$ as approximation for $\boldsymbol{\alpha}$ the following equations have to be iterated

$$\mathbf{B}'\tilde{\mathbf{M}}\mathbf{B}\boldsymbol{\alpha} = \mathbf{B}'\tilde{\mathbf{M}}(\mathbf{y} - \tilde{\boldsymbol{\mu}}) + \mathbf{B}'\tilde{\mathbf{M}}\mathbf{B}\tilde{\boldsymbol{\alpha}} \qquad (21)$$

A roughness penalty on $\boldsymbol{\alpha}$ can be added straightforwardly

$$(\mathbf{B}'\tilde{\mathbf{M}}\mathbf{B} + \lambda \mathbf{D}_3'\mathbf{D}_3)\boldsymbol{\alpha} = \mathbf{B}'\tilde{\mathbf{M}}(\mathbf{y} - \tilde{\boldsymbol{\mu}}) + \mathbf{B}'\tilde{\mathbf{M}}\mathbf{B}\tilde{\boldsymbol{\alpha}} \qquad (22)$$

Figure 8 shows how this works out for the data in Figure 7.

The asymmetric penalty can be made strong enough to fit a unimodal curve through patently non-unimodal data. An example is shown in Figure 9, where simulated trimodal data are smoothed. It also helps to increase λ.

The shift to modeling the logarithm of the signal is more important than the shape penalty. This is illustrated in Figure 10 showing two peaks. Log-concave smoothing would not make sense here. Compared to linear smoothing less rounding at the top of the peaks occur and less wiggles in the valleys.

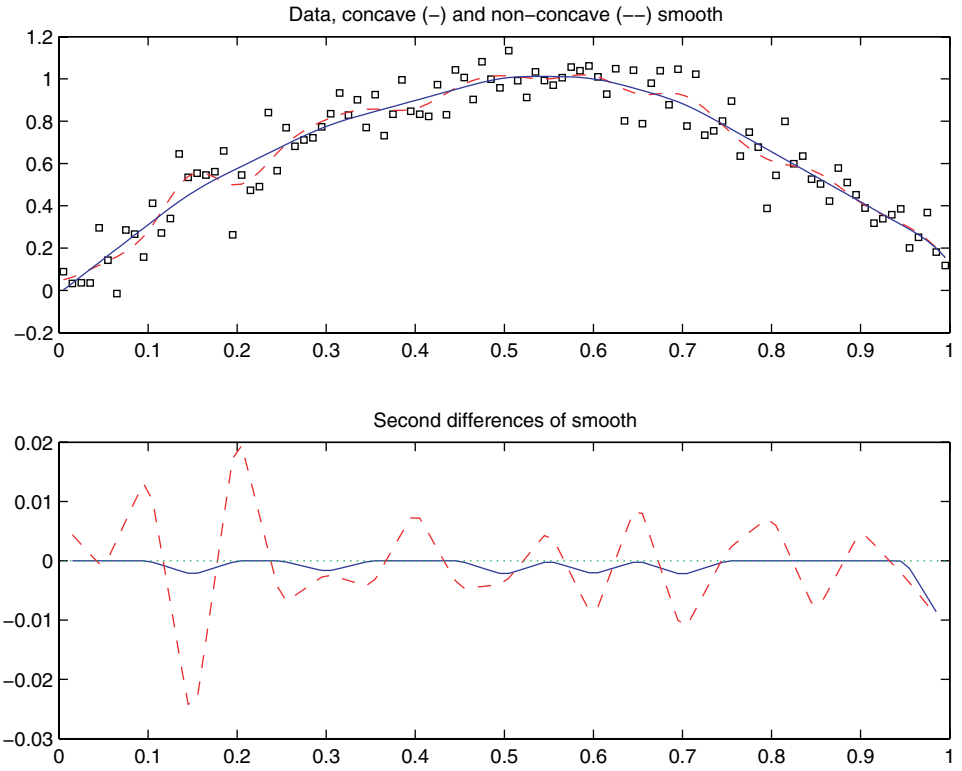

Figure 6. Smoothing of simulated data. The broken line is the result of cubic P-splines with 10 segments on the domain from 0 to 1, a second order difference penalty and $\lambda = 0.01$. The solid line is obtained with an asymmetric second order difference penalty ($\kappa = 10^6$, $\boldsymbol{\delta} = 0$). Top: data and smooth curves. Bottom: second differences of the fitted curves.

