Stat Comput (2017) 27:985–989
DOI 10.1007/s11222-016-9666-x

P-splines with derivative based penalties and tensor product smoothing of unevenly distributed data

Simon N. Wood[1]

Received: 5 February 2016 / Accepted: 29 April 2016 / Published online: 18 May 2016

Abstract The P-splines of Eilers and Marx (Stat Sci 11:89–121, 1996) combine a B-spline basis with a discrete quadratic penalty on the basis coefficients, to produce a reduced rank spline like smoother. P-splines have three properties that make them very popular as reduced rank smoothers: (i) the basis and the penalty are sparse, enabling efficient computation, especially for Bayesian stochastic simulation; (ii) it is possible to flexibly 'mix-and-match' the order of B-spline basis and penalty, rather than the order of penalty controlling the order of the basis as in spline smoothing; (iii) it is very easy to set up the B-spline basis functions and penalties. The discrete penalties are somewhat less interpretable in terms of function shape than the traditional derivative based spline penalties, but tend towards penalties proportional to traditional spline penalties in the limit of large basis size. However part of the point of P-splines is not to use a large basis size. In addition the spline basis functions arise from solving functional optimization problems involving derivative based penalties, so moving to discrete penalties for smoothing may not always be desirable. The purpose of this note is to point out that the three properties of basis-penalty sparsity, mix-and-match penalization and ease of setup are readily obtainable with B-splines subject to derivative based penalization. The penalty setup typically requires a few lines of code, rather than the two lines typically required for P-splines, but this one off disadvantage seems to be the only one associated with using derivative based penalties. As an example application, it is shown how basis-penalty sparsity enables efficient computation with tensor product smoothers of scattered data.

✉ Simon N. Wood
simon.wood@bath.edu

[1] School of Mathematics, University of Bristol, Bristol, UK

Keywords Reduced rank spline · P-spline · Smoothing spline · Derivative penalty · Tensor product smooth

1 Computing arbitrary derivative penalties for B-splines

The main purpose of this note is to show that reduced rank spline smoothers with derivative based penalties can be set up almost as easily as the P-splines of Eilers and Marx (1996; see also Eilers et al. 2015), while retaining sparsity of the basis and penalty and the ability to mix-and-match the orders of spline basis functions and penalties. The key idea is that we want to represent a smooth function $f(x)$ using a rank k spline basis expansion $f(x) = \sum_{j=1}^{k} \beta_j B_{m_1,j}(x)$, where $B_{m_1,j}(x)$ is an order m_1 B-spline basis function, and β_j is a coefficient to be estimated. In this paper order $m_1 = 3$ will denote a cubic spline. Associated with the spline will be a derivative based penalty

$$J = \int_a^b f^{[m_2]}(x)^2 dx$$

where $f^{[m_2]}(x)$ denotes the m_2th derivative of f with respect to x, and $[a, b]$ is the interval over which the spline is to be evaluated. It is assumed that $m_2 \leq m_1$, otherwise the penalty is formulated in terms of a derivative that is not properly defined for the basis functions, which makes no sense. It is possible to write $J = \boldsymbol{\beta}^{\mathsf{T}}\mathbf{S}\boldsymbol{\beta}$ where $\mathbf{S}$ is a band diagonal matrix of known coefficients. Computation of $\mathbf{S}$ is the only part of setting up the smoother that presents any difficulty, since standard routines for evaluating B-splines basis functions (and their derivatives) are readily and widely available, and in any case the recursion for basis function evaluation is straightforward.

