Divided Differences, Falling Factorials, and Discrete Splines

Another Look at Trend Filtering and Related Problems

Ryan J. Tibshirani

Contents

Abstract

This paper reviews a class of univariate piecewise polynomial functions known as *discrete splines*, which share properties analogous to the better-known class of spline functions, but where continuity in derivatives is replaced by (a suitable notion of) continuity in *divided differences*. As it happens, discrete splines bear connections to a wide array of developments in applied mathematics and statistics, from divided differences and Newton interpolation (dating back to over 300 years ago) to trend filtering (from the last 15 years). We survey these connections, and contribute some new perspectives and new results along the way.

1 Introduction

Nonparametric regression is a fundamental problem in statistics, in which we seek to flexibly estimate a smooth trend from data without relying on specific assumptions about its form or shape. The standard setup is to assume that data comes from a model (often called the "signal-plus-noise" model):

$$y_i = f_0(x_i) + \epsilon_i, \quad i = 1, \dots, n.$$

Here, $f_0 : \mathcal{X} \to \mathbb{R}$ is an unknown function to be estimated, referred to as the *regression function*; $x_i \in \mathcal{X}$, $i = 1, \dots, n$ are *design points*, often (though not always) treated as nonrandom; $\epsilon_i \in \mathbb{R}$, $i = 1, \dots, n$ are random errors, usually assumed to be i.i.d. (independent and identically distributed) with zero mean; and $y_i \in \mathbb{R}$, $i = 1, \dots, n$ are referred to as *response points*. Unlike in a *parametric* problem, where we would assume f_0 takes a particular form (for example, a polynomial function) that would confine it to some finite-dimensional function space, in a *nonparametric* problem we make no such restriction, and instead assume f_0 satisfies some broader smoothness properties (for example, it has two bounded derivatives) that give rise to an infinite-dimensional function space.

The modern nonparametric toolkit contains an impressive collection of diverse methods, based on ideas like kernels, splines, and wavelets, to name just a few. Many estimators of interest in nonparametric regression can be formulated as the solutions to optimization problems based on the observed data. At a high level, such optimization-based methods can be divided into two camps. The first can be called the *continuous-time approach*, where we optimize over a function $f : \mathcal{X} \to \mathbb{R}$ that balances some notion of goodness-of-fit (to the data) with another notion of smoothness. The second can be called the *discrete-time approach*, where we optimize over function evaluations $f(x_1), \dots, f(x_n)$ at the design points, again to balance goodness-of-fit with smoothness.[1]

The main difference between these approaches lies in the optimization variable: in the first it is a function f, and in the second it is a vector $\theta = (f(x_1), \dots, f(x_n)) \in \mathbb{R}^n$. Each perspective comes with its advantages. The discrete-time approach is often much simpler, conceptually speaking, as it often requires only a fairly basic level of mathematics in order to explain and understand the formulation at hand. Consider, for example, a setting with $\mathcal{X} = [a, b]$ (the case of univariate design points), where we assume without a loss of generality that $x_1 < x_2 < \cdots < x_n$, and we define an estimator by the solution of the optimization problem:

$$\underset{\theta}{\text{minimize}} \;\; \frac{1}{2} \sum_{i=1}^n (y_i - \theta_i)^2 + \lambda \sum_{i=1}^{n-1} |\theta_i - \theta_{i+1}|. \tag{1}$$

In the above criterion, each θ_i plays the role of a function evaluation $f(x_i)$; the first term measures the goodness-of-fit (via squared error loss) of the evaluations to the responses; the second term measures the jumpiness of the evaluations across neighboring design points, $\theta_i = f(x_i)$ and $\theta_{i+1} = f(x_{i+1})$; and $\lambda \geq 0$ is a tuning parameter determining the relative importance of the two terms for the overall minimization, with a larger λ translating into a higher importance on encouraging smoothness (mitigating jumpiness).

Reasoning about the discrete-time problem (1) can be done without appealing to sophisticated mathematics, both conceptually and formally. Arguably, this could be appropriate for an introductory course on nonparametric statistical estimation. On the other hand, consider the estimator defined by the solution of the optimization problem:[2]

$$\underset{f}{\text{minimize}} \;\; \frac{1}{2} \sum_{i=1}^n \big(y_i - f(x_i)\big)^2 + \lambda \, \mathrm{TV}(f). \tag{2}$$

[1]The use of the word "time" here is completely informal. In some applications, the input $x \in \mathcal{X}$ might actually index time, and thus the names "continuous-time" and "discrete-time" would take on a direct meaning; but in general, they are only to be understood loosely, in reference to the distinction between modeling an entire function, and modeling function evaluations, as in (2) and (1), respectively.

[2]Here and throughout, we say "the solution" only for simplicity. Problem (2), and more generally problem (7), need not admit unique solutions. The discrete-time problems (1) and (3) do, however, always admit unique solutions, because their criteria are strictly convex.

The minimization is taken over functions (for which the criterion is well-defined and finite); the first term measures the goodness-of-fit of the evaluations to the response points, as before; the second term measures the jumpiness of f, now using the total variation operator $\mathrm{TV}(\cdot)$ acting on univariate functions; and $\lambda \geq 0$ is again a tuning parameter. Relative to (1), the continuous-time problem (2) requires an appreciably higher level of mathematical sophistication, in order to develop any conceptual or formal understanding. However, problem (2) does have the distinct advantage of delivering a *function* as its solution, call it $\hat{f}$: this allows us to predict the value of the response at any point $x \in [a, b]$, via $\hat{f}(x)$.

From the solution in (1), call it $\hat{\theta}$, it is not immediately clear how to predict the response value at an arbitrary point $x \in [a, b]$. This is about choosing the "right" method for interpolating (or extrapolating, on $[a, x_1) \cup (x_n, b]$) a set of n function evaluations. To be fair, in the particular case of problem (1), its solution is generically piecewise-constant over its components $\hat{\theta}_i$, $i = 1, \dots, n$, which suggests a natural interpolant. In general, however, the task of interpolating the estimated function evaluations from a discrete-time optimization problem into an entire estimated function is far from clear-cut. Likely for this reason, the statistics literature—which places a strong emphasis, both applied and theoretical, on prediction at a new points $x \in [a, b]$—has focused primarily on the continuous-time approach to optimization-based nonparametric regression. While the discrete-time approach is popular in signal processing and econometrics, the lines of work on discrete- and continuous-time smoothing seem to have evolved mostly in parallel, with limited interplay.

The optimization problems in (1), (2) are not arbitrary examples of the discrete- and continuous-time perspectives, respectively; they are in fact deeply related to the main points of study in this paper. Interestingly, problems (1), (2) are equivalent in the sense that their solutions, denoted $\hat{\theta}$, $\hat{f}$ respectively, satisfy $\hat{\theta}_i = \hat{f}(x_i)$, $i = 1, \dots, n$. In other words, the solution in (1) reproduces the evaluations of the solution in (2) at the design points. The common estimator here is well-known, called *total variation denoising* (Rudin et al., 1992) in some parts of applied mathematics, and the *fused lasso* (Tibshirani et al., 2005) in statistics.

The equivalence between (1), (2) is a special case of a more general equivalence between classes of discrete- and continuous-time optimization problems, in which the differences $\theta_i - \theta_{i+1}$ in (1) are replaced by higher-order discrete derivatives (based on divided differences), and $\mathrm{TV}(f)$ in (2) is replaced by the total variation of a suitable derivative of f. The key mathematical object powering this connection is a linear space of univariate piecewise polynomials called *discrete splines*, which is the central focus of this paper. We dive into the details, and explain the importance of such equivalences, in the next subsection.

1.1 Motivation

The jumping-off point for the developments that follow is a generalization of the discrete-time total variation denoising problem (1), proposed independently by Steidl et al. (2006); Kim et al. (2009) (though similar ideas were around earlier, see Section 2.6), defined for an integer $k \geq 0$ by:

$$\underset{\theta}{\text{minimize}} \;\; \frac{1}{2}\|y - \theta\|_2^2 + \lambda \|\mathbb{C}_n^{k+1}\theta\|_1. \tag{3}$$

Here, $\lambda \geq 0$ is a tuning parameter, $y = (y_1, \dots, y_n) \in \mathbb{R}^n$ is the vector of response points, $\mathbb{C}_n^{k+1} \in \mathbb{R}^{(n-k-1)\times n}$ is an explicit banded matrix that corresponds to a weighted $(k+1)$st order discrete derivative operator (this can be defined in terms of the $(k+1)$st order divided difference coefficients across the design points; see the construction in (68)–(72)), and $\|\cdot\|_2$ and $\|\cdot\|_1$ are the standard ℓ_2 and ℓ_1 norms acting on vectors.

The estimator defined by solving problem (3) is known as *kth order trend filtering*. A important aspect to highlight right away is computational: since $\mathbb{C}_n^{k+1}$ is a banded matrix (with bandwidth $k + 2$), the trend filtering problem (3) can be solved efficiently using various convex optimization techniques that take advantage of this structure (see, for example, Kim et al. (2009); Arnold and Tibshirani (2016); Ramdas and Tibshirani (2016)). The original papers on trend filtering Steidl et al. (2006); Kim et al. (2009) considered the special case of evenly-spaced design points, $x_{i+1} - x_i = v > 0$, $i = 1, \dots, n-1$, where the penalty term in (3) takes a perhaps more familiar form:

$$\|\mathbb{C}_n^{k+1}\theta\|_1 = \begin{cases} \displaystyle\frac{1}{v}\sum_{i=1}^{n-1} |\theta_i - \theta_{i+1}| & \text{if } k = 0 \\ \displaystyle\frac{1}{v^2}\sum_{i=1}^{n-2} |\theta_i - 2\theta_{i+1} + \theta_{i+2}| & \text{if } k = 1 \\ \displaystyle\frac{1}{v^3}\sum_{i=1}^{n-3} |\theta_i - 3\theta_{i+1} + 3\theta_{i+2} - \theta_{i+3}| & \text{if } k = 2, \end{cases} \tag{4}$$

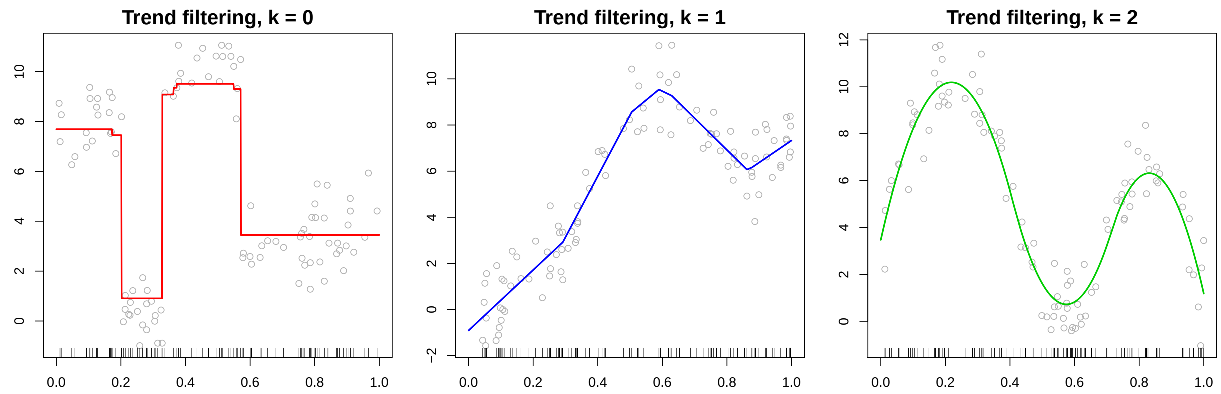

Figure 1: (Adapted from Tibshirani (2014).) Example trend filtering estimates for $k = 0$, $k = 1$, and $k = 2$, exhibiting piecewise constant, piecewise linear, and piecewise quadratic behavior, respectively. In each panel, the $n = 100$ design points are marked by ticks on the horizontal axis (note that they are not evenly-spaced).

and so forth, where for a general $k \geq 0$, the penalty is a $1/v^{k+1}$ times a sum of absolute $(k+1)$st forward differences. (The factor of $1/v^{k+1}$ can always be abosrbed into the tuning parameter λ; and so we can see that (3) reduces to (1) for $k = 0$, modulo a rescaling of λ). The extension of trend filtering to arbitrary (unevenly-spaced) design points is due to Tibshirani (2014). The continuous-time (functional) perspective on trend filtering is also due to Tibshirani (2014), which we describe next.

Connections to continuous-time. To motivate the continuous-time view, consider $\mathbb{C}_n^{k+1}\theta$, the vector of (weighted) $(k+1)$st discrete derivatives of θ across the design points: since discrete differentiation is based on iterated differencing, we can equivalently interpret $\mathbb{C}_n^{k+1}\theta$ as a vector of *differences* of kth discrete derivatives of θ at adjacent design points. By the sparsity-inducing property of the ℓ_1 norm, the penalty in problem (3) thus drives the kth discrete derivatives of θ to be equal at adjacent design points, and the trend filtering solution $\hat{\theta}$ generically takes on the structure of a kth degree piecewise polynomial (as its kth discrete derivative will be piecewise constant), with adaptively-chosen knots (points at which the kth discrete derivative changes). This intuition is readily confirmed by empirical examples; see Figure 1.

These ideas were formalized in Tibshirani (2014), and then developed further in Wang et al. (2014). These papers introduced what were called *kth degree falling factorial basis*, a set of functions defined as

$$
\begin{aligned}
h_j^k(x) &= \frac{1}{(j-1)!} \prod_{\ell=1}^{j-1} (x - x_\ell), \quad j = 1, \dots, k+1, \\
h_j^k(x) &= \frac{1}{k!} \prod_{\ell=j-k}^{j-1} (x - x_\ell) \cdot 1\{x > x_{j-1}\}, \quad j = k+2, \dots, n.
\end{aligned}
\tag{5}
$$

(Note that this basis depends on the design points $x_1, \dots, x_n$, though this is notationally suppressed.) The functions in (5) are kth degree piecewise polynomials, with knots at $x_{k+1}, \dots, x_{n-1}$. Here and throughout, we interpret the empty product to be equal to 1, for convenience (that is, $\prod_{i=1}^{0} a_i = 1$). Note the similarity of the above basis and the standard truncated power basis for splines, with knots at $x_{k+1}, \dots, x_{n-1}$ (see (14)); in fact, when $k = 0$ or $k = 1$, the two bases are equal, and the above falling factorial functions are exactly splines; but when $k \geq 2$, this is no longer true—the above falling factorial functions are piecewise polynomials with *discontinuities* in their derivatives of orders $1, \dots, k-1$ (see (53), (54)), and thus span a different space than that of kth degree splines.

The key result connecting (5) and (3) was given in Lemma 5 of Tibshirani (2014) (see also Lemma 2 of Wang et al. (2014)), and can be explained as follows. For each $\theta \in \mathbb{R}^n$, there is a function in the span of the falling factorial basis, $f \in \mathrm{span}\{h_1^k, \dots, h_n^k\}$, with two properties: first, f interpolates each θ_i at x_i, which we write as $\theta = f(x_{1:n})$, where $f(x_{1:n}) = (f(x_1), \dots, f(x_n)) \in \mathbb{R}^n$ denotes the vector of evaluations of f at the design points; and second

$$
\mathrm{TV}(D^k f) = \big\| \mathbb{C}_n^{k+1} f(x_{1:n}) \big\|_1. \tag{6}
$$

On the right-hand side is the trend filtering penalty, which, recall, we can interpret as a sum of absolute differences of kth discrete derivatives of f over the design points, and therefore as a type of total variation penalty on the kth discrete derivative. On the left-hand side above, we denote by $D^k f$ the kth derivative of f (which we take to mean the kth left derivative when this does not exist), and by $\mathrm{TV}(\cdot)$ the usual total variation operator on functions. Hence, taking total variation of the kth derivative as our smoothness measure, the property in (6) says that the interpolant f of θ is *exactly as smooth* in continuous-time as θ is in discrete-time.

Reflecting on this result, the first property—that f interpolates θ_i at x_i, for $i = 1, \dots, n$—is of course not special in it of itself. Any rich enough function class, of dimension at least n, will admit such a function. However, paired with the second property (6), the result becomes interesting, and even somewhat surprising. Said differently, any function f lying in the span of the kth degree falling factorial basis has the property that its discretization to the design points is *lossless* with respect to the total variation smoothness functional $\mathrm{TV}(D^k f)$: this information is exactly preserved by $\theta = f(x_{1:n})$. Denoting by $\mathcal{H}_n^k = \mathrm{span}\{h_1^k, \dots, h_n^k\}$ the span of falling factorial functions, we thus see that the trend filtering problem (3) is equivalent to the variational problem:

$$\underset{f \in \mathcal{H}_n^k}{\text{minimize}} \;\; \frac{1}{2} \sum_{i=1}^n \big(y_i - f(x_i) \big)^2 + \lambda \, \mathrm{TV}(D^k f), \tag{7}$$

in the sense that at the solutions $\hat{\theta}, \hat{f}$ in problems (3), (7), respectively, we have $\hat{\theta} = \hat{f}(x_{1:n})$. Moreover, it turns out that forming $\hat{f}$ from $\hat{\theta}$ is straightforward: starting with the falling factorial basis expansion $\hat{f} = \sum_{j=1}^n \hat{\alpha}_j h_j^k$, and then writing the coefficient vector in block form $\hat{\alpha} = (\hat{a}, \hat{b}) \in \mathbb{R}^{k+1} \times \mathbb{R}^{n-k-1}$, the piecewise polynomial basis coefficients are given by $\hat{b} = \mathbb{C}_n^{k+1} \hat{\theta}$, and the polynomial basis coefficients $\hat{a}$ can also be expressed simply in terms of lower-order discrete derivatives. This shows that $\hat{f}$ is a kth degree piecewise polynomial, with knots occurring at the nonzeros of $\mathbb{C}_n^{k+1} \hat{\theta}$, that is, at changes in the kth discrete derivative of $\hat{\theta}$, formally justifying the intuition about the structure of $\hat{\theta}$ given above.

Reflections on the equivalence. One might say that the developments outlined above bring trend filtering closer to the "statistical mainstream": we move from being able to estimate the values of the regression function f_0 at the design points $x_1, \dots, x_n$ to being able to estimate f_0 itself. This has several uses: practical—we can use the interpolant $\hat{f}$ to estimate $f_0(x)$ at unseen values of x; conceptual—we can better understand what kinds of "shapes" trend filtering is inclined to produce, via the representation in terms of falling factorial functions; and theoretical—we can tie (7) to an unconstrained variational problem, where we minimize the same criterion over *all* functions f (for which the criterion is well-defined and finite):

$$\underset{f}{\text{minimize}} \;\; \frac{1}{2} \sum_{i=1}^n \big(y_i - f(x_i) \big)^2 + \lambda \, \mathrm{TV}(D^k f). \tag{8}$$

This minimization is in general computationally difficult, but its solution, called the *locally adaptive regression spline* estimator (Mammen and van de Geer, 1997) has favorable theoretical properties, in terms of its rate of estimation of f_0 (see Section 2.5 for a review). By showing that the falling factorial functions are "close" to certain splines, Tibshirani (2014); Wang et al. (2014) showed that the solution in (7) is "close" to that in (8), and thus trend filtering inherits the favorable estimation guarantees of the locally adaptive regression spline (which is important because trend filtering is computationally easier; for more, see Sections 2.5 and 2.6).

The critical device in all of this were the falling factorial basis functions (5), which provide the bridge between the discrete and continuous worlds. This now brings us to the motivation for the current paper. One has to wonder: did we somehow get "lucky" with trend filtering and this basis? Do the falling factorial functions have other properties aside from (6), that is, aside from equating (3) and (7)? At the time of writing Tibshirani (2014); Wang et al. (2014) (and even in subsequent work on trend filtering), we were not fully aware of the relationship of the falling factorial functions and what appears to be fairly classical work in numerical analysis. First and foremost:

> The span $\mathcal{H}_n^k = \mathrm{span}\{h_1^k, \dots, h_n^k\}$ of the kth degree falling factorial basis functions is a special space of piecewise polynomials known as *kth degree discrete splines*.

Discrete splines have been studied since the early 1970s by applied mathematicians, beginning with Mangasarian and Schumaker (1971, 1973). The current paper recasts some of our previous work on trend filtering to better connect it to the discrete spline literature, reviews some relevant existing results on discrete splines and discusses the implications for trend filtering and related problems, and lastly, contributes some new results and perspectives on discrete splines.

1.2 Summary

An outline and summary of this paper is as follows.

- In Section 2, we provide relevant background and historical remarks.
- In Section 3, we give a new perspective on how to construct the falling factorial basis "from scratch". We start by defining a natural discrete derivative operator and its inverse, a discrete integrator. We then show that the falling factorial basis functions are given by kth order discrete integration of appropriate step functions (Theorem 2).
- In Section 4, we verify that the span of the falling factorial basis is indeed a space of discrete splines (Lemma 3), and establish that functions in this span satisfy a key matching derivatives property: their kth discrete derivative matches their kth derivative everywhere, and moreover, they are the *only* kth degree piecewise polynomials with this property (Corollary 1).
- In Section 5, we give a dual basis to the falling factorial basis, based on evaluations of discrete derivatives. As a primary use case, we show how to use such a dual basis to perform efficient interpolation in the falling factorial basis, which generalizes Newton's divided difference interpolation formula (Theorem 3). We also show that this interpolation formula can be recast in an implicit manner, which reveals that interpolation using discrete splines can be done in *constant-time* (Corollary 2), and further, discrete splines are uniquely determined by this implicit result: they are the *only* functions that satisfy such an implicit interpolation formula (Corollary 3).
- In Section 6, we present a matrix-centric view of the results given in previous sections, drawing connections to the way some related results have been presented in past papers. We review specialized methods for fast matrix operations with discrete splines from Wang et al. (2014).
- In Section 7, we present a new discrete B-spline basis for discrete splines (it is new for arbitrary designs, and our construction here is a departure from the standard one): we first define these basis functions as discrete objects, by fixing their values at the design points, and we then define them as continuum functions, by interpolating these values within the space of discrete splines, using the implicit interpolation view (Lemma 8). We show how this discrete B-spline basis can be easily modified to provide a basis for discrete natural splines (Lemma 9).
- In Section 8, we demonstrate how the previous results and developments can be ported over to the case where the knot set that defines the space of discrete splines is an arbitrary (potentially sparse) subset of the design points. An important find here is that the discrete B-spline basis provides a much more stable (better-conditioned) basis for solving least squares problems involving discrete splines.
- In Section 9, we present two representation results for discrete splines. First, we review a result from Tibshirani (2014); Wang et al. (2014) on representing the total variation functional $\mathrm{TV}(D^k f)$ for a kth degree discrete spline f in terms of a sum of absolute differences of its kth discrete derivatives (Theorem 4). (Recall that we translated this in (6).) Second, we establish a new result on representing the L_2-Sobolev functional $\int_a^b (D^m f)(x)^2 \, dx$ for a $(2m-1)$st degree discrete spline f in terms of a quadratic form of its mth discrete derivatives (Theorem 5).
- In Section 10, we derive some simple (crude) approximation bounds for discrete splines, over bounded variation spaces.
- In Section 11, we revisit trend filtering. We discuss some potential computational improvements, stemming from the development of discrete B-splines and their stability properties. We also show that the optimization domain in trend filtering can be further restricted to the space of discrete natural splines by adding simple linear constraints to the original problem, and that this modification can lead to better boundary behavior.
- In Section 12, we revisit Bohlmann-Whittaker (BW) filtering. In the case of arbitrary design points, we propose a simple modification of the BW filter using a weighted penalty, which for $m = 1$ reduces to the linear smoothing spline. For $m = 2$, we derive a deterministic bound on the ℓ_2 distance between the weighted cubic BW filter and the cubic smoothing spline (Theorem 7). We use this, in combination with classical nonparametric regression theory for smoothing splines, to prove that the weighted BW filter attains minimax optimal estimation rates over the appropriate L_2-Sobolev classes (Corollary 4).

Most proofs are deferred to Appendix B. Other relevant technical details (background and otherwise) are deferred to Appendices C and D.

1.3 Notation

Here is an overview of some general notation used in this paper. For integers $a \le b$, we use $z_{a:b} = \{z_a, z_{a+1}, \dots, z_b\}$. For a set C, we use 1_C for the indicator function of C, that is, $1_C(x) = 1\{x \in C\}$. We write $f|_C$ for the restriction of a function f to C. We use D for the differentiation operator, and I for the integration operator: acting on functions f on $[a, b]$, we take If to itself be a function on $[a, b]$, defined by

$$(If)(x) = \int_a^x f(t)\, dt.$$

For a nonnegative integer k, we use D^k and I^k to denote k repeated applications (that is, k times composition) of the differentiation and integration operators, respectively. In general, when the derivative of a function f does not exist, we interpret Df to mean the *left* derivative, assuming the latter exists, and the same with $D^k f$.

An important note: we refer to a kth degree piecewise polynomial that has $k-1$ continuous derivatives as a spline of *degree* k, whereas much of the classical literature refers to this as a spline of *order* $k+1$; we specifically avoid the use of the word "order" when it comes to such functions or functions spaces, to avoid confusion.

Finally, throughout, we use "blackboard" fonts for matrices (such as $\mathbb{F}, \mathbb{G}$, etc.), in order to easily distinguish them from operators that act on functions (for which we use F, G, etc.). The only exceptions are that we reserve $\mathbb{R}$ to denote the set of real numbers and $\mathbb{E}$ to denote the expectation opterator.

For a more detailed summary of notation, and discrete-continuum analogies or equivalences, see Appendix A.

2 Background

We provide background on various topics that will play important roles in the remainder of the paper. Of course, we do not intend to give a comprehensive review of any of the subjects covered, just the basic elements needed for what follows. We also use this space to make historical remarks and discuss related work.

2.1 Divided differences

Divided differences have a very old, rich history in mathematics, and are usually attributed to Newton (due to Newton (1687, 1711)). They also serve a one of the primary building blocks in classical numerical analysis (for example, see Whittaker and Robinson (1924)). For a beautiful review of divided differences, their properties, and connections, see de Boor (2005). Given a univariate function f, the divided difference of f at distinct points z_1, z_2 is defined by

$$f[z_1, z_2] = \frac{f(z_2) - f(z_1)}{z_2 - z_1},$$

and more generally, for an integer $k \ge 1$, the kth order divided difference at distinct $z_1, \dots, z_{k+1}$ is defined by

$$f[z_1, \dots, z_{k+1}] = \frac{f[z_2, \dots, z_{k+1}] - f[z_1, \dots, z_k]}{z_{k+1} - z_1}.$$

(For this to reduce to the definition in the previous display, when $k = 1$, we take by convention $f[z] = f(z)$.) We refer to the points $z_1, \dots, z_{k+1}$ used to define the divided difference above as *centers*. Note that these centers do not need to be in sorted order for this definition to make sense, and the definition of a divided difference is invariant to the ordering of centers: $f[z_1, \dots, z_{k+1}] = f[z_{\sigma(1)}, \dots, z_{\sigma(k+1)}]$ for any permutation σ acting on $\{1, \dots, k+1\}$. (We also note that requiring the centers to be distinct is not actually necessary, but we will maintain this assumption for simplicity; for a more general definition that allows for repeated centers, see, for example, Definition 2.49 in Schumaker (2007).)

A notable special case is when the centers are evenly-spaced, say, $z + iv$, $i = 0, \dots, k$, for some spacing $v > 0$, in which case the divided difference becomes a (scaled) *forward difference*, or equivalently a (scaled) *backward difference*,

$$k! \cdot f[z, \dots, z + kv] = \frac{1}{v^k}(F_v^k f)(z) = \frac{1}{v^k}(B_v^k f)(z + kv),$$

where we use F_v^k, B_v^k to denote the kth order forward and backward difference operators, respectively; to be explicit, we recall that $(F_v^k f)(z) = \sum_{i=0}^k (-1)^{k-i} \binom{k}{i} f(z + iv)$.

Linear combination formulation. It is not hard to see that divided differences are linear combinations of function evaluations. A simple calculation reveals the exact form of the coefficients in this linear combination, for example,

$$\begin{aligned} f[z_1, z_2, z_3] &= \frac{f[z_1, z_2]}{z_1 - z_3} + \frac{f[z_2, z_3]}{z_3 - z_1} \\ &= \frac{f(z_1)}{(z_1 - z_2)(z_1 - z_3)} + \frac{f(z_2)}{(z_2 - z_1)(z_1 - z_3)} + \frac{f(z_2)}{(z_2 - z_3)(z_3 - z_1)} + \frac{f(z_3)}{(z_3 - z_2)(z_3 - z_1)} \\ &= \frac{f(z_1)}{(z_1 - z_2)(z_1 - z_3)} + \frac{f(z_2)}{(z_2 - z_1)(z_2 - z_3)} + \frac{f(z_3)}{(z_3 - z_2)(z_3 - z_1)}. \end{aligned}$$

By an inductive argument (whose inductive step is similar to the calculation above), we may also write for a general order $k \geq 1$,

$$f[z_1, \ldots, z_{k+1}] = \sum_{i=1}^{k+1} \frac{f(z_i)}{\prod_{j \in \{1,\ldots,k+1\} \setminus \{i\}} (z_i - z_j)}. \tag{9}$$

This expression is worth noting because it is completely explicit, but it is not often used, and the recursive formulation given previously is the more common view of divided differences.

Newton interpolation. For distinct points $t_{1:r} = \{t_1, \ldots, t_r\}$, we denote the *Newton polynomial* based on $t_{1:r}$ by

$$\eta(x; t_{1:r}) = \prod_{j=1}^{r} (x - t_j). \tag{10}$$

Here, when $r = 0$, we set $t_{1:0} = \emptyset$ and $\eta(x; t_{1:0}) = 1$ for notational convenience. It is important to note that the pure polynomial functions in the falling factorial basis, given in the first line of (5), are simply Newton polynomials, and the piecewise polynomial functions, given in the second line of (5), are *truncated* Newton polynomials:

$$\begin{aligned} h_j^k(x) &= \frac{1}{(j-1)!} \eta(x; x_{1:j}), \quad j = 1, \ldots, k+1, \\ h_j^k(x) &= \frac{1}{k!} \eta(x; x_{(j-k):(j-1)}) \cdot 1\{x > x_{j-1}\}, \quad j = k+2, \ldots, n. \end{aligned}$$

In this light, it would also be appropriate to call the basis in (5) the *truncated Newton polynomial basis*, but we stick to the name falling factorial basis for consistency with our earlier work (and Chapter 8.5 of Schumaker (2007)).

Interestingly, Newton polynomials and divided differences are closely connected, via *Newton's divided difference interpolation formula* (see, for example, Proposition 7 in de Boor (2005)), which says that for a polynomial p of degree k, and any centers $t_1, \ldots, t_{k+1}$,

$$p(x) = \sum_{j=1}^{k+1} p[t_1, \ldots, t_j] \cdot \eta(x; t_{1:(j-1)}). \tag{11}$$

One of our main developments later, in Theorem 3, may be seen as extending (11) to interpolation with truncated Newton polynomials (that is, with the falling factorial basis). In particular, compare (11) and (64).

An important fact about the representation in (11) is that it is unique (meaning, any kth degree polynomial can only be written as a linear combination of Newton polynomials in one particular way, which is given by (11)). This property has the following implication for divided differences of Newton polynomials (that we will use extensively in later parts of this paper): for any integer $r \geq 0$, and any centers $t_1, \ldots, t_j$,

$$\eta(\cdot; t_{1:r})[t_1, \ldots, t_j] = \begin{cases} 1 & \text{if } j = r+1 \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

The result is clear when $j = r + 1$ and $j > r + 1$ (in these cases, it is a statement about a jth order divided difference of a polynomial of degree at most j, for example, see (58)). However, it is perhaps less obvious for $j < r + 1$ (in this case it is a statement about a jth order divided difference of a polynomial of degree greater than j).

2.2 Splines

Splines play a central role in numerical analysis, approximation theory, and nonparametric statistics. The "father" of spline theory is widely considered to be Schoenberg (due to Schoenberg (1946a,b), where Schoenberg also introduces the terminology "spline function"). It should be noted that in the early 1900s, there were many papers written about splines (without using this name), and piecewise polynomial interpolation, more generally; for a survey of this work, see Greville (1944). For two wonderful books on splines, see de Boor (1978); Schumaker (2007). We will draw on the latter book extensively throughout this paper.

In simple terms, a spline is a piecewise polynomial having continuous derivatives of all orders lower than the degree of the polynomial. We can make this definition more precise as follows.

Definition 1. For an integer $k \geq 0$, and knots $a = t_0 < t_1 < \cdots < t_r < t_{r+1} = b$, we define the space of *kth degree splines* on $[a, b]$ with knots $t_{1:r}$, denoted $\mathcal{S}^k(t_{1:r}, [a, b])$, to contain all functions f on $[a, b]$ such that

$$\begin{aligned} &\text{for each } i = 0, \dots, r, \text{ there is a } k\text{th degree polynomial } p_i \text{ such that } f|_{I_i} = p_i|_{I_i}, \text{ and} \\ &\text{for each } i = 1, \dots, r, \text{ it holds that } (D^\ell p_{i-1})(t_i) = (D^\ell p_i)(t_i),\ \ell = 0, \dots, k-1, \end{aligned} \tag{13}$$

where $I_0 = [t_0, t_1]$ and $I_i = (t_i, t_{i+1}]$, $i = 1, \dots, r$.

We write Definition 1 in this particular way because it makes it easy to compare the definition of discrete splines in Definition 2 (and in Definition 3 for the case of arbitrary design points). The simplest basis for the space $\mathcal{S}^k(t_{1:r}, [a, b])$ is the *kth degree truncated power basis*, defined by

$$\begin{aligned} g_j^k(x) &= \frac{1}{(j-1)!} x^j, \quad j = 1, \dots, k+1, \\ g_{j+k+1}^k(x) &= \frac{1}{k!}(x - t_j)_+^k, \quad j = 1, \dots, r, \end{aligned} \tag{14}$$

where $x_+ = \max\{x, 0\}$. When $k = 0$, we interpret $(x - t)_+^0 = 1\{x > t\}$; this choice (strict versus nonstrict inequality) is arbitrary, but convenient, and consistent with our choice for the falling factorial basis in (5).

An alternative basis for splines, which has local support and is therefore highly computationally appealing, is given by the *B-spline* basis. In fact, most authors view B-splines as *the* basis for splines—not only for computational reasons, but also because building splines out of linear combinations of B-splines makes so many of their important properties transparent. To keep this background section (relatively) short, we defer discussion of B-splines until Appendix C.1.

2.3 Discrete splines

Discrete splines were introduced by Mangasarian and Schumaker (1971, 1973), then further developed by Schumaker (1973); Lyche (1975); de Boor (1976), among others. As far as we know, the most comprehensive summary of discrete splines and their properties appears to be Chapter 8.5 of Schumaker (2007).

In words, a discrete spline is similar to a spline, except in the required smoothness conditions, forward differences are used instead of derivatives. This can be made precise as follows.

Definition 2. For an integer $k \geq 0$, design points $[a, b]_v = \{a, a + v, \dots, b\}$ with $v > 0$ and $b = a + Nv$, and knots $a = t_0 < t_1 < \cdots < t_r < t_{r+1} = b$ with $t_{1:r} \subseteq [a, b]_v$ and $t_r \leq b - kv$, we define the space of *kth degree discrete splines* on $[a, b]$ with knots $t_{1:r}$, denoted $\mathcal{DS}_v^k(t_{1:r}, [a, b]_v)$, to contain all functions f on $[a, b]_v$ such that

$$\begin{aligned} &\text{for each } i = 0, \dots, r, \text{ there is a } k\text{th degree polynomial } p_i \text{ such that } f|_{I_{i,v}} = p_i|_{I_{i,v}}, \text{ and} \\ &\text{for each } i = 1, \dots, r, \text{ it holds that } (F_v^\ell p_{i-1})(t_i) = (F_v^\ell p_i)(t_i),\ \ell = 0, \dots, k-1, \end{aligned} \tag{15}$$

where $I_{0,v} = [t_0, t_1] \cap [a, b]_v$ and $I_{i,v} = (t_i, t_{i+1}] \cap [a, b]_v$, $i = 1, \dots, r$.

Remark 1. Comparing the conditions in (15) and (13), we see that when $k = 0$ or $k = 1$, the space $\mathcal{DS}_v^k(t_{1:r}, [a, b]_v)$ of kth degree discrete splines with knots $t_{1:r}$ is the essentially equivalent to the space $\mathcal{S}^k(t_{1:r}, [a, b])$ of kth degree splines with knots $t_{1:r}$ (precisely, for $k = 0$ and $k = 1$, functions in $\mathcal{DS}_v^k(t_{1:r}, [a, b]_v)$ are the restriction of functions in $\mathcal{S}^k(t_{1:r}, [a, b])$ to $[a, b]_v$). This is not true for $k \geq 2$, in which case the two spaces are genuinely different.

As covered in Chapter 8.5 of Schumaker (2007), various properties of discrete splines can be developed in a parallel fashion to splines. For example, instead of the truncated power basis (14), the following is a basis for $\mathcal{DS}_v^k(t_{1:r}, [a,b]_v)$ (Theorem 8.51 of Schumaker (2007)):

$$
\begin{aligned}
f_j^k(x) &= \frac{1}{(j-1)!}(x-a)_{j-1,v} \quad j = 1, \dots, k+1, \\
f_j^k(x) &= \frac{1}{k!}(x-t_j)_{k,v} \cdot 1\{x > t_j\}, \quad j = 1, \dots, r,
\end{aligned}
\tag{16}
$$

where we write $(x)_{\ell,v} = x(x-v)\cdots(x-(\ell-1)v)$ for the falling factorial polynomial of degree ℓ with gap v, which we take to be equal to 1 when $\ell = 0$. Note that the above basis is an evenly-spaced analog of the falling factorial basis in (5); in fact, Schumaker refers to f_j^k, $j = 1, \dots, r+k+1$ as "one-sided factorial functions", which is (coincidentally) a very similar name to that we gave to (5), in our previous papers. In addition, a local basis for $\mathcal{DS}_v^k(t_{1:r}, [a,b]_v)$, akin to B-splines and hence called *discrete B-splines*, can be formed in an analogous fashion to that for splines; we defer discussion of this until Appendix C.2.