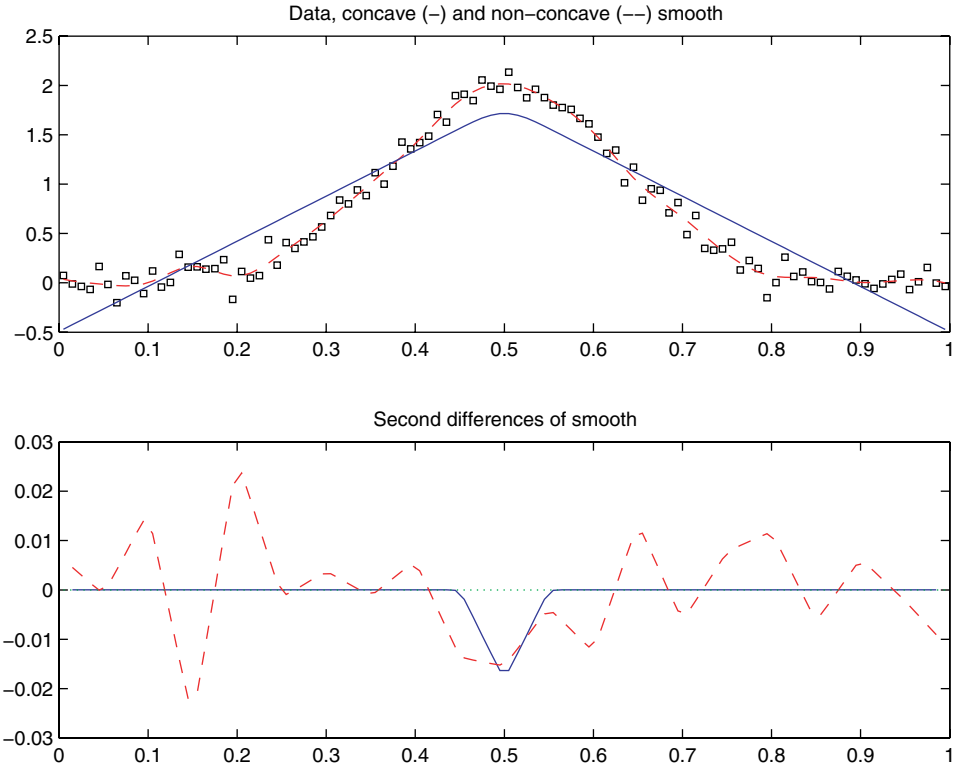

Figure 7. Smoothing of simulated data to show complications with concave smoothing. The broken line is the result of cubic P-splines with 10 segments on the domain from 0 to 1, a second order difference penalty and $\lambda = 0.01$. The solid line is obtained with an asymmetric second order difference penalty ($\kappa = 10^6$, $\boldsymbol{\delta} = 0$). Top: data and smooth curves. Bottom: second differences of the fitted curves.

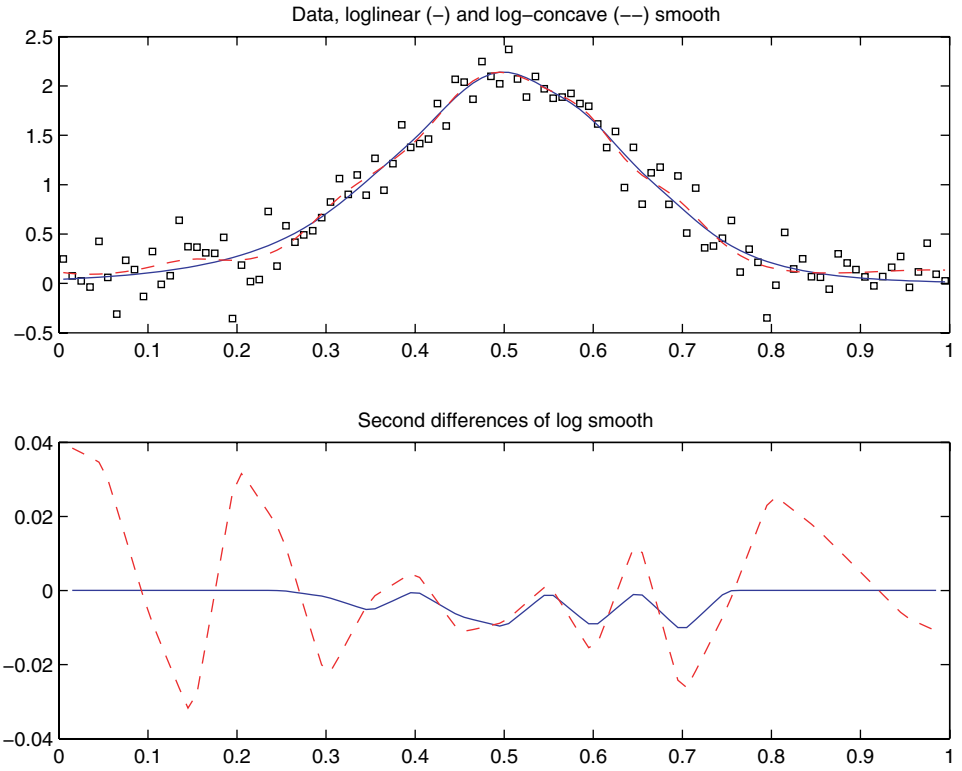

Figure 8. Log-linear and log-concave smoothing. The broken line is the result of cubic P-splines with 20 segments on the domain from 0 to 1, a third order difference penalty and $\lambda = 0.01$. The solid line is obtained with an asymmetric second order difference penalty ($\kappa = 10^6$, $\boldsymbol{\delta} = 0$). Top: data and smooth curves. Bottom: second differences of the fitted curves.