The derivation of the algorithm for computing $\mathbf{S}$ is quite straightforward. Given the basis expansion we have that

$$S_{ij} = \int_a^b B_{m_1,i}^{[m_2]}(x) B_{m_1,j}^{[m_2]}(x) dx.$$

However by construction $B_{m_1,i}^{[m_2]}(x)$ is made up of order $p = m_1 - m_2$ polynomial segments between adjacent knots, x_l, of the spline. So we are really interested in integrals of the form

$$\begin{aligned} S_{ijl} &= \int_{x_l}^{x_{l+1}} B_{m_1,i}^{[m_2]}(x) B_{m_1,j}^{[m_2]}(x) dx \\ &= \frac{h_l}{2} \int_{-1}^{1} \sum_{i=0}^{p} a_i x^i \sum_{j=0}^{p} d_j x^j dx \end{aligned}$$

for some polynomial coefficients a_i and d_j, where $h_l = x_{l+1} - x_l$. The polynomial coefficients are the solution obtained by evaluating $B_{m_1,i}^{[m_2]}(x)$ at $p+1$ points spaced evenly from x_l to x_{l+1}, to obtain a vector of evaluated derivatives, $\mathbf{g}_a$, and then solving $\mathbf{Pa} = \mathbf{g}_a$, where $P_{ij} = (-1 + 2(i-1)/p)^j$ ($\mathbf{d}$ is obtained from $\mathbf{g}_d$ similarly). Then $S_{ij} = \sum_l S_{ijl}$.

Given that $\int_{-1}^{1} x^q dx = (1 + (-1)^q)/(q+1)$ it is clear that $S_{ijl} = h_l \mathbf{a}^\mathsf{T} \mathbf{H} \mathbf{d}/2$ where $H_{ij} = (1 + (-1)^{i+j-2})/(i+j-1)$ (i and j start at 1). In terms of the evaluated gradient vectors,

$$S_{ijl} = h_l \mathbf{g}_a^\mathsf{T} \mathbf{P}^{-\mathsf{T}} \mathbf{H} \mathbf{P}^{-1} \mathbf{g}_d/2.$$

Letting $\mathbf{G}$ denote the matrix mapping $\boldsymbol{\beta}$ to the concatenated (and duplicate deleted) gradient vectors for all intervals, while $\mathbf{W}$ is just the overlapping-block diagonal matrix with blocks given by $h_l \mathbf{P}^{-\mathsf{T}} \mathbf{H} \mathbf{P}^{-1}/2$ (see step 4 below), we have that $S_{ij} = \mathbf{G}_i^\mathsf{T} \mathbf{W} \mathbf{G}_j$, where $\mathbf{G}_i$ is the ith row of $\mathbf{G}$. Notice that the construction applies equally well to a cyclic smoother: all that changes is $\mathbf{G}$.

The algorithm for finding $\mathbf{S}$ in general is then as follows. $p = m_1 - m_2$ denotes the order of piecewise polynomial defining the m_2th derivative of the spline. Let $x_1, x_2 \ldots x_{k-m+1}$ be the (ordered) 'interior knots' defining the B-spline basis, that is the knots within whose range the spline and its penalty are to be evaluated (so $a = x_1$ and $b = x_{k-m+1}$). Let the inter-knot distances be $h_j = x_{j+1} - x_j$, for $0 < j \leq k - m$.

1. For each interval $[x_j, x_{j+1}]$, generate $p+1$ evenly spaced points within the interval. For $p = 0$ the point should be at the interval centre, otherwise the points always include the end points x_j and x_{j+1}. Let $\mathbf{x}'$ contain the unique x values so generated, in ascending order.
2. Obtain the matrix $\mathbf{G}$ mapping the spline coefficients to the m_2th derivative of the spline at the points $\mathbf{x}'$.
3. If $p = 0$, $\mathbf{W} = \mathrm{diag}(\mathbf{h})$.
4. It $p > 0$, let $p+1 \times p+1$ matrices $\mathbf{P}$ and $\mathbf{H}$ have elements $P_{ij} = (-1 + 2(i-1)/p)^j$ and $H_{ij} = (1 + (-1)^{i+j-2})/(i+j-1)$ (i and j start at 1). Then compute matrix $\tilde{\mathbf{W}} = \mathbf{P}^{-\mathsf{T}} \mathbf{H} \mathbf{P}^{-1}$. Now compute $\mathbf{W} = \sum_{q=1}^{k-m} \mathbf{W}^q$ where each $\mathbf{W}^q$ is zero everywhere except at $W^q_{i+pq-p, j+pq-p} = h_q \tilde{W}_{ij}/2$, for $i = 1, \ldots, p+1$, $j = 1, \ldots, p+1$. $\mathbf{W}$ is banded with $2p+1$ non-zero diagonals.
5. The diagonally banded penalty coefficient matrix is $\mathbf{S} = \mathbf{G}^\mathsf{T} \mathbf{W} \mathbf{G}$.
6. Optionally, compute the diagonally banded Cholesky decomposition $\mathbf{R}^\mathsf{T} \mathbf{R} = \mathbf{W}$, and form diagonally banded matrix $\mathbf{D} = \mathbf{R}\mathbf{G}$, such that $\mathbf{S} = \mathbf{D}^\mathsf{T} \mathbf{D}$.