It should be noted that most of the classical literature, as well as Chapter 8.5 of Schumaker (2007), studies discrete splines in the special case of evenly-spaced design points $[a,b]_v$. Furthermore, the classical literature treats discrete splines as discrete objects, that is, as *vectors*: see Definition 2, which is concerned only with the evaluations of f over the discrete set $[a,b]_v$. The assumption of evenly-spaced design points is not necessary, and in the current paper we consider discrete splines with arbitrary design points. We also treat discrete splines as continuum objects, namely, as *functions* defined over the continuum interval $[a,b]$. To be clear, we do not intend to portray such extensions alone as particularly original or important contributions. Rather, it is the *perspective* that we offer on discrete splines that (we believe) is important—this starts with constructing a basis via discrete integration of indicator functions in Section 3, which then leads to the development of new properties, such as the matching derivatives property in Section 4.2, and the implicit interpolation formula in Section 5.4.

2.4 Smoothing splines

Let $x_{1:n} = \{x_1, \dots, x_n\} \in [a,b]$ be design points, assumed to be ordered, as in $x_1 < \dots < x_n$, and let $y_1, \dots, y_n$ be associated response points. For an odd integer $k = 2m-1 \geq 1$, the *kth degree smoothing spline* estimator is defined as the solution of the variational optimization problem:

$$
\underset{f}{\text{minimize}} \; \sum_{i=1}^n \big(y_i - f(x_i)\big)^2 + \lambda \int_a^b (D^m f)(x)^2 \, dx,
\tag{17}
$$

where $\lambda \geq 0$ is a regularization parameter, and the domain of the minimization in (17) is all functions f on $[a,b]$ that are m times weakly differentiable, with $\int_a^b (D^m f)(x)^2 \, dx < \infty$; this is known as as the L_2-Sobolev space of order m, and denoted $\mathcal{W}^{m,2}([a,b])$. The smoothing spline estimator was first proposed by Schoenberg (1964), where he asserts (appealing to logic from previous work on spline interpolation) that the solution in (17) is unique, and is a kth degree spline belonging to $\mathcal{S}^k(x_{1:n}, [a,b])$. In fact, the solution in (17) is a special type of spline that reduces to a polynomial of degree $m-1$ on the boundary intervals $[a, x_1]$ and $[x_n, b]$, which is called a *natural spline* of degree $k = 2m-1$. To fix notation, we will denote the space of kth degree natural splines on $[a,b]$ with knots $x_{1:n}$ by $\mathcal{NS}^k(x_{1:n}, [a,b])$.

Following Schoenberg's seminal contributions, smoothing splines have become the topic of a vast body of work in both applied mathematics and statistics, with work in the latter community having been pioneered by Grace Wahba and coauthors; see, for example, Craven and Wahba (1978) for a notable early paper. Two important books on the statistical perspective underlying smoothing splines are Wahba (1990); Green and Silverman (1993). Today, smoothing splines are undoubtedly one of the most widely used tools for univariate nonparametric regression.

Connections to discrete-time. An interesting historical note, which is perhaps not well-known (or at least it seems to have been largely forgotten in discussions on motivation for the smoothing spline from a modern point of view), is that in creating the smoothing spline, Schoenberg was motivated by the much earlier discrete-time smoothing (graduation) approach of Whittaker (1923), stating this explicitly in Schoenberg (1964). Whittaker's approach, see (34), estimates smoothed values by minimizing the sum of a squared loss term and a penalty term of squared mth divided differences (Whittaker takes $m = 3$); meanwhile, Schoenberg's approach (17), "in an attempt to combine [spline interpolation ...] with Whittaker's idea", replaces mth divided differences with mth derivatives. Thus, while Schoenberg was motivated to move from a discrete-time to a continuous-time perspective on smoothing, we are, as one of the main themes in this paper, interested in returning to the discrete-time perspective, and ultimately, connecting the two.

Given this, it is not really a surprise that Schoenberg himself derived the first concrete connection between the two perspectives, continuous and discrete. Next we transcribe his result from Schoenberg (1964), and we include a related result from Reinsch (1967).

Theorem 1 (Schoenberg (1964); Reinsch (1967))**.** *For any odd integer $k = 2m-1 \geq 1$, and any kth degree natural spline $f \in \mathcal{NS}^k(x_{1:n}, [a,b])$ with knots in $x_{1:n}$, it holds that*

$$\int_a^b (D^m f)(x)^2 \, dx = \big\|(\mathbb{K}_n^m)^{\frac{1}{2}} \mathbb{D}_n^m f(x_{1:n})\big\|_2^2, \tag{18}$$

where $f(x_{1:n}) = (f(x_1), \dots, f(x_n)) \in \mathbb{R}^n$ is the vector of evaluations of f at the design points, and $\mathbb{D}_n^m \in \mathbb{R}^{(n-m)\times n}$ is the mth order discrete derivative matrix, as in (70). Furthermore, $\mathbb{K}_n^m \in \mathbb{R}^{(n-m)\times(n-m)}$ is a symmetric matrix (that depends only on $x_{1:n}$), with a banded inverse of bandwidth $2m-1$. If we abbreviate, for $i = 1, \dots, n-m$, the function $P_i^{m-1} = P^{m-1}(\cdot; x_{i:(i+m)})$, which is the degree $m-1$ B-spline with knots $x_{i:(i+m)}$, defined in (183) in Appendix C.1, then we can write the entries of $(\mathbb{K}_n^m)^{-1}$ as

$$(\mathbb{K}_n^m)^{-1}_{ij} = m^2 \int_a^b P_i^{m-1}(x) P_j^{m-1}(x) \, dx. \tag{19}$$

For $m = 1$, this matrix is diagonal, with entries

$$(\mathbb{K}_n)^{-1}_{ii} = \frac{1}{x_{i+1} - x_i}. \tag{20}$$

For $m = 2$, this matrix is tridiagonal, with entries

$$(\mathbb{K}_n^2)^{-1}_{ij} = \begin{cases} \dfrac{4}{3(x_{i+2} - x_i)} & \text{if } i = j \\ \dfrac{2(x_{i+1} - x_i)}{3(x_{i+2} - x_i)(x_{i+1} - x_{i-1})} & \text{if } i = j+1. \end{cases} \tag{21}$$

The matrix $\mathbb{D}_n^m \in \mathbb{R}^{(n-m)\times n}$ appearing in Theorem 1 is to be defined (and studied in detail) later, in (70). Acting on a vector $f(x_{1:n}) = (f(x_1), \dots, f(x_n)) \in \mathbb{R}^n$, it gives $m!$ times the appropriate divided differences of f, namely,

$$\big(\mathbb{D}_n^m f(x_{1:n})\big)_i = m! \cdot f[x_i, \dots, x_{i+m}], \quad i = 1, \dots, n-m. \tag{22}$$

Schoenberg (1964) states the result in (19) without proof. Reinsch (1967) derives the explicit form in (21), for $m = 2$, using a somewhat technical proof that stems from the Euler-Lagrange conditions for the variational problem (17). We give a short proof all results (19), (20), (21) in Theorem 1 in Appendix B.1, based on the Peano representation for the B-spline (to be clear, we make no claims of originality, this is simply done for completeness).

Remark 2. Theorem 1 reveals that the variational smoothing spline problem (17) can be recast as a finite-dimensional convex quadratic program (relying on the fact that the solution in this problem lies in $\mathcal{NS}^k(x_{1:n}, [a,b])$ for $k = 2m-1$):

$$\underset{\theta}{\text{minimize}} \;\; \|y - \theta\|_2^2 + \lambda \big\|(\mathbb{K}_n^m)^{\frac{1}{2}} \mathbb{D}_n^m \theta\big\|_2^2, \tag{23}$$

for $y = (y_1, \dots, y_n) \in \mathbb{R}^n$, and $\mathbb{K}_n^m$ as defined in (19). The solutions $\hat\theta, \hat f$ in problems (23), (17), respectively, satisfy $\hat\theta = \hat f(x_{1:n})$. Furthermore, from (23), the solution is easily seen to be

$$\hat\theta = \big(\mathbb{I}_n + \lambda (\mathbb{D}_n^m)^\mathsf{T} \mathbb{K}_n^m \, \mathbb{D}_n^m\big)^{-1} y, \tag{24}$$

where $\mathbb{I}_n$ denotes the $n \times n$ identity matrix. Despite the fact that $\mathbb{K}_n^m$ is itself dense for $m \geq 2$ (recall that its inverse is banded with bandwidth $2m-1$), the smoothing spline solution $\hat\theta$ in (24) can be computed in linear-time using a number of highly-efficient, specialized approaches (see, for example, Chapter XIV of de Boor (1978)).

Remark 3. It is interesting to compare (23) and what we call the Bohlmann-Whittaker (BW) filter (37) (note that the traditional case studied by Bohlmann and Whittaker was unit-spaced design points, as in (36), and problem (37) was Whittaker's proposed extension to arbitrary design points). We can see that the smoothing spline problem reduces to a modified version of the discrete-time BW problem, where $\|\mathbb{D}_n^m \theta\|_2^2$ is replaced by the quadratic form $\|(\mathbb{K}_n^m)^{\frac{1}{2}} \mathbb{D}_n^m \theta\|_2^2$, for a matrix $\mathbb{K}_n^m$ having a banded inverse. To preview one of our later results, in Theorem 5: by restricting the domain in problem (17) to discrete splines, it turns out we can obtain another variant of the BW filter where the corresponding matrix $\mathbb{K}_n^m$ is now *itself* banded.

2.5 Locally adaptive splines

Smoothing splines have many strengths, but adaptivity to changes in the local level of smoothness is not one of them. That is, if the underlying regression function f_0 is smooth in some parts of its domain and wiggly in other parts, then the smoothing spline will have trouble estimating f_0 adequately throughout. It is not alone: any *linear smoother*—meaning, an estimator $\hat{f}$ of f_0 whose fitted values $\hat{\theta} = \hat{f}(x_{1:n})$ are a linear function of the responses y—will suffer from the same problem, as made precise by the influential work of Donoho and Johnstone (1998). (From (24), it is easy to check that the smoothing spline estimator is indeed a linear smoother.) We will explain this point in more detail shortly.

Aimed at addressing this very issue, Mammen and van de Geer (1997) proposed an estimator based on solving the variational problem (8), which recall, for a given integer $k \geq 0$, is known as the kth degree locally adaptive regression spline estimator. (It is worth noting that the same idea was proposed earlier by Koenker et al. (1994), who studied total variation smoothing of the first derivative, $k = 1$, in nonparametric quantile regression.) We can see that (8) is like the smoothing spline problem (17), but with the L_2-Sobolev penalty is replaced by a (higher-order) total variation penalty on f. Note that when f is $k+1$ times weakly differentiable on an interval $[a, b]$, we have

$$\mathrm{TV}(D^k f) = \int_a^b |(D^{k+1} f)(x)| \, dx. \tag{25}$$

In this sense, we can interpret problem (8) as something like the L_1 analog of problem (17). Importantly, note that the fitted values $\hat{\theta} = \hat{f}(x_{1:n})$ from the locally adaptive regression spline estimator are *not* a linear function of y, that is, the locally adaptive regression spline estimator is not a linear smoother.

Local adaptivity. True to its name, the locally adaptive regression spline estimator is more attuned to the local level of smoothness in f_0 compared to the smoothing spline. This is evident both empirically and theoretically. See Figure 2 for an empirical example. In terms of theory, there are clear distinctions in the optimality properties belonging to linear and nonlinear methods. In classical nonparametric regression, linear smoothers such as smoothing splines are typically analyzed for their rates of estimation of an underlying function f_0 when the latter is assumed to lie in a function class like a Sobolev or Holder class. In a minimax sense, smoothing splines (as well as several other linear methods, such as kernel smoothers) are rate optimal for Sobolev or Holder classes (for example, see Chapter 10 of van de Geer (2000)). But for "larger" function classes like certain total variation, Besov, or Triebel classes, they are notably suboptimal.

As an example, the following is an implication of the results in Donoho and Johnstone (1998) (see Section 5.1 of Tibshirani (2014) for an explanation). Let $\mathcal{V}^k([a, b])$ denote the space of functions f on $[a, b]$ that are k times weakly differentiable, with $\mathrm{TV}(D^k f) < \infty$; and denote the associated seminorm ball of radius $C > 0$ by

$$\mathcal{V}^k(C; [a, b]) = \Big\{ f : [a, b] \to \mathbb{R} : \mathrm{TV}(D^k f) \leq C \Big\}.$$

Abbreviating $\mathcal{V}^k = \mathcal{V}^k(C; [a, b])$ for fixed C, a, b (not depending on n), and placing standard assumptions on the data generation model (that is, assumptions on the design points x_i, $i = 1, \ldots, n$, and errors $\epsilon = y_i - f_0(x_i)$, $i = 1, \ldots, n$), the minimax rate in mean squared L_2 error over the design points is

$$\inf_{\hat{f}} \sup_{f_0 \in \mathcal{V}^k} \mathbb{E}\bigg[\frac{1}{n} \big\| \hat{f}(x_{1:n}) - f_0(x_{1:n}) \big\|_2^2 \bigg] \lesssim n^{-\frac{2k+2}{2k+3}}, \tag{26}$$

where the infimum above is taken over all estimators $\hat{f}$. However, the minimax *linear* rate is

$$\inf_{\hat{f} \text{ linear}} \sup_{f_0 \in \mathcal{V}^k} \mathbb{E}\bigg[\frac{1}{n} \big\| \hat{f}(x_{1:n}) - f_0(x_{1:n}) \big\|_2^2 \bigg] \gtrsim n^{-\frac{2k+1}{2k+2}}, \tag{27}$$

where the infimum above is taken over all linear smoothers $\hat{f}$. (Here, we use $a_n \lesssim b_n$ to mean $a_n \leq c b_n$ for a constant $c > 0$ and large enough n, and $a_n \gtrsim b_n$ to mean $1/a_n \lesssim 1/b_n$.) Mammen and van de Geer (1997) proved that locally adaptive regression splines achieve the optimal rate in (26) (note that wavelet smoothing also achieves the optimal rate, as shown by Donoho and Johnstone (1998)). Importantly, from (27), we can see that smoothing splines—and further, any linear smoother whatsoever—are suboptimal.

For a concrete case, we can take $k = 0$, and then the rates (26) and (27) are $n^{-\frac{2}{3}}$ and $n^{-\frac{1}{2}}$, respectively, which we can interpret as follows: for estimating a function of bounded variation, the smoothing spline requires (on the order of) $n^{\frac{4}{3}}$ data points to achieve the same error guarantee that the locally adaptive regression spline has on n data points. See Figure 3 for an illustration of the rates for general k.

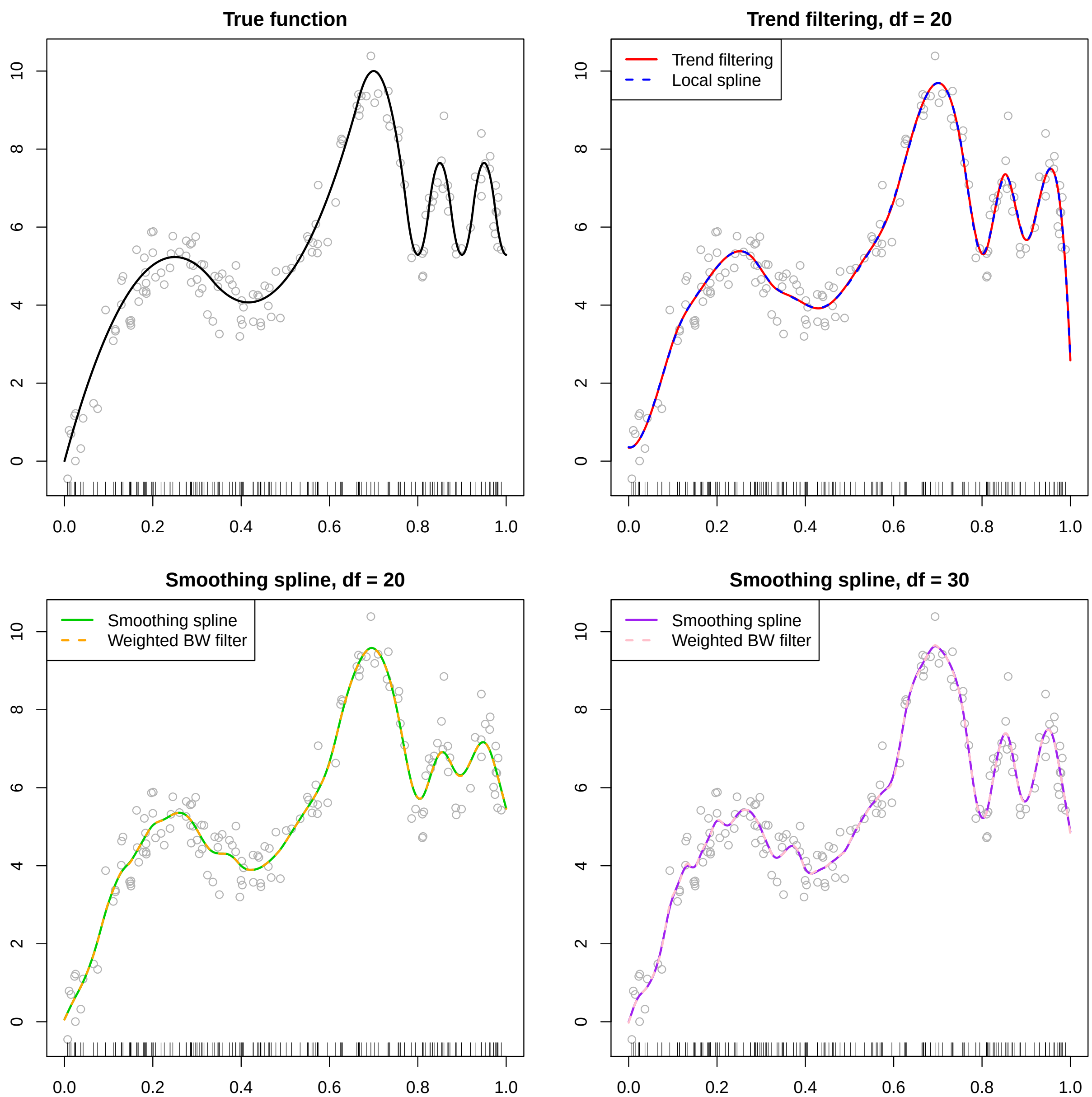

Figure 2: (Adapted from Tibshirani (2014).) Comparison of trend filtering and smoothing splines on an example with heterogeneous smoothness. The top left panel shows the true underlying regression function and $n = 150$ sampled response points, with the design points are marked by ticks on the horizontal axis (they are not evenly-spaced). The top right panel shows the cubic trend filtering solution ($k = 3$), in solid blue, with a "hand-picked" value of the tuning parameter λ. This solution results in an estimated degrees of freedom (df) of 20 (see Tibshirani and Taylor (2011)). Note that it adapts well to the smooth part of the true function on the left side of the domain, as well as the wiggly part on the right side. Also plotted is the restricted locally adaptive regression spline solution ($k = 3$), in dashed red, at the same value of λ, which looks visually identical. The bottom left panel is the cubic smoothing spline solution ($m = 2$), in solid green, whose df is matched to that of the trend filtering solution; notice that it oversmooths on the right side of the domain. The bottom right panel is the smoothing spline solution when its df has been increased to 30, the first point at which it begins appropriately pick up the two peaks on the right side; but note that it now undersmooths on the left side. Finally, in the bottom two panels, the cubic BW filter ($m = 2$) is also plotted, in dotted orange and dotted pink—to be clear, this is actually our proposed weighted extension of the BW filter to arbitrary designs, as given in Section 12. In each case it uses same value of λ as the smoothing spline solution, and looks identical to the latter.

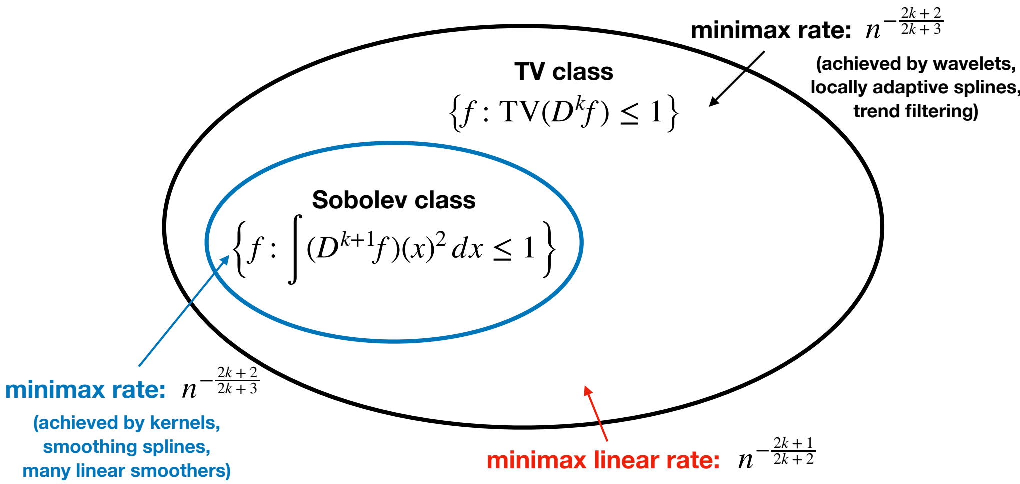

Figure 3: Minimax rates for the unit ball in the TV space of order k, $\mathcal{V}^k(1;[0,1]) = \{f : [0,1] \to \mathbb{R} : \mathrm{TV}(D^k f) \le 1\}$, and for the unit ball in the Sobolev space of order $k+1$, $\mathcal{W}^{k+1,2}(1;[0,1]) = \{f : [0,1] \to \mathbb{R} : \int_0^1 (D^{k+1}f)(x)^2\,dx \le 1\}$. Observe that

$$\mathcal{V}^k(1;[0,1]) \supseteq \mathcal{W}^{k+1,1}(1;[0,1]) \supseteq \mathcal{W}^{k+1,2}(1;[0,1]);$$

the first containment is due to the equality in (25) for $k+1$ times weakly differentiable functions, and the second containment is due to the relation between L_1 and L_2 norms on $[0,1]$. The minimax rates between $\mathcal{V}^k(1;[0,1])$ and the $\mathcal{W}^{k+1,2}(1;[0,1])$ are the same; but critically, linear smoothers, which can be optimal on the smaller set $\mathcal{W}^{k+1,2}(1;[0,1])$, *cannot* be optimal on the larger set $\mathcal{V}^k(1;[0,1])$, which contains functions that display more heterogeneous smoothness.

Computational difficulties. Mammen and van de Geer (1997) proved that the solution $\hat{f}$ in (8) is a kth degree spline. For $k=0$ or $k=1$, they show that the knots in $\hat{f}$ must lie in particular subset of the design points, denoted $T_{n,k} \subseteq x_{1:n}$, with cardinality $|T_{n,k}| = n-k-1$; that is, for $k=0$ or $k=1$, we know that $\hat{f} \in \mathcal{S}^k(T_{n,k},[a,b])$, which reduces (8) to a finite-dimensional problem. But for $k \ge 2$, this is no longer true; the knots in $\hat{f}$ may well lie outside of $x_{1:n}$, and (8) remains an infinite-dimensional problem (since we have to optimize over all possible knot sets).

As a proposed fix, for a general degree $k \ge 0$, Mammen and van de Geer (1997) defined (what we refer to as) the *kth degree restricted locally adaptive regression spline* estimator, which solves

$$\underset{f \in \mathcal{G}_n^k}{\text{minimize}} \;\; \frac{1}{2}\sum_{i=1}^n \big(y_i - f(x_i)\big)^2 + \lambda\, \mathrm{TV}(D^k f), \tag{28}$$

for a certain space $\mathcal{G}_n^k = \mathcal{S}^k(T_{n,k},[a,b])$ of kth degree splines with knots $T_{n,k} \subseteq x_{1:n}$, where $|T_{n,k}| = n-k-1$ (they define $T_{n,k}$ by excluding $k+1$ points at the extremes of the design). To be clear, for $k=0$ and $k=1$, problems (28) and (8) are equivalent; but for $k \ge 2$, they are not, and the former is an approximation of the latter.

The proposal in (28) is useful because it is equivalent to a finite-dimensional convex optimization problem: letting $\mathbb{G}_n^k \in \mathbb{R}^{\times n}$ be the truncated power basis matrix, with entries $(\mathbb{G}_n^k)_{ij} = g_j^k(x_i)$, where g_j^k, $j=1,\dots,n$ are the truncated power basis (14) for $\mathcal{G}_n^k$, we can rewrite (28) as

$$\underset{\alpha}{\text{minimize}} \;\; \frac{1}{2}\big\|y - \mathbb{G}_n^k \alpha\big\|_2^2 + \lambda \sum_{j=k+2}^n |\alpha_j|, \tag{29}$$

where the solutions $\hat{\alpha}, \hat{f}$ in (29), (28), respectively, satisfy $\hat{f} = \sum_{j=1}^n \hat{\alpha}_j g_j^k$. Mammen and van de Geer (1997) proved that the restricted locally adaptive regression spline estimator (under weak conditions on the design points) still achieves the optimal rate in (26). Readers familiar with the high-dimensional regression literature will recognize (29) as a type of *lasso* problem (Tibshirani, 1996; Chen et al., 1998), for which many efficient algorithms exist (for just one example, see Friedman et al. (2007)). But for large sample sizes n, it can still be computationally difficult to solve, owing to the fact that the design matrix $\mathbb{G}_n^k$ is dense (it is actually lower-triangular, but generally poorly-conditioned, which causes trouble for first-order optimization algorithms).

2.6 Trend filtering

Building on the background and motivation for trend filtering given in the introduction, and the motivation for locally adaptive regression splines just given, we arrive at the following perspective. Trend filtering is an approximation to the locally adaptive regression spline problem (8), which is similar to the proposal for restricted locally adaptive regression splines in (28), but with a different restriction for the optimization domain: it uses the kth degree discrete spline space $\mathcal{H}_n^k$, as we saw in (7), rather than the kth degree spline space $\mathcal{G}_n^k$. To retrieve an equivalent lasso form, similar to (29), we can let $\mathbb{H}_n^k \in \mathbb{R}^{n \times n}$ denote the falling factorial basis matrix, with entries $(\mathbb{H}_n^k)_{ij} = h_j^k(x_i)$, where h_j^k, $j = 1, \dots, n$ are as in (5), and then (30) becomes

$$\underset{\alpha}{\text{minimize}} \;\; \frac{1}{2}\big\|y - \mathbb{H}_n^k \alpha\big\|_2^2 + \lambda \sum_{j=k+2}^{n} |\alpha_j|. \tag{30}$$

where the solutions $\hat{\alpha}, \hat{f}$ in problems (30), (7), respectively, are related by $\hat{f} = \sum_{j=1}^n \hat{\alpha}_j h_j^k$. Fortunately, trend filtering retains (under mild conditions on the design) the minimax optimal rate in (26). This was shown in Tibshirani (2014); Wang et al. (2014) by bounding the distance between solutions in (30), (29).

Finally—and critically for practical use—the problem (30) has an equivalent form given in (3). The latter, original form of trend filtering is more amenable to efficient computation, thanks to the structured, banded nature of its penalty term; computation here scales considerably better than that in either (30) or the restricted locally adaptive regression spline problem (28). (We have found that in most empirical examples, trend filtering and restricted locally adaptive spline solutions are more or less visually identical anyway; see Figure 2.)

On the topic of (3), we remark that this problem can be equivalently written as

$$\underset{\theta}{\text{minimize}} \;\; \frac{1}{2}\|y - \theta\|_2^2 + \lambda \big\|\mathbb{W}_n^{k+1} \mathbb{D}_n^{k+1} \theta\big\|_1, \tag{31}$$

with $\mathbb{D}_n^{k+1} \in \mathbb{R}^{(n-k-1) \times n}$ the $(k+1)$st order discrete derivative matrix to be defined in (70) (recall, this matrix acts by producing divided differences over the design points), and $\mathbb{W}_n^{k+1} \in \mathbb{R}^{(n-k-1) \times (n-k-1)}$ is the $(k+1)$st order diagonal weight matrix to be defined in (69). The penalty in the above problem is hence

$$\big\|\mathbb{W}_n^{k+1} \mathbb{D}_n^{k+1} \theta\big\|_1 = \sum_{i=1}^{n-k-1} \big|(\mathbb{D}_n^{k+1}\theta)_i\big| \cdot \frac{x_{i+k+1} - x_i}{k+1}. \tag{32}$$

Thus (3) versus (31) is a matter of whether the natural operator is viewed as $\mathbb{C}_n^{k+1}$ or $\mathbb{D}_n^{k+1}$. We should note that when the design points are evenly-spaced, we have $\mathbb{W}_n^{k+1} = \mathbb{I}_{n-k-1}$, the identity matrix, so this choice makes no difference; in general though, it does, and we now view (31) as a more natural way of presenting trend filtering, which differs from the choice (3) that we made in Tibshirani (2014); Wang et al. (2014) and our subsequent work. In Remarks 12 and 18, and Section 11, we return to this point.

Historical remarks. As already mentioned, trend filtering for evenly-spaced designs was independently proposed by Steidl et al. (2006); Kim et al. (2009). However, similar ideas were around much earlier. Kim et al. (2009) were clear about being motivated by Hodrick and Prescott (1981), who considered an ℓ_2 analog of trend filtering, that is, with an ℓ_2 penalty on forward differences, rather than an ℓ_1 penalty. (Actually, such ℓ_2 analogs were proposed over 100 years ago, long before Hodrick and Prescott, first by Bohlmann and then by Whittaker, as we discuss in the next subsection.) Moreover, Schuette (1978) and Koenker et al. (1994) studied estimators defined using piecewise linear ($k = 1$) trend filtering penalties, but where the squared ℓ_2 loss is replaced with an ℓ_1 loss or quantile loss, respectively. Lastly, we remark again that for the piecewise constant case ($k = 0$), trend filtering reduces to what is known as total variation denoising (Rudin et al., 1992) in signal processing, and the fused lasso in statistics (Tibshirani et al., 2005).

In writing Tibshirani (2014), we were motivated by Kim et al. (2009); these authors called their method "ℓ_1 trend filtering", which we shortened to "trend filtering" in our work. At this time, we had not heard of discrete splines, but we were aware that the trend filtering solution displayed a kind of continuity in its lower-order discrete derivatives: this was demonstrated empirically in Figure 3 of Tibshirani (2014). By the time of our follow-up paper Wang et al. (2014), we learned that Steidl et al. (2006) had proposed the same idea as Kim et al. (2009). It was in the former paper that we first learned of discrete splines and the foundational work by Mangasarian and Schumaker (1971, 1973) on the topic, but it was not until much later—until we read the book by Schumaker (2007), where the development of discrete splines is laid out systematically in a parallel fashion to the development of splines—that we truly appreciated the connection between discrete splines and trend filtering, and the value that such a connection can bring to both lines of work. The current paper grew from an attempt to pay homage to discrete splines and to make all such connections explicit.

2.7 Bohlmann-Whittaker filtering

Over 120 years ago, Bohlmann (1899) studied the solution of the problem:

$$\underset{\theta}{\text{minimize}} \;\; \|y-\theta\|_2^2 + \lambda \sum_{i=1}^{n-1} (\theta_i - \theta_{i+1})^2, \tag{33}$$

as a smoother of responses y_i, $i = 1, \dots, n$ observed at evenly-spaced (unit-spaced) design points $x_i = i$, $i = 1, \dots, n$. This is one of the earliest references that we know of for discrete-time smoothing (or smoothing of any kind) based on optimization. Over 20 years after this, Whittaker (1923) proposed a variant of (33) where first differences are replaced by third differences:

$$\underset{\theta}{\text{minimize}} \;\; \|y-\theta\|_2^2 + \lambda \sum_{i=1}^{n-3} (\theta_i - 3\theta_{i+1} + 3\theta_{i+2} - \theta_{i+3})^2. \tag{34}$$

Whittaker seems to have been unaware of the work by Bohlmann, and unfortunately, Bohlmann's work has remained relatively unknown (it is still not cited in most references on discrete-time smoothing and its history). Meanwhile, the work of Whittaker (1923) was quite influtential and led a long line of literature, centered in the actuarial community, where (34) is often called the *Whittaker-Henderson method of graduation*, honoring the contributions of Henderson (1924). Moreover, as explained previously, recall it was Whittaker's work that inspired Schoenberg (1964) to develop the smoothing spline.

Almost 60 years after this, Hodrick and Prescott (1981) proposed a variation on (34) in which third differences are replaced by second differences:

$$\underset{\theta}{\text{minimize}} \;\; \|y-\theta\|_2^2 + \lambda \sum_{i=1}^{n-2} (\theta_i - 2\theta_{i+1} + \theta_{i+2})^2. \tag{35}$$

Hodrick and Prescott were aware of the work of Whittaker, but not of Bohlmann. The paper by Hodrick and Prescott (1981), which was later published as Hodrick and Prescott (1997), has become extremely influential in econometrics, where (35) is known as the *Hodrick-Prescott filter*. Recall, as explained previously, that it was Hodrick and Prescott's work that inspired Kim et al. (2009) to develop trend filtering.

Generalizing (33), (34), (35), consider for an integer $m \ge 0$, the problem:

$$\underset{\theta}{\text{minimize}} \;\; \|y-\theta\|_2^2 + \lambda \sum_{i=1}^{n-m} (F^m\theta)(i)^2 \tag{36}$$

where $(F^m\theta)(i) = \sum_{\ell=0}^{k} (-1)^{k-\ell} \binom{k}{\ell} \theta_{i+\ell}$ is the standard (integer-based) mth order forward differences of θ starting at an integer i. To honor their early contributions, we call the solution in (36) the *Bohlmann-Whittaker (BW) filter*.

Arbitrary designs. For a set of arbitrary design points $x_{1:n}$, it would seem natural to use divided differences in place of forward differences in (36), resulting in

$$\underset{\theta}{\text{minimize}} \;\; \|y-\theta\|_2^2 + \lambda \big\|\mathbb{D}_n^m \theta\big\|_2^2, \tag{37}$$

where $\mathbb{D}_n^m \in \mathbb{R}^{(n-m)\times n}$ is the mth order discrete derivative matrix defined in (70). In fact, such an extension (37) for arbitrary designs was suggested by Whittaker (1923), in a footnote of his paper. This idea caught on with many authors, including Schoenberg (1964), who in describing Whittaker's method as the source of inspiration for his creation of the smoothing spline, used the form (37).

In Section 12, we argue that for arbitrary designs it is actually in some ways more natural to replace the penalty in (37) by a weighted squared ℓ_2 penalty,

$$\big\|(\mathbb{W}_n^m)^{\frac{1}{2}} \mathbb{D}_n^m \theta\big\|_2^2 = \sum_{i=1}^{n-m} (\mathbb{D}_n^m\theta)_i^2 \cdot \frac{x_{i+m} - x_i}{m}. \tag{38}$$

Here $\mathbb{W}_n^m \in \mathbb{R}^{(n-m)\times(n-m)}$ is the mth order diagonal weight matrix, defined later in (69). Notice the close similarity between the weighting in (38) and in the trend filtering penalty (32). The reason we advocate for the penalty (38) is that the resulting estimator admits a close tie to the smoothing spline: when $m = 1$, these two exactly coincide (recall (18) and (20) from Theorem 1), and when $m = 2$, they are provably "close" in ℓ_2 distance (for appropriate values of their tuning parameters), as we show later in Theorem 7. Moreover, empirical examples support the idea that the estimator associated with the weighted penalty (38) can be closer than the solution in (37) to the smoothing spline.

Finally, unlike trend filtering, whose connection to discrete splines is transparent and clean (at least in hindsight), the story with the BW filter is more subtle. This is covered in Section 12.3.

3 Falling factorials

In this section, we define a discrete derivative operator based on divided differences, and its inverse operator, a discrete integrator, based on cumulative sums. We use these discrete operators to construct the falling factorial basis for discrete splines, in a manner analogous to the construction of the truncated power basis for splines.

3.1 Discrete differentiation

Let f be a function defined on an interval $[a, b]$[3], and let $a \le x_1 < \cdots < x_n \le b$. To motivate the discrete derivative operator that we study in this subsection, consider the following question: given a point $x \in [a, b]$, how might we use $f(x_1), \dots, f(x_n)$, along with one more evaluation $f(x)$, to approximate the kth derivative $(D^k f)(x)$, of f at x?

A natural answer to this question is given by divided differences. For an integer $k \ge 1$, we write $\Delta^k(\cdot; x_{1:n})$ for an operator that maps a function f to a function $\Delta^k(f; x_{1:n})$, which we call the *kth discrete derivative* (or the *discrete kth derivative*) of f, to be defined below. A remark on notation: $\Delta^k(f; x_{1:n})$ emphasizes the dependence on the underlying design points $x_{1:n} = \{x_1, \dots, x_n\}$; henceforth, we abbreviate $\Delta_n^k f = \Delta^k(f; x_{1:n})$ (and the underlying points $x_{1:n}$ should be clear from the context). Now, we define the function $\Delta_n^k f$ at a point $x \in [a, b]$ as

$$(\Delta_n^k f)(x) = \begin{cases} k! \cdot f[x_{i-k+1}, \dots, x_i, x] & \text{if } x \in (x_i, x_{i+1}],\ i \ge k \\ i! \cdot f[x_1, \dots, x_i, x] & \text{if } x \in (x_i, x_{i+1}],\ i < k \\ f(x) & \text{if } x \le x_1. \end{cases} \tag{39}$$

Here and throughout, we use $x_{n+1} = b$ for notational convenience. Note that, on "most" of the domain $[a, b]$, that is, for $x \in (x_k, b]$, we define $(\Delta_n^k f)(x)$ in terms of a (scaled) kth divided difference of f, where the centers are the k points immediately to the left of x, and x itself. Meanwhile, on a "small" part of the domain, that is, for $x \in [a, x_k]$, we define $(\Delta_n^k f)(x)$ to be a (scaled) divided difference of f of the highest possible order, where the centers are the points to the left of x, and x itself.