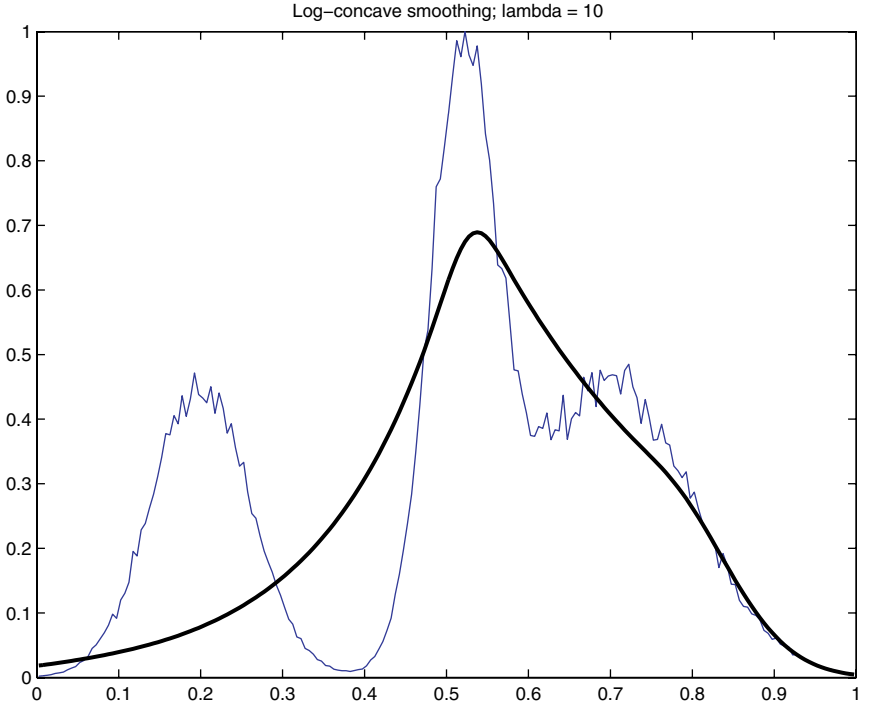

Figure 9. Log-concave smoothing of simulated data, to show the power of the asymmetric penalty. The thick line is the result of cubic P-splines with 50 segments on the domain from 0 to 1, a third order difference penalty and $\lambda = 10$ ($\kappa = 10^6$, $\boldsymbol{\delta} = 0$).