Step 2 can be accomplished by standard routines for generating B-spline bases and their derivatives of arbitrary order: for example, the function `splines:splineDesign` in `R`. Alternatively see the appendix. All that is required to produce a cyclic spline is to substitute the appropriate cyclic routine at step 2: for example `mgcv:cSplineDes` in `R`. Step 4 requires no more than a single rank $p+1$ matrix inversion of $\mathbf{P}$. $\mathbf{P}$ is somewhat ill conditioned for $p \geq 20$, with breakdown for $p > 30$. However it is difficult to imagine any sane application for which p would even be as high as 10, and for $p \leq 10$, $\mathbf{P}$'s condition number is $< 2 \times 10^4$. Of course $\mathbf{W}$ is formed without explicitly forming the $\mathbf{W}^q$ matrices. Step 6 can be accomplished by a banded Cholesky decomposition such as `dpbtrf` from LAPACK (accessible via routine `mgcv:bandchol` in `R`, for example). Alternatively see the appendix. However for applications with k less than 1000 or so, a dense Cholesky decomposition might be deemed efficient enough. Note that step 6 is preferable to construction of $\mathbf{D}$ by decomposition of $\mathbf{S}$, since $\mathbf{W}$ is positive definite by construction, while, for $m_2 > 0$, $\mathbf{S}$ is only positive semi-definite. As in the case of a discrete P-spline penalty, the leading order computational cost of evaluating $\mathbf{S}$ (or $\mathbf{D}$) is $O(bk)$ where b is the number of bands in $\mathbf{S}$: the constant of proportionality is lower for a discrete penalty of course, but in either case the cost is trivial relative to that of model fitting (which is $O(nk^2)$ using dense matrix methods).

The simplicity of the algorithm rests on the ease with which $\mathbf{G}$ and $\mathbf{W}$ can be computed. Note that the construction is more general than that of Wand and Ormerod (2008), in allowing m_1 and m_2 to be chosen freely (rather than m_1 determining m_2), and treating even m_1 as well as odd.

2 Tensor product smoothing of unevenly distributed data

An example where a compactly supported basis and sparse penalty is computationally helpful is in tensor product smoothing of unevenly distributed data. A three dimensional