Linear combination formulation. As divided differences are linear combinations of function evaluations, it is not hard to see from its definition in (39) that $(\Delta_n^k f)(x)$ is a linear combination of (a subset of size at most $k + 1$ of) the evaluations $f(x_1), \dots, f(x_n)$ and $f(x)$. In fact, from the alternative representation for divided differences in (9), we can rewrite (39) as

$$(\Delta_n^k f)(x) = \begin{cases} \displaystyle\sum_{j=i-k+1}^{i} \frac{k! \cdot f(x_j)}{\big(\prod_{\ell \in \{i-k+1,\dots,i\} \setminus \{j\}} (x_j - x_\ell)\big)(x_j - x)} + \frac{k! \cdot f(x)}{\prod_{\ell=i-k+1}^{i} (x - x_\ell)} & \text{if } x \in (x_i, x_{i+1}],\ i \ge k \\ \displaystyle\sum_{j=1}^{i} \frac{i! \cdot f(x_j)}{\big(\prod_{\ell \in \{1,\dots,i\} \setminus \{j\}} (x_j - x_\ell)\big)(x_j - x)} + \frac{i! \cdot f(x)}{\prod_{\ell=1}^{i} (x - x_\ell)} & \text{if } x \in (x_i, x_{i+1}],\ i < k \\ f(x) & \text{if } x \le x_1. \end{cases} \tag{40}$$

It is worth presenting this formula as it is completely explicit. However, it is not directly used in the remainder of the paper. On the other hand, a more useful formulation can be expressed via recursion, as we develop next.

[3] There is no real need to consider an interval $[a, b]$ containing the points $x_1, \dots, x_n$. We introduce this interval simply because we think it may be conceptually helpful when defining the discrete derivative and integral operators, but the same definitions make sense, with minor modifcations, when we consider f as a function on all of $\mathbb{R}$.

Recursive formulation. The following is an equivalent recursive formulation for the discrete derivative operators in (39). We start by explicitly defining the first order operator Δ_n (omitting the superscript here, for $k = 1$, which we will do commonly henceforth) by

$$(\Delta_n f)(x) = \begin{cases} \dfrac{f(x) - f(x_i)}{x - x_i} & \text{if } x \in (x_i, x_{i+1}] \\ f(x) & \text{if } x \le x_1. \end{cases} \tag{41}$$

For $k \ge 2$, due to the recursion obeyed by divided differences, we can equivalently define the kth discrete derivative operator by

$$(\Delta_n^k f)(x) = \begin{cases} \dfrac{(\Delta_n^{k-1} f)(x) - (\Delta_n^{k-1} f)(x_i)}{(x - x_{i-k+1})/k} & \text{if } x \in (x_i, x_{i+1}] \\ (\Delta_n^{k-1} f)(x) & \text{if } x \le x_k. \end{cases} \tag{42}$$

To express this recusion in a more compact form, we define the simple difference operator $\overline{\Delta}_n = \overline{\Delta}(\cdot; x_{1:n})$ by

$$(\overline{\Delta}_n f)(x) = \begin{cases} f(x) - f(x_i) & \text{if } x \in (x_i, x_{i+1}] \\ f(x) & \text{if } x \le x_1. \end{cases} \tag{43}$$

and for $k \ge 1$, we define the weight map $W_n^k = W^k(\cdot; x_{1:n})$ by

$$(W_n^k f)(x) = \begin{cases} f(x) \cdot (x - x_{i-k+1})/k & \text{if } x \in (x_i, x_{i+1}],\, i \ge k \\ f(x) & \text{if } x \le x_k. \end{cases} \tag{44}$$

Then the recursion in (41), (42) can be rewritten as

$$\begin{aligned} \Delta_n &= (W_n)^{-1} \circ \overline{\Delta}_n, \\ \Delta_n^k &= (W_n^k)^{-1} \circ \overline{\Delta}_{n-k+1} \circ \Delta_n^{k-1}, \quad \text{for } k \ge 2. \end{aligned} \tag{45}$$

An important note: here, we denote by $\overline{\Delta}_{n-k+1} = \overline{\Delta}(\cdot; x_{k:n})$, the simple difference operator in (43) when we use the $n - k + 1$ underlying points $x_{k:n} = \{x_k, \dots, x_n\}$ (rather than the original n points $x_{1:n} = \{x_1, \dots, x_n\}$).

The compact recursive formulation in (45) is quite useful, since it allows us to define a certain discrete integrator, which acts as the inverse to discrete differentiation, to be described in the next subsection.

Evenly-spaced design points. When the design points are evenly-spaced, $x_{i+1} - x_i = v > 0$, for $i = 1, \dots, n-1$, the discrete derivative operator (39) can be expressed at design points as a (scaled) forward difference, or equivalently a (scaled) backward difference,

$$(\Delta_n^k f)(x_i) = \begin{cases} F_v^k(x_i - kv) = B_v^k(x_i) & \text{if } i \ge k+1 \\ F_v^i(x_1) = B_v^i(x_i) & \text{if } i < k+1. \end{cases}$$

where recall we use F_v^k, B_v^k for the kth order forward and backward difference operators, respectively. In the case of evenly-spaced design points, there are some special properties of discrete derivatives (forward/backward differences), such as

$$(\Delta_n^k f)(x_i) = (\Delta_{n-k+d}^d \, \Delta_n^{k-d} f)(x_i),$$

for all i and all $0 \le d \le k$. This unfortunately does not hold more generally (for arbitrary designs); from (41), (42), we see that for arbitrary $x_{1:n}$, the above property holds at x_i with $d = 1$ if and only if $(x_i - x_{i-k})/k = x_i - x_{i-1}$. (Further, it should be noted that the above property never holds—whether in the evenly-spaced case, or not—at points $x \notin x_{1:n}$.)

3.2 Discrete integration

Consider the same setup as the last subsection, but now with the following question as motivation: given $x \in [a, b]$, how might we use $f(x_1), \dots, f(x_n)$, along with $f(x)$, to approximate the kth integral $(I^k f)(x)$, of f at x?

We write $S^k(\cdot; x_{1:n})$ to denote an operator that maps a function f to a function $S^k(f; x_{1:n})$, which we call the *kth discrete integral* (or the *discrete kth integral*) of f, to be defined below. As before, we abbreviate $S_n^k = S^k(\cdot; x_{1:n})$. To define the function $S^k_n f$, we take a recursive approach, mirroring our approach in (43), (44), (45). We start by defining the simple cumulative sum operator $\overline{S}_n = \overline{S}(\cdot; x_{1:n})$ by

$$(\overline{S}_n f)(x) = \begin{cases} \displaystyle\sum_{j=1}^{i} f(x_j) + f(x) & \text{if } x \in (x_i, x_{i+1}] \\ f(x) & \text{if } x \le x_1. \end{cases} \tag{46}$$

We then define the discrete integral operators by

$$\begin{aligned} S_n &= \overline{S}_n \circ W_n, \\ S^k_n &= S^{k-1}_n \circ \overline{S}_{n-k+1} \circ W^k_n, \quad \text{for } k \ge 2. \end{aligned} \tag{47}$$

An important note: as before, we abbreviate $\overline{S}_{n-k+1} = \overline{S}(\cdot; x_{k:n})$ the discrete integral operator in (47) over the $n-k+1$ underlying points $x_{k:n} = \{x_k, \dots, x_n\}$ (instead of over the original n points $x_1, \dots, x_n$).

Linear combination formulation. As with discrete derivatives (recall (9)), the discrete integral of a function f can be written in terms of linear combinations of evaluations of f. This can be seen by working through the definitions (46) and (47), which would lead to a formula for $(S^k_n f)(x)$ as a linear combination of $f(x_1), \dots, f(x_n)$ and $f(x)$, with the coefficients being kth order cumulative sums of certain gaps between the design points $x_1, \dots, x_n$ and x.

A subtle fact is that this linear combination can be written in a more explicit form, that does not involve cumulative sums at all. Letting h^{k-1}_j, $j = 1, \dots, n$ denote the falling factorial basis functions as in (5), but of degree $k-1$, it holds that

$$(S^k_n f)(x) = \begin{cases} \displaystyle\sum_{j=1}^{k} h^{k-1}_j(x) \cdot f(x_j) \;+\; \sum_{j=k+1}^{i} h^{k-1}_j(x) \cdot \frac{x_j - x_{j-k}}{k} \cdot f(x_j) \;+\; h^{k-1}_{i+1}(x) \cdot \frac{x - x_{i-k+1}}{k} \cdot f(x) & \\ & \text{if } x \in (x_i, x_{i+1}],\, i \ge k \\ \displaystyle\sum_{j=1}^{i} h^{k-1}_j(x) \cdot f(x_j) \;+\; h^{k-1}_{i+1}(x) \cdot f(x) & \text{if } x \in (x_i, x_{i+1}],\, i < k \\ f(x) & \text{if } x \le x_1, \end{cases} \tag{48}$$

The above is a consequence of results that we will develop in subsequent parts of this paper: the inverse relationship between discrete differentation and discrete integration (Lemma 1, next), and the dual relationship between discrete differentiation and the falling factorial basis (Lemmas 4 and 5, later). We defer its proof to Appendix B.2. As with the discrete derivative result (40), it is worth presenting (48) because its form is completely explicit. However, again, we note that this linear combination formulation is not itself directly used in the remainder of this paper.

Inverse relationship. The next result shows an important relationship between discrete differentiation and discrete integration: they are precisely inverses of each other. The proof follows by induction and is given in Appendix B.3.

Lemma 1. *For any $k \ge 1$, it holds that $(\Delta^k_n)^{-1} = S^k_n$, that is, $\Delta^k_n S^k_n f = f$ and $S^k_n \Delta^k_n f = f$ for all functions f.*

Remark 4. It may be surprising, at first glance, that the kth order discrete derivative operator Δ^k_n has an inverse at all. In continuous-time, by comparison, the kth order derivative operator D^k annihilates all polynomials of degree k, thus we clearly cannot have $I^k D^k f = f$ for all f. Viewed as an operator over all functions with sufficient regularity, D^k only has a right inverse, that is, $D^k I^k f = f$ for all f (by the fundamental theorem of calculus). The fact that Δ^k_n has a proper (both left and right) inverse S^k_n is due to the special way in which $\Delta^k_n f$ is defined towards the left side of the underlying domain: recall that $(\Delta^k_n f)(x)$ does not involve a divided difference of order k for $x \in [a, x_k]$, but rather, a divided difference of order $k-1$ for $x \in (x_{k-1}, x_k]$, of order $k-2$ for $x \in (x_{k-2}, x_{k-1}]$, etc. This "fall off" in the order of the divided difference being taken, as x approaches the left boundary point a, is what renders Δ^k_n invertible. For example, when p is kth degree polynomial, we have $(\Delta^k_n p)(x) = 0$ for $x \in (x_k, b]$, and yet the lower order divided differences, $(\Delta^k_n p)(x)$ for $x \in [a, x_k]$, encode enough information that we can recover p via discrete integration.

3.3 Constructing the basis

We recall a simple way to construct the truncated power basis for splines. Let us abbreviate $1_t = 1_{(t,b]}$, that is, the step function with step at $t \in [a,b]$,

$$1_t(x) = 1\{x > t\}.$$

(The choice of left-continuous step function is arbitrary, but convenient for our development). It can be easily checked by induction that for all $k \geq 0$,

$$(I^k 1_t)(x) = \frac{1}{k!}(x-t)_+^k,$$

where recall $x_+ = \max\{x, 0\}$, and we denote by $I^0 = \mathrm{Id}$, the identity map, for notational convenience. We can thus see that the truncated power basis in (14), for the space $\mathcal{S}^k(t_{1:r}, [a,b])$ of kth degree splines with knot set $t_{1:r}$, can be constructed by starting with the polynomials x^{j-1}, $j = 1, \dots, k+1$ and including the kth order antiderivatives of the appropriate step functions,

$$\frac{1}{k!}(x-t_j)_+^k = (I^k 1_{t_j})(x), \quad j = 1, \dots, r. \tag{49}$$

We now show that an analogous construction gives rise to the falling factorial basis functions in (5).

Theorem 2. *For any $k \geq 0$, the piecewise polynomials in the kth degree falling factorial basis, given in the second line of (5), satisfy*

$$\frac{1}{k!}\prod_{\ell=j-k}^{j-1}(x - x_\ell) \cdot 1\{x > x_{j-1}\} = (S_n^k 1_{x_{j-1}})(x), \quad j = k+2, \dots, n. \tag{50}$$

Here, we use $S_n^0 = \mathrm{Id}$, the identity map, for notational convenience.

Theorem 2 shows that the falling factorial basis functions arise from k times discretely integrating step functions with jumps at $x_{k+1}, \dots, x_{n-1}$. These are nothing more than truncated Newton polynomials, with the left-hand side in (50) being $\eta(x; x_{(j-k):(j-1)}) 1\{x > x_{j-1}\}/k!$, using the compact notation for Newton poynomials, as defined in (10).

Recalling that the discrete integrators are defined recursively, in (47), one might guess that the result in (50) can be established by induction on k. While this is indeed true, the inductive proof for Theorem 2 does not follow a standard approach that one might expect: it is not at all clear from the recursion in (47) how to express each h_j^k in terms of a discrete integral of h_j^{k-1}. Instead, it turns out that we can derive what we call a *lateral* recursion, where we express h_j^k as a weighted sum of h_ℓ^{k-1} for $\ell \geq j$, and similarly for their discrete derivatives. This is the key driver behind the proof of Theorem 2, and is stated next.

Lemma 2. *For any $k \geq 1$, the piecewise polynomials in the kth degree falling factorial basis, given in the second line of (5), satisfy the following recursion. For each $d \geq 0$, $j \geq k+2$, and $x \in (x_i, x_{i+1}]$, where $i \geq j-1$,*

$$(\Delta_n^d h_j^k)(x) = \sum_{\ell=j}^{i} (\Delta_n^d h_\ell^{k-1})(x) \cdot \frac{x_\ell - x_{\ell-k}}{k} + (\Delta_n^d h_{i+1}^{k-1})(x) \cdot \frac{x - x_{i-k+1}}{k}. \tag{51}$$

Here, we use $\Delta_n^0 = \mathrm{Id}$, the identity map, for notational convenience.

The proof of Lemma 2 is elementary and is deferred until Appendix B.4. We now show how it can be used to prove Theorem 2.

Proof of Theorem 2. Note that, by the invertibility of S_n^k, from Lemma 1, it suffices to show that for all $k \geq 0$,

$$(\Delta_n^k h_j^k)(x) = 1\{x > x_{j-1}\}, \quad j = k+2, \dots, n. \tag{52}$$

We proceed by induction on k. When $k = 0$, the result is immediate from the definition of the falling factorial basis functions in (5). Assume the result holds for the degree $k-1$ falling factorial basis. Fix $j \geq k+2$. If $x \leq x_{j-1}$, then it is easy to check that $(\Delta_n^k h_j^k)(x) = 0$. Thus let $x \in (x_i, x_{i+1}]$ where $i \geq j-1$. By the recursive representation (51),

$$\begin{aligned}
&(\Delta_n^{k-1} h_j^k)(x) - (\Delta_n^{k-1} h_j^k)(x_i) \\
&= \sum_{\ell=j}^{i} (\Delta_n^{k-1} h_\ell^{k-1})(x) \cdot \frac{x_\ell - x_{\ell-k}}{k} + (\Delta_n^{k-1} h_{i+1}^{k-1})(x) \cdot \frac{x - x_{i-k+1}}{k} - \sum_{\ell=j}^{i} (\Delta_n^{k-1} h_\ell^{k-1})(x_i) \cdot \frac{x_\ell - x_{\ell-k}}{k}
\end{aligned}$$

$$= \sum_{\ell=j}^{i} \big((\Delta_n^{k-1} h_\ell^{k-1})(x) - (\Delta_n^{k-1} h_\ell^{k-1})(x_i)\big) \cdot \frac{x_\ell - x_{\ell-k}}{k} + (\Delta_n^{k-1} h_{i+1}^{k-1})(x) \cdot \frac{x - x_{i-k+1}}{k}$$

$$= \sum_{\ell=j}^{i} \big(1\{x > x_{\ell-1}\} - 1\{x_i > x_{\ell-1}\}\big) \cdot \frac{x_\ell - x_{\ell-k}}{k} + 1\{x > x_i\} \cdot \frac{x - x_{i-k+1}}{k},$$

where in the last line we used the inductive hypothesis. As all indicators in above line are equal to 1, the sum is equal to 0, and hence by the definition in (42),

$$\begin{aligned}(\Delta_n^k h_j^k)(x) &= \frac{(\Delta_n^{k-1} h_j^k)(x) - (\Delta_n^{k-1} h_j^k)(x_i)}{(x - x_{i-k+1})/k} \\ &= \frac{(x - x_{i-k+1})/k}{(x - x_{i-k+1})/k} = 1.\end{aligned}$$

This completes the proof. □

Now that we have constructed the piecewise polynomials in the kth degree falling factorial basis functions, using kth order discrete integration of step functions in (50), we can add any set of $k+1$ linearly independent kth degree polynomials to these piecewise polynomials to form an equivalent basis: the falling factorial basis. For example, the monomials x^{j-1}, $j = 1, \ldots, k+1$ would be a simple choice. However, as originally defined in (5), we used a different set of kth degree polynomials: Newton polynomials of degrees $0, \ldots, k$. This is a natural pairing, because the falling factorial basis can be seen as a set of truncated Newton polynomials; furthermore, as we show later in Section 5, this choice leads to a convenient dual basis to the falling factorials h_j^k, $j = 1, \ldots, n$.

Evenly-spaced design points. When the design points are evenly-spaced, $x_{i+1} - x_i = v > 0$, for $i = 1, \ldots, n-1$, the falling factorial basis functions in (5) reduce to

$$\begin{aligned}h_j^k(x) &= \frac{1}{(j-1)!}(x - x_1)_{j-1,v}, \quad j = 1, \ldots, k+1, \\ h_j^k(x) &= \frac{1}{k!}(x - x_{j-k})_{k,v} \cdot 1\{x > x_{j-1}\}, \quad j = k+2, \ldots, n,\end{aligned}$$

where recall we write $(x)_{\ell,v} = x(x-v)\cdots(x-(\ell-1)v)$ for the falling factorial polynomial of degree ℓ with gap v, which we interpret to be equal to 1 when $\ell = 0$. This connection inspired the name of these basis functions as given in Tibshirani (2014); Wang et al. (2014). Further, it follows by a simple inductive argument (for example, see Lemma 2 in Tibshirani (2014)) that, evaluated at a design point x_i, the basis functions become

$$\begin{aligned}h_j^k(x_i) &= v^{j-1} \sigma_{i-j+1}^{j-1} \cdot 1\{i > j-1\}, \quad j = 1, \ldots, k+1, \\ h_j^k(x_i) &= v^k \sigma_{i-j+1}^k \cdot 1\{i > j-1\}, \quad j = k+2, \ldots, n,\end{aligned}$$

where we define $\sigma_i^0 = 1$ for all i and $\sigma_i^\ell = \sum_{j=1}^i \sigma_j^{\ell-1}$, the ℓth order cumulative sum of $1, \ldots, 1$ (repeated i times).

4 Smoothness properties

We study some properties relating to the structure and smoothness of functions in the span of the falling factorial basis. To begin, we point out an important *lack of* smoothness in the usual sense: the piecewise polynomial falling factorial basis functions h_j^k, $j = k+2, \ldots, n$, given in the second line of (5), do not have continuous derivatives. To see this, write, for each $j \geq k+2$,

$$h_j^k(x) = \frac{1}{k!}\eta(x; x_{(j-k):(j-1)}) \cdot 1\{x > x_{j-1}\},$$

where recall $\eta(x; x_{(j-k):(j-1)}) = \prod_{i=j-k}^{j-1}(x - x_i)!$ is the kth degree Newton polynomial, as introduced in (10). Note that for any $0 \leq d \leq k$, and $x < x_{j-1}$, we have $(D^d h_j^k)(x) = 0$, whereas for $x > x_{j-1}$,

$$(D^d h_j^k)(x) = \frac{d!}{k!} \sum_{\substack{I \subseteq (j-k):(j-1) \\ |I| = k-d}} \eta(x; x_I). \tag{53}$$

where for a set I, we let $x_I = \{x_i : i \in i\}$. We can hence see that, for $d \geq 1$,

$$\lim_{x \to x_{j-1}^+} (D^d h_j^k)(x) = \frac{d!}{k!} \sum_{\substack{I \subseteq (j-k):(j-2) \\ |I|=k-d}} \eta(x_{j-1}; x_I) > 0, \tag{54}$$

which is strictly positive because the design points are assumed to be distinct, and hence the left and right derivatives do not match at x_{j-1}.

In other words, we have just shown that the falling factorial basis functions h_j^k, $j = 1, \ldots, n$, when $k \geq 2$, are not kth degree splines, as their derivatives lack continuity at the knot points. On the other hand, as we show next, the falling factorial functions are not void of smoothness, it is simply expressed in a different way: their *discrete* derivatives end up being continuous at the knot points.

4.1 Discrete splines

We begin by extending the definition of discrete splines in Definition 2 to the setting of arbitrary design points, where naturally, divided differences appear in place of forward differences.

Definition 3. For an integer $k \geq 0$, design points $a \leq x_1 < \cdots < x_n \leq b$ (that define the operators $\Delta_n^\ell = \Delta^\ell(\cdot; x_{1:n})$, $\ell = 1, \ldots, k-1$), and knots $a = t_0 < t_1 < \cdots < t_r < t_{r+1} = b$ such that $t_{1:r} \subseteq x_{1:n}$ and $t_1 \geq x_{k+1}$, we define the space of *kth degree discrete splines* on $[a, b]$ with knots $t_{1:r}$, denoted $\mathcal{DS}_n^k(t_{1:r}, [a, b])$, to contain all functions f on $[a, b]$ such that

$$\begin{gathered} \text{for each } i = 0, \ldots, r, \text{ there is a } k\text{th degree polynomial } p_i \text{ such that } f|_{I_{i,v}} = p_i, \text{ and} \\ \text{for each } i = 1, \ldots, r, \text{ it holds that } (\Delta_n^\ell p_{i-1})(t_i) = (\Delta_n^\ell p_i)(t_i),\ \ell = 0, \ldots, k-1, \end{gathered} \tag{55}$$

where $I_0 = [t_0, t_1]$ and $I_i = (t_i, t_{i+1}]$, $i = 1, \ldots, r$.

Remark 5. It is worth emphasizing again that we treat (in Definition 3, and throughout) a discrete spline as a *function*, defined on the continuum interval $[a, b]$, whereas the classical literature (recall Definition 2) treats a discrete spline as a *vector*: a sequence of function evaluations made on a discrete (and evenly-spaced) subset $[a, b]_v \subseteq [a, b]$.

Remark 6. When $k = 0$ or $k = 1$, the space $\mathcal{DS}_n^k(t_{1:r}, [a, b])$ of kth degree discrete splines with knots $t_{1:r}$ is equal to the space $\mathcal{S}^k(t_{1:r}, [a, b])$ of kth degree splines with knots $t_{1:r}$, as the conditions in (13) and (55) match (for $k = 0$, there is no smoothness condition at the knots, and for $k = 1$, there is only continuity at the knots). When $k \geq 2$, this is no longer true, and the two spaces are different; however, they contain "similar" piecewise polynomial functions for large n, r, which will be made precise in Section 10.

Now denote the linear span of the kth degree falling factorial basis functions defined in (5) by

$$\mathcal{H}_n^k = \mathrm{span}\{h_1^k, \ldots, h_n^k\} = \left\{ \sum_{j=1}^n \alpha_j h_j^k : \alpha_j \in \mathbb{R},\ j = 1, \ldots, n \right\}. \tag{56}$$

Next we show that the span of falling factorial basis functions is a space of discrete splines. The arguments are similar to those for the case of evenly-spaced design points, see, for example, Theorem 8.51 of Schumaker (2007).

Lemma 3. *For any $k \geq 0$, the span $\mathcal{H}_n^k$ of the kth degree falling factorial basis functions, in (56), can be equivalently represented as*

$$\mathcal{H}_n^k = \mathcal{DS}_n^k(x_{(k+1):(n-1)}, [a, b]),$$

the space of kth degree discrete splines on $[a, b]$ with knots in $x_{(k+1):(n-1)} = \{x_{k+1}, \ldots, x_{n-1}\}$.

Proof. We first show that each basis function h_j^k, $j = 1, \ldots, n$ is an element of $\mathcal{DS}_n^k(x_{(k+1):(n-1)}, [a, b])$. Note that h_j^k, $j = 1, \ldots, k+1$ are clearly kth degree discrete splines because they are kth degree polynomials. Fix $j \geq k+2$. The function h_j^k has just one knot to consider, at x_{j-1}. Observe

$$h_j^k|_{[a, x_{j-1}]} = 0, \quad \text{and} \quad h_j^k|_{[x_{j-1}, b]} = \frac{1}{k!} \eta(\cdot; x_{(j-k):(j-1)}).$$

Recall the property (12) of divided differences of Newton polynomials; this gives $(\Delta_n^\ell \eta(\cdot; x_{(j-k):(j-1)}))(x_{j-1}) = 0$ for $\ell = 0, \dots, k-1$, certifying the required property (55) for a kth degree discrete spline.

It is straightforward to show from the structure of their supports that h_j^k, $j = 1, \dots, n$ are linearly independent (we can evaluate them at the design points $x_{1:n}$, yielding a lower triangular matrix, which clearly has linearly independent columns). Furthermore, a standard dimensionality argument shows that the linear space $\mathcal{DS}_n^k(x_{(k+1):(n-1)}, [a,b])$ has dimension $(n-k-1) + (k+1) = n$ (we can expand any function in this space as a linear combination of piecewise polynomials the segments $I_0, \dots, I_{n-k-1}$ then subtract the number of constraints at the knot points). Thus the span of h_j^k, $j = 1, \dots, n$ is all of $\mathcal{DS}_n^k(x_{(k+1):(n-1)}, [a,b])$, completing the proof. □

As we saw in (54), functions in the span of the falling factorial basis do not have continuous derivatives, and thus lack the smoothness of splines, in this particular sense. However, as Lemma 3 reveals, functions in this span are in fact discrete splines; therefore they have an equal number of constraints (as splines) on their degrees of freedom, and this is just expressed in a different way (using discrete derivatives in place of derivatives).

4.2 Matching derivatives

In this subsection, we investigate which kinds of functions f have discrete kth derivatives that everywhere match their kth derivatives,

$$(\Delta_n^k f)(x) = (D^k f)(x), \quad \text{for } x \in (x_k, b]. \tag{57}$$

Notice that, although we call the property (57) an "everywhere" match of derivatives, we restrict our consideration to $x \in (x_k, b]$. This is because for the kth discrete derivative operator (39), recall, it is only for $x \in (x_k, b]$ that $\Delta_n^k(f)$ is defined in terms of a kth divided difference (for $x \in [a, x_k]$, it is defined in terms of a lower order divided difference for the purposes of invertibility).

It is a well-known fact that a kth degree polynomial, $p(x) = \sum_{j=0}^k c_j x^j$, has a kth divided difference equal to its leading coefficient, with respect to any choice of $k+1$ distinct centers $z_1, \dots, z_{k+1}$,

$$p[z_1, \dots, z_{k+1}] = c_k = \frac{1}{k!}(D^k p)(x), \quad \text{for all } x. \tag{58}$$

(See, for example, Theorem 2.51 in Schumaker (2007).) Hence degree k polynomials satisfy the matching derivatives property (57) (note that this covers degree ℓ polynomials, with $\ell \le k$, for which both sides in (57) are zero).

What about piecewise polynomials? By the same logic, a kth degree piecewise polynomial function f will have a discrete kth derivative matching its kth derivative at a point x, *provided that f evaluates to a single polynomial over the centers $x_{i-k+1}, \dots, x_i, x$* used to define $(\Delta_n^k f)(x)$. But, if $x_{i-k+1}, \dots, x_i, x$ straddle (at least) two neighboring segments on which f is a different polynomial, then this will not generally be true. Take as an example the truncated power function $g(x) = (x-t)_+^k / k!$, for $k \ge 2$. Let $x \in (x_i, x_{i+1}]$. Consider three cases. In the first, $x \le t$. Then[4]

$$(\Delta_n^k g)(x) = (D^k g)(x) = 0.$$

In the second case, $x > t$ and $x_{i-k+1} \ge t$. Then

$$(\Delta_n^k g)(x) = (D^k g)(x) = 1.$$

In the third case, $x > t$ and $x_{i+k-1} < t$.[5] Then $(D^k f)(x) = 1$, but $(\Delta_n^k f)(x)$ will vary between 0 and 1. See Figure 4 for a simple empirical example. To summarize: if x is far enough from the underlying knot t in the truncated power function—either to the left of t, or to the right of t and separated by k underlying design points—then the kth discrete derivative and kth derivative at x will match; otherwise, they will not. (This restriction is quite problematic once we think about trying to match derivatives (57) for a kth degree spline with with knots at the design points.)

A remarkable fact about the kth degree falling factorial basis functions (5) is that their kth discrete derivatives and kth derivatives match *at all* x, regardless of how close x lies to their underlying knot points. This result was actually already established in (52), in the proof of Theorem 2 (this is for the piecewise polynomial basis functions, and for the polynomial basis functions, it follows from the property (58) on discrete derivatives of polynomials). For emphasis, we state the full result next as a corollary. We also prove a converse result.

[4]Here we are taking $(D^k g)(x) = 1\{x > t\}$, the choice of left-continuous step function being arbitrary but convenient, and consistent with our treatment of the falling factorial functions.

[5]Note that if t is one of the design points $x_{1:n}$, then this case can only occur when $k \ge 2$ (when $k = 1$, we have $x_{i+k-1} = x_i$, which is defined to be the largest design point strictly less than x, thus we cannot have $x_i < t < x$).

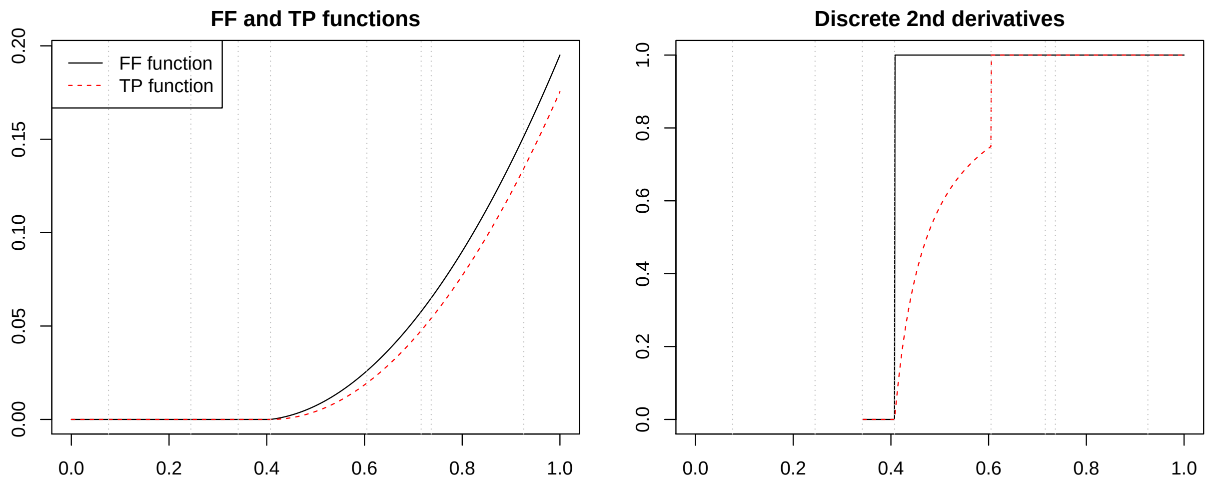

Figure 4: Left panel: falling factorial (FF) function $\frac{1}{2}(x-x_3)(x-x_4)_+$ in black and truncated power (TP) function $\frac{1}{2}(x-x_4)_+^2$ in dashed red. The $n=8$ design points are marked by dotted gray vertical lines. Both functions have a knot at x_4, and both have 2nd derivative equal to $1\{x > x_4\}$. Right panel: discrete 2nd derivatives of the same functions (FF in black, TP in dashed red), for $x \in [x_3, 1]$. Note the discrete 2nd derivative of the FF function matches its 2nd derivative everywhere (this being $1\{x > x_4\}$), but this is not true for the TP function, specifically, it fails for $x \in (x_4, x_5]$.

Corollary 1. *For any $k \geq 0$, each of the kth degree falling factorial basis functions in (5) have matching kth discrete derivatives and kth derivatives, at all $x > x_k$, as in (57). Hence, by linearity, any function in the span $\mathcal{H}_n^k$ of the kth degree falling factorial basis (56), that is, any kth degree discrete spline with knots in $x_{(k+1):(n-1)}$, also satisfies (57).*

Conversely, if f is a kth degree piecewise polynomial with knots in $x_{(k+1):(n-1)}$ and f satisfies property (57), then f must be in the span $\mathcal{H}_n^k$ of the kth degree falling factorial basis (56), that is, f must be a kth degree discrete spline.

Proof. As already discussed, the first statement was already shown, for the piecewise polynomial basis functions, in (52) in the proof of Theorem 2, and for the polynomial basis functions, it is a reflection of the basic fact (58). To prove the converse statement, observe that if f is a kth degree piecewise polynomial and has knots in $x_{(k+1):(n-1)}$, then its kth derivative is piecewise constant with knots in $x_{(k+1):(n-1)}$, and thus can be written as

$$(D^k f)(x) = \alpha_0 + \sum_{j=k+2}^{n} \alpha_j 1\{x > x_{j-1}\}, \quad \text{for } x \in [a, b].$$

for coefficients $\alpha_0, \alpha_{k+2}, \dots, \alpha_n$. But because f satisfies property (57), we have

$$(\Delta_n^k f)(x) = \alpha_0 + \sum_{j=k+2}^{n} \alpha_j 1\{x > x_{j-1}\}, \quad \text{for } x \in (x_k, b].$$

Inverting using Lemma 1, then using linearity of S_n^k, and Theorem 2, we have

$$f(x) = \alpha_0 (S_n^k 1)(x) + \sum_{j=k+2}^{n} \alpha_j h_j^k(x), \quad \text{for } x \in (x_k, b].$$

A staightforward inductive argument shows that $S_n^k 1$ is a kth degree polynomial. Therefore, the above display, along with the fact that f must be a kth degree polynomial on $[a, x_k]$ (it is a kth degree piecewise polynomial and its first knot is at x_{k+1}), shows that f lies in the span of the kth degree falling factorial basis (5). $\square$

Remark 7. In light of the discussion preceeding Corollary 1, it is somewhat remarkable that a kth degree piecewise polynomial with knots at each $x_{k+1}, \dots, x_{n-1}$ can have a matching kth discrete derivative and kth derivative, at all $x \in (x_k, b]$. Recall that for a kth degree polynomial, its kth discrete derivative and kth derivative match at all points, stemming from the property (58) of divided differences of polynomials. For a kth degree piecewise polynomial f with knots $x_{k+1}, \dots, x_{n-1}$, we have *just one evaluation of f on each segment in which f is a polynomial*, yet Corollary 1 says that the kth divided difference $f[x_{i-k+1}, \dots, x_i, x]$ still perfectly reflects the local structure of f around x, in such a way that $(\Delta_n^k f)(x) = (D^k f)(x)$. This is a very different situation than that in (58), and only happens when f has a particular piecewise polynomial structure—given by the span of falling factorial functions.

Remark 8. It is interesting to emphasize the second part of Corollary 1. As highlighted in (54), the kth degree falling factorial functions (5) have discontinuous lower order derivatives at their knots, and hence so do functions in their span (56), that is, so do kth degree discrete splines with knots in $x_{(k+1):(n-1)}$. This may seem like an undesirable property of a piecewise polynomial (although discrete splines do enjoy continuity in discrete derivatives across their knot points). However, if we want our piecewise polynomial to satisfy the matching derivatives property (57), then Corollary 1 tells us that such discontinuities are inevitable, as discrete splines are the *only* ones that satisfy this property.

Remark 9. The result in (52) can be shown to hold at the design points $x = x_i$, $i = k+1, \dots, n$ by directly invoking the fact in (12), on divided differences of Newton polynomials. In other words, that the matching derivatives property (57) holds for the kth degree falling factorial functions at $x = x_i$, $i = k+1, \dots, n$ has a simple proof based on the fact they are truncated Newton polynomials, and (12). However, the fact that it is true *for all* $x \in (x_k, b]$ is much less straightforward, and is due to the lateral recursion obeyed by these basis functions, from Lemma 2.

5 Dual basis

In this section, we construct a natural dual basis to the falling factorial basis in (5), based on discrete derivatives. We begin by building on the matching kth order derivatives property (52) of the piecewise polynomial functions in the kth degree falling factorial basis, to investigate discrete $(k+1)$st order discrete derivatives of such functions.

5.1 Discrete differentiation of one "extra" order

The next lemma reveals a special form for the $(k+1)$st order discrete derivatives of the piecewise polynomials in the kth degree falling factorial basis.