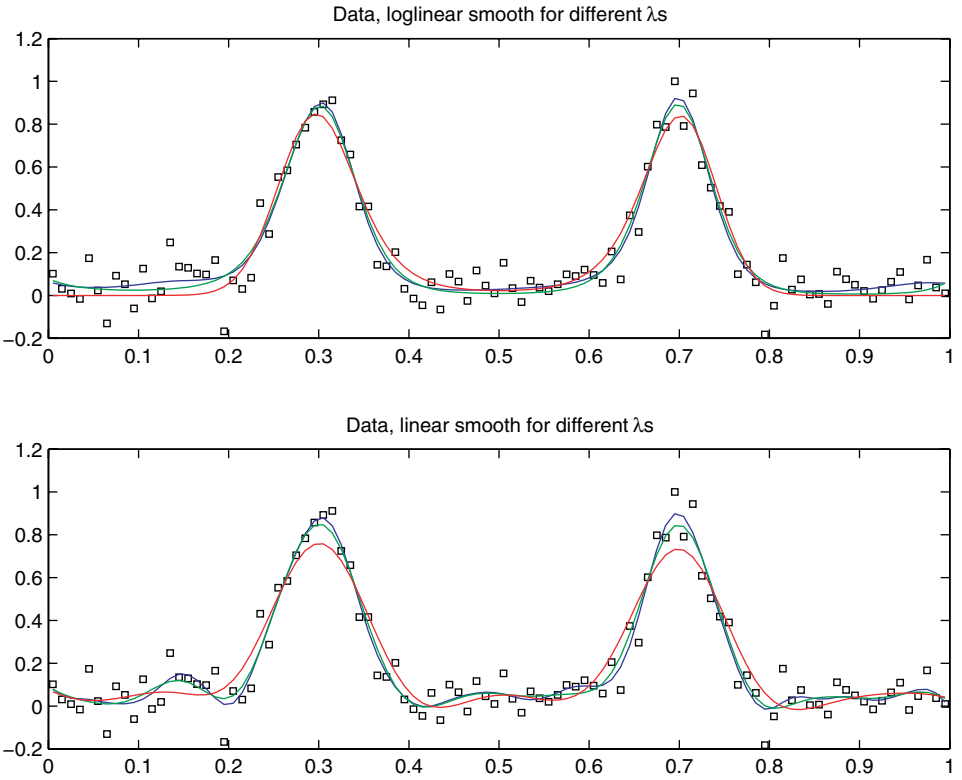

Figure 10. Log-linear smoothing with $\lambda = 0.1,1,10$ (top) and linear smoothing with $\lambda = 0.1,1,10$ (bottom).

In linear smoothing scaling of $\mathbf{y}$ has no influence, because the penalty will scale to the same amount as the sum of squares. This is not the case with log-linear smoothing: scaling of $\mathbf{y}$ will shift the logarithm by a certain amount, which will have no influence on differences. Consequently the size of the penalty would have to be changed by scaling λ (with the square of the scale factor). It is advisable to always scale $\mathbf{y}$ to the same order of magnitude, say a maximum value of 1, perform the smoothing, and then scale the result back.

The baseline can also lead to problems. It is impossible to get a negative fitted value. A very small positive number is the nearest possibility. This may lead to large negative values of the logarithm, which may take a large number of iterations to converge. Of course, a difference between -10 and -20 or -40 in the logarithm is meaningless: all these numbers will lead to a fitted value essentially equal to zero. In extreme cases even numerical instabilities may occur, when the log of the fitted values gets down below -200 or so. To avoid such complications, it is wise to shift the data a little in the positive direction and shift the fitted curve back. An extra safeguard is the introduction of a small ridge penalty to fight potential collinearity.

The number of user-defined parameters may look bewildering at first view. In practice, most of them can be set to default values. A basis of 50 cubic B-spline is flexible enough for most applications. Third order differences in the roughness penalty work well. For the penalty parameter, $\lambda = 0.1$ is a good starting choice. One lowers it if the curve is too smooth or increases it if results are too wiggly. For the unimodality penalty, $\kappa = 10^6$ and $\boldsymbol{\delta} = 0$ are good defaults. There is only a need to change $\boldsymbol{\delta}$ if one has very strict ideas about a lower bound on the second derivative of the log of the fitted curve. This choice of default parameters is based on the assumption that signals have been scaled so that their maximum value is 1.

5. APPLICATIONS

Riu and Bro [7] present an analysis of synthetic samples, consisting of four analytes (hydroquinone, tryptophan, phenylalanine, and dopa) in water. The data can be found on the KVL website (www.models.kvl.dk/research/data/dorrit). Figure 11 shows emission spectra for the first sample (only hydroquinone) at four different excitation wavelengths, as well as the fitted curves (using the default choice of parameters as described at the end of the previous section). Notice that missing data in the upper left panel (apparently due to clipping) have been interpolated automatically.

Bro and Sidiropoulos [2] presented a simulation of 25 signals, each a random multiple of a Gaussian curve shape plus noise. The model to be fitted is $\hat{y}_{ij} = \beta_{i\gamma j}$, with $\boldsymbol{\beta}$ unimodal and either $\boldsymbol{\beta}$ or $\boldsymbol{\gamma}$ scaled to some nominal value. Without shape constraints alternating least squares is a simple and effective algorithm. The resulting estimates of $\boldsymbol{\beta}$, for four different noise levels, are shown in Figure 12. It is straightforward to model the logarithm of $\boldsymbol{\beta}$ as a sum of B-splines and to introduce penalties for smoothness and concave shape. The result is also shown in Figure 12. Experience