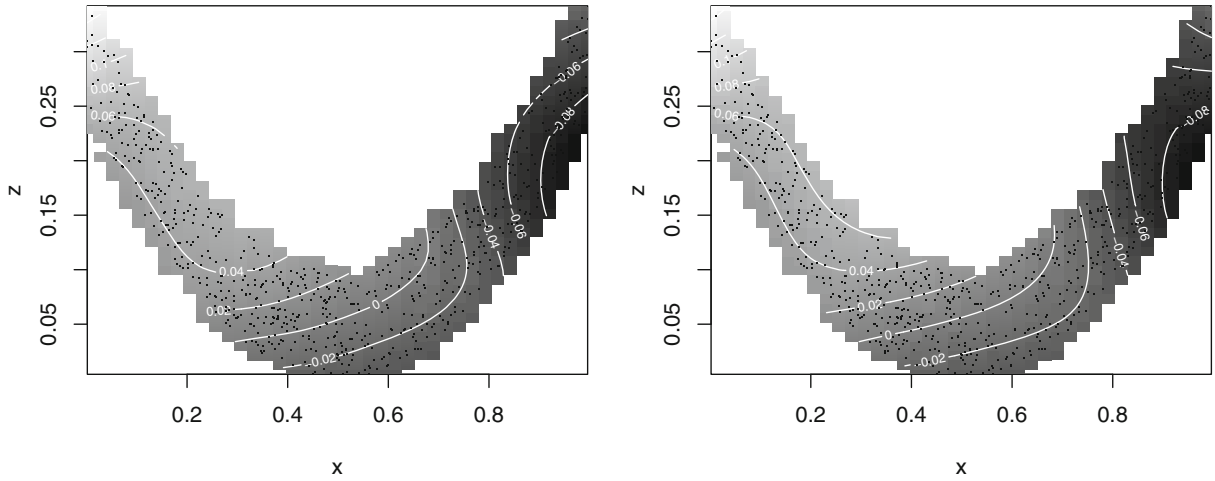

Fig. 1 *Left* conventional tensor product smooth reconstruction of the example function given in the text, based on noisy samples at the x, z locations shown as *black dots*. *Right* as left, but using the reduced basis described in Sect. 2.

example suffices to illustrate how tensor product smooths are constructed from one dimensional bases. Suppose we want to smooth with respect to z_1, z_2 and z_3. Firstly B-spline bases are constructed for smooth functions of each covariate separately. Suppressing subscripts for order, let $B_{j1}(z_j)$, $B_{j2}(z_j)$, ... denote the basis for the smooth function of z_j, and let $\mathbf{D}_j$ denote the corresponding 'square root' penalty matrix. The smooth function of all three variables is then represented as

$$f(\mathbf{z}) = \sum_{ijl} \beta_{ijl} B_{1i}(z_1) B_{2j}(z_2) B_{3l}(z_3)$$

where the β_{ijl} are coefficients. Notice that the tensor product basis functions, $B_{1i}(z_1)B_{2j}(z_2)B_{3l}(z_3)$, inherit compact support form the marginal basis functions. Now write the coefficients in 'column major' order in one vector $\boldsymbol{\beta}^{\mathsf{T}} = (\beta_{111}, \beta_{112}, \ldots, \beta_{11k_1}, \beta_{121}, \beta_{122}, \ldots \beta_{k_1k_2k_3})$, where k_j is the dimension of the jth basis. The tensor product smoother then has three associated penalties, $\boldsymbol{\beta}^{\mathsf{T}}\mathbf{S}_j\boldsymbol{\beta}$ (each with its own smoothing parameter), where $\mathbf{S}_j = \tilde{\mathbf{D}}_j^{\mathsf{T}}\tilde{\mathbf{D}}_j$,

$$\tilde{\mathbf{D}}_1 = \mathbf{D}_1 \otimes \mathbf{I}_{k_2} \otimes \mathbf{I}_{k_3}, \tilde{\mathbf{D}}_2 = \mathbf{I}_{k_1} \otimes \mathbf{D}_2 \otimes \mathbf{I}_{k_3} \text{ and } \tilde{\mathbf{D}}_3 = \mathbf{I}_{k_1} \otimes \mathbf{I}_{k_2} \otimes \mathbf{D}_3.$$

This construction generalizes to other numbers of dimensions in the obvious way (see e.g. Wood 2006).

By construction the domain of the tensor product smooth is a rectangle, cuboid or hypercuboid, but it is often the case that the covariates to be smoothed over occupy only part of this domain. In this case it is possible for some basis functions to evaluate to zero at every covariate observation, and there is often little point in retaining these basis functions and their associated coefficients. Let ι denote the index of a coefficient to be dropped from $\boldsymbol{\beta}$ (along with its corresponding basis function). The naïve approach of dropping row and column ι of each $\mathbf{S}_j$ is equivalent to setting β_ι to zero when evaluating $\boldsymbol{\beta}^{\mathsf{T}}\mathbf{S}_j\boldsymbol{\beta}$, which is not usually desirable. Rather than setting $\beta_\iota = 0$ in the penalty, we would like to omit those components of the penalty dependent on β_ι. This is easily achieved by dropping every row κ from $\tilde{\mathbf{D}}_j$ for which $\tilde{D}_{j,\kappa\iota} \neq 0$. Notice (i) this construction applies equally well to P-splines, and (ii) that without $\mathbf{D}$ being diagonally banded this would be a rather drastic reduction of the penalty.