Lemma 4. *For any $k \geq 0$, the piecewise polynomials in the kth degree falling factorial basis, given in the second line of (5), satisfy for each $j \geq k+2$ and $x \in [a, b]$,*

$$(\Delta_n^{k+1} h_j^k)(x) = \begin{cases} \frac{k+1}{x - x_{j-k-1}} & \text{if } x \in (x_{j-1}, x_j] \\ 0 & \text{otherwise}. \end{cases} \tag{59}$$

Proof. For $x \leq x_{j-1}$, it is easy to see $\Delta_n^{k+1} h_j^k(x) = 0$. Thus consider $x \in (x_i, x_{i+1}]$ with $i \geq j-1$. By definition,

$$\begin{aligned} (\Delta_n^{k+1} h_j^k)(x) &= \frac{(\Delta_n^k h_j^k)(x) - (\Delta_n^k h_j^k)(x_i)}{(x - x_{i-k})/(k+1)} \\ &= \frac{1\{x > x_{j-1}\} - 1\{x_i > x_{j-1}\}}{(x - x_{i-k})/(k+1)}. \end{aligned}$$

where in the second line we used property (52), from the proof of Theorem 2. When $x > x_j$, we have $i \geq j$, and both indicators above are equal to 1, so $(\Delta_n^{k+1} h_j^k)(x) = 0$. Otherwise, when $x \in (x_{j-1}, x_j]$, we have $i = j-1$, and only the first indicator above is equal to 1, therefore we get $(\Delta_n^{k+1} h_j^k)(x) = (k+1)/(x - x_{j-k-1})$, as claimed. ☐

Meanwhile, for the pure polynomials in the kth degree falling factorial basis, their $(k+1)$st order discrete derivatives take an even simpler form.

Lemma 5. *For any $k \geq 0$, the polynomial functions in the kth degree falling factorial basis, given in the first line of (5), satisfy for each $j \leq k+1$,*

$$\Delta_n^{k+1} h_j^k = \begin{cases} 1_{(x_{j-1}, x_j]} & \text{if } j \geq 2 \\ 1_{[a, x_1]} & \text{if } j = 1. \end{cases} \tag{60}$$

Proof. Fix any $j \le k+1$. If $x > x_j$, then $(\Delta_n^{k+1} h_j^k)(x)$ is given by a $(j+1)$st order divided difference of the $(j-1)$st degree polynomial h_j^k, and is hence equal to 0. If $x \in (x_i, x_{i+1}]$ for $i < j$ (or $x \in [a, x_1]$ when $i = 0$), then

$$(\Delta_n^{k+1} h_j^k)(x) = \eta(\cdot; x_{1:(j-1)})[x_1, \dots, x_i, x] = \begin{cases} 1 & \text{if } i = j-1 \\ 0 & \text{otherwise,} \end{cases}$$

where we have used the important property of divided differences of Newton polynomials in (12). Observe that $i = j-1$ implies $x \in (x_{j-1}, x_j]$ (or $x \in [a, x_1]$ when $j = 1$), which completes the proof. $\square$

5.2 Constructing the dual basis

Simply identifying natural points of evaluation for the discrete derivative results in Lemmas 4 and 5 gives us a dual basis for the kth degree falling factorial basis. The proof of the next lemma is immediate and hence omitted.

Lemma 6. *For any $k \ge 0$, define the linear functionals λ_i^k, $i = 1, \dots, n$ according to*

$$\begin{aligned} \lambda_i^k f &= (\Delta_n^{k+1} f)(x_i), \quad i = 1, \dots, k+1, \\ \lambda_i^k f &= (\Delta_n^{k+1} f)(x_i) \cdot \frac{x_i - x_{i-k-1}}{k+1}, \quad i = k+2, \dots, n. \end{aligned} \tag{61}$$

Then λ_i^k, $i = 1, \dots, n$ is a dual basis to the kth degree falling factorial basis in (5), in the sense that for all i, j,

$$\lambda_i^k h_j^k = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \tag{62}$$

One general property of a dual basis is that it allows us to explicitly compute coefficients in a corresponding basis expansion: if $f = \sum_{i=1}^n \alpha_i h_i^k$, then for each $i = 1, \dots, n$, applying the linear functional λ_i^k to both sides gives $\alpha_i = \lambda_i^k f$, by (62). Next we develop the implications of this for interpolation with the falling factorial basis.

5.3 Falling factorial interpolation

An immediate consequence of the dual basis developed in Lemma 6 is the following interpolation result.

Theorem 3. *Let y_i, $i = 1, \dots, n$ be arbitrary. For any $k \ge 0$, we can construct a kth degree discrete spline interpolant $f \in \mathcal{H}_n^k$ with knots in $x_{(k+1):(n-1)}$, satisfying $f(x_i) = y_i$, $i = 1, \dots, n$, via*

$$f(x) = \sum_{i=1}^{k+1} (\Delta_n^{k+1} f)(x_i) \cdot h_i^k(x) \;+ \sum_{i=k+2}^{n} (\Delta_n^{k+1} f)(x_i) \cdot \frac{x_i - x_{i-k-1}}{k+1} \cdot h_i^k(x). \tag{63}$$

(Note that the discrete derivatives $(\Delta_n^{k+1} f)(x_i)$, $i = 1, \dots, n$ above, though notationally dependent on f, actually only depend on the points y_i, $i = 1, \dots, n$.) Moreover, the representation in (63) is unique.

Proof. As $\mathcal{H}_n^k$ is n-dimensional, we can find a unique interpolant f passing through any n points y_i, $i = 1, \dots, n$. Let $f = \sum_{i=1}^n \alpha_i h_i^k$. As explained after Lemma 6, for each $i = 1, \dots, n$, we have $\alpha_i = \lambda_i^k f$ for the dual basis defined in (61), that is, $\alpha_i = (\Delta_n^{k+1} f)(x_i)$ for $i \le k+1$ and $\alpha_i = (\Delta_n^{k+1} f)(x_i) \cdot (x_i - x_{i-k-1})/(k+1)$ for $i \ge k+2$. $\square$

Remark 10. The result in (63) can be written in a more explicit form, namely,

$$f(x) = \sum_{i=1}^{k+1} f[x_1, \dots, x_i] \cdot \eta(x; x_{1:(i-1)}) \;+ \sum_{i=k+2}^{n} f[x_{i-k-1}, \dots, x_i] \cdot (x_i - x_{i-k-1}) \cdot \eta_+(x; x_{(i-k):(i-1)}), \tag{64}$$

where we introduce the notation $\eta_+(x; t_{1:r}) = \eta(x; t_{1:r}) \cdot 1\{x > \max(t_{1:r})\}$ for a truncated Newton polynomial. In this form, we can see it as a natural extension of Newton interpolation in (11). The latter (11) constructs a polynomial of degree $n-1$ passing through any n points, whereas the former (64) separates the degree of the polynomial from the number of points, and allows us to construct a piecewise polynomial (specifically, a discrete spline) of degree k, with $n-k-1$ knots, passing through any n points. A nice feature of this generalization is that it retains the property of the classical Newton formula that the coefficients in the interpolatory expansion are simple, explicit, and easy to compute (they are just based on sliding divided differences).

5.4 Implicit form interpolation

To proceed in an opposite direction from our last remark, we now show that the interpolation result in Theorem 3 can be written in a more implicit form.

Corollary 2. *Let y_i, $i = 1, \dots, n$ be arbitrary. For any $k \geq 0$, we can construct a kth degree discrete spline interpolant $f \in \mathcal{H}_n^k$ with knots in $x_{(k+1):(n-1)}$, satisfying $f(x_i) = y_i$, $i = 1, \dots, n$, in the following manner. For $x \in [a, b] \setminus x_{1:n}$, if $x > x_{k+1}$ and i is the smallest index such that $x_i > x$ (with $i = n$ when $x > x_n$), then $f(x)$ is the unique solution of the linear system*

$$f[x_{i-k}, \dots, x_i, x] = 0. \tag{65}$$

If instead $x < x_{k+1}$, then $f(x)$ is the unique solution of the linear system

$$f[x_1, \dots, x_{k+1}, x] = 0. \tag{66}$$

We note that (65), (66) are each linear systems in just one unknown, $f(x)$.

Proof. First consider the case $x > x_{k+1}$, and $x \in (x_{i-1}, x_i)$ with $i \leq n$. Define sequences of augmented design points and target points by

$$\begin{aligned} &\tilde{x}_1 = x_1, \;\; \dots, \;\; \tilde{x}_{i-1} = x_{i-1}, \;\; \tilde{x}_i = x, \qquad \tilde{x}_{i+1} = x_i, \;\; \dots, \;\; \tilde{x}_{n+1} = x_n, \\ &\tilde{y}_1 = y_1, \;\; \dots, \;\; \tilde{y}_{i-1} = y_{i-1}, \;\; \tilde{y}_i = f(x), \;\; \tilde{y}_{i+1} = y_i, \;\; \dots, \;\; \tilde{y}_{n+1} = y_n. \end{aligned} \tag{67}$$

In what follows, we use a subscript $n+1$ (in place of a subscript n) to denote the "usual" quantities of interest defined with respect to design points $\tilde{x}_{1:(n+1)}$ (instead of $x_{1:n}$). In particular, we use Δ_{n+1}^{k+1} to denote the $(k+1)$st order discrete derivative operator defined using $\tilde{x}_{1:(n+1)}$, and $\mathcal{H}_{n+1}^k$ to denote the space of kth degree discrete splines with knots in $\tilde{x}_{(k+1):n}$. By Theorem 3, we can construct an interpolant $f \in \mathcal{H}_{n+1}^k$ passing through $\tilde{y}_{1:(n+1)}$ at $\tilde{x}_{1:(n+1)}$. Note that, by construction, f is also the unique interpolant in $\mathcal{H}_n^k$ passing through $y_{1:n}$ at $x_{1:n}$. Denote the falling factorial basis for $\mathcal{H}_{n+1}^k$ by

$$\tilde{h}_j^k, \;\; j = 1, \dots, n+1.$$

As $f \in \mathcal{H}_n^k$, the coefficient of $\tilde{h}_{i+1}^k$ in the basis expansion of f with respect to $\tilde{h}_j^k$, $j = 1, \dots, n+1$ must be zero (this is because $\tilde{h}_{i+1}^k$ has a knot at $\tilde{x}_i = x$, so if its coefficient is nonzero, then f will also have a knot at x and cannot be in $\mathcal{H}_n^k$). By (63) (applied to $\tilde{x}_{1:(n+1)}$), this means $(\Delta_{n+1}^{k+1} f)(\tilde{x}_{i+1}) = 0$, or equivalently by (64) (applied to $\tilde{x}_{1:(n+1)}$), this means $f[x_{i-k}, \dots, x_{i-1}, x, x_i] = 0$. The desired result (65) follows by recalling that divided differences are invariant to the ordering of the centers.

For the case $x > x_n$, a similar argument applies, but instead of augmenting the design and target points as in (67) we simply append x to the end of $x_{1:n}$ and $f(x)$ to the end of $y_{1:n}$.

For the case $x < x_{k+1}$, note that as f is simply a kth degree polynomial on $[a, x_{k+1}]$, it hence satisfies (66) (any $(k+1)$st order divided difference with centers in $[a, x_{k+1}]$ is zero, recall (58)). This completes the proof. ☐

Remark 11. A key feature of the implicit representation for the discrete spline interpolant as described in Corollary 2 is that it reveals $f(x)$ can be computed in *constant-time*[6], or more precisely, in $O(k)$ operations (independent of the number of knots in the interpolant, and hence of n). This is because we can always express a kth order divided difference as a linear combination of function evaluations (recall (9)): writing $f[x_{i-k}, \dots, x_i, x] = \sum_{j=1}^{k+1} \omega_j f(x_{i-k-1+j}) + \omega_{k+2} f(x)$, we see that (65) reduces to $f(x) = -(\sum_{j=1}^{k+1} \omega_j f(x_{i-k-1+j}))/\omega_{k+2}$, and similarly for (66).

Interestingly, as we prove next, discrete splines are the *only* interpolatory functions satisfying (65), (66) for all x. In other words, equations (65), (66) uniquely define f, which is reminiscent of the implicit function theorem (and serves as further motivation for us to call the approach in Corollary 2 an "implicit" form of interpolation).

Corollary 3. *Given any evaluations $f(x_i)$, $i = 1, \dots, n$ and $k \geq 0$, if for all $x \in [a, b] \setminus x_{1:n}$, the function f satisfies (65) for $x > x_{k+1}$ (where i is the smallest index such that $x_i > x$, with $i = n$ when $x > x_n$), and (66) for $x < x_{k+1}$, then $f \in \mathcal{H}_n^k$, that is, f must be the kth degree discrete spline with knots in $x_{(k+1):(n-1)}$ that interpolates $y_{1:n}$.*

[6] This is not including the time it takes to rank x among the design points: finding the index i before solving (65) will have a computational cost that, in general, depends on n; say, $O(\log n)$ if the design points are sorted and we use binary search. However, note that this would be constant-time if the design points are evenly-spaced, and we use integer divison.

Proof. This proof is similar to the proof of the converse statement in Corollary 1. First, note that (65) implies that the kth discrete derivative of f is piecewise constant on $[x_{k+1}, b]$ with knots in $x_{(k+1):(n-1)}$. Moreover, a simple inductive argument (deferred until Lemma 20 in Appendix B.5) shows that (66) implies f is a kth degree polynomial on $[a, b]$. Therefore we may write

$$(\Delta_n^k f)(x) = \alpha_0 + \sum_{j=k+2}^{n} \alpha_j 1\{x > x_{j-1}\}, \quad \text{for } x \in [a, b],$$

and proceeding as in the proof of Corollary 1 (inverting using Lemma 1, using linearity of S_n^k, then Theorem 2) shows that f is in the span of the falling factorial basis, completing the proof. □

6 Matrix computations

We translate several of our definitions and results derived thus far to a slightly different perspective. While there will be no new results established in this section, phrasing our results in terms of matrices (which act on function values at the design points) will help draw clearer connections to results in previous papers (Tibshirani, 2014; Wang et al., 2014), and will be notationally convenient for some subsequent parts of the paper. We remind the reader that we use "blackboard" fonts for matrices (as in $\mathbb{A}, \mathbb{B}$, etc.), in order to easily distinguish them from operators that act on functions.

6.1 Discrete differentiation

First define the simple difference matrix $\overline{\mathbb{D}}_n \in \mathbb{R}^{(n-1)\times n}$ by

$$\overline{\mathbb{D}}_n = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ \vdots & & & & & \\ 0 & 0 & 0 & \dots & -1 & 1 \end{bmatrix}, \tag{68}$$

and for $k \geq 1$, define the weight matrix $\mathbb{W}_n^k \in \mathbb{R}^{(n-k)\times(n-k)}$ by

$$\mathbb{W}_n^k = \operatorname{diag}\bigg(\frac{x_{k+1} - x_1}{k}, \dots, \frac{x_n - x_{n-k}}{k}\bigg). \tag{69}$$

Then we define the kth order discrete derivative matrix $\mathbb{D}_n^k \in \mathbb{R}^{(n-k)\times n}$ by the recursion

$$\begin{aligned} \mathbb{D}_n &= (\mathbb{W}_n)^{-1}\overline{\mathbb{D}}_n, \\ \mathbb{D}_n^k &= (\mathbb{W}_n^k)^{-1}\overline{\mathbb{D}}_{n-k+1}\, \mathbb{D}_n^{k-1}, \quad \text{for } k \geq 2. \end{aligned} \tag{70}$$

We emphasize that $\overline{\mathbb{D}}_{n-k+1}$ above denotes the $(n-k)\times(n-k+1)$ version of the simple difference matrix in (68). For a function f, denote by $f(x_{1:n}) = (f(x_1), \dots, f(x_n)) \in \mathbb{R}^n$ the vector of its evaluations at the design points $x_{1:n}$. It is not hard to see that the kth discrete derivative matrix $\mathbb{D}_n^k$, applied to $f(x_{1:n})$, yields the vector of the kth discrete derivatives of f at the points $x_{(k+1):n}$, that is,

$$\mathbb{D}_n^k f(x_{1:n}) = (\Delta_n^k f)(x_{(k+1):n}). \tag{71}$$

Lastly, we note that $\mathbb{D}_n^k$ is a banded matrix, with bandwidth $k+1$.

Remark 12. Our definition of the discrete derivative matrices in (70) differs from that in Tibshirani (2014); Wang et al. (2014) and subsequent papers on trend filtering. In these papers, the discrete derivative matrices are defined as

$$\begin{aligned} \mathbb{C}_n &= \overline{\mathbb{D}}_n, \\ \mathbb{C}_n^k &= \overline{\mathbb{D}}_{n-k+1}(\mathbb{W}_n^{k-1})^{-1}\mathbb{C}_n^{k-1}, \quad \text{for } k \geq 2. \end{aligned} \tag{72}$$

We can hence see that $\mathbb{D}_n^k = (\mathbb{W}_n^k)^{-1}\mathbb{C}_n^k$ for each $k \geq 1$, that is, the discrete derivative matrices in (72) are just like those in (70), but without the leading (inverse) weight matrices. The main purpose of (72) in Tibshirani (2014); Wang et al. (2014) was to derive a convenient formula for the total variation of derivatives of discrete splines (represented in terms of discrete derivatives), and as we will see in Theorem 4, and we will arrive at the same formula using (70) (see also Remark 18). In this sense, the discrepancy between (70) and (72) is not problematic (and if the design points are evenly-spaced, then the two definitions coincide). However, in general, we should note that the current definition (70) offers a more natural perspective on discrete derivatives: recalling (71), we see that it connects to Δ_n^k and therefore to divided differences, a celebrated and widely-studied discrete analogue of differentiation.

6.2 Extended discrete differentiation

We can extend the construction in (68), (69), (70) to yield discrete derivatives at all points $x_{1:n}$, as follows.

For $k \geq 1$, define an extended difference matrix $\overline{\mathbb{B}}_{n,k} \in \mathbb{R}^{n \times n}$ by

$$\overline{\mathbb{B}}_{n,k} = \left[\begin{array}{cccccccccc} 1 & 0 & \dots & 0 & & & & & & \\ 0 & 1 & \dots & 0 & & & & 0 & & \\ \vdots & & & & & & & & & \\ 0 & 0 & \dots & 1 & & & & & & \\ & & & -1 & 1 & 0 & \dots & 0 & 0 \\ & 0 & & 0 & -1 & 1 & \dots & 0 & 0 \\ & & & \vdots & & & & & \\ & & & 0 & 0 & 0 & \dots & -1 & 1 \end{array}\right] \begin{array}{l} \left.\vphantom{\begin{array}{c}1\\0\\\vdots\\0\end{array}}\right\} k \text{ rows} \\ \left.\vphantom{\begin{array}{c}1\\0\\\vdots\\0\end{array}}\right\} n-k \text{ rows} \end{array} \tag{73}$$

(note that the top-left $k \times k$ submatrix is the identity matrix $\mathbb{I}_k$, and the bottom-right $(n-k) \times (n-k+1)$ submatrix is $\overline{\mathbb{D}}_{n-k+1}$), and also define an extended weight matrix $\mathbb{Z}_n^k \in \mathbb{R}^{n \times n}$ by

$$\mathbb{Z}_n^k = \operatorname{diag}\bigg(\underbrace{1, \dots, 1}_{k \text{ times}}, \frac{x_{k+1} - x_1}{k}, \dots, \frac{x_n - x_{n-k}}{k}\bigg). \tag{74}$$

Then we define the extended kth order discrete derivative matrix $\mathbb{B}_n^k \in \mathbb{R}^{n \times n}$ by the recursion

$$\begin{aligned} \mathbb{B}_n &= (\mathbb{Z}_n)^{-1} \overline{\mathbb{B}}_{n,1}, \\ \mathbb{B}_n^k &= (\mathbb{Z}_n^k)^{-1} \overline{\mathbb{B}}_{n,k} \, \mathbb{B}_n^{k-1}, \quad \text{for } k \geq 2. \end{aligned} \tag{75}$$

The construction (73), (74), (75) is precisely analogous to what was done in (43), (44), (45), but it is just specialized to the design points, and yields

$$\mathbb{B}_n^k f(x_{1:n}) = (\Delta_n^k f)(x_{1:n}), \tag{76}$$

which is the extension of property (71) to the full set of the design points $x_{1:n}$. Lastly, we note that $\mathbb{B}_n^k$ is again banded, with bandwidth $k+1$, and that the discrete derivative matrix $\mathbb{D}_n^k$ is simply given by the last $n-k$ rows of the extended matrix $\mathbb{B}_n^k$.

6.3 Falling factorial basis

Now define for $k \geq 0$ the falling factorial basis matrix $\mathbb{H}_n^k \in \mathbb{R}^{n \times n}$ to have entries

$$(\mathbb{H}_n^k)_{ij} = h_j^k(x_i), \tag{77}$$

where h_j^k, $j = 1, \dots, n$ are the falling factorial basis functions in (5). The lateral recursion in Lemma 2 implies

$$\mathbb{H}_n^k = \mathbb{H}_n^{k-1} \mathbb{Z}_n^k \begin{bmatrix} \mathbb{I}_k & 0 \\ 0 & \mathbb{L}_{n-k} \end{bmatrix}, \tag{78}$$

where $\mathbb{I}_k$ denotes the $k \times k$ identity matrix, and $\mathbb{L}_{n-k}$ denotes the $(n-k) \times (n-k)$ lower triangular matrix of all 1s. Furthermore, the dual result between discrete differentiation and the falling factorial basis in Lemma 6 can be written as

$$\mathbb{Z}_n^{k+1} \, \mathbb{B}_n^{k+1} \, \mathbb{H}_n^k = \mathbb{I}_n. \tag{79}$$

We note that the results in (78) and (79) were already established in Lemmas 1 and 2 of Wang et al. (2014) (and for the case of evenly-spaced design points, in Lemmas 2 and 4 of Tibshirani (2014)). To be clear, the analogous results in the current paper (Lemmas 2, 4, 5, and 6) are slightly more general, as they hold for arbitrary x, and not just at the design points. (Their proofs are also simpler; in particular Lemma 2, whose proof is quite different and considerably simpler than the proof of Lemma 1 in Wang et al. (2014).)

6.4 Fast matrix multiplication

A nice consequence of (78) and (79), as developed by Wang et al. (2014), is that matrix-vector multiplication using any of $\mathbb{H}_n^k$, $(\mathbb{H}_n^k)^{-1}$, $(\mathbb{H}_n^k)^\mathsf{T}$, $(\mathbb{H}_n^k)^{-\mathsf{T}}$ can be done in $O(nk)$ operations using simple, in-place algorithms, based on iterated scaled cumulative sums, and iterated scaled differences—to be precise, each of these algorithms requires at most $4nk$ flops (kn additions, subtractions, multiplications, and divisions). For convenience, we recap the details in Appendix D.

7 Discrete B-splines

We develop a local basis for $\mathcal{H}_n^k$, the space of kth degree discrete splines with knots in $x_{(k+1):(n-1)}$. This basis bears similarities to the B-spline basis for splines, and is hence called the *discrete B-spline basis*. In this section (as we do throughout this paper), we consider discrete splines with arbitrary design points $x_{1:n}$, defining the underlying discrete derivative operators $\Delta_n^\ell = \Delta^\ell(\cdot; x_{1:n})$, $\ell = 1, \dots, k-1$. For the construction of discrete B-splines, in particular, this presents an interesting conceptual challenge (that is absent in the case of evenly-spaced design points).

To explain this, we note that a key to the construction of B-splines, reviewed in Appendix C.1, is a certain kind of symmetry possessed by the truncated power functions. At its core, the kth degree B-spline with knots $z_1 < \cdots < z_{k+2}$ is defined by a pointwise divided difference of a truncated power function; this is given in (182), but for convenience, we copy it here:

$$P^k(x; z_{1:(k+2)}) = (\cdot - x)_+^k[z_1, \dots, z_{k+2}]. \tag{80}$$

To be clear, here the notation $(\cdot - x)_+^k[z_1, \dots, z_{k+2}]$ means that we are taking the divided difference of the function $z \mapsto (z-x)_+^k$ with respect to the centers $z_1, \dots, z_{k+2}$. The following are two critical observations. First, for fixed x, the map $z \mapsto (z-x)_+^k$ is a kth degree polynomial for $z > x$, and thus if $z_1 > x$, then the divided difference at centers $z_1, \dots, z_{k+2}$ will be zero (this is a $(k+1)$st order divided difference of a kth degree polynomial, recall (58)). Trivially, we also have that the divided difference will be zero if $z_{k+2} < x$, because then we will be taking a divided difference of all zeros. This shows that $P^k(\cdot; z_{1:(k+2)})$ is supported on $[z_1, z_{k+2}]$ (see also (185)). Second (and this is where the symmetry property is invoked), for fixed z, the map $x \mapsto (z-x)_+^k$ is a kth degree spline that has a single knot at z, and hence $P^k(\cdot; z_{1:(k+2)})$, a linear combination of such functions, is a kth degree spline with knots $z_{1:(k+2)}$.

For evenly-spaced design points, an analogous construction goes through for discrete splines, replacing truncated power functions with truncated rising factorial polynomials, as reviewed in Appendix C.2. The key is again symmetry: now $(z-x)(z-x+v)\cdots(z-x+(k-1)v) \cdot 1\{z > x\}$, for fixed x, acts as a polynomial in z over $z > x$, giving the desired support property (when we take divided differences); and for fixed z, it acts as a truncated falling factorial function in x, giving the desired discrete spline property (again after divided differences).

But for arbitrary design points, there is no apparent way to view the argument and the knots in a truncated Newton polynomial in a symmetric fashion. Therefore it is unclear how to proceed in the usual manner as outlined above (and covered in detail in Appendices C.1 and C.2). Our solution is to first define a discrete B-spline at the design points only (which we can do in analogous way to the usual construction), and then prove that the discrete spline interpolant of such values has the desired support structure. For the latter step, the interpolation results in Theorem 3 and Corollary 2 (especially the implicit result in Corollary 2) end up being very useful.

7.1 Construction at the design points

Here we define discrete B-splines directly at the design points $x_{1:n}$. We begin by defining boundary design points

$$x_{-(k-1)} < \cdots < x_{-1} < x_0 = a, \quad \text{and} \quad x_{n+1} = b.$$

(Any such values for $x_{-(k-1)}, \dots, x_{-1}$ will suffice for our ultimate purpose of defining a basis.) For a degree $k \geq 0$, and for each $j = 1, \dots, n$, now define evaluations of a function Q_j^k at the design points by

$$Q_j^k(x_i) = \eta_+(\cdot; x_{(i-k+1):i})[x_{j-k}, \dots, x_{j+1}], \quad i = 1, \dots, n. \tag{81}$$

where recall $\eta_+(x; t_{1:r}) = \eta(x; t_{1:r}) \cdot 1\{x > \max(t_{1:r})\}$ denotes a truncated Newton polynomial, and the notation $\eta_+(\cdot; x_{(i-k+1):i})[x_{j-k}, \dots, x_{j+1}]$ means that we are taking the divided difference of the map $z \mapsto \eta_+(z; x_{(i-k+1):i})$ with respect to the centers $x_{j-k}, \dots, x_{j+1}$. Comparing (80) and (81), we see that Q_j^k, $j = 1, \dots, n$ are defined (over the design points) in a similar manner to $P^k(\cdot; z_{1:(k+2)})$, using sliding sets of centers for the divided differences, and with truncated Newton polynomials instead of truncated power functions.[7]

It is often useful to deal with a normalized version of the function evaluations in (81). Define, for $j = 1, \dots, n$, the function N_j^k at the design points by

$$N_j^k(x_i) = (x_{j+1} - x_{j-k}) \cdot \eta_+(\cdot; x_{(i-k+1):i})[x_{j-k}, \dots, x_{j+1}], \quad i = 1, \dots, n. \tag{82}$$

Next we show a critical property of these normalized function evaluations.

Lemma 7. *For any $k \geq 0$, the function evaluations in (82) satisfy:*

$$N_j^k(x_i) = \delta_{ij}, \quad i, j, = 1, \dots, n, \tag{83}$$

where $\delta_{ij} = 1$ if $i = j$, and $\delta_{ij} = 0$ otherwise.

Proof. Fix any $j = 1, \dots, n$. For $i \geq j + 1$, we have $x_{j-k} < \cdots < x_{j+1} \leq x_i$, hence $N_j^k(x_i)$ is defined by a divided difference of all zeros, and is therefore zero. For $j \geq i + 1$, we claim that

$$\eta_+(x_\ell; x_{(i-k+1):i}) = \eta(x_\ell; x_{(i-k+1):i}), \quad \ell = j - k, \dots, j + 1.$$

This is true because for $\ell = j - k, \dots, i$, the left-hand side is zero (by truncation), but the right-hand side is also zero, as $(j - k){:}i \subseteq (i - k + 1){:}i$. The above display implies

$$N_j^k(x_i) = (x_{j+1} - x_{j-k}) \cdot \eta(\cdot; x_{(i-k+1):i})[x_{j-k}, \dots, x_{j+1}] = 0,$$

with the last equality due to the fact that a $(k + 1)$st order divided difference of a kth order polynomial is zero (recall, for example, (58)). It remains to consider $j = i$. In this case, writing $f[x_{i-k}, \dots, x_{i+1}] = \sum_{\ell=1}^{k+2} \omega_\ell f(x_{i-k-1+\ell})$ by linearity of divided differences (recall (9)), we have

$$N_i^k(x_i) = \omega_{k+2}(x_{i+1} - x_{i-k}) \cdot \eta(x_{i+1}; x_{(i-k+1):i}) = 1,$$

where we have used the explicit form of ω_{k+2} from (9). $\square$

7.2 Interpolation to $[a, b]$

We now interpolate the values defined in (82) to a discrete spline defined on all $[a, b]$. In particular, for $j = 1, \dots, n$, let

$$N_j^k \text{ be the interpolant in } \mathcal{H}_n^k \text{ passing through } \delta_{ij} \text{ at } x_i, \text{ for } i = 1, \dots, n. \tag{84}$$

We refer to the resulting functions N_j^k, $j = 1, \dots, n$ as kth degree normalized *discrete B-splines* or DB-splines. Since $\mathcal{H}_n^k$, the space of kth degree discrete splines with knots $x_{(k+1):(n-1)}$, is an n-dimensional linear space, and each N_j^k is determined by interpolating n values, it is well-defined. We also note that N_j^k, $j = 1, \dots, n$ are linearly independent (this is clear from Lemma 7), and thus they form a basis for $\mathcal{H}_n^k$.

Next we establish that key property the functions N_j^k, $j = 1, \dots, n$ have local supports.

Lemma 8. *For any $k \geq 0$, the kth degree normalized DB-spline basis functions, as defined in (84), have the following support structure:*

$$N_j^k \textit{ is supported on } \begin{cases} [a, x_{j+k}] & \textit{if } j \leq k + 1 \\ [x_{j-1}, x_{j+k}] & \textit{if } k + 2 \leq j \leq n - k - 1 \\ [x_{j-1}, b] & \textit{if } j \geq n - k. \end{cases} \tag{85}$$

Furthermore, for each $j = 1, \dots, n$, we have the explicit expansion in the falling factorial basis:

$$N_j^k = \sum_{i=j}^{(j+k+1) \wedge n} (\mathbb{Z}_n^{k+1} \, \mathbb{B}_n^{k+1})_{ij} \cdot h_i^k, \tag{86}$$

where $\mathbb{B}_n^{k+1} \in \mathbb{R}^{n \times n}$ is the $(k + 1)$st order extended discrete derivative matrix, as in (75), and $\mathbb{Z}_n^{k+1} \in \mathbb{R}^{n \times n}$ is the $(k + 1)$st order extended diagonal weight matrix, as in (74); also, we use the abbreviation $x \wedge y = \min\{x, y\}$.

[7] Moreover, our definition in (81) is in the same spirit (at the design points) as the standard definition of a discrete B-spline in the evenly-spaced case, as given in Appendix C.2. It is not exactly equivalent, as the standard definition (190) uses a truncated rising factorial polynomial, whereas our preference is to use truncated Newton polynomial that more closely resembles a truncated falling factorial in the evenly-spaced case. In the end, this just means that our discrete B-splines look like those from Appendix C.2 after reflection about the vertical axis; compare Figures 5 and 9.

Proof. We will apply the implicit interpolation result from Corollary 2. First consider the middle case, $k+2 \le j \le n-k-1$. If $x > x_{k+1}$ and i is the smallest index such that $x_i > x$, then by (65) we know that $N_j^k(x)$ is determined by solving the linear system

$$N_j^k[x_{i-k}, \dots, x_i, x] = 0.$$

But Lemma 7 tells us that N_j^k, restricted to the design points, is only nonzero at x_j. Therefore the above linear system will have all $N_j^k(x_{i-k}) = \cdots = N_j^k(x_i) = 0$, and thus trivially $N_j^k(x) = 0$ as the solution, unless $i-k \le j \le i$, that is, unless $x \in [x_{j-1}, x_{j+k}]$. If $x < x_{k+1}$, then by (66) we know that $N_j^k(x)$ is determined by solving the linear system

$$N_j^k[x_1, \dots, x_{k+1}, x] = 0.$$

But $N_j^k(x_{i-k}) = \cdots = N_j^k(x_i) = 0$, and again $N_j^k(x) = 0$ is the solution, since $j \ge k+2$. This proves the middle case in (85).

Now consider the first case, $j \le k+1$. If $x > x_{k+1}$ and i is the smallest index such that $x_i > x$, then by (65) we know that $N_j^k(x)$ is determined by solving the linear system in the second to last display, but this gives $N_j^k(x) = 0$ unless $i-k \le j \le i$. As $i \ge k+2$ (since we are assuming $x > x_{k+1}$) and $j \le k+1$, the condition $j \le i$ is always satisfied. The condition $i-k \le j$ translates into $x \le x_{j+k}$, as before, which proves the first case in (85). The last case, $j \ge n-k$, is similar.

Finally, the result in (86) is a direct consequence of the explicit interpolation result in (63) from Theorem 3. □

Remark 13. Lemma 8 shows the kth degree DB-spline basis functions are supported on intervals that each contain at most $k+2$ knots: for $j = k+2, \dots, n-k-1$, N_j^k is supported on $[x_{j-1}, x_{j+k}]$, which contain knots $x_{(j-1):(j+k)}$; and for $j = 1, \dots, k+1$ or $j = n-k, \dots, n$, N_j^k is supported on $[a, x_{j+k}]$ or $[x_{j-1}, b]$, respectively, which contain knots $x_{(k+1):(j+k)}$ or $x_{(j-1):(n-1)}$, respectively. This matches the "support width" of the usual B-splines: recall that the kth degree B-spline basis functions are also supported on intervals containing at most $k+2$ knots, see (186). In fact, when $k=0$ or $k=1$, the normalized DB-spline basis N_j^k, $j = 1, \dots, n$ is "almost" the same as the normalized B-spline basis M_j^k, $j = 1, \dots, n$ defined in (186); it only differs in the left side of the supports of the first $k+1$ basis functions, and the right side of the supports of the last $k+1$ basis functions. This should not be a surprise, as discrete splines of degrees $k=0$ and $k=1$ are simply splines.

Remark 14. A curious fact about DB-splines, as defined in (84), is that they are not always positive on their support. This is in contrast to the usual B-splines, which are always positive when nonzero, see (185). (However, it is consistent with the behavior of standard DB-splines for evenly-spaced design points, see Appendix C.2.) For $k \ge 2$, DB-splines have a negative "ripple" close to their rightmost knot point. See Figure 5 for examples of DB-splines of degree 2.

Remark 15. As DB-splines are discrete splines, in $\mathcal{H}_n^k$ (by construction in (84), via interpolation within this function space), they have the property that their kth derivatives and kth discrete derivatives match everywhere, by Corollary 1. This means that for each $j = 1, \dots, n$, the piecewise constant function $\Delta_n^k N_j^k$ shares the local support of N_j^k, as given by Lemma 8. Figure 5 confirms this numerically. For $k \ge 2$, B-splines—being splines and not discrete splines—do not share their property, as also confirmed in the figure.

The discrete B-spline basis developed in this section finds two primary applications in the remainder of this paper. First, it can be easily modified to provide a basis for the space of discrete natural splines, which we describe in the next subsection. Second, it provides a significantly more stable (that is, better-conditioned) basis for solving least squares problems in discrete splines, described later in Section 8.4.

7.3 Discrete natural splines

Similar to the usual definition of natural splines, we can modify the definition of discrete splines to require lower-degree polynomial behavior on the boundaries, as follows.

Definition 4. As in Definition 3, but with $k = 2m-1 \ge 1$ constrained to be odd, we define the space of *kth degree discrete natural splines* on $[a,b]$ with knots $t_{1:r}$, denoted $\mathcal{DNS}_n^k(t_{1:r}, [a,b])$, to contain all functions f on $[a,b]$ such that (55) holds, and additionally,

$$(\Delta_n^\ell p_0)(t_1) = 0 \text{ and } (\Delta_n^\ell p_r)(t_r) = 0, \; \ell = m, \dots, k. \tag{87}$$

We note that this is equivalent to restricting p_0 and p_r to be polynomials of degree $m-1$.