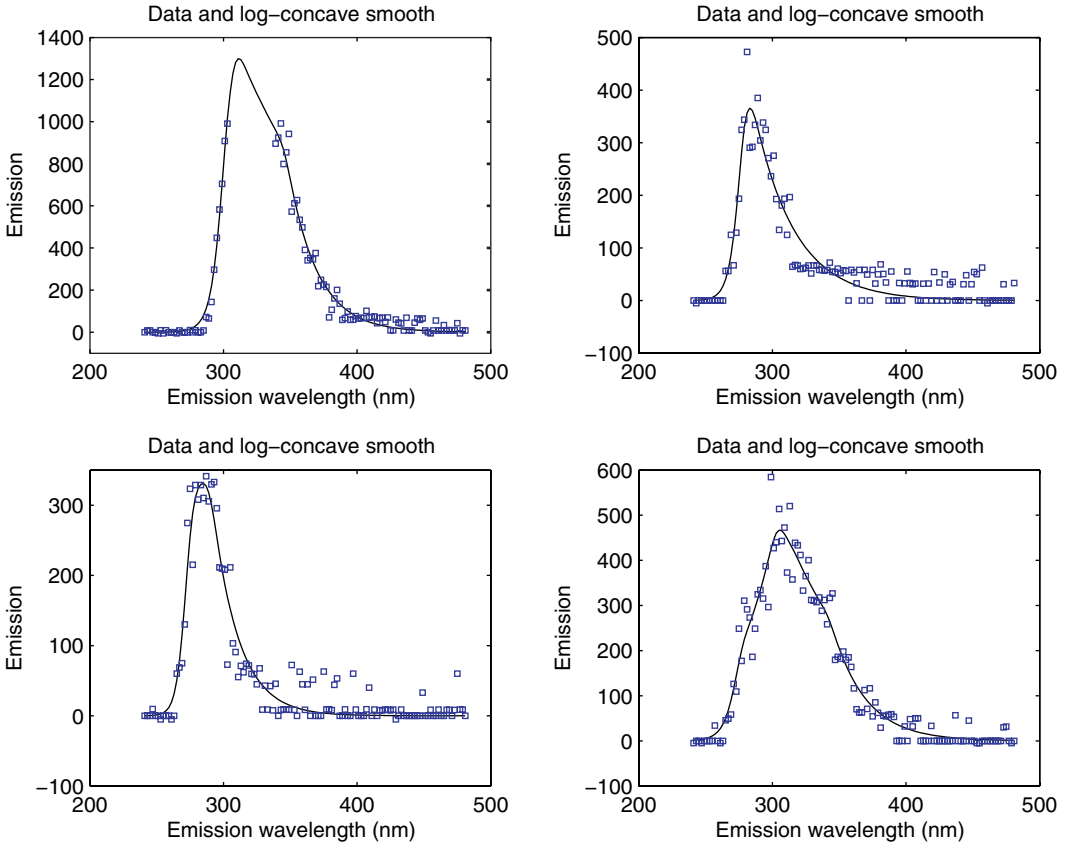

Figure 11. Four fluorescence spectra of hydroquinone, with fitted unimodal curves.

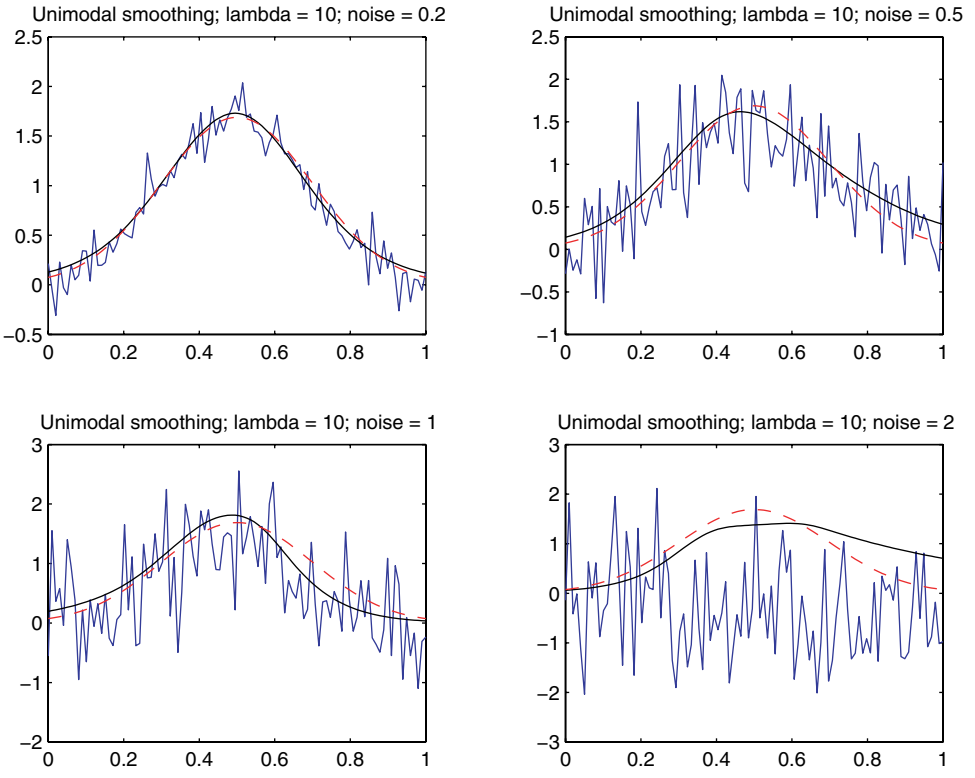

Figure 12. Smoothing in a rank-one model for a 100 by 25 matrix at different noise levels. The broken line shows the curve used for the simulation. The thin line shows the unconstrained curve estimate and the solid line the smooth log-concave estimate.