As an illustration 700 data were generated from the model

$$y_i = \exp\left\{-(z_i - 0.3)^2/2 - (x_i - 0.2)^2/4\right\} + \epsilon_i, \text{ where } \epsilon_i \sim N(0, 0.1^2)$$

at the x, z locations shown as black dots in Fig. 1. The figure shows the reconstruction of the test function using a tensor product smoother, based on cubic spline marginals with second derivative penalties. The left figure is for the full smoother, which had 625 coefficients, while the right figure is for the reduced version which had 358 coefficients. Since the $\mathbf{S}$ matrix of a smoother can be viewed as the prior precision matrix of a Gaussian random field, the smoothing parameters can be estimated by marginal likelihood maximization (e.g. Wahba 1985), and the computational method of Wood (2011) was used for this purpose. Including smoothing parameter selection the reduced rank fit took around 1/8 of the computation time of the full rank fit, as a result of the reduction in basis size. The correlation between the fitted values for the two fits is 0.999. In the example the reduced rank fit has marginally smaller mean square reconstruction error than the full rank version, a feature that seems to be robust under repeated replication of the experiment.

In this example penalty sparsity was important in order to be able to reduce the penalty appropriately when eliminating irrelevant basis functions, but the sparsity was not further exploited in the computations. In contrast, Bayesian stochastic simulation with such models is much more efficient if the basis matrix, $\mathbf{X}$, is sparse, so that the likelihood is efficiently computable at each step, and the penalty is sparse, so that computations involving $(\mathbf{X}^{\mathsf{T}}\mathbf{X} + \lambda\mathbf{S})^{-1}$ can be used to make efficient proposals. Similarly for direct estimation in big data settings, sparse evaluation of the REML expressions given in Wood (2011), for example, rests on evaluation of terms like $\log|\mathbf{X}^{\mathsf{T}}\mathbf{X} + \lambda\mathbf{S}|$, which can be made more efficient if both $\mathbf{X}$ and $\mathbf{S}$ are sparse. See Davis (2006) for more on the computational exploitation of sparsity.

3 Conclusions

Given that the theoretical justification for using spline bases for smoothing is that they arise as the solutions to variational problems with derivative based penalties (see e.g. Wahba 1990; Duchon 1977), it is sometimes appealing to be able to use derivative based penalties for reduced rank smoothing also. Since the derivative based penalty is not reliant on even knot spacing there may also be practical advantages when uneven spacing of knots is desirable (e.g. Whitehorn et al. 2013). However if a sparse smoothing basis and penalty were required alongside the ability to mix-and-match penalty order and basis order, then the apparent complexity of obtaining the penalty matrix for derivative based penalties has hitherto presented an obstacle to their use. This note removes this obstacle, allowing the statistician an essentially free choice whether to use derivative based penalties or discrete penalties. Notice that there is nothing to prevent computation of several different orders of penalty for the same smoother, thereby facilitating the use of more general differential operators as penalties (e.g. Ramsay et al. 2007).

The splines described here are available in `R` package `mgcv` from version 1.8–12. They could be referred to as 'D-splines', but a new name is probably un-necessary. This work was supported by EPSRC grant EP/K005251/1, and I am grateful to two anonymous referees for several helpful suggestions.

Acknowledgements Open access funding provided by University of Bristol.

Appendix: Standard recursions

B-spline bases, their derivatives and banded Cholesky decompositions are readily available in standard software libraries and packages such as `R` and `Matlab`. However, for completeness the required recursions are included here.