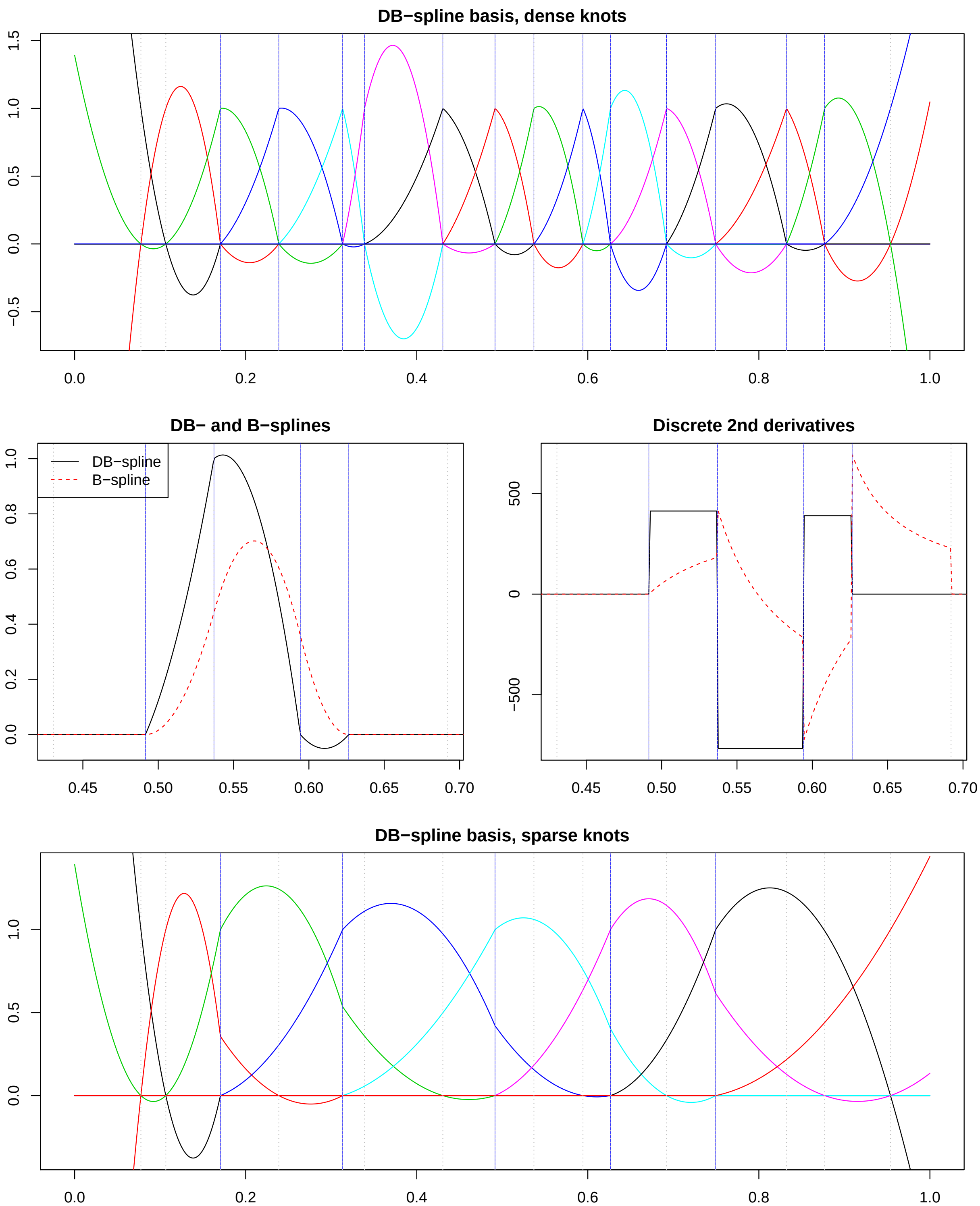

Figure 5: Top row: normalized DB-spline basis for $\mathcal{H}_n^k = \mathcal{DS}_n^k(x_{(k+1):(n-1)}, [a,b])$, where $k = 2$, and the $n = 16$ design points marked by dotted gray vertical lines. The knots are marked by blue vertical lines. We can see that the DB-splines have a negative "ripple" near their rightmost knot point. Middle row, left panel: comparison of a single DB-spline basis function in black, and its B-spline counterpart in dashed red. Right panel: their discrete 2nd derivatives; notice the discrete 2nd derivative matches the 2nd derivative for the DB-spline, but not for the B-spline. Bottom row: normalized DB-spline basis for $\mathcal{H}_n^k = \mathcal{DS}_n^k(t_{1:r}, [a,b])$, for a sparse subset $t_{1:r}$ of the design points of size $r = 5$.

As has been our focus thus far, we consider in this subsection the knot set $x_{(k+1):(n-1)}$, and study the kth degree discrete natural spline space $\mathcal{N}_n^k = \mathcal{DNS}_n^k(x_{(k+1):(n-1)}, [a,b])$. (In the next section, we will discuss the case in which $t_{1:r}$ is an arbitrary subset of the design points, in generality.) On the one hand, since $\mathcal{N}_n^k \subseteq \mathcal{H}_n^k$ by construction, many properties of $\mathcal{H}_n^k$ carry over automatically to $\mathcal{N}_n^k$: for example, the matching derivatives property in Corollary 1 and the interpolation results in Theorem 3 and Corollary 2 all hold for discrete natural splines. On the other hand, other aspects require some work: for example, constructing a basis for $\mathcal{N}_n^k$ is nontrivial. Certainly, it seems to be highly nontrivial to modify the falling factorial basis h_j^k, $j = 1, \dots, n$ in (5) for $\mathcal{H}_n^k$ in order to obtain a basis for $\mathcal{N}_n^k$. Fortunately, as we show in the next lemma, it is relatively easy to modify the DB-spline basis N_j^k, $j = 1, \dots, n$ in (84) (written explicitly in (86)) to form a basis for $\mathcal{N}_n^k$.

Lemma 9. *For any odd $k = 2m - 1 \geq 1$, the space $\mathcal{N}_n^k = \mathcal{DNS}_n^k(x_{(k+1):(n-1)}, [a,b])$ of kth degree discrete natural splines on $[a,b]$ with knots $x_{(k+1):(n-1)}$ is spanned by the following $n-k-1$ functions:*

$$
\begin{aligned}
&L_j^k = \sum_{i=1}^{k+1} x_i^{j-1} \cdot N_i^k, \quad j = 1, \dots, m, \\
&N_j^k, \quad j = k+2, \dots, n-k-1, \\
&R_j^k = \sum_{i=n-k}^{n} (x_i - x_{n-k-1})^{j-1} \cdot N_i^k, \quad j = 1, \dots, m,
\end{aligned}
\tag{88}
$$

where recall N_j^k, $j = 1, \dots, n$ are the DB-spline basis functions in (84).

Proof. A dimensionality argument shows that the linear space $\mathcal{N}_n^k$ has dimension $n-k-1$. Clearly, the functions N_j^k, $j = k+2, \dots, n-k-1$ are kth degree discrete natural splines: each such N_j^k is zero on $[a, x_{j-1}] \supseteq [a, x_{k+1}]$ and is thus a polynomial of degree $m-1$ on this interval; further, it evaluates to zero over the points $x_{(j+1):n} \supseteq x_{(n-k):(n-1)}$ and hence its restriction to $[x_{n-1}, b]$ can also be taken to be a polynomial of degree $m-1$.

It remains to show that the functions L_j^k, $j = 1, \dots, m$ and R_j^k, $j = 1, \dots, m$ defined in the first and third lines of (88) are discrete natural splines, since, given the linear independence of the $n-k-1$ functions in (88) (an implication of the structure of their supports), this would complete the proof. Consider the "left" side functions L_j^k, $j = 1, \dots, m$ (which will have local supports on the left side of the domain). Suppose we seek a linear combination $\sum_{j=1}^{k+1} \alpha_j N_j^k$ of the first $k+1$ DB-splines that meet the conditions in (87); since these DB-splines will evaluate to zero on $x_{(n-k):(n-1)}$, we only need to check the first condition in (87), that is,

$$
\bigg(\Delta_n^\ell \sum_{j=1}^{k+1} \alpha_j N_j^k\bigg)(x_{k+1}) = 0, \quad \ell = m, \dots, k,
$$

Using linearity of the discrete derivative operator, and recalling that $N_j^k(x_i) = \delta_{ij}$ by definition in (84), we conclude the above condition is equivalent to $\mathbb{F}\alpha = 0$, where $\mathbb{F} \in \mathbb{R}^{m \times (k+1)}$ has entries $\mathbb{F}_{\ell-m+1,j} = (\mathbb{B}_n^\ell)_{k+1,j}$ for $\ell = m, \dots, k$ and $j = 1, \dots, k+1$, and where $\mathbb{B}_n^\ell \in \mathbb{R}^{n \times n}$ is the ℓth order extended discrete derivative matrix, as in (75). The null space of $\mathbb{F}$ is simply given by evaluating all degree $m-1$ polynomials over the design points $x_{1:(k+1)}$ (each such vector is certainly in the null space, because its mth through kth discrete derivatives are zero, and there are m such linearly independent vectors, which is the nullity of $\mathbb{F}$). Thus with $\mathbb{P} \in \mathbb{R}^{(k+1)\times m}$ defined to have entries $\mathbb{P}_{ij} = x_i^{j-1}$, we may write any $\alpha \in \mathbb{R}^{k+1}$ such that $\mathbb{F}\alpha = 0$ as $\alpha = \mathbb{P}\beta$ for some $\beta \in \mathbb{R}^m$, and any linear combination satisfying the above condition (in the last display) must therefore be of the form

$$
\sum_{j=1}^{k+1} \alpha_j N_j^k = \sum_{j=1}^{m} \beta_j \sum_{i=1}^{k+1} \mathbb{P}_{ij} N_i^k,
$$

which shows that L_j^k, $j = 1, \dots, m$ are indeed kth degree natural splines. The argument for the "right" side functions R_j^k, $j = 1, \dots, m$ follows similarly. ☐

Later in Section 11, we discuss restricting the domain in trend filtering problem (31) (equivalently, (7)) to the space of discrete natural splines $\mathcal{N}_n^k$, and give an empirical example where this improves its boundary behavior. See Figure 7, where we also plot the discrete natural B-spline basis in (88) of degree 3.

8 Sparse knot sets

While our focus in this paper is the space $\mathcal{H}_n^k = \mathcal{DS}_n^k(x_{(k+1):(n-1)}, [a,b])$, of kth degree discrete splines with knots in $x_{(k+1):(n-1)}$, all of our developments thus far can be appropriately generalized to the space $\mathcal{DS}_n^k(t_{1:r}, [a,b])$, for arbitrary knots $t_{1:r} \subseteq x_{1:n}$ (this knot set could be a sparse subset of the design points, that is, with r much smaller than n). We assume (without a loss of generality) that $t_1 < \cdots < t_r$, where $t_1 \geq x_{k+1}$ (as in Definition 3), and $t_r \leq x_{n-1}$ (for simplicity). Defining i_j, $j = 1, \ldots, r$ such that

$$t_j = x_{i_j}, \quad j = 1, \ldots, r,$$

it is not hard to see that a falling factorial basis h_j^k, $j = 1, \ldots, r+k+1$ for $\mathcal{DS}_n^k(t_{1:r}, [a,b])$ is given by

$$\begin{aligned} h_j^k(x) &= \frac{1}{(j-1)!} \prod_{\ell=1}^{j-1} (x - x_\ell), \quad j = 1, \ldots, k+1, \\ h_{j+k+1}^k(x) &= \frac{1}{k!} \prod_{\ell=i_j-k+1}^{i_j} (x - x_\ell) \cdot 1\{x > x_{i_j}\}, \quad j = 1, \ldots, r. \end{aligned} \tag{89}$$

(In the "dense" knot set case, we have $t_{1:r} = x_{(k+1):(n-1)}$, thus $r = n-k-1$ and $i_j = j+k$, $j = 1, \ldots, n-k-1$, in which case (89) matches (5).) Further, as $\mathcal{DS}_n^k(t_{1:r}, [a,b]) \subseteq \mathcal{H}_n^k$, many results on $\mathcal{H}_n^k$ carry over immediately to the "sparse" knot set case: we can still view the basis in (89) from the same constructive lens (via discrete integration of step functions) as in Theorem 2; functions in $\mathcal{DS}_n^k(t_{1:r}, [a,b])$ still exhibit the same matching derivatives property as in Corollary 1; a dual basis to (89) is given by a subset of the functions in (61) from Lemma 6 (namely, the functions corresponding to the indices $1, \ldots, k+1$ and i_j, $j = 1, \ldots, r$); and interpolation within the space $\mathcal{DS}_n^k(t_{1:r}, [a,b])$ can be done efficiently, precisely as in Theorem 3 or Corollary 2 (assuming we knew evaluations of f at the design points, $f(x_i)$, $i = 1, \ldots, n$).

Meanwhile, other developments—such as key matrix computations involving the falling factorial basis matrix, and the construction of discrete B-splines—do not carry over trivially, and require further explanation; we give the details in the following subsections.

8.1 Matrix computations

The fact that the dual basis result from Lemma 6 implies $\mathbb{Z}_n^{k+1} \mathbb{B}_n^{k+1}$ is the inverse of $\mathbb{H}_n^k$, as in (79), hinges critically on the fact that these matrices are square, which would not be the case for a general knot set $t_{1:r}$, where the corresponding basis matrix would have dimension $n \times (r+k+1)$. However, as we show next, this inverse result can be suitably and naturally extended to a rectangular basis matrix.

Lemma 10. *For any $k \geq 0$, and knots $t_1 < \cdots < t_r$ with $t_{1:r} \subseteq x_{(k+1):(n-1)}$, let us abbreviate $T = t_{1:r}$ and let $\mathbb{H}_T^k \in \mathbb{R}^{n \times (r+k+1)}$ denote the kth degree falling factorial basis matrix with entries*

$$(\mathbb{H}_T^k)_{ij} = h_j^k(x_i),$$

where h_j^k, $j = 1, \ldots, r+k+1$ are the falling factorial basis functions in (89) for $\mathcal{DS}_n^k(t_{1:r}, [a,b])$.

Let $\mathbb{H}_n^k \in \mathbb{R}^{n \times n}$ denote the "usual" kth degree falling factorial basis matrix (77), defined over $x_{(k+1):(n-1)}$, and let $J = \{1, \ldots, k+1\} \cup \{i_j + 1 : j = 1, \ldots, r\}$, where $t_j = x_{i_j}$ for $j = 1, \ldots, r$. Observe that

$$\mathbb{H}_T^k = (\mathbb{H}_n^k)_J, \tag{90}$$

where we write $(\mathbb{H}_n^k)_S$ to represent the submatrix defined by retaining the columns of $\mathbb{H}_n^k$ in a set S. Furthermore, let $\mathbb{A}_n^{k+1} = \mathbb{Z}_n^{k+1} \mathbb{B}_n^{k+1}$, where $\mathbb{Z}_n^{k+1}, \mathbb{B}_n^{k+1} \in \mathbb{R}^{n \times n}$ are the "usual" $(k+1)$st order extended weight and extended discrete derivative matrix, defined over the knot set $x_{(k+1):(n-1)}$, as in (74) and (75), respectively. Then the (Moore-Penrose) generalized inverse of $\mathbb{H}_T^k$ can be expressed as

$$(\mathbb{H}_T^k)^\dagger = (\mathbb{A}_n^{k+1})_J \big(\mathbb{I}_n - (\mathbb{A}_n^{k+1})_{J^c}^\dagger (\mathbb{A}_n^{k+1})_{J^c}\big), \tag{91}$$

where we use $(\mathbb{A}_n^{k+1})_S$ to denote the submatrix formed by retaining the rows of $\mathbb{A}_n^{k+1}$ in a set S, and recall we use $\mathbb{I}_n$ for the $n \times n$ identity matrix. A direct consequence of the above is

$$\mathrm{col}\big((\mathbb{H}_n^k)_J\big) = \mathrm{null}\big((\mathbb{A}_n^{k+1})_{J^c}\big), \tag{92}$$

where we use $\mathrm{col}(\mathbb{M})$ and $\mathrm{null}(\mathbb{M})$ to denote the column space and null space of a matrix $\mathbb{M}$, respectively.

Proof. We abbreviate $\mathbb{H} = \mathbb{H}_n^k$, $\mathbb{A} = \mathbb{A}_n^{k+1}$, and further, $\mathbb{H}_1 = (\mathbb{H}_n^k)_J$, $\mathbb{H}_2 = (\mathbb{H}_n^k)_{J^c}$, $\mathbb{A}_1 = (\mathbb{A}_n^{k+1})_J$, $\mathbb{A}_2 = (\mathbb{A}_n^{k+2})_{J^c}$ for notational simplicity. Let $y \in \mathbb{R}^n$ be arbitrary, and consider solving the linear system

$$\mathbb{H}_1^\mathsf{T} \mathbb{H}_1 \alpha = \mathbb{H}_1^\mathsf{T} y.$$

We can embed this into a larger linear system

$$\begin{bmatrix} \mathbb{H}_1^\mathsf{T} \mathbb{H}_1 & \mathbb{H}_1^\mathsf{T} \mathbb{H}_2 \\ \mathbb{H}_2^\mathsf{T} \mathbb{H}_1 & \mathbb{H}_2^\mathsf{T} \mathbb{H}_2 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \mathbb{H}_1^\mathsf{T} y \\ z \end{bmatrix},$$

which will yield the same solution α as our original system provided we choose z so that we have $\beta = 0$ at the solution in the larger system. Now inverting (using $\mathbb{A}\mathbb{H} = \mathbb{I}_n$), the above system is equivalent to

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \mathbb{A}_1 \mathbb{A}_1^\mathsf{T} & \mathbb{A}_1 \mathbb{A}_2^\mathsf{T} \\ \mathbb{A}_2 \mathbb{A}_1^\mathsf{T} & \mathbb{A}_2 \mathbb{A}_2^\mathsf{T} \end{bmatrix} \begin{bmatrix} \mathbb{H}_1^\mathsf{T} y \\ z \end{bmatrix},$$

that is,

$$\begin{aligned} \alpha &= \mathbb{A}_1 y + \mathbb{A}_1 \mathbb{A}_2^\mathsf{T} z, \\ \beta &= \mathbb{A}_2 y + \mathbb{A}_2 \mathbb{A}_2^\mathsf{T} z. \end{aligned}$$

Setting the second line equal to zero gives $z = -\mathbb{A}_2^\dagger \mathbb{A}_2 y$, and plugging this back into the first gives $\alpha = \mathbb{A}_1(y - \mathbb{A}_2^\dagger \mathbb{A}_2 y)$. As y was arbitrary, this proves the desired result. □

Remark 16. An important implication of (91) is that we can reduce least squares problems in the falling factorial basis $\mathbb{H}_T^k$ to linear systems involving discrete derivatives. This is important for two reasons: first, these discrete derivative systems can be solved in linear-time, due to the bandedness of the discrete derivative matrices; second, these discrete derivative systems are typically much better-conditioned than falling factorial systems. However, it should be noted that these discrete derivative systems can still suffer from poor conditioning for large problem sizes, and discrete B-splines, as developed in Section 8.2, offer a much more stable computational route. This is demonstrated in Section 8.4.

Remark 17. Given the relationship in (90), it is clear that multiplication by $\mathbb{H}_T^k$ and $(\mathbb{H}_T^k)^\mathsf{T}$ can be done in linear-time, using the specialized, in-place algorithms described in Appendix D. To see this, note that for any $\alpha \in \mathbb{R}^{r+k+1}$ we can write $\mathbb{H}_T^k \alpha = \mathbb{H}_n^k \beta$, where we set the entries of $\beta \in \mathbb{R}^n$ on J according to $\beta_J = \alpha$, and we set $\beta_{J^c} = 0$. Also, for any $y \in \mathbb{R}^n$ we can write $(\mathbb{H}_T^k)^\mathsf{T} y = ((\mathbb{H}_n^k)^\mathsf{T} y)_J$.

Owing to (91), multiplication by $(\mathbb{H}_T^k)^\dagger$ and $((\mathbb{H}_T^k)^\dagger)^\mathsf{T}$ can also be done in linear-time; but it is unclear if these can be done entirely with specialized, in-place algorithms. For multiplication by $(\mathbb{H}_T^k)^\dagger$, we can see that this reduces to multiplication by $((\mathbb{A}_n^{k+1})_J$, $(\mathbb{A}_n^{k+1})_{J^c}$, and $(\mathbb{A}_n^{k+1})_{J^c}^\dagger$; while the first two are handled by the algorithms in Appendix D, the third requires solving a linear system in the banded matrix $(\mathbb{A}_n^{k+1})_{J^c}(\mathbb{A}_n^{k+1})_{J^c}^\mathsf{T}$, which as far as we can tell, cannot be done in-place in generality. Multiplication by $((\mathbb{H}_T^k)^\dagger)^\mathsf{T}$ is similar.

8.2 Discrete B-splines

To construct a discrete B-spline or DB-spline basis for $\mathcal{DS}_n^k(t_{1:r}, [a, b])$, we assume that $r \geq k + 2$ (otherwise it would not be possible to construct kth degree DB-splines that have local support). First, we define boundary design points

$$b = x_{n+1} < x_{n+2} < \cdots < x_{n+k+2},$$

a boundary endpoint $\tilde{b} > x_{n+k+2}$, and boundary knots

$$t_{r+1} = x_{n+1}, t_{r+2} = x_{n+2}, \ldots, t_{r+k+1} = x_{n+k+1}.$$

(Any such choice of $x_{n+2}, \ldots, x_{n+k+2}, \tilde{b}$ will suffice; though our construction may appear to have different boundary considerations compared to the "dense" case in Section 7.1, these differences are only notational, and our construction in what follows will reduce exactly to the previous DB-splines when $t_{1:r} = x_{(k+1):(n-1)}$.) Now, for $j = 1, \ldots, k+1$, we define the normalized DB-spline N_j^k as follows:

$$\begin{aligned} &N_j^k \text{ is the unique function } f \in \mathcal{DS}_n^k(t_{1:j}, [a, b]) \text{ satisfying} \\ &\quad f(x_1) = \cdots = f(x_{j-1}) = 0, \quad f(x_j) = 1, \quad \text{and} \\ &\quad f(x_{i_j-k+1}) = \cdots = f(x_{i_j}) = f(x_{i_j+1}) = 0. \end{aligned} \tag{93}$$

(Note that the space $\mathcal{DS}_n^k(t_{1:j},[a,b])$ is $(j+k+1)$-dimensional, and above there are $j+k+1$ linearly independent constraints, hence this system has a unique solution.) Moreover, for $j=1,\dots,r$, we define the normalized DB-spline N_{j+k+1}^k as follows:

$$
\begin{gathered}
N_{j+k+1}^k = f|_{[a,b]} \text{ for the unique function } f \in \mathcal{DS}_{n+k+2}^k(t_{j:(j+k+1)},[a,\tilde{b}]) \text{ satisfying} \\
f(x_{i_j-k}) = \cdots = f(x_{i_j-1}) = f(x_{i_j}) = 0, \quad f(x_{i_{j+1}}) = 1, \quad \text{and} \\
f(x_{i_{j+k+1}-k+1}) = \cdots = f(x_{i_{j+k+1}-1}) = f(x_{i_{j+k+1}}) = f(x_{i_{j+k+1}+1}) = 0,
\end{gathered} \tag{94}
$$

where $\mathcal{DS}_{n+k+2}^k(t_{j:(j+k+1)},[a,\tilde{b}])$ is defined over the extended design points $x_{1:(n+k+2)}$. (Note again this is $(2k+3)$-dimensional, and the above system has $2k+3$ linearly independent constraints, so it has a unique solution.)

The normalized DB-splines N_j^k, $j=1,\dots,r+k+1$, defined above in (93), (94), form a basis for $\mathcal{DS}_n^k(t_{1:r},[a,b])$ (they lie in this space by design; and their defining evaluations imply linear independence). The next result establishes the key local support property; we omit its proof, as it follows from arguments similar to Lemma 8.

Lemma 11. *For any $k \geq 0$, the kth degree normalized DB-spline basis functions, as defined in (93), (94), have support structure:*

$$
N_j^k \textit{ is supported on } \begin{cases} [a,t_j] & \textit{if } j \leq k+1 \\ [t_{j-k-1}, t_j \wedge b] & \textit{if } j \geq k+2, \end{cases} \tag{95}
$$

where recall we abbreviate $x \wedge y = \min\{x,y\}$.

Just as before, in the "dense" knot set case, we see that each kth degree DB-spline is supported on at most $k+2$ knot points. Furthermore, all of the other remarks following Lemma 8 carry over appropriately to the current setting. We refer back to Figure 5 for examples of DB-splines of degree 2, with a "sparse" knot set.

8.3 Evaluation at the design points

This subsection covers a critical computational development: starting from the definitions (93), (94), we can fill in the evaluations of each basis function N_j^k at the design points $x_{1:n}$ using an entirely "local" scheme involving discrete derivative systems. This "local" scheme is both numerically stable (much more stable than solving (93), (94) using say the falling factorial basis) and linear-time.

Fix $j \geq k+2$, and consider the following "local" strategy for computing $N_j^k(x_{1:n})$. First recall that $N_j^k(x_i)=0$ for $x_i \leq t_{j-k-1}$ and $x_i \geq t_j \wedge b$, by (95) in Lemma 11, so we only need to calculate $N_j^k(x_i)$ for $t_{j-k-1} < x_i < t_j \wedge b$. For notational simplicity, and without a loss of generality, set $j=k+2$, and abbreviate $f=N_{k+2}^k$. Between the first knot and second knot, t_1 and t_2, note that we can compute the missing evaluations by solving the linear system:

$$
\begin{gathered}
f[x_{i_1-k+1},\dots,x_{i_1+1},x_{i_1+2}] = 0, \\
f[x_{i_1-k+2},\dots,x_{i_1+2},x_{i_1+3}] = 0, \\
\vdots \\
f[x_{i_2-k-1},\dots,x_{i_2-1},x_{i_2}] = 0.
\end{gathered} \tag{96}
$$

This has i_2-i_1-1 equations and the same number of unknowns, $f(x_{(i_1+1):(i_2-1)})$. Between the second and last knot, t_2 and t_{k+2}, we can set up a similar linear system in order to perform interpolation. From (94), recall that $f(x_i)=0$ for $x_i \geq x_{i_{k+2}-k+1}$, and thus we only need to interpolate from x_{i_2+1} to $x_{i_{k+2}-k}$. Our linear system of discrete derivatives is comprised of the equations:

$$
f[x_{\ell-k},\dots,x_{\ell+1}] = 0, \quad \text{for } i_2+1 \leq \ell \leq i_{k+2}-1,\ \ell \notin \{i_m : m=3,\dots,k+1\}. \tag{97}
$$

These are discrete derivatives at each $x_{\ell+1}$ such that x_ℓ is *not* a knot point. There are exactly $i_{k+2}-i_2-1-(k-1) = i_{k+2}-i_2-k$ such equations and the same number of unknowns, $f(x_{(i_2+1):(i_{k+2}-k)})$. Hence, putting this all together, we have shown how to compute all of the unknown evaluations of $f=N_{k+2}^k$.

For $j \leq k+1$, the "local" strategy for computing $N_j^k(x_{1:n})$ is similar but even simpler. Abbreviating $f=N_j^k$, we solve the linear system:

$$
f[x_{\ell-k},\dots,x_{\ell+1}] = 0, \quad \text{for } k+1 \leq \ell \leq i_j-1,\ \ell \notin \{i_m : m=1,\dots,j-1\}. \tag{98}
$$

This has $i_j - k - 1 - (j-1) = i_j - k - j$ equations and the same number of unknowns, $f(x_{(j+1):(i_j-k)})$.

A critical feature of the linear systems (96), (97), (98) that we must solve in order to calculate the evaluations of the DB-spline basis functions is that they are "local", meaning that they are defined by discrete derivatives over a local neighborhood of $O(k)$ design points. Therefore these systems will be numerically stable to solve, as the conditioning of the discrete derivative matrices of such a small size (just $O(k)$ rows) will not be an issue. Furthermore, since each design point x_i appears in the support of at most $k+2$ basis functions, computing all evaluations of all basis functions, $N_j^k(x_i)$ for $j = 1, \dots, r+k+1$ and $i = 1, \dots, n$, takes linear-time.

8.4 Least squares problems

Finally, we investigate solving least squares problems in the falling factorial basis, of the form

$$\underset{\alpha}{\text{minimize}} \ \|y - \mathbb{H}_T^k \alpha\|_2^2$$

In particular, suppose we are interested in the least squares projection

$$\hat{y} = \mathbb{H}_T^k (\mathbb{H}_T^k)^\dagger y. \tag{99}$$

By (92) in Lemma 10, we know that we can alternatively compute this by projecting onto $\mathrm{null}((\mathbb{A}_n^{k+1})_{J^c})$,

$$\hat{y} = \big(\mathbb{I}_n - (\mathbb{A}_n^{k+1})_{J^c}^\dagger (\mathbb{A}_n^{k+1})_{J^c}\big) y. \tag{100}$$

Another alternative is to use the DB-spline basis constructed in the last subsection. Denoting by $\mathbb{N}_T^k \in \mathbb{R}^{n \times (r+k+1)}$ the matrix with entries $(\mathbb{N}_T^k)_{ij} = N_j^k(x_i)$, where N_j^k, $j = 1, \dots, r+k+1$ are defined in (93), (94) (recall from the last subsection that their evaluations can be computed in linear-time), we have

$$\hat{y} = \mathbb{N}_T^k (\mathbb{N}_T^k)^\dagger y. \tag{101}$$

Naively, solving the falling factorial linear system (99) requires $O(n(r+k)^2)$ operations. A larger issue is that this system will be typically very poorly-conditioned. The discrete derivative linear system (100) gives an improvement in both computation time and conditioning: it requires $O(nk^2)$ operations (because it requires us to solve a linear system in the banded matrix $(\mathbb{A}_n^{k+1})_{J^c}((\mathbb{A}_n^{k+1})_{J^c})^\mathsf{T}$), and will typically be better-conditioned than the falling factorial system. Finally, the DB-spline linear system (101) is computationally the same but improves conditioning even further: it again takes $O(nk^2)$ operations (as it requires us to solve a linear system in the banded matrix $(\mathbb{N}_T^k)^\mathsf{T}(\mathbb{N}_T^k)$), and will typically be much better-conditioned than the discrete derivative system.

To substantiate these claims about conditioning, we ran an empirical experiment with the following setup. For each problem size $n = 100, 200, 500, 1000, 2000, 5000$, we considered both a fixed evenly-spaced design $x_{1:n}$ on $[0,1]$, and a random design given by sorting i.i.d. draws from the uniform distribution $[0,1]$. In each case (fixed or random design), we then selected $r = n/10$ points to serve as knots, drawing these uniformly at random from the allowable set of design points $x_{(k+1):(n-1)}$, where $k = 3$. Next we formed the key matrices $\mathbb{H}_T^k$, $(\mathbb{A}_n^{k+1})_{J^c}$, $\mathbb{N}_n^k$ appearing in the linear systems (99), (100), (101), and computed their condition numbers, where we define the condition number of a matrix $\mathbb{M}$ by

$$\kappa(\mathbb{M}) = \frac{\lambda_{\max}(\mathbb{M}^\mathsf{T}\mathbb{M})}{\lambda_{\min}(\mathbb{M}^\mathsf{T}\mathbb{M})}$$

(with $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$ returning the maximum and minimum eigenvalues of their arguments). We set $\kappa(\mathbb{M}) = \infty$ when $\lambda_{\min}(\mathbb{M}^\mathsf{T}\mathbb{M}) < 0$ due to numerical inaccuracy. Figure 6 plots the condition numbers for these systems versus the problem size n, where the results are aggregated over multiple repetitions: for each n, we took the median condition number over 30 repetitions of forming the design points and choosing a subset of knots. We see that for evenly-spaced design points (fixed design case), the falling factorial systems degrade quickly in terms of conditioning, with an infinite median condition number after $n = 500$; the discrete derivative and DB-spline systems are much more stable, and the latter marks a huge improvement over the former (for example, its median condition is more than 2000 times smaller for $n = 5000$). For unevenly-spaced design points (random design case), the differences are even more dramatic: now both the falling factorial and discrete derivative systems admit an infinite median condition number at some point (after $n = 200$ and $n = 1000$, respectively), yet the DB-spline systems remain stable throughout.

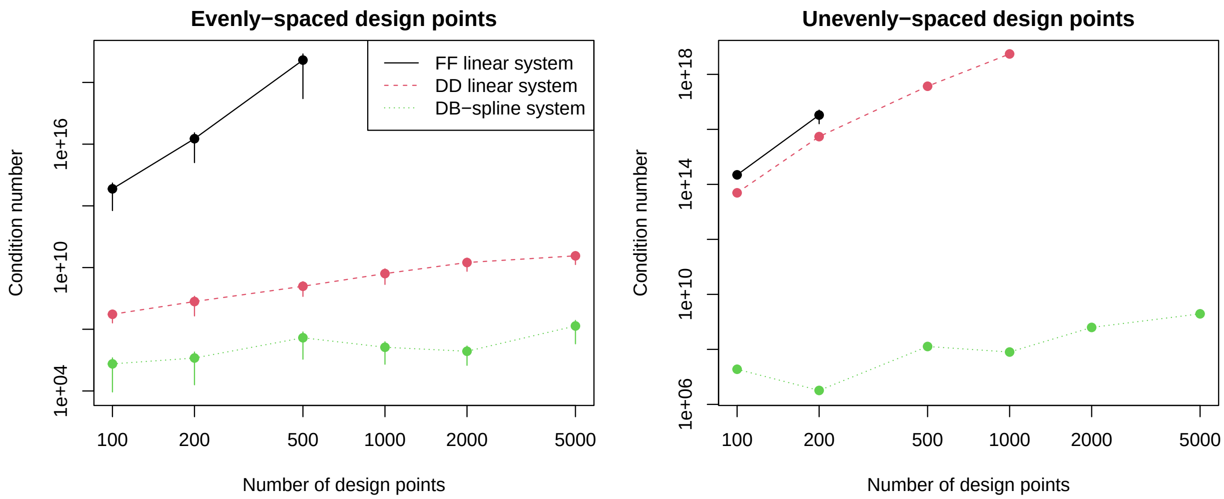

Figure 6: Comparison of condition numbers for three equivalent linear systems, based on falling factorials (FF), discrete derivatives (DD), and discrete B-splines (DB-splines), for projecting onto a space of cubic discrete splines with n design points and $r = n/10$ knots, as n varies from 100 to 5000. The left panel shows a case where the design points are evenly-spaced on $[0, 1]$, and the right shows a case where they are sorted i.i.d. draws from the uniform distribution on $[0, 1]$. In each case, the knots are assigned to be a random subset of the design points, and the median condition number is computed over 30 repetitions. Median absolute deviations are also shown as vertical lines. We can see that the FF systems, in solid black, are by far the worse in either case; the DD systems, in dashed red, provide a marked improvement over FF systems in the evenly-spaced case, but just a moderate improvement in the unevenly-spaced case; and the DB-spline systems, in dotted green, provide a considerable improvement over both, in either case.

9 Representation

In this section, we study the representational properties of discrete splines with respect to two smoothness functionals: total variation and Sobolev seminorms, which serve as the penalty functionals in the variational optimization problems for locally adaptive regression splines and smoothing splines, respectively. In particular, we show that such smoothness functionals, for a discrete spline f, have exact representations in terms of discrete derivatives of f at the design points. We recall that for a kth degree discrete spline f, and $\ell = 1, \dots, k$, we use $(D^\ell f)(x)$ to denotes the ℓth derivative at x when this exists, and the ℓth left derivative when it does not (when x is one of the knot points supporting f).

9.1 Total variation functionals

Below we show that for a kth degree discrete spline, the total variation of its kth derivative can be written in terms of a weighted ℓ_1 norm of its $(k+1)$st discrete derivatives at the design points. Recall that the total variation of a function f on an interval $[a, b]$ is defined by

$$\mathrm{TV}(f) = \sup_{a=z_0<z_1<\cdots<z_N=b} \sum_{i=1}^{N} |f(z_i) - f(z_{i-1})|.$$

The next result is an implication of Corollary 1. It serves as one of the main motivating points behind trend filtering (as an approximation to locally adaptive regression splines); essentially the same result can be found in Lemma 5 of Tibshirani (2014) (for evenly-spaced design points), and Lemma 2 of Wang et al. (2014) (for arbitrary design points).

Theorem 4. *For any $k \geq 0$, and any kth degree discrete spline $f \in \mathcal{H}_n^k$ (with knots in $x_{(k+1):(n-1)}$), as defined in (56), it holds that*

$$\mathrm{TV}(D^k f) = \sum_{i=k+2}^{n} \big|(\Delta_n^{k+1} f)(x_i)\big| \cdot \frac{x_i - x_{i-k-1}}{k+1}. \tag{102}$$

Equivalently, with $f(x_{1:n}) = (f(x_1), \dots, f(x_n)) \in \mathbb{R}^n$ denoting the vector of evaluations of f at the design points,

$$\mathrm{TV}(D^k f) = \big\|\mathbb{W}_n^{k+1} \mathbb{D}_n^{k+1} f(x_{1:n})\big\|_1, \tag{103}$$

where $\mathbb{D}_n^{k+1} \in \mathbb{R}^{(n-k-1)\times n}$ *is the* $(k+1)$*st order discrete derivative matrix, as in* (70)*, and* $\mathbb{W}_n^{k+1} \in \mathbb{R}^{(n-k-1)\times(n-k-1)}$ *is the* $(k+1)$*st order diagonal weight matrix, as in* (69)*.*

Proof. As $D^k f$ is piecewise constant with knots in $x_{(k+1):(n-1)}$ (and our convention is to treat it as left-continuous),

$$\mathrm{TV}(D^k f) = \sum_{i=k+2}^{n} \big|(D^k f)(x_i) - (D^k f)(x_{i-1})\big| \tag{104}$$

$$= \sum_{i=k+2}^{n} \big|(\Delta_n^k f)(x_i) - (\Delta_n^k f)(x_{i-1})\big|, \tag{105}$$

where in the second line we used the matching derivatives result from Corollary 1. Recalling the recursive formulation for Δ_n^{k+1} from (42) establishes the result. □

Remark 18. As discussed previously, recall that Tibshirani (2014); Wang et al. (2014) defined the discrete derivative operators differently, specifically, they defined the operators according the recursion (72) (compare this to the recursion (70) in the current paper). These papers also expressed the total variation result in (103) differently, recall (6), where the modified operator $\mathbb{C}_n^{k+1} = \mathbb{W}_n^{k+1}\mathbb{D}_n^{k+1}$ results from the construction in (72). While the results (103) and (6) are equivalent, the latter is arguably a more natural presentation of the same result, as it invokes the more natural notion of discrete differentiation from this paper (recall Remark 12). Using this notion, it then represents the total variation functional via differences of discrete derivatives (which equal differences of derivatives, recall (105) in the proof of Theorem 4).