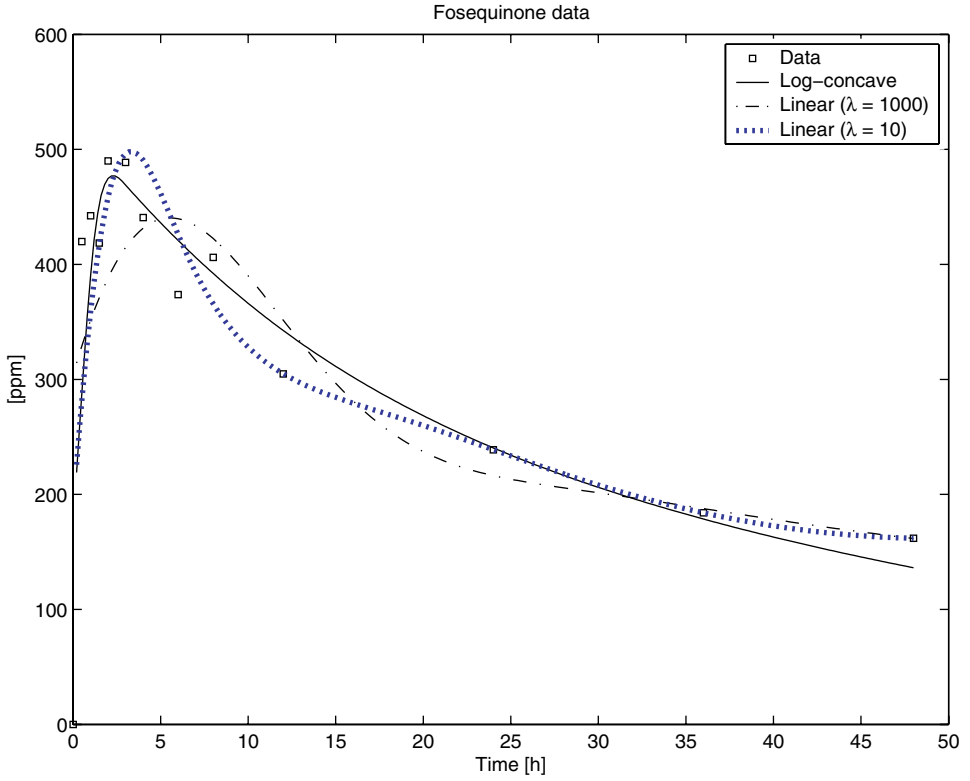

Figure 13. Smoothing data from a pharmacokinetic experiment. Dashed (dot-dashed) line: linear smoothing with $\lambda = 1$ ($\lambda = 10$). Solid line: log-concave smoothing with $\lambda = 1000$ and $\kappa = 10^8$.

showed that smoothing was enough in most cases. Only when the noise was very strong did the shape penalty spring into action.

An advantage of the use of P-splines is that the data can be arbitrarily positioned and that one can interpolate on any desired scale. This shows its value in the next example. Lindsey [8] analyzes pharmacokinetic experiments with fosequinone. A typical curve is shown in Figure 13. The curves have been fitted with 43 cubic B-splines using linear smoothing ($\lambda = 1$ and $\lambda = 10$ or with log-linear smoothing ($\lambda = 1000$ and $\kappa = 10^8$, no scaling of $\mathbf{y}$). With linear smoothing either one gets wiggles in the right tail, or misses the peak. Log-concave smoothing gives a meaningful fit to the data. From the background knowledge of the experiment it is expected that a unimodal concentration curve will occur.

The log-concave shape constraint is also effective in curve resolution. The value of unimodality has been emphasized in multidimensional data analysis [2,3]. But even in one-dimensional data benefits can be reaped. Figure 14 shows simulated data consisting of the sum of three Gaussian peaks with multiplicative noise. A sum of three log-concave components was fitted with a 'backfitting' approach: the estimate of each component in turn is improved. To this end the partial residuals $(\mathbf{y} - \Sigma_j \tilde{\boldsymbol{\mu}}_j + \tilde{\boldsymbol{\mu}}_k)$ are being regressed on $\tilde{\mathbf{M}}_k\mathbf{B}$ to improve $\boldsymbol{\alpha}_k$. Here the subscripts j and k (running from 1 to 3) refer to one of the component curves $(\tilde{\boldsymbol{\mu}}_k = \exp(\mathbf{B}\boldsymbol{\alpha}_k))$. Starting values have been obtained as the logarithms of Gaussian curves centered at the three observed peaks, with standard deviation 0.1 and heights at the observed peaks. After 50 iterations the maximal change in the $\boldsymbol{\alpha}$ components was less than 10^{-4}. This took one second, using Matlab on a 1000 MHz Pentium III computer. Speed improvements are possible, by updating all three $\boldsymbol{\alpha}$ vectors at the same time in a larger regression model. However, backfitting apparently is fast enough and has an appealing simplicity.