To define a k dimensional B-spline basis of order m we need to define $k + m + 1$ knots $x_1 < x_2 < \cdots < x_{k+m+1}$. The interval over which the spline is to be evaluated is $[x_{m+1}, x_{k+1}]$ so the locations of knots outside this interval are rather unimportant. The B-spline basis functions are defined recursively as

$$B_{m,i}(x) = \frac{x - x_i}{x_{i+m} - x_i} B_{m-1,i}(x) + \frac{x_{i+m+1} - x}{x_{i+m+1} - x_{i+1}} B_{m-1,i+1}(x), \quad i = 1, \ldots, k, \ m > 0$$

where

$$B_{0,i}(x) = \begin{cases} 1 & x_i \le x < x_{i+1} \\ 0 & \text{otherwise.} \end{cases}$$

It turns out that the derivative with respect to x of a B-spline of order m can be expressed in terms of a B-spline basis of order $m - 1$ as follows

$$\sum_j \beta_j B'_{m,j}(x) = (m-1) \sum_j \frac{\beta_j - \beta_{j-1}}{x_{j+m} - x_j} B_{m-1,j}(x).$$

This can be applied recursively to obtain higher order derivatives. For more on both of these recursions see p. 89 and p. 116 of de Boor (2001) (or de Boor 1978).

Now consider the banded Cholesky decomposition of a symmetric positive definite matrix $\mathbf{A}$ with $2p - 1$ non-zero diagonals (clustered around the leading diagonal). We have

$$R_{ii} = \sqrt{A_{ii} - \sum_{k=i-p}^{i-1} R_{ki}^2}, \text{ and}$$

$$R_{ij} = \frac{A_{ij} - \sum_{k=i-p}^{i-1} R_{ki} R_{kj}}{R_{ii}}, \quad i < j < i + p.$$

all other elements of Cholesky factor $\mathbf{R}$ being 0. The expressions are used one row at a time, starting from row 1, and working across the columns from left to right. See any matrix algebra book for Cholesky decomposition (e.g. Golub and Van Loan 2013).

References

Davis, T.A.: Direct Methods for Sparse Linear Systems. SIAM, Philadelphia (2006)

de Boor, C.: A Practical Guide to Splines. Springer, New York (1978)

de Boor, C.: A Practical Guide to Splines, Revised edn. Springer, New York (2001)

Duchon, J.: Splines minimizing rotation-invariant semi-norms in Solobev spaces. In: Schemp, W., Zeller, K. (eds.) Construction Theory of Functions of Several Variables, pp. 85–100. Springer, Berlin (1977)

Eilers, P.H., Marx, B.D., Durbán, M.: Twenty years of p-splines. SORT-Stat. Oper. Res. Trans. **39**(2), 149–186 (2015)

Eilers, P.H.C., Marx, B.D.: Flexible smoothing with B-splines and penalties. Stat. Sci. **11**(2), 89–121 (1996)

Golub, G.H., Van Loan, C.F.: Matrix Computations, 4th edn. Johns Hopkins University Press, Baltimore (2013)

Ramsay, J.O., Hooker, G., Campbell, D., Cao, J.: Parameter estimation for differential equations: a generalized smoothing approach. J. R. Stat. Soc.: Ser. B (Stat. Methodol.) **69**(5), 741–796 (2007)

Wahba, G.: A comparison of GCV and GML for choosing the smoothing parameter in the generalized spline smoothing problem. Ann. Stat. **13**, 1378–1402 (1985)

Wahba, G.: Spline Models for Observational Data. SIAM, Philadelphia (1990)

Wand, M., Ormerod, J.: On semiparametric regression with O'Sullivan penalized splines. Aust. N. Z. J. Stat. **50**(2), 179–198 (2008)

Whitehorn, N., van Santen, J., Lafebre, S.: Penalized splines for smooth representation of high-dimensional monte carlo datasets. Comput. Phys. Commun. **184**(9), 2214–2220 (2013)

Wood, S.N.: Low-rank scale-invariant tensor product smooths for generalized additive mixed models. Biometrics **62**(4), 1025–1036 (2006)

Wood, S.N.: Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. J. R. Stat. Soc.: Ser. B (Stat. Methodol.) **73**(1), 3–36 (2011)