Remark 19. Once we assume f lies in an n-dimensional linear space of kth degree piecewise polynomials with knots in $x_{1:n}$, the fact that the representation (103) holds for *some* matrix $\mathbb{W}_n^{k+1}$ is essentially definitional. To see this, we can expand f in a basis for this linear space, $f = \sum_{j=1}^n \alpha_j g_j$, then observe that, for some matrix $\mathbb{Q} \in \mathbb{R}^{n\times n}$ (that depends on this basis, but not on f),

$$\begin{aligned}\mathrm{TV}(D^k f) &= \mathrm{TV}\bigg(\sum_{j=1}^n \alpha_j D^k g_j\bigg) \\ &= \|\mathbb{Q}\alpha\|_1 \\ &= \big\|\mathbb{Q}\,\mathbb{G}^{-1} f(x_{1:n})\big\|_1.\end{aligned}$$

In the second line we used the fact that each $D^k g_j$ is a piecewise constant function (with knots in $x_{1:n}$), and in the third line we simply multiplied by $\mathbb{G} \in \mathbb{R}^{n\times n}$ and its inverse, which has entries $\mathbb{G}_{ij} = g_j(x_i)$. Now in the last line above, if we multiplied by $\mathbb{D}_n^m$ and its "inverse" (in quotes, since this matrix is not square, thus strictly speaking, not invertible), then this would yield a result as in (103) for a particular matrix $\mathbb{W}_n^{k+1}$ (defined in terms of $\mathbb{Q}, \mathbb{G}^{-1}$, and the "inverse" of $\mathbb{D}_n^m$). But to be clear, the fact that (103) holds for a *diagonal* matrix $\mathbb{W}_n^{k+1}$ is what makes the result special, and is tied to the matching derivatives property that is uniquely satisfed kth degree discrete splines. For example, the corresponding matrix $\mathbb{W}_n^{k+1}$ would not be diagonal for kth degree splines.

9.2 L_2-Sobolev functionals

Now we show that for a kth degree discrete spline, where $k = 2m-1$, the integral of the square of its mth derivative can be written in terms of a certain quadratic form of its mth discrete derivatives at the design points. This integral is (the square of) the seminorm naturally associated with the L_2-Sobolev space $\mathcal{W}^{m,2}([a,b])$.

Theorem 5. *For any odd* $k = 2m-1 \geq 1$*, and any* k*th degree discrete spline* $f \in \mathcal{H}_n^k$ *(with knots in* $x_{(k+1):(n-1)}$*), as defined in* (56)*, it holds that*

$$\int_a^b (D^m f)(x)^2 \, dx = \big\|(\mathbb{V}_n^m)^{\frac{1}{2}} \mathbb{D}_n^m f(x_{1:n})\big\|_2^2, \tag{106}$$

where $f(x_{1:n}) = (f(x_1), \dots, f(x_n)) \in \mathbb{R}^n$ *is the vector of evaluations of* f *at the design points, and* $\mathbb{D}_n^m \in \mathbb{R}^{(n-m)\times n}$ *is the* m*th order discrete derivative matrix, as in* (70)*. Moreover,* $\mathbb{V}_n^m \in \mathbb{R}^{(n-m)\times(n-m)}$ *is a symmetric banded matrix (that depends only on* $x_{1:n}$*) of bandwidth* $2m-1$*.*

The proof of Theorem 5 is somewhat intricate and is deferred to Appendix B.6. It relies on several key properties underlying discrete splines, specifically, the recursive property of the falling factorial basis in Lemma 2, and the dual relationship in Lemma 6.

Remark 20. As before (similar to Remark 19 on the total variation representation result), once we assume that f lies in an n-dimensional linear space, the fact the representation (106) holds for *some* matrix $\mathbb{V}_n^m$ is essentially definitional. We can see this by expanding f in terms of a basis for this linear space, $f = \sum_{j=1}^n \alpha_j g_j$, then observing that

$$\begin{aligned}\int_a^b (D^m f)(x)^2 \, dx &= \int_a^b \sum_{i,j=1}^n \alpha_i \alpha_j (D^m g_i)(x)(D^m g_j)(x)\, dx \\ &= \alpha^\mathsf{T} \mathbb{Q} \alpha \\ &= f(x_{1:n})^\mathsf{T} \mathbb{G}^{-\mathsf{T}} \mathbb{Q}\, \mathbb{G}^{-1} f(x_{1:n}),\end{aligned}$$

where $\mathbb{Q}, \mathbb{G} \in \mathbb{R}^{n\times n}$ have entries $\mathbb{Q}_{ij} = \int_a^b (D^m g_i)(x)(D^m g_j)(x)\, dx$ and $\mathbb{G}_{ij} = g_j(x_i)$. In the last line above, if we multiplied by $\mathbb{D}_n^m$ and its "inverse" (in quotes, because this matrix is not square, hence not invertible), then this would yield a result as in (106) for a particular matrix $\mathbb{V}_n^m$ (defined in terms of $\mathbb{Q}, \mathbb{G}^{-1}$, and the "inverse" of $\mathbb{D}_n^m$). To be clear, the fact that (106) holds for a *banded* matrix $\mathbb{V}_n^m$ is highly nontrivial, and this appears to be special to the space of kth degree discrete splines. For example, the corresponding matrix $\mathbb{V}_n^m$ would not be banded for kth degree splines. On the other hand, for splines, the *inverse* of this matrix turns out to be banded; recall Theorem 1.

Remark 21. It is worth noting that the nature of the result in Theorem 5 is, at a high level, quite different from previous results in this paper. Thus far, the core underlying property enjoyed by kth degree discrete splines has been the fact that their kth derivatives and kth discrete derivatives match everywhere, as stated in Corollary 1. This led to the dual basis result in Lemma 6, the implicit form interpolation result in Corollary 2, and the total variation representation result in Theorem 4. Meanwhile, the L_2-Sobolev representation result in Theorem 5 is a statement about connecting a functional of mth derivatives of kth degree discrete splines, where $k = 2m-1$, to their mth discrete derivatives. In other words, this connects derivatives and discrete derivatives whose order does not match the degree of the piecewise polynomial. That this is still possible (and yields a relatively simple and computationally efficient form) reveals another new feature of discrete splines, and brings hope that discrete splines may harbor even more results of this type (discrete-continuous connections) that are yet to be discovered.

The form of the matrix $\mathbb{V}_n^m$ in (106) can be made explicit. This is a consequence of the proof of Theorem 5.

Lemma 12. *The matrix $\mathbb{V}_n^m \in \mathbb{R}^{(n-m)\times(n-m)}$ from Theorem 5 can be defined via recursion, in the following manner. First define a matrix $\mathbb{M} \in \mathbb{R}^{(n-m)\times(n-m)}$ to have entries*

$$\mathbb{M}_{ij} = \int_a^b (D^m h^k_{i+m})(x)(D^m h^k_{j+m})(x)\, dx, \tag{107}$$

where recall h_j^k, $j = 1,\dots,n$ are the falling factorial basis functions in (5). For a matrix $\mathbb{A}$ and positive integers i, j, introduce the notation

$$\mathbb{A}(i,j) = \begin{cases} \mathbb{A}_{ij} & \textit{if } \mathbb{A} \textit{ has at least } i \textit{ rows and } j \textit{ columns} \\ 0 & \textit{otherwise}, \end{cases}$$

as well as $\delta^r_{ij}(\mathbb{A}) = \mathbb{A}(i,j) - \mathbb{A}(i+1,j)$ and $\delta^c_{ij}(\mathbb{A}) = \mathbb{A}(i,j) - \mathbb{A}(i,j+1)$. Then $\mathbb{V}_n^m = \mathbb{V}^{m,m}$ is the termination point of a $2m$-step recursion, initialized at $\mathbb{V}^{0,0} = \mathbb{M}$, and defined as follows:

$$\mathbb{V}^{\ell,0}_{ij} = \begin{cases} \mathbb{V}^{\ell-1,0}_{ij} & \textit{if } i \le m-\ell \\ \delta^r_{ij}(\mathbb{V}^{\ell-1,0}) \cdot \dfrac{2m-\ell}{x_{i+m} - x_{i-(m-\ell)}} & \textit{if } i > m-\ell, \end{cases} \qquad \textit{for } \ell = 1,\dots,m-1, \tag{108}$$

$$\mathbb{V}^{m,0}_{ij} = \delta^r_{ij}(\mathbb{V}^{m-1,0}) \tag{109}$$

$$\mathbb{V}^{m,\ell}_{ij} = \begin{cases} \mathbb{V}^{m,\ell-1}_{ij} & \textit{if } j \le m-\ell \\ \delta^c_{ij}(\mathbb{V}^{m,\ell-1}) \cdot \dfrac{2m-\ell}{x_{j+m} - x_{j-(m-\ell)}} & \textit{if } j > m-\ell, \end{cases} \qquad \textit{for } \ell = 1,\dots,m-1, \tag{110}$$

$$\mathbb{V}^{m,m}_{ij} = \delta^c_{ij}(\mathbb{V}^{m,m-1}). \tag{111}$$

Furthermore, as we show next, the matrix $\mathbb{M}$ in (107) can be expressed in an explicit form (circumventing the need for numerical integration). The proof is an application of integration by parts and is given in Appendix B.7.

Lemma 13. *The entries of the matrix $\mathbb{M} \in \mathbb{R}^{(n-m)\times(n-m)}$ from Lemma 12 can be written explicitly, for $i \geq j$, as*

$$\mathbb{M}_{ij} = \begin{cases} \left(\displaystyle\sum_{\ell=1}^{i-1} (-1)^{\ell-1} (D^{m+\ell-1} h^k_{i+m})(x)(D^{m-\ell} h^k_{j+m})(x) + (-1)^{i-1} (D^{m-i} h^k_{j+m})(x) \right) \Bigg|_a^b & \text{if } i \leq m \\ \left(\displaystyle\sum_{\ell=1}^{m-1} (-1)^{\ell-1} (D^{m+\ell-1} h^k_{i+m})(x)(D^{m-\ell} h^k_{j+m})(x) + (-1)^{m-1} h^k_{j+m}(x) \right) \Bigg|_{x_{i+m-1}}^b & \text{if } i > m, \end{cases} \tag{112}$$

where recall the derivatives of the falling factorial basis functions are given explicitly in (53), and we use the notation

$$f(x)\Big|_s^t = \big(f^-(t) - f^+(s)\big).$$

as well as $f^-(x) = \lim_{t \to x^-} f(t)$ and $f^+(x) = \lim_{t \to x^+} f(t)$.

We conclude this subsection by generalizing Theorem 5. Inspection of its proof shows that the only property of the integration operator (defining the Sobolev functional) that is actually used in Theorem 5 (and Lemma 12) is linearity; we can therefore substantially generalize this representational result as follows.

Theorem 6. *Let L be a linear functional (acting on functions over $[a, b]$). For any odd $k = 2m - 1 \geq 1$, and any kth degree discrete spline $f \in \mathcal{H}^k_n$, as defined in (56), it holds that*

$$L(D^m f)^2 = \big\| (\mathbb{V}^m_{n,L})^{\frac{1}{2}} \mathbb{D}^m_n f(x_{1:n}) \big\|_2^2, \tag{113}$$

where $f(x_{1:n}) = (f(x_1), \ldots, f(x_n)) \in \mathbb{R}^n$ is the vector of evaluations of f at the design points, and $\mathbb{D}^m_n \in \mathbb{R}^{(n-m)\times n}$ is the mth order discrete derivative matrix, as in (70). Further, $\mathbb{V}^m_{n,L} \in \mathbb{R}^{(n-m)\times(n-m)}$ is a symmetric banded matrix (depending only on $x_{1:n}$ and L) of bandwidth $2m - 1$. As before, it can be defined recursively: $\mathbb{V}^m_{n,L}$ is the termination point of the recursion in (108)–(111), but now initialized at the matrix $\mathbb{M}$ with entries

$$\mathbb{M}_{ij} = L(D^m h^k_{i+m})(D^m h^k_{j+m}), \tag{114}$$

where h^k_j, $j = 1, \ldots, n$ are the falling factorial basis functions in (5).

Remark 22. Theorem 6 allows for a generic linear operator L, and hence covers, for example, a weighted L_2-Sobolev functional of the form $\int_a^b (D^m f)(x)^2 w(x)\,dx$ for a weight function w. We could further generalize this to a functional defined by integration with respect to an arbitrary measure μ on $[a, b]$ (Lebesgue-Stieltjes integration). For such a class of functionals, some version of integration by parts, and thus an explicit result for the entries of $\mathbb{M}$ in (114), analogous to Lemma 12, would still be possible.

We emphasize once more that the proof of Theorem 6 follows immediately from that of Theorem 5. It is not clear to us that the spline result in (18) from Theorem 1, due to Schoenberg (1964), would extend as seamlessly to an arbitrary linear functional L. The proof is closely tied to the Peano representation of the B-spline, and therefore for an arbitrary linear functional L, the B-spline itself would need to be replaced by an appropriate kernel.

10 Approximation

Approximation theory is a vast subject, and is particularly well-developed for splines; see, for example, Chapters 6 and 7 of Schumaker (2007); or Chapters 5, 12, and 13 of DeVore and Lorentz (1993). Assuming an evenly-spaced design, Chapter 8.5 of Schumaker (2007) develops approximation results for discrete splines that are completely analogous to standard spline approximation theory. Roughly speaking, Schumaker shows that discrete splines obtain the same order of approximation as splines, once we measure approximation error and smoothness in suitable discrete-time notions.

Extending these results to arbitrary design points seems nontrivial, although it is reasonable to expect that similar approximation results should hold in this case. Instead of pursuing this line of argument, in this section, we give some very simple (crude) approximation results for discrete splines, by bounding their distance to splines and then invoking standard spline approximation results. The intent is not to give approximation results that are of the optimal order—in fact, the approximation rates obtained will be grossly suboptimal—but "good enough" for typical use in nonparametric statistical theory (for example, for bounding the approximation error in trend filtering, as discussed in the next section). A finer analysis of discrete spline approximation may be the topic of future work.

10.1 Proximity of truncated power and falling factorial bases

We can easily bound the L_∞ distance between certain truncated power and falling factorial basis functions, as we show next. Denote by $\mathcal{G}_n^k = \mathcal{S}^k(x_{(k+1):(n-1)}, [a,b])$, the space of kth degree splines on $[a,b]$ with knots in $x_{(k+1):(n-1)}$. As a basis for $\mathcal{G}_n^k$, recall that we have the truncated power basis g_j^k, $j=1,\dots,n$, as in (14), but with $t_{1:r} = x_{(k+1):(n-1)}$ (to be explicit, $g_j^k(x) = (x-x_{j-1})_+^k/k!$, for each $j=k+2,\dots,n$). The first part (115) of the result below is a trivial strengthening of Lemma 4 in Wang et al. (2014), and the second part (116) can be found in the proof of Lemma 13 in Sadhanala and Tibshirani (2019).

Lemma 14. *For design points $a \le x_1 < \cdots < x_n \le b$, let $\delta_n = \max_{i=1,\dots,n-1}\,(x_{i+1}-x_i)$ denote the maximum gap between adjacent points. For $k \ge 0$, let g_j^k, $j=1,\dots,n$ denote the truncated power basis for $\mathcal{G}_n^k$, as in (14) (but with $t_{1:r} = x_{(k+1):(n-1)}$), and h_j^k, $j=1,\dots,n$ denote the falling factorial basis for $\mathcal{H}_n^k$, as in (5). For $k=0$ or $k=1$, and each $j=k+2,\dots,n$, recall that $g_j^k = h_j^k$, and hence $\mathcal{G}_n^k = \mathcal{H}_n^k$. Meanwhile, for $k \ge 2$, and each $j=k+2,\dots,n$,*

$$\|g_j^k - h_j^k\|_{L_\infty} \le \frac{k(b-a)^{k-1}}{(k-1)!}\delta_n, \tag{115}$$

where $\|f\|_{L_\infty} = \sup_{x\in[a,b]}\,|f(x)|$ denotes the L_∞ norm of a function f on $[a,b]$. Hence for each spline $g \in \mathcal{G}_n^k$, there exists a discrete spline $h \in \mathcal{H}_n^k$ such that

$$\mathrm{TV}(D^k h) = \mathrm{TV}(D^k g), \quad \textit{and} \quad \|g-h\|_{L_\infty} \le \frac{k(b-a)^{k-1}}{(k-1)!}\delta_n \cdot \mathrm{TV}(D^k g). \tag{116}$$

Proof. The proof is simple. For each $j=k+2,\dots,n$, consider for $x > x_{j-1}$,

$$\begin{aligned} k! \cdot |g_j^k(x) - h_j^k(x)| &= \prod_{\ell=j-k}^{j-1}(x-x_\ell) - (x-x_{j-1})^k \\ &\le (x-x_{j-k})^k - (x-x_{j-1})^k \\ &= (x_{j-1}-x_{j-k})\sum_{\ell=1}^{k}(x-x_{j-k})^{\ell-1}(x-x_{j-1})^{k-\ell} \\ &\le k\delta_n(x-x_{j-k})^{k-1} \\ &\le k^2\delta_n(b-a)^{k-1}. \end{aligned} \tag{117}$$

This proves the first part (115). As for the second part (116), write $g = \sum_{j=1}^n \alpha_j g_j^k$, and then define

$$h = \sum_{j=1}^{k+1}\frac{\alpha_j}{(j-1)!}x^{j-1} + \sum_{j=k+2}^{n}\alpha_j h_j^k,$$

Note that $h \in \mathcal{H}_n^k$, and we have specified its polynomial part to match that of g. We have

$$\mathrm{TV}(D^k h) = \mathrm{TV}(D^k g) = \|\alpha_{(k+2):n}\|_1.$$

Furthermore, using (117), for any $x \in [a,b]$,

$$|g(x)-h(x)| \le \sum_{j=k+2}^{n}|\alpha_j||g_j^k(x) - h_j^k(x)| \le \frac{k}{(k-1)!}\delta_n \cdot \mathrm{TV}(D^k f),$$

which completes the proof. $\square$

10.2 Approximation of bounded variation functions

Next we show how to couple Lemma 14 with standard spline approximation theory to derive discrete spline approximation results for functions whose derivatives are of bounded variation. First we state the spline approximation result; for completeness we give its proof in Appendix B.8 (similar arguments were used in the proof of Proposition 7 of Mammen and van de Geer (1997)).

Lemma 15. *Let f be a function that is k times weakly differentiable on $[0,1]$, such that $D^k f$ is of bounded variation. Also let $0 \le x_1 < \cdots < x_n \le 1$ be arbitrary design points. Then there exists a kth degree spline $g \in \mathcal{G}_n^k$, with knots in $x_{(k+1):(n-1)}$, such that for $k=0$ or $k=1$,*

$$\mathrm{TV}(D^k g) \le \mathrm{TV}(D^k f), \quad \textit{and} \quad g(x_i) = f(x_i),\ i = 1, \dots, n, \tag{118}$$

and for $k \ge 2$,

$$\mathrm{TV}(D^k g) \le a_k \mathrm{TV}(D^k f), \quad \textit{and} \quad \|f - g\|_{L_\infty} \le b_k \delta_n^k \cdot \mathrm{TV}(D^k f), \tag{119}$$

where $\delta_n = \max_{i=1,\dots,n-1} (x_{i+1} - x_i)$ denotes the maximum gap between adjacent design points, and $a_k, b_k > 0$ are constants that depend only on k.

Combining Lemmas 14 and 15 and using the triangle inequality leads immediately to the following result.

Lemma 16. *Let f be a function that is k times weakly differentiable on $[0,1]$, such that $D^k f$ is of bounded variation. Also let $0 \le x_1 < \cdots < x_n \le 1$ be arbitrary design points. Then there exists a kth degree discrete spline $h \in \mathcal{H}_n^k$, with knots in $x_{(k+1):(n-1)}$, such that for $k=0$ or $k=1$,*

$$\mathrm{TV}(D^k h) \le \mathrm{TV}(D^k f), \quad \textit{and} \quad h(x_i) = f(x_i),\ i = 1, \dots, n, \tag{120}$$

and for $k \ge 2$,

$$\mathrm{TV}(D^k h) \le a_k \mathrm{TV}(D^k f), \quad \textit{and} \quad \|f - h\|_{L_\infty} \le c_k \delta_n \cdot \mathrm{TV}(D^k f), \tag{121}$$

where $\delta_n = \max_{i=1,\dots,n-1} (x_{i+1} - x_i)$ denotes the maximum gap between adjacent design points, and $a_k, c_k > 0$ are constants that depend only on k (note a_k is the same constant as in Lemma 15).

Remark 23. The approximation bound for discrete splines in (121) scales with δ_n, which is weaker than the order δ_n^k approximation we can obtain with splines, in (119). It is reasonable to believe that discrete splines can also obtain an order δ_n^k approximation, with a finer analysis. Before we discuss this further, we emphasize once more that an order δ_n approximation is "good enough" for our eventual statistical purposes, as discussed in the next section, because it will be on the order of $\log n/n$ with high probability when the design points are sorted i.i.d. draws from a continous distribution on $[0,1]$ (for example, Lemma 5 in Wang et al. (2014)), and this is of (much) smaller order than the sought estimation error rates, which (on the L_2 scale, not squared L_2 scale) will always be of the form n^{-r} for $r < 1/2$.

Now, the culprit—the reason that (121) "suffers" a rate of δ_n and not δ_n^k—is the use of truncated power and falling factorial bases in Lemma 14. Fixing any $j \ge k+2$, the fact g_j^k, h_j^k do not have local support means that the factor of $(x - x_{j-k})^{k-1}$ in the line preceding (117) can grow to a large (constant) order, as x moves away from the shared knot point x_{j-k}, and thus in a uniform sense over all $x > x_{j-k}$ (and all $j \ge k+2$), we can only bound it by $(b-a)^{k-1}$, as done in (117). A way to fix this issue would be to instead consider locally-supported bases, that is, to switch over to comparing B-splines and discrete B-splines: with the appropriate pairing, each basis function (B-spline and DB-spline) would be supported on the same interval containing $k+2$ design points, which would have width at most $(k+2)\delta_n$. This should bring the L_∞ distance between pairs of basis functions down to the desired order of δ_n^k.

However, a better way forward, to refining approximation results, seems to be to analyze discrete splines directly (not just analyze their approximation capacity via their proximity to splines). For this, we imagine DB-splines should also play a prominent role: for example, it is not hard to see that the map P defined by $Pf = \sum_{i=1}^n f(x_i) N_i^k$, where N_i^k, $i = 1, \dots, n$ is the DB-spline basis in (84) (written explicitly in (86)), is a bounded linear projector onto the space $\mathcal{H}_n^k$. (We mean bounded with respect to the L_∞ norm, that is, $\|P\| = \sup_{\|g\|_{L_\infty} \le 1} \|Pg\|_{L_\infty} < \infty$.) Thus it achieves within a global constant factor of the optimal approximation error (pointwise for each function f): for any $h \in \mathcal{H}_n^k$, we have $\|f - Pf\|_{L_\infty} \le \|f - h\|_{L_\infty} + \|Pf - Ph\|_{L_\infty} \le (1 + \|P\|)\|f - h\|_{L_\infty}$, which implies

$$\|f - Pf\|_{L_\infty} \le (1 + \|P\|) \cdot \inf_{h \in \mathcal{H}_n^k} \|f - h\|_{L_\infty}.$$

11 Trend filtering

In this section, we revisit trend filtering, in light of our developments on discrete splines in the previous sections. The following subsections outline some computational improvements, and then introduce a variant of trend filtering based on discrete natural splines (which often shows better boundary behavior). Before this, we briefly revisit some aspects of its interpretation and estimation theory, to highlight the application of the matching derivatives result (from Corollary 1) and approximation guarantees (from Lemma 16).

Penalizing differences of kth discrete derivatives. In the trend filtering problem (31), where $\mathbb{D}_n^{k+1} \in \mathbb{R}^{(n-k-1)\times n}$ is the $(k+1)$st order discrete derivative matrix, as in (70), and $\mathbb{W}_n^{k+1} \in \mathbb{R}^{(n-k-1)\times(n-k-1)}$ the $(k+1)$st order diagonal weight matrix, as in (69), note that its penalty can be written as

$$\begin{aligned} \big\|\mathbb{W}_n^{k+1}\mathbb{D}_n^{k+1}\theta\big\|_1 &= \sum_{i=1}^{n-k-1} \big|(\mathbb{D}_n^{k+1}\theta)_i\big| \cdot \frac{x_{i+k+1}-x_i}{k+1} &(122)\\ &= \sum_{i=1}^{n-k-1} \big|(\mathbb{D}_n^{k}\theta)_{i+1} - (\mathbb{D}_n^{k}\theta)_i\big|. &(123)\end{aligned}$$

The first line was given previously in (32), and we copy it here for convenience; the second line is due to the recursive definition (70) of the discrete derivative matrices. In other words, we can precisely interpret the trend filtering penalty as an absolute sum of *differences* of kth discrete derivatives of θ at adjacent design points. This provides the most direct path to the continuous-time formulation of trend filtering: for the unique kth degree discrete spline f with $f(x_{1:n}) = \theta$, it is immedate that (123) is an absolute sum of its kth derivatives at adjacent design points, once we recall the matching derivatives property from Corollary 1; and as $D^k f$ is piecewise constant with knots at the design points, it is easy to see that this equals $\mathrm{TV}(D^k f)$. That is, it is easy to work backwards from (123) through the steps (105), (104), and (102). The conclusion is, of course, as before: the trend filtering problem (31) is equivalent the variational problem (7), where we restrict the optimization domain in the locally adaptive regression spline problem to the space $\mathcal{H}_n^k$ of kth degree splines with knots in $x_{(k+1):(n-1)}$.

Estimation theory via oracle inequalities. Tibshirani (2014) established estimation error bounds for trend filtering by first proving that the trend filtering and (restricted) locally adaptive regression spline estimators, in (7) and (28), are "close" (in the ℓ_2 distance defined with respect to the design points $x_{1:n}$), and then invoking existing estimation results for the (restricted) locally adaptive regression spline estimator from Mammen and van de Geer (1997). These bounds were refined for arbitrary design points in Wang et al. (2014). The conclusion is that the trend filtering estimator $\hat{f}$ in (7) achieves (under mild conditions on the design points) the minimax error rate in (26), over the class of functions $\mathcal{V}^k$ whose kth weak derivative has total variation bounded by a constant $C < \infty$.

It was later shown in Sadhanala and Tibshirani (2019) that the same result could be proved more directly, without a need to bound the distance between the trend filtering and (restricted) locally adaptive spline estimators. The setting in Sadhanala and Tibshirani (2019) is more general (additive models, where the dimension of the design points is allowed to grow with n); here we relay the implication of their results, namely, Theorem 1 and Corollary 1, for (univariate) trend filtering, and explain where the approximation result from Lemma 16 enters the picture. If x_i, $i = 1,\dots,n$ are sorted i.i.d. draws from a continuous distribution $[0,1]$, and $y_i = f_0(x_i) + \epsilon_i$, $i = 1,\dots,n$ for uniformly sub-Gaussian errors ϵ_i, $i = 1,\dots,n$ with mean zero and variance-proxy $\sigma^2 > 0$, then there are constants $c_1, c_2, c_3, n_0 > 0$ depending only on k, σ such that for all $c \geq c_1$, $n \geq n_0$, and $\lambda \geq cn^{\frac{1}{2k+3}}$, the solution $\hat{f}$ in the trend filtering problem (7) satisfies[8]

$$\frac{1}{n}\big\|\hat{f}(x_{1:n}) - f_0(x_{1:n})\big\|_2^2 \leq \frac{1}{n}\big\|h(x_{1:n}) - f_0(x_{1:n})\big\|_2^2 + \frac{6\lambda}{n}\max\{1, \mathrm{TV}(D^k h)\}. \tag{124}$$

with probability at least $1 - \exp(-c_2 c) - \exp(-c_3\sqrt{n})$, simultaneously over all $h \in \mathcal{H}_n^k$ such that $(1/n)\|h(x_{1:n}) - f_0(x_{1:n})\|_2^2 \leq C$. The first term on the right-hand side in (124) is the approximation error, and can be controlled using Lemma 16. When $k = 0$ or $k = 1$, we can see from (120) that we can set it exactly to zero. When $k \geq 2$, assuming the underlying regression function f_0 satisfies $\mathrm{TV}(D^k f_0) \leq 1$, we can see from (121) that we can choose h so that

$$\frac{1}{n}\big\|h(x_{1:n}) - f_0(x_{1:n})\big\|_2^2 \leq \big\|h(x_{1:n}) - f_0(x_{1:n})\big\|_{L_\infty}^2 \leq c_k^2\delta_n^2.$$

When the density of the design points is bounded below by a positive constant, it can be shown (see Lemma 5 of Wang et al. (2014)) that δ_n is on the order of $\log n/n$ with high probability. The right-hand side in the display above is thus on the order of $(\log n/n)^2$ with high probability, and so the first term in (124) is negligible compared to the second. All in all, for any $k \geq 0$, we get that for $\mathrm{TV}(D^k f_0) \leq 1$ and $\lambda = cn^{\frac{1}{2k+3}}$, we can choose h so that the first term in (124) is negligible and the second term is on the order of $n^{-\frac{2k+2}{2k+3}}$ (where we have used the bound on $\mathrm{TV}(D^k h)$ from (120) or (121)). This establishes that trend filtering achieves the desired minimax estimation error rate.

[8]To be clear, the result in (124) is of a somewhat classical oracle-inequality-type flavor, and similar results can be found in many other papers; the theoretical novelty in Sadhanala and Tibshirani (2019) lies in the analysis of additive models with growing dimension, which is given in their Theorem 2 and Corollary 2.

11.1 Computational improvements

We discuss computational implications of our developments on discrete splines for trend filtering.

Efficient interpolation. To state the obvious, both the explicit and implicit interpolation formulae, from Theorem 3 and Corollary 2, respectively, can be applied directly to trend filtering. Starting with the discrete-time solution $\hat{\theta}$ from (31), we can efficiently compute the unique kth degree discrete spline interpolant $\hat{f}$ to these values, that is, efficiently evaluate $\hat{f}(x)$ at any point x. The two different perspectives each have their strengths, explained below.

- To use the explicit formula (63), note that we only need to store the $k+1$ polynomial coefficients, $(\Delta_n^{k+1}\hat{f})(x_i)$, $i = 1, \dots, k+1$, and the coefficients corresponding to the active knots, $(\Delta_n^{k+1}\hat{f})(x_i)$, $i \in I$, where

$$I = \Big\{ i \geq k+2 : (\Delta_n^{k+1}\hat{f})(x_i) \neq 0 \Big\}.$$

  As for the design points, in order to use (63), we similarly only need to store $x_{1:(k+1)}$ as well as $x_{(i-k-1):i}$, $i \in I$. Thus for $r = |I|$ active knots, we need $O(r+k)$ memory and $O((r+k)k)$ operations to compute $\hat{f}(x)$ via (63).

- To use the implicit formulae (65), (66), we need to store all evaluations $\hat{\theta} = \hat{f}(x_{1:n})$, and all design points $x_{1:n}$, that is, we require $O(n)$ memory. Given this, to compute $\hat{f}(x)$ we then need to locate x among the design points, which is at most $O(\log n)$ operations (via binary search), and solve a single linear system in one unknown, which costs $O(k)$ operations to set up. Hence the total cost of finding $\hat{f}(x)$ via (65), (66) is $O(\log n + k)$ operations (or even smaller, down to $O(k)$ operations if the design points are evenly-spaced, because then locating x among the design points could be done with integer division). The implicit interpolation strategy is therefore more efficient when memory is not a concern and the number of active knots r is large (at least $r = \Omega(\log n)$).

DB-spline polishing. Given the trend filtering solution $\hat{\theta}$ in (31), let $\mathbb{C}_n^{k+1} = \mathbb{W}_n^{k+1}\,\mathbb{D}_n^{k+1}$, and define the set of active coordinates $I = \{i : (\mathbb{C}_n^{k+1}\hat{\theta})_i \neq 0\}$ and vector of active signs $s = \mathrm{sign}((\mathbb{C}_n^{k+1}\hat{\theta})_I)$. Based on the Karush-Kuhn-Tucker (KKT) conditions for (31) (see Tibshirani and Taylor (2011) or Tibshirani and Taylor (2012)), it can be shown that

$$\hat{\theta} = \big(\mathbb{I}_n - (\mathbb{C}_n^{k+1})_{I^c}^\dagger (\mathbb{C}_n^{k+1})_{I^c}\big)\big(y - (\mathbb{C}_n^{k+1})_I^\mathsf{T} s\big), \tag{125}$$

where $(\mathbb{C}_n^{k+1})_S$ denotes the submatrix formed by retaining the rows of $\mathbb{C}_n^{k+1}$ in a set S. Recall the extended version of $\mathbb{C}_n^{k+1}$, namely, $\mathbb{A}_n^{k+1} = \mathbb{Z}_n^{k+1}\,\mathbb{B}_n^{k+1}$ from Lemma 10, and define a set $J = \{1, \dots, k+1\} \cup \{i+k+1 : i \in I\}$. Then $(\mathbb{C}_n^{k+1})_{I^c} = (\mathbb{A}_n^{k+1})_{J^c}$, and (125) is the projection of $y - (\mathbb{C}_n^{k+1})_I^\mathsf{T} s$ onto $\mathrm{null}((\mathbb{A}_n^{k+1})_{J^c})$. Thus, by the same logic as that in Section 8.4 (recall the equivalence of (100) and (101)), we can rewrite (125) as

$$\hat{\theta} = \mathbb{N}_T^k (\mathbb{N}_T^k)^\dagger \big(y - (\mathbb{C}_n^{k+1})_I^\mathsf{T} s\big), \tag{126}$$

where $T = \{t_j : j = 1, \dots, r\}$ is the active knot set, with $r = |I|$ and $t_j = x_{j+k}$, $j \in I$, and where $\mathbb{N}_T^k \in \mathbb{R}^{n \times (r+k+1)}$ is the DB-spline basis matrix with entries $(\mathbb{N}_T^k)_{ij} = N_j^k(x_i)$, for N_j^k, $j = 1, \dots, n$ as defined in (93), (94). We argued in Section 8.4 that linear systems in DB-splines (like (126)) have the same computational cost yet a significantly better degree of stability than linear systems in discrete derivatives (like (125)). Hence, a very simple idea for improving the numerical accuracy in trend filtering solutions is as follows: form a candidate solution $\hat{\theta}$, keep only the active set I and active signs s, and then *polish* the solution using DB-splines (126) (note that this requires $O(nk^2)$ operations, due to the bandedness of $\mathbb{N}_T^k$).

DB-spline ADMM. Instead of just using DB-splines post-optimization (to polish an already-computed trend filtering solution), a more advanced idea would be to use DB-splines to improve stability over the course of optimization directly. As an example, we consider a specialized augmented Lagrangian method of multipliers (ADMM) for trend filtering due to Ramdas and Tibshirani (2016). To derive this algorithm, we first rewrite (31), using the recursion (70), as

$$\underset{\theta, z}{\mathrm{minimize}} \;\; \frac{1}{2}\|y - \theta\|_2^2 + \lambda \big\|\overline{\mathbb{D}}_{n-k} z\big\|_1 \quad \text{subject to} \quad z = \mathbb{D}_n^k \theta, \tag{127}$$

and define the augmented Lagrangian, for a parameter $\rho > 0$,

$$L(\theta, z, u) = \frac{1}{2}\|y - \theta\|_2^2 + \lambda \big\|\overline{\mathbb{D}}_{n-k} z\big\|_1 + \frac{\rho}{2}\big\|z - \mathbb{D}_n^k \theta + u\big\|_2^2 - \frac{\rho}{2}\|u\|_2^2. \tag{128}$$

Minimizing over θ, then z, then taking a gradient ascent step with respect to the dual variable u, gives the updates

$$\theta^+ = \big(\mathbb{I}_n + \rho(\mathbb{D}_n^k)^\mathsf{T}\mathbb{D}_n^k\big)^{-1}\big(y + (\mathbb{D}_n^k)^\mathsf{T}(z+u)\big), \tag{129}$$

$$z^+ = \mathop{\mathrm{argmin}}_z \bigg\{\frac{1}{2}\big\|z - \mathbb{D}_n^k\theta + u\big\|_2^2 + \frac{\lambda}{\rho}\big\|\overline{\mathbb{D}}_{n-k}\, z\big\|_1\bigg\}, \tag{130}$$

$$u^+ = u + z - \mathbb{D}_n^k\theta. \tag{131}$$

The z-update in (130) may look at first like the most expensive step, but it can be done with super-efficient, linear-time algorithms for total variation denoising (such algorithms take advantage of the simple pairwise difference structure in the ℓ_1 penalty), for example, based on dynamic programming (Johnson, 2013). The θ-update in (129) is just a banded linear system solve, which is again linear-time, but it is (perhaps surprisingly) the more problematic update in practice due to poor conditioning of the discrete derivative matrices.

Recalling the notable empirical benefits in using DB-splines for similar systems (see Figure 6 in Section 8.4), it is reasonable to believe that DB-splines could provide a big improvement in stability if used within this ADMM algorithm as well. The trick is to first define a *working active set* based on an intermediate value of z, namely,

$$I = \{i : (\overline{D}_{n-k}\, z)_i \neq 0\}.$$

This could, for example, be computed after running a handful of the ADMM iterations in (129)–(131). We then restrict our attention to optimization over $z \in \mathrm{null}((\overline{D}_{n-k})_{I^c}$ and $\theta \in \mathrm{null}((\mathbb{D}_n^{k+1})_{I^c})$ (and upon convergence, we check the KKT conditions for the full problem (127); if not satisfied then we increase the working active set appropriately and repeat). With this restriction, the z-update (130) just becomes a lower-dimensional total variation denoising problem that can still be solved by dynamic programming. More importantly, the θ-update (129) can be now rewritten as

$$\theta^+ = \mathbb{N}_T^k\Big((\mathbb{N}_T^k)^\mathsf{T}\big(\mathbb{I}_n + \rho(\mathbb{D}_n^k)^\mathsf{T}\mathbb{D}_n^k\big)\mathbb{N}_T^k\Big)^{-1}(\mathbb{N}_T^k)^\mathsf{T}\big(y + \rho(\mathbb{D}_n^k)^\mathsf{T}(z+u)\big), \tag{132}$$

Here $\mathbb{N}_T^k \in \mathbb{R}^{n\times(r+k)}$ is the DB-spline basis matrix defined with respect to the active knots $T = \{t_j : j = 1,\dots,r\}$, with $r = |I|$ and $t_j = x_{j+k}$, $j \in I$. The step (132) is still a banded linear system solve, and thus still linear-time, but is much better-conditioned (the DB-spline basis matrix $\mathbb{N}_T^k$ acts something like a rectangular preconditioner). Careful implementation and comparisons are left to future work.