6. DISCUSSION

Log-linear smoothing and a simple asymmetric penalty for concaveness work together nicely to give a powerful and efficient algorithm for unimodal smoothing. The B-spline coefficients give a compact representation of the estimated curve, allowing evaluation at any desired resolution. This is attractive when fitting sparse, arbitrarily positioned data, like in pharmacokinetics or pharmacodynamics. Peak values and positions, as well as slopes, can be estimated accurately this way.

In contrast to other algorithms, the smooth unimodal shape constraint is formulated as a global property, by means of a penalty, not as (non-smooth) monotonicity of separate tails, implemented by 'merging adjacent violators'.

The algorithm seems general enough to be plugged into any alternating least squares algorithm for multi-way modeling, like curve resolution or PARAFAC. A very simple example was given in a rank-one model. More research is needed to establish the performance of the proposed algorithm in this area.

It was shown that unimodality constraints allow one-dimensional curve resolution (as a sum of unimodal

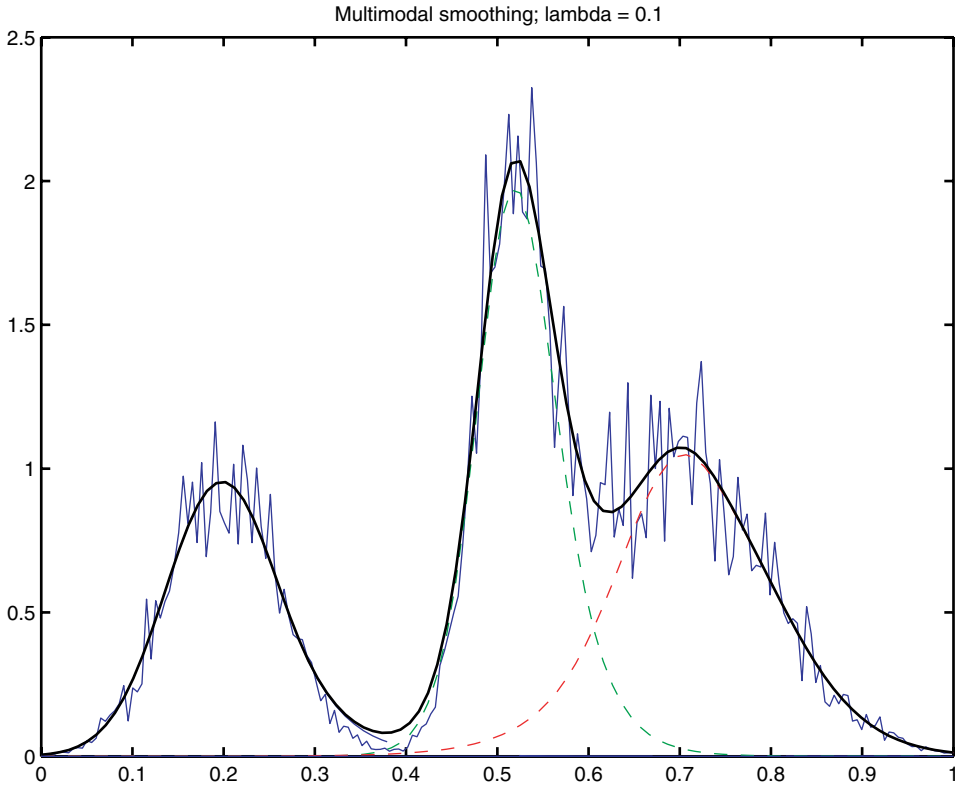

Figure 14. Fitting (simulated) data as a sum of smooth log-concave components. Thin solid line: data; thick line: sum of components; broken lines: individual components.

components) in special cases. It will be interesting to extend this approach to data consisting of a large number of peaks, like chromatograms or electrophoretic DNA profiles [9]. In principle backfitting will work, but with many pulses it might be inefficient to model each (local) peak as a complete curve. Working with (moving) segments or adaptive selection schemes could improve efficiency.

More work is needed to extend the power of this algorithm. It would be attractive to have speedy algorithms for (leave-one-out) cross-validation. That would be a useful building block for automatic choice of the smoothness parameter λ.

Prior weights can be incorporated easily. An interesting line for further research will be along the lines of quasi-likelihood in generalized linear models (GLM) [10]. The variance is computed from the fitted expected value $\boldsymbol{\mu}$, for example, as $\sigma^2 + \gamma^2\boldsymbol{\mu}^2$, where σ is the standard deviation of the absolute error and γ the proportionality constant for the relative error. Actually, without the penalties, the log-linear model as presented here is a special type of GLM, with a Gaussian response and a logarithmic link function.

Finally, a variety of 'tailor-made' shape constraints can easily be implemented. The adaptive weights in a difference penalty can be multiplied by prior weights. If monotone or concave (convex) behavior is only needed in part of the x domain, this can be coded into a corresponding pattern of zeros and ones in these prior weights. Also, one can form combinations like concave shape and monotone decrease, by introducing two asymmetric penalties.

REFERENCES

1. Eilers PHC, Marx BD. Flexible smoothing with B-splines and penalties (with discussion). *Stat. Sci.* 1996; **11**: 89–121.
2. Bro R, Sidiropoulos ND. Least squares algorithms under unimodality and non-negativity constraints. *J. Chemometrics* 1998; **12**: 223–247.
3. Gemperline PJ, Cash E. Advantages of soft versus hard constraints in self-modeling curve resolution problems. alternating least squares with penalty functions. *Anal. Chem.* 2003; **75**: 4236–4243.
4. De Boor C. *A Practical Guide to Splines* Springer, 2001.
5. O'Sullivan F. A statistical perspective on ill-posed inverse problems. *Stat. Sci.* 1986; **1**: 502–518.
6. Eilers PHC. A perfect smoother. *Anal. Chem.* 2003; **75**: 3631–3636.
7. Riu J, Bro R. Jack-knife estimation of standard errors and outlier detection in parafac models. *Chemometrics Intell. Lab. Sys.* 2003; **65**: 35–49.
8. Lindsey JK. *Nonlinear Models in Medical Statistics*. Oxford University Press: Oxford, UK, 2001.
9. Li LM, Speed TP. Deconvolution of sparse positive spikes. *J. Comput. Graph. Stat.* 2004; **13**: 853–870.
10. McCullagh P, Nelder J. *Generalized Linear Models*. Chapman and Hall: London, UK, 1990.

APPENDIX

Here follows the code for unimodal smoothing of the fosequinone data. To simplify the presentation of the algorithm no convergence test is performed, but a fixed number of iterations is used.

```
% Unimodal smoothing demo; fosequinone data (Lindsey)

% Get the data and scale them
x = [0 0.5 1 1.5 2 3 4 6 8 12 24 36 48]';
y0 = [0 420 442 419 490 489 441 374 406 305 239 184 162]';
ysc = max(y0);
y = y0 / ysc;

% Set B-spline parameters
nseg = 50;
deg = 3;
xlo = -1;
xhi = 50;
B = bbase(x, xlo, xhi, nseg, deg);

% Penalty matrices
lambda = 100;
n = size(B, 2);
D3 = diff(eye(n), 3);
D2 = diff(eye(n), 2);
P = lambda * D3' * D3;
kappa = 1e6;
R = 1e-8 * eye(n); % small ridge penalty

% Convex fit
v = zeros(n - 2, 1);
it = 0;
alpha = log(mean(y)) * ones(n, 1);
for it = 1:20
  eta = B * alpha;
  mu = exp(eta);
  G = B. * repmat(mu, 1, n);
  Q = D' * diag(v) * D;
  A = (G' * G + P + kappa * Q + R);
  r = G' * (y - mu + mu. * eta);
  alpha = A \ r;
  v = D * alpha > 0;
end

% Compute curve on grid
u = linspace(0, 50, 200)';
Bu = bbase(u, xlo, xhi, nseg, deg);
z = ysc * exp(Bu * alpha);

% Plot data and smooth curves
plot(x, y0, 's', 'MarkerSize', 4);
line(u, z);
```

The B-spline basis is computed with the following function:

```
function B = bbase(x, xmin, xmax, nseg, deg)
% Compute a B-spline basis using differences of truncated power functions
% Input
% x: evaluation points;
% xmin: left boundary
% xmax: right boudary
% nseg: number of inter-knot segments between xmin and xmax
% deg: degree of B-splines
dx = (xmax - xmin) / nseg;
knots = (x_min - deg * dx):dx:(xmax + deg * dx);
m = length(x);
n = length(knots);
X = repmat(x, 1, n);
T = repmat(knots, m, 1);
P = (X - T). ^ deg.* (X > T);
n = size(P, 2);
D = diff(eye(n), deg + 1) / (gamma(deg + 1) * dx ^ deg);
B = (-1) ^ (deg + 1) * P * D';
```