11.2 Natural trend filtering

For odd $k = 2m-1 \geq 1$, consider further restricting the domain in the continuous-time trend filtering problem (7) to the space $\mathcal{N}_n^k$ of kth degree discrete natural splines on $[a,b]$ with knots $x_{(k+1):(n-1)}$ (as defined in Section 7.3):

$$\mathop{\mathrm{minimize}}_{f\in\mathcal{N}_n^k} \;\; \frac{1}{2}\sum_{i=1}^n \big(y_i - f(x_i)\big)^2 + \lambda\, \mathrm{TV}(D^k f). \tag{133}$$

As motivation for this, recall the smoothing spline problem (17) inherently gives rise to a natural spline as its solution, which can have better boundary behavior (than a normal spline without any boundary constraints). In fact, looking back at Figure 2, we can see evidence of this: despite deficiencies in coping with heterogeneous smoothness, the smoothing spline estimates (bottom row) have better boundary behavior than trend filtering (top right)—see, in particular, the very right side of the domain.

The estimator defined by (133), which we call *natural trend filtering*, can be recast in a familiar discrete-time form:

$$\begin{aligned} \mathop{\mathrm{minimize}}_{\theta} \quad & \frac{1}{2}\|y-\theta\|_2^2 + \lambda\big\|\mathbb{W}_n^{k+1}\mathbb{D}_n^{k+1}\theta\big\|_1 \\ \text{subject to} \quad & \theta_{1:m} = \mathbb{P}_1^m \theta_{(m+1):(k+1)} \\ & \theta_{(n-m+1):n} = \mathbb{P}_2^m \theta_{(n-k):(n-m)}, \end{aligned} \tag{134}$$

Here $\mathbb{P}_1^m \in \mathbb{R}^{m\times m}$ is a matrix that performs polynomial extrapolation from function values on $x_{(m+1):(k+1)}$ to values on $x_{1:m}$, that is, for any polynomial p of degree $m-1$,

$$p(x_{1:m}) = \mathbb{P}_1^m p(x_{(m+1):(k+1)}),$$

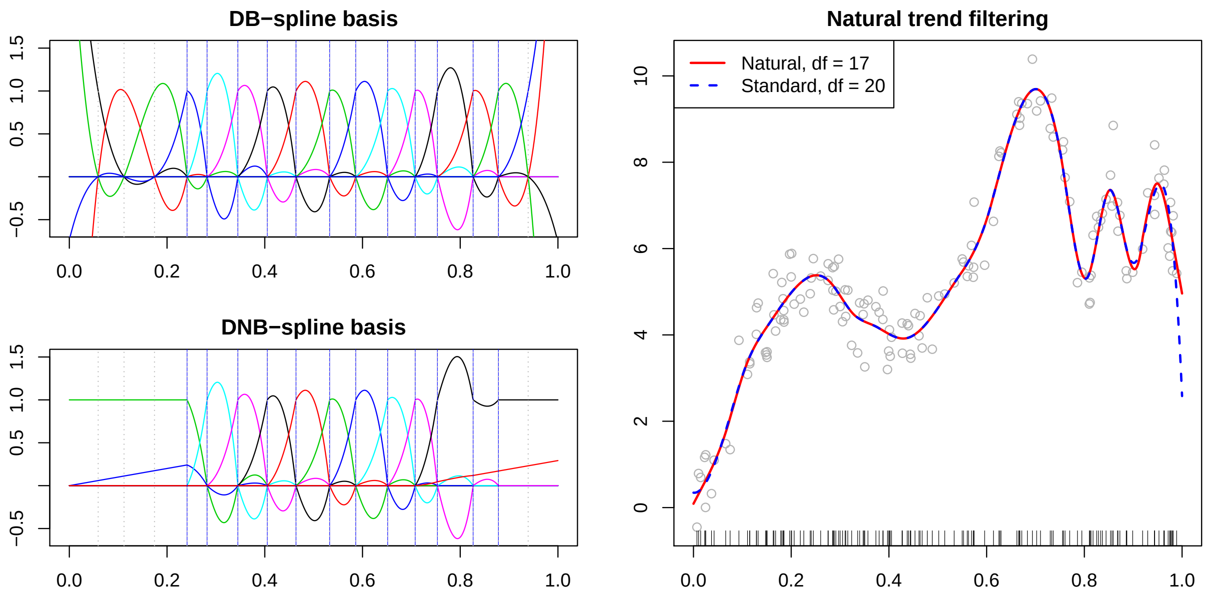

Figure 7: Left panel: comparison of discrete B-spline (DB-spline) and discrete natural B-spline (DNB-spline) bases, for $\mathcal{H}_n^k$ and $\mathcal{N}_n^k$, respectively, where $k = 3$ and the $n = 16$ design points marked by dotted gray vertical lines. The knots are marked by blue vertical lines. We can see that the middle $n - 4$ DB-splines are also DNB-splines; and the key difference is that the two leftmost and rightmost DNB-splines are linear beyond the boundary knots. Right panel: comparison of natural trend filtering and trend filtering on the same data example as in Figure 2. They are computed using the same value of λ; we can see that natural trend filtering, in solid red, essentially matches trend filtering on the entirety of the domain, but it has better boundary behavior and uses fewer degrees of freedom (informally, it does not "waste" degrees of freedom in fitting at the extreme ends of the domain)).

and similarly, $\mathbb{P}_2^m \in \mathbb{R}^{m \times m}$ performs polynomial extrapolation from $x_{(n-k):(n-m)}$ to $x_{(n-m+1):n}$. Observe that (134) is just a standard trend filtering problem where the first m and last m coordinates of θ are just linear combinations of the second m and second-to-last m, respectively. Computationally, this is only a small variant on trend filtering (that is, it would require only a small tweak on existing optimization approaches for trend filtering). Thanks to the development of the DB-spline basis for discrete natural splines (see (88) in Lemma 9), the stability advantages of using DB-splines for trend filtering, as outlined in the last subsection, should carry over here as well. Finally, Figure 7 displays natural trend filtering fitted to the same data as in Figure 2, where we can indeed see that the boundary behavior improves on the right side of the domain.

12 BW filtering

We revisit Bohlmann-Whittaker (BW) filtering, focusing on the case of arbitrary design points. We first define a (slight) variant of the classical BW filter with a weighted penalty, then develop connections to the smoothing spline.

12.1 Weighted BW filtering

Recall that for a unit-spaced design, the BW filter is defined in terms of an quadratic program with a squared ℓ_2 penalty on forward differences, as given in (36). For an arbitrary design $x_{1:n}$, Whittaker proposed to replace forward differences by divided differences, in a footnote of his famous 1923 paper (Whittaker, 1923), resulting in (37). As we alluded to in Section 2.7, we will argue in what follows that it is in several ways more natural to replace the penalty in (37) with the weighted version (38), so that the problem becomes

$$\underset{\theta}{\text{minimize}} \ \|y - \theta\|_2^2 + \lambda \big\| (\mathbb{W}_n^m)^{\frac{1}{2}} \mathbb{D}_n^m \theta \big\|_2^2. \tag{135}$$

Here, $\mathbb{D}_n^m \in \mathbb{R}^{(n-m) \times n}$ is the mth order discrete derivative matrix, as in (70), and $\mathbb{W}_n^m \in \mathbb{R}^{(n-m) \times (n-m)}$ is the mth order diagonal weight matrix, as in (69). For convenience, we copy over (38), to emphasize once again that the form of the penalty in (135) is

$$\big\|(\mathbb{W}_n^m)^{\frac{1}{2}}\mathbb{D}_n^m\theta\big\|_2^2 = \sum_{i=1}^{n-m} (\mathbb{D}_n^m\theta)_i^2 \cdot \frac{x_{i+m}-x_i}{m}. \tag{136}$$

We note once again the strong similarity between the weighted BW filter in (135) and trend filtering in (31), that is, the strong similarity between their penalties in (136) and (122), respectively—the latter uses a weighted squared ℓ_2 norm of discrete derivatives (divided differences), while the former uses a weighted ℓ_1 norm.

We now list three reasons why the weighted BW problem (135) may be preferable to the classical unweighted one (37) for arbitrary designs (for evenly-spaced design points, the two penalties are equal up to a global constant, which can be absorbed into the tuning parameter; that is, problems (135) and (37) are equivalent modulo a rescaling of λ).

1. For $m=1$ and $k=1$, Theorem 1 tells us for any natural linear spline f on $[a,b]$ with knots at the design points $x_{1:n}$, we have the exact representation

$$\int_a^b (Df)(x)^2\,dx = \sum_{i=1}^{n-1} \big(\mathbb{D}_n f(x_{1:n})\big)_i^2 \cdot (x_{i+1}-x_i). \tag{137}$$

   This means that for $m=1$, the smoothing spline problem (17) is equivalent to the weighted BW problem (135). That is, to be perfectly explicit (and to emphasize the appealing simplicity of the conclusion), the following two problems are equivalent:

$$\underset{f}{\text{minimize}} \; \sum_{i=1}^n \big(y_i - f(x_i)\big)^2 + \lambda \int_a^b (Df)(x)^2\,dx,$$

$$\underset{\theta}{\text{minimize}} \; \|y-\theta\|_2^2 + \lambda \sum_{i=1}^{n-1} (\theta_i - \theta_{i+1})^2 \cdot (x_{i+1}-x_i),$$

   in the sense that their solutions satisfy $\hat\theta = \hat f(x_{1:n})$.

2. For $m=2$ and $k=3$, we prove in Theorem 7 in the next subsection that the weighted BW filter and smoothing spline are "close" in ℓ_2 distance (for enough large values of their tuning parameters). This enables the weighted BW filter to inherit the favorable estimation properties of the smoothing spline (over the appropriate L_2-Sobolev classes), as we show in Corollary 4.

3. Empirically, the weighted BW filter seems to track the smoothing spline more closely than the unweighted BW filter does, for arbitrary design points. The differences here are not huge (both versions of the discrete-time BW filter are typically quite close to the smoothing spline), but still, the differences can be noticeable. Figure 8 gives an example.

12.2 Bounds on the distance between solutions

To study the distance between smoothing spline and weighted BW filtering solutions, it helps to first recall a notion of similarity between matrices: positive semidefinite matrices $\mathbb{A},\mathbb{B} \in \mathbb{R}^{n\times n}$ are said to be (σ,τ)-*spectrally-similar*, for $0<\tau\le 1\le\sigma$, provided that

$$\tau u^\mathsf{T}\mathbb{B}u \le u^\mathsf{T}\mathbb{A}u \le \sigma u^\mathsf{T}\mathbb{B}u, \quad \text{for all } u\in\mathbb{R}^n. \tag{138}$$

Spectral similarity is commonly studied in certain areas of theoretical computer science, specifically in the literature on graph sparsification (see, for example, Batson et al. (2013) for a nice review). The next result is both a simplification and sharpening of Theorem 1 in Sadhanala et al. (2016). Its proof follows from direct examination of the stationarity conditions for optimality and application of (138), and is given in Appendix B.9.

Lemma 17. *Let $\mathbb{A},\mathbb{B}$ be (σ,τ)-spectrally-similar, and let $\hat\theta_a,\hat\theta_b$ denote solutions in the quadratic problems*

$$\underset{\theta}{\text{minimize}} \; \|y-\theta\|_2^2 + \lambda_a \theta^\mathsf{T}\mathbb{A}\theta, \tag{139}$$

$$\underset{\theta}{\text{minimize}} \; \|y-\theta\|_2^2 + \lambda_b \theta^\mathsf{T}\mathbb{B}\theta, \tag{140}$$

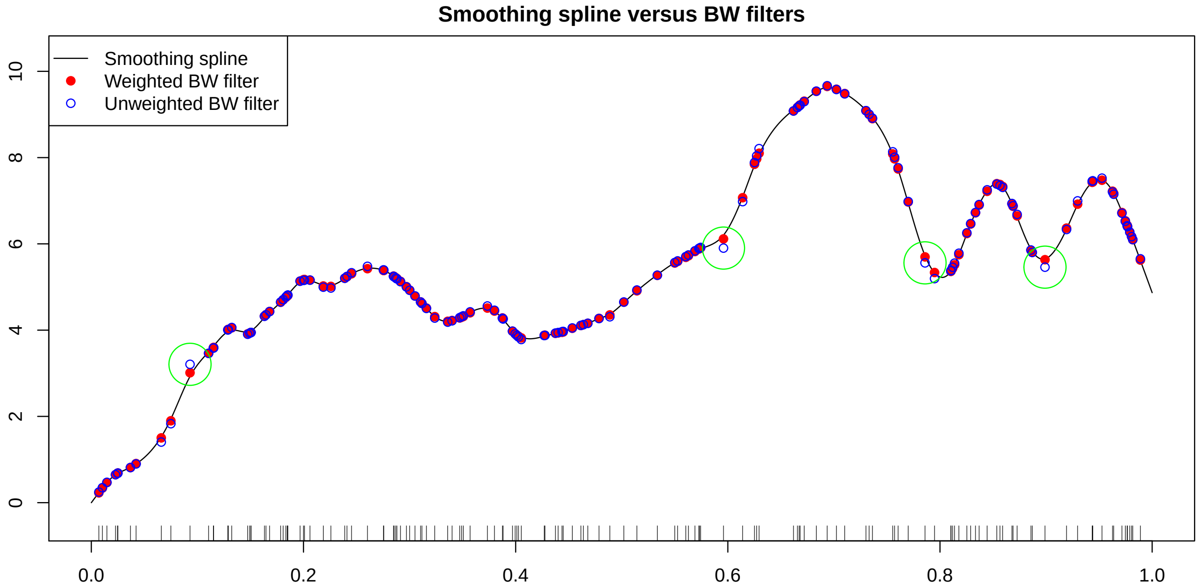

Figure 8: Comparison of spline smoothing and BW filtering, both weighted (see (135)) and unweighted (see (37)) variants, on the same example data as in Figure 2, with $m = 2$ in all cases. Note that the $n = 150$ design points, marked by ticks on the horizontal axis, are not evenly-spaced, and hence there is a meaningful difference between the two versions of the BW filter. The smoothing spline solution, plotted as a black line, was fit at the same tuning parameter value λ as in the bottom right plot of Figure 2 (where it had 30 df). Both BW filter solutions are shown as a sequence of discrete points, at $(x_i, \hat{\theta}_i)$, $i = 1, \ldots, n$; being truly discrete-time estimators, this is the purest representation of their behavior. The weighted BW filter, plotted as solid red points, was fit at the same tuning parameter value λ as the smoothing spline; the unweighted BW filter, plotted as hollow blue points, was fit at the value λ/n (which seems to be the single best choice of tuning parameter to adjust for the difference in scale of its penalty). We can see that, for the most part, the hollow blue points are essentially directly on top of the solid red points, which are on top of the smoothing spline curve. However, in a few locations (specifically, around 0.1, 0.6, 0.8, and 0.9 on the x-axis, highlighted by green circles), the hollow blue points are noticeably far from the solid red points, and in these locations the former is farther than the latter from the smoothing spline curve.

respectively. Then for any $\lambda_a, \lambda_b \geq 0$, it holds that

$$\|\hat{\theta}_a - \hat{\theta}_b\|_2^2 \leq \frac{1}{2}(\lambda_b/\tau - \lambda_a)\hat{\theta}_a^\mathsf{T} \mathbb{A} \hat{\theta}_a + \frac{1}{2}(\sigma\lambda_a - \lambda_b)\hat{\theta}_b^\mathsf{T} \mathbb{B} \hat{\theta}_b. \tag{141}$$

In particular, for any $\lambda_b \geq \sigma\lambda_a$, it holds that

$$\|\hat{\theta}_a - \hat{\theta}_b\|_2^2 \leq \frac{1}{2}(1/\tau - 1/\sigma)\lambda_b \hat{\theta}_a^\mathsf{T} \mathbb{A} \hat{\theta}_a. \tag{142}$$

We now show that the matrices featured in the quadratic penalties in the smoothing spline and weighted BW filtering problems, (23) and (135), are spectrally similar for $m = 2$, and then apply Lemma 17 to bound the ℓ_2 distance between the corresponding solutions. The proof is given in Appendix B.10.[9]

Theorem 7. *For $m = 2$, and any (distinct) set of design points $x_{1:n}$, the tridiagonal matrix $\mathbb{K}_n^2$ defined in (21) and the diagonal matrix $\mathbb{W}_n^2 = \mathrm{diag}((x_3 - x_1)/2, \ldots, (x_n - x_{n-2})/2)$ are $(3, 1)$-spectrally-similar. Thus Lemma 17 gives the following conclusion: if $\hat{f}$ is the solution in the cubic smoothing spline problem (17) with tuning parameter λ_a, and $\hat{\theta}$ is the solution in the weighted cubic BW filtering problem (135) with tuning parameter $\lambda_b \geq 3\lambda_a$, then*

$$\big\|\hat{f}(x_{1:n}) - \hat{\theta}\big\|_2^2 \leq \frac{\lambda_b}{3} \int_a^b (D^2 \hat{f})(x)^2 \, dx. \tag{143}$$

[9] We thank Yining Wang for his help with the spectral similarity result.

Remark 24. To achieve the bound in (143) in Theorem 7, we take the weighted BW filter tuning parameter λ_b to be at least three times the smoothing spline tuning parameter λ_a. This is the result of applying (142) in Lemma 17. Of course, empirically, and conceptually, we are more likely to believe that taking $\lambda_a = \lambda_b$ will lead to the most similar solutions; with this choice, the result in (141) translates to (in the context of the smoothing spline and weighted BW filtering):

$$\big\|\hat{f}(x_{1:n}) - \hat{\theta}\big\|_2^2 \le \lambda_a \big\|(\mathbb{W}_n^2)^{\frac{1}{2}} \mathbb{D}_n^2 \hat{\theta}\big\|_2^2, \tag{144}$$

which might also be a useful bound. However, the reason we chose to state (143) in the theorem, rather than (144), is that the former has the L_2-Sobolev penalty of $\hat{f}$ on the right-hand side, which can be controlled by leveraging classical nonparametric regression theory, as we show next.

Our next result uses known bounds on the estimation error of the cubic smoothing spline over L_2-Sobolev classes, along with (144) and the triangle inequality, to establish a similar result for the weighted cubic BW filter.

Corollary 4. *Assume that the design points x_i, $i = 1, \dots, n$ are drawn from a continuous distribution on $[0, 1]$, and that the responses follow the model*

$$y_i = f_0(x_i) + \epsilon_i, \quad i = 1, \dots, n,$$

for uniformly sub-Gaussian errors ϵ_i, $i = 1, \dots, n$ with mean zero and unit variance, independent of the design points. Further assume that f_0 has two weak derivatives, and that $\int_0^1 (D^2 f_0)(x)^2 \, dx \le C_n^2$ for $C_n \ge 1$. Recall that there are universal constants $c_1, c_2, c_3, n_0 > 0$ such that for all $c \ge c_1$ and $n \ge n_0$, the cubic smoothing spline solution in (17) (that is, $m = 2$) with $\lambda \ge c n^{\frac{1}{5}} C_n^{-\frac{8}{5}}$ satisfies

$$\frac{1}{n}\big\|\hat{f}(x_{1:n}) - f_0(x_{1:n})\big\|_2^2 \le \frac{8\lambda}{n} C_n^2, \tag{145}$$

$$\int_0^1 (D^2 \hat{f})(x)^2 \, dx \le 5 C_n^2, \tag{146}$$

with probability at least $1 - \exp(-c_2 c) - \exp(-c_3\sqrt{n})$. Setting $\lambda = c n^{\frac{1}{5}} C_n^{-\frac{8}{5}}$, the right-hand side in (145) becomes $8 c n^{-\frac{4}{5}} C_n^{\frac{2}{5}}$ which matches the minimax optimal error rate (in squared L_2 norm) for estimation over the space

$$\mathcal{W}^{2,2}(C_n; [0,1]) = \bigg\{ f : [0,1] \to \mathbb{R} : \int_0^1 (D^2 f_0)(x)^2 \, dx \le C_n^2 \bigg\}.$$

A consequence of the above result and Theorem 7 is as follows: for all $c \ge c_1$ and $n \ge n_0$, the weighted cubic BW filtering solution in (135) (that is, $m = 2$) with $\lambda = 3 c n^{\frac{1}{5}} C_n^{-\frac{8}{5}}$ satisfies

$$\frac{1}{n}\big\|\theta(x_{1:n}) - f_0(x_{1:n})\big\|_2^2 \le 26 c n^{-\frac{4}{5}} C_n^{\frac{2}{5}}, \tag{147}$$

with probability at least $1 - \exp(-c_2 c) - \exp(-c_3\sqrt{n})$, again matching the minimax error rate over $\mathcal{W}^{2,2}(C_n; [0,1])$.

We omit the proof of Corollary 4, as (147) follows immediately from (143), (145), (146), and the simple inequality $\|u + v\|_2^2 \le 2\|u\|_2^2 + 2\|v\|_2^2$. To be clear, the smoothing spline error bound (145), penalty bound (146), and claims of minimax optimality are well-known (and are not intended to be portrayed as original contributions in the corollary); for example, see Chapter 10.1 of van de Geer (2000) for a statement of (145), (146) in O_P (bounded in probability) form; the results in Corollary 4, written in finite-sample form, are a consequence of Theorem 1 in Sadhanala and Tibshirani (2019). For the minimax lower bound over the L_2-Sobolev class $\mathcal{W}^{2,2}(C_n; [0,1])$, see, for example, Chapter 2.6.1 of Tsybakov (2009). It is not really suprising that the weighted BW filter achieves minimax optimal error rates over the appropriate Sobolev classes, however it is of course reassuring to know that this is the case. As far as we can tell, this seems to be a new result, despite the fact that the BW filter has a very long history.

12.3 Connections to discrete splines

Unlike trend filtering, which bears a very clear connection to discrete splines, the connections between the (weighted) BW filter and discrete splines appear to be more subtle. Recall that the ℓ_1 case, for a kth degree discrete spline f, the total variation penalty $\mathrm{TV}(D^k f)$ is simply the trend filtering penalty (122) acting on $\theta = f(x_{1:n})$ (Theorem 4). In the ℓ_2 case, for a kth degree discrete spline f, with $k = 2m-1$, the L_2-Sobolev penalty $\int_a^b (D^m f)(x)^2 \, dx$ is a quadratic form of the mth discrete derivatives of $\theta = f(x_{1:n})$ (Theorem 5), but this quadratic form is *not* the BW penalty, either unweighted $\|\mathbb{D}_n^m \theta\|_2^2$, or weighted (136). It is instead $\|(\mathbb{V}_n^m)^{\frac{1}{2}} \mathbb{D}_n^m \theta\|_2^2$, where $\mathbb{V}_n^m \in \mathbb{R}^{(n-m)\times(n-m)}$ is a banded matrix (a function of $x_{1:n}$ only), of bandwidth $2m-1$.

For completeness, recall that for that a kth degree spline, the penalty $\int_a^b (D^m f)(x)^2 \, dx$ is also a quadratic form of the mth discrete derivatives of $\theta = f(x_{1:n})$ (Theorem 1), of the form $\|(\mathbb{K}_n^m)^{\frac{1}{2}} \mathbb{D}_n^m \theta\|_2^2$, where $\mathbb{K}_n^m \in \mathbb{R}^{(n-m)\times(n-m)}$ is a matrix (a function of $x_{1:n}$ only) with a banded *inverse*, and is therefore itself dense.

One way to roughly interpret and compare these penalties on discrete derivatives is as follows. Both can be seen as

$$\big\|\mathbb{A}^{\frac{1}{2}} \mathbb{D}_n^m \theta\big\|_2^2 = \sum_{i,\tau=-\infty}^{\infty} \mathbb{A}_{i,i-\tau} (\mathbb{D}_n^m \theta)_i (\mathbb{D}_n^m \theta)_{i-\tau}. \tag{148}$$

for a symmetric matrix $\mathbb{A} \in \mathbb{R}^{(n-m)\times(n-m)}$, where for notational convenience we simply set the entries of $\mathbb{A}$ or $\mathbb{D}_n^m \theta$ to zero when we index beyond their inherent ranges. That is, when the continuous-time penalty $\int_a^b (D^m f)(x)^2 \, dx$ gets translated into discrete-time, we see that the discrete-time equivalent (148) "blurs" the derivatives before it aggregates them; more precisely, the discrete-time equivalent (148) measures the weighted ℓ_2 norm of the product of $\mathbb{D}_n^m \theta$ and its convolution, weighted here by a (two-dimensional) kernel $\mathbb{A}$. The weighted BW penalty $\|(\mathbb{W}_n^m)^{\frac{1}{2}} \mathbb{D}_n^m \theta\|_2^2$ performs no such "blurring" (it measures the weighted ℓ_2 norm of $\mathbb{D}_n^m \theta$ times itself). Therefore we might view the discrete spline discretization of the Sobolev penalty, $\|(\mathbb{V}_n^m)^{\frac{1}{2}} \mathbb{D}_n^m \theta\|_2^2$, as being "closer" to the weighted BW penalty, as its kernel $\mathbb{V}_n^m$ performs less "blurring" (it has bandwidth $2m-1$), versus the spline discretization, $\|(\mathbb{K}_n^m)^{\frac{1}{2}} \mathbb{D}_n^m \theta\|_2^2$, whose kernel $\mathbb{K}_n^m$ performs more "blurring" (it is supported everywhere).

An important exception is the linear case, $m = 1$, in which all three penalties—from the weighted BW filter, spline discretization, and discrete spline discretization—coincide. The equivalence of the first two was already noted in (137). The next lemma gives the equivalence of the third, by calculating the explicit form of $\mathbb{V}_n^m$ for $m = 1$.[10] Its proof is elementary and is deferred until Appendix B.11.

Lemma 18. *For $m = 1$, the matrix $\mathbb{V}_n \in \mathbb{R}^{(n-1)\times(n-1)}$ from Theorem 5 has entries*

$$(\mathbb{V}_n)_{ii} = \begin{cases} x_2 - a & \text{if } i = 1 \\ x_{i+1} - x_i & \text{if } i \geq 2. \end{cases} \tag{149}$$

Thus when $a = x_1$, we see that the matrix $\mathbb{V}_n$ in (149) is the same as the matrix $\mathbb{K}_n$ in (20), which is the same as $\mathbb{W}_n$ in (69) with $m = 1$.

The next case to consider would of course be the cubic case, $m = 2$. As it turns out, deriving the explicit form of $\mathbb{V}_n^m$ for $m = 2$ requires a formidable calculation. The recursion in Lemma 12—though conceptually straightforward—is practically challenging to carry out, since it involves some rather complicated algebraic calculations. However, it can be done for evenly-spaced design points $x_{i+1} - x_i = v > 0$, $i = 1, \dots, n-1$, with $a = x_1$ and $b = x_n$: [11]

$$\mathbb{V}_n^2 = \begin{bmatrix} 3 & -3/2 & 0 & 0 & \dots & 0 & 0 & 0 \\ -3/2 & 10/3 & -5/6 & 0 & \dots & 0 & 0 & 0 \\ 0 & -5/6 & 8/3 & -5/6 & \dots & 0 & 0 & 0 \\ \vdots & & & & & & & \\ 0 & 0 & 0 & 0 & \dots & -5/6 & 8/3 & -5/6 \\ 0 & 0 & 0 & 0 & \dots & 0 & -5/6 & 7/3 \end{bmatrix} \cdot v. \tag{150}$$

For comparison, in this case, we have from (21):

$$\mathbb{K}_n^2 = \begin{bmatrix} 2/3 & 1/6 & 0 & 0 & \dots & 0 & 0 & 0 \\ 1/6 & 2/3 & 1/6 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1/6 & 2/3 & 1/6 & \dots & 0 & 0 & 0 \\ \vdots & & & & & & & \\ 0 & 0 & 0 & 0 & \dots & 1/6 & 2/3 & 1/6 \\ 0 & 0 & 0 & 0 & \dots & 0 & 1/6 & 2/3 \end{bmatrix}^{-1} \cdot v. \tag{151}$$

[10] This should not be a surprise: for degree $k = 1$, discrete splines are splines, and Lemma 18 is really just a sanity check.

[11] We thank Pratik Patil for his help in checking the result (150).

For the case of arbitrary design points, we can carry out the recursion defining $\mathbb{V}_n^m$ in Lemma 12 with symbolic computation software. Our current attempts have resulted in somewhat compact expressions for the elements of $\mathbb{V}_n^m$, but they do not appear simple enough to be useful (amenable to further interpretation or analysis). We may report on this in more detail at a future time.

13 Discussion

This paper began as an attempt to better understand the connections between trend filtering and discrete splines, and it grew into something broader: an attempt to better understand some fundamental properties of discrete splines, and offer some new perspectives on them. Though discrete splines were first studied 50 years ago, there still seems to be some fruitful directions left to explore. For example, the approximation results in Section 10 are weak (though recall, they are sufficient for the intended statistical applications) and could most certainly be improved. The use of discrete B-splines within trend filtering optimization algorithms, described in Section 11.1, should be investigated thoroughly, as it should improve their stability. As for more open directions, it may be possible to use discrete splines to approximately (and efficiently) solve certain differential equations. Lastly, the multivariate case is of great interest and importance.

Software

An R package (with C++ backend) that implements some of the core numerical methods for discrete splines as described in this paper is available at `https://github.com/ryantibs/dspline`.

Acknowledgements

We are grateful to Yu-Xiang Wang for his many, many insights and inspiring conversations over the years. His lasting enthusiasm helped fuel our own interest in “getting to the bottom” of the falling factorial basis, and putting this paper together. We also thank our other collaborators on trend filtering papers: Aaditya Ramdas, Veeranjaneyulu Sadhanala, James Sharpnack, and Alex Smola. Finally, we are grateful to Addison Hu, Alden Green, Pratik Patil, Veeranjaneyulu Sadhanala, and Yu-Xiang Wang for their helpful comments and feedback on this paper as a whole; to Yining Wang for his help with Section 12.2; and to Pratik Patil for his help with Section 12.3. This paper is based upon work supported by the National Science Foundation under Grant No. DMS-1554123.

A Notation table

Discrete object	Reference	Continuum object	Reference	Notes
Operators				
$\Delta^k_n = \Delta^k(\cdot; x_{1:n})$, kth order discrete differentiation with respect to design point $x_{1:n}$	(39)	D^k, kth order differentiation	–	$(\Delta^k_n f)(x) = (D^k f)(x)$, for $f \in \mathcal{H}^k_n$ and $x > x_k$ (Corollary 1)
$S^k_n = S^k(\cdot; x_{1:n})$, kth order discrete integration	(46), (47)	I^k, kth order integration	–	$S^k_n = (D^k_n)^{-1}$ (Lemma 1)
Spaces				
$\mathcal{DS}^k_n(t_{1:r})$, kth degree discrete splines with knots $t_{1:r}$ and design points $x_{1:n}$	Definition 3	$\mathcal{S}^k(t_{1:r})$, kth degree splines with knots $t_{1:r}$	Definition 1	These spaces coincide for $k=0$ and $k=1$
$\mathcal{H}^k_n = \mathcal{DS}^k_n(x_{(k+1):(n-1)})$	–	$\mathcal{G}^k_n = \mathcal{S}^k(x_{(k+1):(n-1)})$	–	Abbreviations for the "canonical" spaces, with knots $x_{(k+1):(n-1)}$
Bases				
h^k_j, $j = 1,\dots,n$, kth degree falling factorial basis for $\mathcal{H}^k_n$	(5)	g^k_j, $j = 1,\dots,n$, kth degree truncated power basis for $\mathcal{G}^k_n$	(14)	Falling factorials can be seen as truncated Newton polynomials, and have dual relationship to discrete differentiation (Lemma 6)
Q^k_j and N^k_j, $j = 1,\dots,n$, unnormalized and normalized kth degree DB-spline basis for $\mathcal{H}^k_n$	(81), (82), (84)	P^k_j and M^k_j, $j = 1,\dots,n$, unnormalized and normalized kth degree B-spline basis for $\mathcal{G}^k_n$	(183), (184), (186)	The basis in (186) is actually defined for an arbitrary knot set $t_{1:r}$; for arbitrary knots in the DB-spline setting, see (93), (94)
Matrices				
$\mathbb{D}^k_n$, kth order discrete derivative matrix with respect to design points $x_{1:n}$	(68), (69), (70)	–	–	Multiplying by a vector of evaluations gives discrete derivatives at design points $x_{(k+1):n}$, as in (71)
$\mathbb{B}^k_n$, kth order extended discrete derivative matrix with respect to design points $x_{1:n}$	(73), (74), (75)	–	–	Multiplying by a vector of evaluations gives discrete derivatives at all design points $x_{1:n}$, as in (76)
$\mathbb{H}^k_n$, kth degree falling factorial basis matrix with respect to design points $x_{1:n}$	Basis in kth degree trend filter (30)	$\mathbb{G}^k_n$, kth degree truncated power basis matrix with respect to design points $x_{1:n}$	Basis in kth degree restricted locally adaptive regression spline (29)	$\mathbb{H}^k_n = (\mathbb{Z}^{k+1}_n \mathbb{B}^{k+1}_n)^{-1}$, see (79); results in fast algorithms for matrix computations in $\mathbb{H}^k_n$, see Appendix D
Smoothness functionals				
$\sum_{i=1}^{n-k-1} \lvert(\mathbb{D}^k_n\theta)_{i+1} - (\mathbb{D}^k_n\theta)_i\rvert = \lVert\mathbb{W}^{k+1}_n \mathbb{D}^{k+1}_n \theta\rVert_1$, kth order discrete total variation of vector θ	Penalty in kth degree trend filter (31)	$\mathrm{TV}(D^k f)$, kth order total variation of function f	Penalty in kth degree locally adaptive regression spline (8)	Equal for $\theta = f(x_{1:n})$ and $f \in \mathcal{H}^k_n$ (Theorem 4)
$\sum_{i=1}^{n-m} (\mathbb{D}^m_n\theta)^2_i (x_{i+m} - x_i)/m = \lVert(\mathbb{W}^m_n)^{\frac{1}{2}} \mathbb{D}^m_n \theta\rVert_2^2$, mth order discrete Sobolev seminorm of vector θ	Penalty in kth degree BW filter (135), for $k = 2m-1$	$\int_a^b (D^m f)(x)^2 \, dx$, mth order Sobolev seminorm of f	Penalty in kth degree smoothing spline (17), for $k = 2m-1$	Equal for $\theta = f(x_{1:n})$ and $m = 1$ (Lemma 18), but not in general; see also Theorem 5

Table 1: Main notation, and discrete-continuum analogies/equivalences in this paper. We omit notational dependence on the domain $[a, b]$ for simplicity.

B Proofs

B.1 Proof of Theorem 1

Since f is a natural spline of degree $2m-1$ with knots in $x_{1:n}$, we know that $D^m f$ is a spline of degree $m-1$ with knots in $x_{1:n}$, and moreover, it is supported on $[x_1, x_n]$. Thus we can expand $D^m f = \sum_{i=1}^{n-m} \alpha_i P_i^{m-1}$ for coefficients α_i, $i = 1, \dots, n-m$, and

$$\int_a^b (D^m f)(x)^2 \, dx = \alpha^\mathsf{T} \mathbb{Q} \alpha, \tag{152}$$

where $\mathbb{Q} \in \mathbb{R}^{(n-m)\times(n-m)}$ has entries $\mathbb{Q}_{ij} = \int_a^b P_i^{m-1}(x) P_j^{m-1}(x) \, dx$. But we can also write

$$\begin{aligned} \int_a^b (D^m f)(x)^2 \, dx &= \int_a^b (D^m f)(x) \sum_{i=1}^{n-m} \alpha_i P_i^{m-1}(x) \, dx \\ &= \sum_{i=1}^{n-m} \alpha_i \int_a^b (D^m f)(x) P_i^{m-1}(x) \, dx \\ &= \frac{1}{m} \sum_{i=1}^{n-m} \alpha_i \big(\mathbb{D}_n^m f(x_{1:n})\big)_i \\ &= \frac{1}{m} \alpha^\mathsf{T} \mathbb{D}_n^m f(x_{1:n}), \end{aligned} \tag{153}$$

where in the third line, we used the Peano representation for B-splines, as described in (181) in Appendix C.1, which implies that for $i = 1, \dots, n-m$,

$$(m-1)! \cdot f[x_i, \dots, x_{i+m}] = \int_a^b (D^m f)(x) P_i^{m-1}(x) \, dx.$$

Comparing (152) and (153), we learn that $\mathbb{Q}\alpha = \mathbb{D}_n^m f(x_{1:n})/m$, that is, $\alpha = \mathbb{Q}^{-1}\mathbb{D}_n^m f(x_{1:n})/m$, and therefore

$$\begin{aligned} \int_a^b (D^m f)(x)^2 \, dx &= \frac{1}{m^2} \big(\mathbb{Q}^{-1}\mathbb{D}_n^m f(x_{1:n})\big)^\mathsf{T} \mathbb{Q} \, \mathbb{Q}^{-1} \mathbb{D}_n^m f(x_{1:n}) \\ &= \frac{1}{m^2} \big(\mathbb{D}_n^m f(x_{1:n})\big)^\mathsf{T} \mathbb{Q}^{-1} \mathbb{D}_n^m f(x_{1:n}), \end{aligned}$$

which establishes (18), (19) with $\mathbb{K}_n^m = (1/m^2)\mathbb{Q}^{-1}$, that is, $(\mathbb{K}_n^m)^{-1} = m^2 \mathbb{Q}$.

When $m = 1$, for each $i = 1, \dots, n-1$, we have the simple form for the constant B-spline:

$$P_i^0(x) = \begin{cases} \dfrac{1}{x_{i+1} - x_i} & \text{if } x \in I_i \\ 0 & \text{otherwise.} \end{cases}$$

where $I_1 = [x_1, x_2]$, and $I_i = (x_i, x_{i+1}]$ for $i = 2, \dots, n-1$. The result (20) comes from straightforward calculation of $\int_a^b P_i^0(x)^2 \, dx$. Lastly, when $m = 2$, for each $i = 1, \dots, n-2$, we have the linear B-spline:

$$P_i^1(x) = \begin{cases} \dfrac{x - x_i}{(x_{i+2} - x_i)(x_{i+1} - x_i)} & \text{if } x \in I_i^- \\ \dfrac{x_{i+2} - x}{(x_{i+2} - x_i)(x_{i+2} - x_{i+1})} & \text{if } x \in I_i^+ \\ 0 & \text{otherwise,} \end{cases}$$

where $I_1^- = [x_1, x_2]$, $I_i^- = (x_i, x_{i+1}]$ for $i = 2, \dots, n-2$, and $I_i^+ = (x_{i+1}, x_{i+2}]$ for $i = 1, \dots, n-2$. The two cases in (21) again come from straightforward calculation of the integrals $\int_a^b P_i^1(x)^2 \, dx$ and $\int_a^b P_i^1(x) P_{i-1}^1(x) \, dx$, which completes the proof.

B.2 Proof of the linear combination formulation (48)

Denote by $g(x)$ the right-hand side of (48). We will show that $\Delta_n^k g = f$. Note by Lemma 1, this would imply $g = S_n^k f$, proving (48). An inductive argument similar to that in the proof of Lemma 2 shows that, for $x \in (x_i, x_{i+1}]$ and $i \geq k$,

$$(\Delta_n^k g)(x) = \sum_{j=1}^{k} (\Delta_n^k h_j^{k-1})(x) \cdot f(x_j) \; + \sum_{j=k+1}^{i} (\Delta_n^k h_j^{k-1})(x) \cdot \frac{x_j - x_{j-k}}{k} \cdot f(x_j) \\ + \; (\Delta_n^k h_{i+1}^{k-1})(x) \cdot \frac{x - x_{i-k+1}}{k} \cdot f(x).$$

By Lemmas 4 and 5, all discrete derivatives here are zero except the last, which is $(\Delta_n^k h_{i+1}^{k-1})(x) = k/(x - x_{i-k+1})$, thus we have shown $(\Delta_n^k g)(x) = f(x)$. Similarly, for $x \in (x_i, x_{i+1}]$ and $i < k$,

$$(\Delta_n^k g)(x) = \sum_{j=1}^{i} (\Delta_n^k h_j^{k-1})(x) \cdot f(x_j) \; + \; (\Delta_n^k h_{i+1}^{k-1})(x) \cdot f(x),$$

and by Lemma 5, all discrete derivatives here are zero except the last, which is $(\Delta_n^k h_{i+1}^{k-1})(x) = 1$. For $x \leq x_1$, we have $g(x) = f(x)$ by definition. This establishes the desired claim and completes the proof.

B.3 Proof of Lemma 1

We use induction, beginning with $k = 1$. Using (46), (44), we can express the first order discrete integral operator S_n more explicitly as

$$(S_n f)(x) = \begin{cases} f(x_1) + \sum_{j=2}^{i} f(x_j)(x_j - x_{j-1}) + f(x)(x - x_i) & \text{if } x \in (x_i, x_{i+1}] \\ f(x) & \text{if } x \leq x_1. \end{cases} \tag{154}$$

Compare (41) and (154). For $x \leq x_1$, clearly $(\Delta_n S_n f)(x) = f(x)$ and $(S_n \Delta_n f)(x) = f(x)$, and for $x \in (x_i, x_{i+1}]$,

$$\begin{aligned} (\Delta_n S_n f)(x) &= \frac{(S_n f)(x) - (S_n f)(x_i)}{x - x_i} \\ &= \frac{f(x_1) + \sum_{j=2}^{i} f(x_j)(x_j - x_{j-1}) + f(x)(x - x_i) - \big(f(x_1) + \sum_{j=2}^{i} f(x_j)(x_j - x_{j-1})\big)}{x - x_i} \\ &= f(x), \end{aligned}$$

and also

$$\begin{aligned} (S_n \Delta_n f)(x) &= f(x_1) + \sum_{j=2}^{i} (\Delta_n f)(x_j) \cdot (x_j - x_{j-1}) + (\Delta_n f)(x) \cdot (x - x_i) \\ &= f(x_1) + \sum_{j=2}^{i} \big(f(x_j) - f(x_{j-1})\big) + f(x) - f(x_i) \\ &= f(x). \end{aligned}$$

Now assume the result is true for the order $k - 1$ operators. Then, we have from (45), (47),

$$\Delta_n^k \circ S_n^k = (W_n^k)^{-1} \circ \overline{\Delta}_{n-k+1} \circ \Delta_n^{k-1} \circ S_n^{k-1} \circ \overline{S}_{n-k+1} \circ W_n^k = \mathrm{Id},$$

and also

$$S_n^k \circ \Delta_n^k = S_n^{k-1} \circ \overline{S}_{n-k+1} \circ W_n^k \circ (W_n^k)^{-1} \circ \overline{\Delta}_{n-k+1} \circ \Delta_n^{k-1} = \mathrm{Id},$$

where Id denotes the identity operator. This completes the proof.

B.4 Proof of Lemma 2

The case $d = 0$**.** Beginning with the case $d = 0$, the desired result in (51) reads

$$\frac{1}{k!}\prod_{m=j-k}^{j-1}(x-x_m) = \sum_{\ell=j}^{i}\frac{1}{(k-1)!}\prod_{m=\ell-k+1}^{\ell-1}(x-x_m)\frac{x_\ell - x_{\ell-k}}{k} + \frac{1}{(k-1)!}\prod_{m=i-k+2}^{i}(x-x_m)\frac{x-x_{i-k+1}}{k},$$

or more succintly,

$$\eta(x; x_{(j-k):(j-1)}) = \sum_{\ell=j}^{i}\eta(x; x_{(\ell-k+1):(\ell-1)})(x_\ell - x_{\ell-k}) + \eta(x; x_{(i-k+2):i}),$$

The above display is a consequence of an elementary result (155) on Newton polynomials. We state and prove this result next, which we note completes the proof for the case $d = 0$.

Lemma 19. *For any* $k \geq 1$*, and points* $t_{1:r}$ *with* $r \geq k$*, the Newton polynomials defined in* (10) *satisfy, at any* x*,*

$$\eta(x; t_{1:k}) - \eta(x; t_{(r-k+1):r}) = \sum_{\ell=k+1}^{r}\eta(x; t_{(\ell-k+1):(\ell-1)})(t_\ell - t_{\ell-k}). \tag{155}$$

Proof. Observe that

$$\eta(x; t_{1:k}) - \eta(x; t_{2:(k+1)}) = \eta(x; t_{2:k})\big((x-t_1) - (x-t_{k+1})\big) = \eta(x; t_{2:k})(t_{k+1} - t_1). \tag{156}$$

Therefore

$$\eta(x; t_{1:k}) - \eta(x; t_{(r-k+1):r}) = \underbrace{\eta(x; t_{1:k}) - \eta(x; t_{2:(k+1)})}_{a_1} + \underbrace{\eta(x; t_{2:(k+1)}) - \eta(x; t_{3:(k+2)})}_{a_2} + \cdots$$
$$+ \underbrace{\eta(x; t_{(r-k):r-1}) - \eta(x; t_{(r-k+1):r})}_{a_{r-k}}.$$

In a similar manner to (156), for each $i = 1, \ldots, k$, we have $a_i = \eta(x; t_{(i+1):(i+k-1)})(t_{i+k} - t_i)$, and the result follows, after making the substitution $\ell = i + k$. □

The case $d \geq 1$**.** We now prove the result (51) for $d \geq 1$ by induction. The base case was shown above, for $d = 0$. Assume the result holds for discrete derivatives of order $d - 1$. If $x \leq x_d$ (or $d > n$), then $(\Delta_n^d f)(x) = (\Delta_n^{d-1} f)(x)$ for all functions f and thus the desired result holds trivially. Hence assume $x > x_d$ (which implies that $i \geq d$). By the inductive hypothesis,

$$\begin{aligned}
&(\Delta_n^{d-1} h_j^k)(x) - (\Delta_n^{d-1} h_j^k)(x_i) \\
&= \sum_{\ell=j}^{i}(\Delta_n^{d-1} h_\ell^{k-1})(x)\cdot\frac{x_\ell - x_{\ell-k}}{k} + (\Delta_n^{d-1} h_{i+1}^{k-1})(x)\cdot\frac{x - x_{i-k+1}}{k} - \sum_{\ell=j}^{i}(\Delta_n^{d-1} h_\ell^{k-1})(x_i)\cdot\frac{x_\ell - x_{\ell-k}}{k} \\
&= \sum_{\ell=j}^{i}\big((\Delta_n^{d-1} h_\ell^{k-1})(x) - (\Delta_n^{d-1} h_\ell^{k-1})(x_i)\big)\cdot\frac{x_\ell - x_{\ell-k}}{k} + \big((\Delta_n^{d-1} h_{i+1}^{k-1})(x) - (\Delta_n^{d-1} h_{i+1}^{k-1})(x_i)\big)\cdot\frac{x - x_{i-k+1}}{k},
\end{aligned}$$

where in the last line we used the fact that $h_{i+1}^{k-1} = 0$ on $[a, x_i]$, and thus $(\Delta_n^{d-1} h_{i+1}^{k-1})(x_i) = 0$. This means, using (42),

$$\begin{aligned}
(\Delta_n^d h_j^k)(x) &= \frac{(\Delta_n^{d-1} h_j^k)(x) - (\Delta_n^{d-1} h_j^k)(x_i)}{(x - x_{i-d+1})/d} \\
&= \sum_{\ell=j}^{i}\frac{(\Delta_n^{d-1} h_\ell^{k-1})(x) - (\Delta_n^{d-1} h_\ell^{k-1})(x_i)}{(x - x_{i-d+1})/d}\cdot\frac{x_\ell - x_{\ell-k}}{k} + \frac{(\Delta_n^{d-1} h_{i+1}^{k-1})(x) - (\Delta_n^{d-1} h_{i+1}^{k-1})(x_i)}{(x - x_{i-d+1})/d}\cdot\frac{x - x_{i-k+1}}{k} \\
&= \sum_{\ell=j}^{i}(\Delta_n^d h_\ell^{k-1})(x)\cdot\frac{x_\ell - x_{\ell-k}}{k} + (\Delta_n^d h_{i+1}^{k-1})(x)\cdot\frac{x - x_{i-k+1}}{k},
\end{aligned}$$

as desired. This completes the proof.

B.5 Lemma 20 (helper result for the proof of Corollary 3)

Lemma 20. *Given distinct points $t_i \in [a,b]$, $i=1,\dots,r$ and evaluations $f(t_i)$, $i=1,\dots,r$, if f satisfies*

$$f[t_1,\dots,t_r,x]=0, \quad \textit{for } x \in [a,b],$$

then f is a polynomial of degree r.

Proof. We will actually prove a more general result, namely, that if f satisfies

$$f[t_1,\dots,t_r,x]=p_\ell(x), \quad \text{for } x\in[a,b], \tag{157}$$

where p_ℓ is a polynomial of degree ℓ, then f is a polynomial of degree $r+\ell$. We use induction on r. For $r=0$, the statement (157) clearly holds for all ℓ, because $f[x]=f(x)$ (a zeroth order divided difference is simply evaluation). Now assume (157) holds for any $r-1$ centers and all degrees ℓ. Then

$$p_\ell(x)=f[t_1,\dots,t_r,x]=\frac{f[t_2,\dots,t_r,x]-f[t_1,\dots,t_r]}{x-t_1},$$

which means $f[t_2,\dots,t_r,x]=(x-t_1)p_\ell(x)+f[t_1,\dots,t_r]$. As the right-hand side is a polynomial of degree $\ell+1$, the inductive hypothesis implies that f is a polynomial of degree $r-1+\ell+1=r+\ell$, completing the proof. □

B.6 Proof of Theorem 5

Let h_j^k, $j=1,\dots,n$ denote the falling factorial basis, as in (5). Consider expanding f in this basis, $f=\sum_{j=1}^n \alpha_j h_j^k$. Define $\mathbb{Q}\in\mathbb{R}^{n\times n}$ to have entries

$$\mathbb{Q}_{ij}=\int_a^b (D^m h_i^k)(x)(D^m h_j^k)(x)\,dx. \tag{158}$$

Observe

$$\begin{aligned}\int_a^b (D^m f)(x)^2\,dx &= \int_a^b \sum_{i,j=1}^n \alpha_i\alpha_j (D^m h_i^k)(x)(D^m h_j^k)(x)\,dx \\ &= \alpha^\mathsf{T}\mathbb{Q}\alpha \\ &= f(x_{1:n})^\mathsf{T}(\mathbb{H}_n^k)^{-\mathsf{T}}\mathbb{Q}(\mathbb{H}_n^k)^{-1}f(x_{1:n}) \\ &= f(x_{1:n})^\mathsf{T}(\mathbb{B}_n^{k+1})^\mathsf{T}\mathbb{Z}_n^{k+1}\,\mathbb{Q}\,\mathbb{Z}_n^{k+1}\,\mathbb{B}_n^{k+1}f(x_{1:n}).\end{aligned} \tag{159}$$

In the third line above we used the expansion $f(x_{1:n})=\mathbb{H}_n^k\alpha$, where $\mathbb{H}_n^k$ is the kth degree falling factorial basis with entries $(\mathbb{H}_n^k)_{ij}=h_j^k(x_i)$, and in the fourth line we applied the inverse relationship in (79), where $\mathbb{B}_n^{k+1}$ is the $(k+1)$st order extended discrete derivative matrix in (75) and $\mathbb{Z}_n^{k+1}$ is the extended weight matrix in (74). Now note that we can unravel the recursion in (75) to yield

$$\mathbb{B}_n^{k+1}=(\mathbb{Z}_n^{k+1})^{-1}\underbrace{\overline{\mathbb{B}}_{n,k+1}(\mathbb{Z}_n^k)^{-1}\overline{\mathbb{B}}_{n,k}\cdots(\mathbb{Z}_n^{m+1})^{-1}\overline{\mathbb{B}}_{n,m+1}}_{\mathbb{F}}\mathbb{B}_n^m, \tag{160}$$

and returning to (159), we get

$$\int_a^b (D^m f)(x)^2\,dx = f(x_{1:n})^\mathsf{T}(\mathbb{B}_n^m)^\mathsf{T}\mathbb{F}^\mathsf{T}\mathbb{Q}\,\mathbb{F}\,\mathbb{B}_n^m f(x_{1:n}). \tag{161}$$

We break up the remainder of the proof up into parts for readability.

Reducing (161) to involve only discrete derivatives. First we show that the right-hand side in (161) really depends on the discrete derivatives $\mathbb{D}_n^m f(x_{1:n})$ only (as opposed to extended discrete derivatives $\mathbb{B}_n^m f(x_{1:n})$). As the first m basis functions $h_1^k,\dots,h_m^k$ are polynomials of degree at most $m-1$, note that their mth derivatives are zero, and hence we can write

$$\mathbb{Q}=\begin{bmatrix}0 & 0\\ 0 & \mathbb{M}\end{bmatrix},$$

where $\mathbb{M} \in \mathbb{R}^{(n-m)\times(n-m)}$ has entries as in (107). Furthermore, note that $\mathbb{F}$ as defined in (160) can be written as

$$\mathbb{F} = \begin{bmatrix} \mathbb{I}_m & 0 \\ 0 & \mathbb{G} \end{bmatrix},$$

for a matrix $\mathbb{G} \in \mathbb{R}^{(n-m)\times(n-m)}$. Therefore

$$\mathbb{F}^\mathsf{T} \mathbb{Q}\, \mathbb{F} = \begin{bmatrix} 0 & 0 \\ 0 & \mathbb{G}^\mathsf{T} \mathbb{M}\, \mathbb{G} \end{bmatrix}, \tag{162}$$

and hence (161) reduces to

$$\int_a^b (D^m f)(x)^2 \, dx = f(x_{1:n})^\mathsf{T} (\mathbb{D}_n^m)^\mathsf{T} \underbrace{\mathbb{G}^\mathsf{T} \mathbb{M}\, \mathbb{G}}_{\mathbb{V}_n^m} \mathbb{D}_n^m f(x_{1:n}), \tag{163}$$

recalling that $\mathbb{D}_n^m$ is exactly given by the last $n-m$ rows of $\mathbb{B}_n^m$.

Casting $\mathbb{F}^\mathsf{T} \mathbb{Q}\, \mathbb{F}$ in terms of scaled differences. Next we prove that $\mathbb{V}_n^m = \mathbb{G}^\mathsf{T} \mathbb{M}\, \mathbb{G}$, as defined in (163), is a banded matrix. To prevent unnecessary indexing difficulties, we will actually just work directly with $\mathbb{F}^\mathsf{T} \mathbb{Q}\, \mathbb{F}$, and then in the end, due to (162), we will be able to read off the desired result according to the lower-right submatrix of $\mathbb{F}^\mathsf{T} \mathbb{Q}\, \mathbb{F}$, of dimension $(n-m)\times(n-m)$. Observe that

$$\mathbb{F}^\mathsf{T} \mathbb{Q}\, \mathbb{F} = (\overline{\mathbb{B}}_{n,m+1})^\mathsf{T} (\mathbb{Z}_n^{m+1})^{-1} \cdots (\overline{\mathbb{B}}_{n,k})^\mathsf{T} (\mathbb{Z}_n^k)^{-1} (\overline{\mathbb{B}}_{n,k+1})^\mathsf{T} \mathbb{Q}\, \overline{\mathbb{B}}_{n,k+1} (\mathbb{Z}_n^k)^{-1} \overline{\mathbb{B}}_{n,k} \cdots (\mathbb{Z}_n^{m+1})^{-1} \overline{\mathbb{B}}_{n,m+1}. \tag{164}$$

To study this, it helps to recall the notation introduced in Lemma 12: for a matrix $\mathbb{A}$ and positive integers i, j, let

$$\mathbb{A}(i,j) = \begin{cases} \mathbb{A}_{ij} & \text{if } \mathbb{A} \text{ has at least } i \text{ rows and } j \text{ columns} \\ 0 & \text{otherwise,} \end{cases}$$

as well as

$$\delta^r_{ij}(\mathbb{A}) = \mathbb{A}(i,j) - \mathbb{A}(i+1,j),$$
$$\delta^c_{ij}(\mathbb{A}) = \mathbb{A}(i,j) - \mathbb{A}(i,j+1).$$

Now to compute (164), we first compute the product

$$\mathbb{F}^\mathsf{T} \mathbb{Q} = (\overline{\mathbb{B}}_{n,m+1})^\mathsf{T} (\mathbb{Z}_n^{m+1})^{-1} \cdots (\overline{\mathbb{B}}_{n,k})^\mathsf{T} (\mathbb{Z}_n^k)^{-1} (\overline{\mathbb{B}}_{n,k+1})^\mathsf{T} \mathbb{Q}.$$

We will work "from right to left". From (73), we have

$$(\overline{\mathbb{B}}_{n,k+1})^\mathsf{T} = \begin{array}{l} \left[\begin{array}{cccc|cccccc} 1 & 0 & \dots & 0 & & & & & & \\ 0 & 1 & \dots & 0 & & & & 0 & & \\ \vdots & & & & & & & & & \\ 0 & 0 & \dots & 1 & & & & & & \\ \hline & & & & 1 & -1 & 0 & \dots & 0 & 0 \\ & & & & 0 & 1 & -1 & \dots & 0 & 0 \\ & & 0 & & \vdots & & & & & \\ & & & & 0 & 0 & 0 & \dots & 1 & -1 \\ & & & & 0 & 0 & 0 & \dots & 0 & 1 \end{array}\right] \end{array} \begin{array}{l} \left.\vphantom{\begin{array}{c}1\\0\\\vdots\\0\end{array}}\right\} k \text{ rows} \\ \left.\vphantom{\begin{array}{c}1\\0\\\vdots\\0\\0\end{array}}\right\} n-k \text{ rows} \end{array}$$

This shows left multiplication by $(\overline{\mathbb{B}}_{n,k+1})^\mathsf{T}$ gives row-wise differences, $((\overline{\mathbb{B}}_{n,k+1})^\mathsf{T} \mathbb{A})_{ij} = \delta^r_{ij}(\mathbb{A})$, for $i > k$. Further, from (74), we can see that left multiplication by $(\mathbb{Z}_n^k)^{-1}$ applies a row-wise scaling, $(\mathbb{Z}_n^k)^{-1} \mathbb{A} = \mathbb{A}_{ij} \cdot k/(x_i - x_{i-k})$, for $i > k$. Thus letting $\mathbb{U}^{1,0} = (\mathbb{Z}_n^k)^{-1} (\overline{\mathbb{B}}_{n,k+1})^\mathsf{T} \mathbb{Q}$, its entries are:

$$\mathbb{U}^{1,0}_{ij} = \begin{cases} \mathbb{Q}_{ij} & \text{if } i \le k \\ \delta^r_{ij}(\mathbb{Q}) \cdot \dfrac{k}{x_i - x_{i-k}} & \text{if } i > k. \end{cases}$$

The next two products to consider are left multiplication by $(\overline{\mathbb{B}}_{n,k})^\mathsf{T}$ and by $(\mathbb{Z}_n^{k-1})^{-1}$, which act similarly (they again produce row-wise differencing and scaling, respectively). Continuing on in this same manner, we get that $\mathbb{F}^\mathsf{T}\mathbb{Q} = \mathbb{U}^{m,0}$, where $\mathbb{U}^{\ell,0}$, $\ell = 1,\dots,m-1$ satisfy the recursion relation (setting $\mathbb{U}^{0,0} = \mathbb{Q}$ for convenience):

$$\mathbb{U}^{\ell,0}_{ij} = \begin{cases} \mathbb{U}^{\ell-1,0}_{ij} & \text{if } i \le k+1-\ell \\ \delta^r_{ij}(\mathbb{U}^{\ell-1,0}) \cdot \dfrac{k+1-\ell}{x_i - x_{i-(k+1-\ell)}} & \text{if } i > k+1-\ell, \end{cases} \tag{165}$$

and where (using $k+1-m=m$):

$$\mathbb{U}^{m,0}_{ij} = \begin{cases} \mathbb{U}^{m-1,0}_{ij} & \text{if } i \le m \\ \delta^r_{ij}(\mathbb{U}^{m-1,0}) & \text{if } i > m. \end{cases} \tag{166}$$

The expressions (165), (166) are equivalent to (108), (109), the row-wise recursion in Lemma 12 (the main difference is that Lemma 12 is concerned with the lower-right $(n-m)\times(n-m)$ submatrices of these matrices, and so these recursive expressions are written with i, j replaced by $i+m, j+m$, respectively).

The other half of computing (164) is of course to compute the product

$$\mathbb{F}^\mathsf{T}\mathbb{Q}\,\mathbb{F} = \mathbb{F}^\mathsf{T}\mathbb{Q}\,\overline{\mathbb{B}}_{n,k+1}(\mathbb{Z}_n^k)^{-1}\overline{\mathbb{B}}_{n,k}\cdots(\mathbb{Z}_n^{m+1})^{-1}\overline{\mathbb{B}}_{n,m+1}.$$

Working now "from left to right", this calculation proceeds analogously to the case just covered, but with column-wise instead of row-wise updates, and we get $\mathbb{F}^\mathsf{T}\mathbb{Q}\,\mathbb{F} = \mathbb{U}^{m,m}$, where $\mathbb{U}^{m,\ell}$, $\ell = 1,\dots,m-1$ satisfy the recursion:

$$\mathbb{U}^{m,\ell}_{ij} = \begin{cases} \mathbb{U}^{m,\ell-1}_{ij} & \text{if } j \le k+1-\ell \\ \delta^c_{ij}(\mathbb{U}^{m,\ell-1}) \cdot \dfrac{k+1-\ell}{x_j - x_{j-(k+1-\ell)}} & \text{if } j > k+1-\ell, \end{cases} \tag{167}$$

and where:

$$\mathbb{U}^{m,m}_{ij} = \begin{cases} \mathbb{U}^{m,m-1}_{ij} & \text{if } j \le m \\ \delta^c_{ij}(\mathbb{U}^{m,m-1}) & \text{if } j > m. \end{cases} \tag{168}$$

Similarly, (167), (168) are equivalent to (110), (111), the column-wise recursion in in Lemma 12 (again, the difference is that Lemma 12 is written in terms of the lower-right $(n-m)\times(n-m)$ submatrices). This establishes the result in Lemma 12.

Exchanging the order of scaled differencing with integration and differentiation. Now that we have shown how to explicitly write the entries of $\mathbb{F}^\mathsf{T}\mathbb{Q}\,\mathbb{F}$ via recursion, it remains to prove bandedness. To this end, for each $x \in [a,b]$, define $\mathbb{Q}^x \in \mathbb{R}^{n\times n}$ to have entries $\mathbb{Q}^x_{ij} = (D^m h^k_i)(x)(D^m h^k_j)(x)$, and note that by linearity of integration,

$$\mathbb{F}^\mathsf{T}\mathbb{Q}\,\mathbb{F} = \int_a^b \mathbb{F}^\mathsf{T}\mathbb{Q}^x\,\mathbb{F}\,dx,$$

where the integral on the right-hand side above is meant to be interpreted elementwise. Furthermore, defining $a^x \in \mathbb{R}^n$ to have entries $a^x_i = (D^m h^k_i)(x)$, we have $\mathbb{Q}^x = a^x(a^x)^\mathsf{T}$, and defining $b^x \in \mathbb{R}^n$ to have entries $b^x_i = h^k_i(x)$, note that by linearity of differentiation,

$$\mathbb{F}^\mathsf{T}a^x = D^m\mathbb{F}^\mathsf{T}b^x,$$

where again the derivative on the right-hand side is meant to be interpreted elementwise. This means that

$$\mathbb{F}^\mathsf{T}\mathbb{Q}^x\,\mathbb{F} = (D^m\mathbb{F}^\mathsf{T}b^x)(D^m\mathbb{F}^\mathsf{T}b^x)^\mathsf{T}.$$

By the same logic as that given above (see the development of (165), (166)), we can view $\mathbb{F}^\mathsf{T}b^x$ as the endpoint of an m-step recursion. First initialize $u^{x,0} = b^x$, and define for $\ell = 1,\dots,m-1$,

$$u^{x,\ell}_i = \begin{cases} u^{x,\ell-1}_i & \text{if } i \le k+1-\ell \\ (u^{x,\ell-1}_i - u^{x,\ell-1}_{i+1}) \cdot \dfrac{k+1-\ell}{x_i - x_{i-(k+1-\ell)}} & \text{if } i > k+1-\ell, \end{cases} \tag{169}$$

as well as

$$u_i^{x,m} = \begin{cases} u_i^{x,m-1} & \text{if } i \le m \\ u_i^{x,m-1} - u_{i+1}^{x,m-1} & \text{if } i > m. \end{cases} \tag{170}$$

Here, we set $u_{n+1}^{x,\ell} = 0$, $\ell = 1, \dots, m$, for convenience. Then as before, this recursion terminates at $u^{x,m} = \mathbb{F}^\mathsf{T} b^x$.

In what follows, we will show that

$$(D^m u_i^{x,m})(D^m u_j^{x,m}) = 0, \quad \text{for } x \in [a,b] \text{ and } |i-j| > m. \tag{171}$$

Clearly this would imply that $(\mathbb{F}^\mathsf{T} \mathbb{Q}^x \mathbb{F})_{ij} = 0$ for $x \in [a,b]$ and $|i-j| > m$, and so $(\mathbb{F}^\mathsf{T} \mathbb{Q} \mathbb{F})_{ij} = 0$ for $|i-j| > m$; focusing on the lower-right submatrix of dimension $(n-m) \times (n-m)$, this would mean $(\mathbb{G}^\mathsf{T} \mathbb{M} \mathbb{G})_{ij} = (\mathbb{V}_n^m)_{ij} = 0$ for $|i-j| > m$, which is the claimed bandedness property of $\mathbb{V}_n^m$.

Proof of the bandedness property (171) for $i > k+1, j > k+1$. Consider $i > k+1$. At the first iteration of the recursion (169), (170), we get

$$u_i^{x,1} = \big(h_i^k(x) - h_{i+1}^k(x)\big) \cdot \frac{k}{x_i - x_{i-k}}, \tag{172}$$

where we set $h_{n+1}^k = 0$ for notational convenience. Next we present a helpful lemma, which is an application of the elementary result in Lemma 19, on differences of Newton polynomials (recall this serves as the main driver behind the proof of Lemma 2). Since (173) is a direct consequence of (155) (more specifically, a direct consequence of the special case highlighted in (156)), we state the lemma without proof.

Lemma 21. *For any $k \ge 1$, the piecewise polynomials in the kth degree falling factorial basis, given in the second line of (5), satisfy for each $k+2 \le i \le n-1$,*

$$h_i^k(x) - h_{i+1}^k(x) = h_i^{k-1}(x) \cdot \frac{x_i - x_{i-k}}{k}, \quad \textit{for } x \notin (x_{i-1}, x_i]. \tag{173}$$

Fix $i \le n-m$. Applying Lemma 21 to (172), we see that for $x \notin (x_{i-1}, x_i]$, we have simply $u_i^{x,1} = h_i^{k-1}(x)$. By the same argument, for $x \notin (x_{i-1}, x_{i+1}]$,

$$\begin{aligned} u_i^{x,2} &= (u_i^{x,1} - u_{i+1}^{x,1}) \cdot \frac{k-1}{x_i - x_{i-(k-1)}} \\ &= \big(h_i^{k-1}(x) - h_{i+1}^{k-1}(x)\big) \cdot \frac{k-1}{x_i - x_{i-(k-1)}} \\ &= h_i^{k-2}(x). \end{aligned}$$

Iterating this argument over $u_i^{x,\ell}$, $\ell = 3, \dots, m$, we get that for $x \notin (x_{i-1}, x_{i+m-1}]$,

$$\begin{aligned} u_i^{x,m} &= u_i^{x,m-1} - u_{i+1}^{x,m-1} \\ &= h_i^m(x) - h_{i+1}^m(x) \\ &= h_i^{m-1}(x) \cdot \frac{x_i - x_{i-m}}{m}. \end{aligned}$$

As $h_i^{m-1} = 0$ on $[a, x_{i-1}]$ and it is a polynomial of degree $m-1$ on $(x_{i-1}, b]$, we therefore conclude that $D^m u_i^{x,m} = 0$ for $x \notin (x_{i-1}, x_{i+m-1}]$.

For $i \ge n-m+1$, note that we can still argue $u_i^{x,m} = 0$ for $x \le x_{i-1}$, as $u_i^{x,m}$ is just a linear combination of the evaluations $h_i^k(x), h_{i+1}^k(x), \dots, h_n^k(x)$, each of which are zero. Thus, introducing the convenient notation $\bar{x}_i = x_i$ for $i \le n-1$ and $\bar{x}_i = b$ for $i \ge n$, we can still write $D^m u_i^{x,m} = 0$ for $x \notin (x_{i-1}, \bar{x}_{i+m-1}]$.

Putting this together, we see that for $i > k+1, j > k+1$, the product $(D^m u_i^{x,m})(D^m u_j^{x,m})$ can only be nonzero if $x \notin (x_{i-1}, \bar{x}_{i+m-1}] \cap (x_{j-1}, \bar{x}_{j+m-1}]$, which can only happen (this intersection is only nonempty) if $|i-j| \le m$. This proves (171) for $i > k+1, j > k+1$.

Proof of the bandedness property (171) **for** $i \le k+1, j > k+1$**.** Consider $i = k+1$. At the first iteration of the recursion (169), (170), we get

$$u_{k+1}^{x,1} = \big(h_{k+1}^k(x) - h_{k+2}^k(x)\big) \cdot \frac{k}{x_{k+1} - x_1}. \tag{174}$$

We give another helpful lemma, similar to Lemma 21. As (175) is again a direct consequence of (155) from Lemma 19 (indeed a direct consequence of the special case in (156)), we state the lemma without proof.

Lemma 22. *For any $k \ge 1$, the last of the pure polynomials and the first of the piecewise polynomials in the kth degree falling factorial basis, given in (5), satisfy*

$$h_{k+1}^k(x) - h_{k+2}^k(x) = h_{k+1}^{k-1}(x) \cdot \frac{x_{k+1} - x_1}{k}, \quad \textit{for } x > x_{k+1}. \tag{175}$$

Applying Lemma 22 to (174), we see that for $x > x_{k+1}$, it holds that $u_{k+1}^{x,2} = h_{k+1}^{k-1}(x)$. Combined with our insights from the recursion for the case $i > k+1$ developed previously, at the next iteration we see that for $x > x_{k+2}$,

$$\begin{aligned} u_{k+1}^{x,2} &= (u_{k+1}^{x,1} - u_{k+2}^{x,1}) \cdot \frac{k-1}{x_{k+1} - x_2} \\ &= \big(h_i^{k-1}(x) - h_{i+1}^{k-1}(x)\big) \cdot \frac{k-1}{x_{k+1} - x_2} \\ &= h_{k+1}^{k-2}(x). \end{aligned}$$

Iterating this argument over $u_i^{x,\ell}$, $\ell = 3, \dots, m$, we get that for $x > x_{k+m}$,

$$\begin{aligned} u_{k+1}^{x,m} &= u_{k+1}^{x,m-1} - u_{k+2}^{x,m-1} \\ &= h_{k+1}^m(x) - h_{k+2}^m(x) \\ &= h_{k+1}^{m-1}(x) \cdot \frac{x_{k+1} - x_{k+1-m}}{m}. \end{aligned}$$

and as before, we conclude that $D^m u_{k+1}^{x,m} = 0$ for $x > x_{k+m}$.

For $i < k+1$, the same argument applies, but just lagged by some number of iterations (for $\ell = 1, \dots, k+1-i$, we stay at $u_i^{x,\ell} = h_i^k(x)$, then for $\ell = k+2-i$, we get $u_i^{x,\ell} = (h_i^k(x) - h_{i+1}^k(x)) \cdot (i-1)/(x_i - x_1)$, so Lemma 22 can be applied, and so forth), which leads us to $D^m u_i^{x,m} = 0$ for $x > x_{i+m-1}$.

Finally, for $i \le k+1$ and $|i-j| > m$, we examine the product $(D^m u_i^{x,m})(D^m u_j^{x,m})$. As $|i-j| > m$, we must have either $j < m$ or $j > k+1$. For $j < m$, we have already shown $(\mathbb{F}^\mathsf{T} \mathbb{Q} \mathbb{F})_{ij} = 0$, and so for our ultimate purpose (of establishing (171) to establish bandedness of $\mathbb{F}^\mathsf{T} \mathbb{Q} \mathbb{F}$), we only need to consider the case $j > k+1$. But then (from our analysis in the last part) we know $(D^m u_j^{x,m}) = 0$ for $x \le x_{j-1}$, whereas (from our analysis in the current part) $(D^m u_i^{x,m}) = 0$ for $x > x_{i+m-1}$, and since $x_{j-1} > x_{i+m-1}$, we end up with $(D^m u_i^{x,m})(D^m u_j^{x,m}) = 0$ for all x. This establishes the desired property (171) over all i, j, and completes the proof of the theorem.

B.7 Proof of Lemma 13

To avoid unnecessary indexing difficulties, we will work directly on the entries of $\mathbb{Q}$, defined in (158), and then we will be able to read off the result for the entries of $\mathbb{M}$, defined in (107), by inspecting the lower-right submatrix of dimension $(n-m) \times (n-m)$. Fix $i \ge j$, with $i > 2m$. Applying integration by parts on each subinterval of $[a, b]$ in which the product $(D^m h_i^k)(D^m h_j^k)$ is continuous, we get

$$\int_a^b (D^m h_i^k)(x)(D^m h_j^k)(x)\, dx = (D^m h_i^k)(x)(D^{m-1} h_j^k)(x)\Big|_{a, x_{j-1}, x_{i-1}}^{x_{j-1}, x_{i-1}, b} - \int_a^b (D^{m+1} h_i^k)(x)(D^{m-1} h_j^k)(x)\, dx,$$

where we use the notation

$$f(x)\Big|_{a_1, \dots, a_r}^{b_1, \dots, b_r} = \sum_{i=1}^r \big(f^-(b_i) - f^+(a_i)\big).$$

as well as $f^-(x) = \lim_{t \to x^-} f(t)$ and $f^+(x) = \lim_{t \to x^+} f(t)$. As h_i^k and h_j^k are supported on $(x_{i-1}, b]$ and $(x_{j-1}, b]$, respectively, so are there derivatives, and as $x_{i-1} \ge x_{j-1}$ (since $i \ge j$) the second to last display reduces to

$$\int_a^b (D^m h_i^k)(x)(D^m h_j^k)(x)\, dx = (D^m h_i^k)(x)(D^{m-1} h_j^k)(x)\Big|_{x_{i-1}}^{b} - \int_a^b (D^{m+1} h_i^k)(x)(D^{m-1} h_j^k)(x)\, dx,$$

Applying integration by parts $m-2$ more times (and using $k=2m-1$) yields

$$\int_a^b (D^m h_i^k)(x)(D^m h_j^k)(x)\,dx$$
$$= \sum_{\ell=1}^{m-1} (-1)^{\ell-1}(D^{m+\ell-1}h_i^k)(x)(D^{m-\ell}h_j^k)(x)\Big|_{x_{i-1}}^{b} + (-1)^{m-1}\int_a^b (D^k h_i^k)(x)(Dh_j^k)(x)\,dx \tag{176}$$
$$= \sum_{\ell=1}^{m-1} (-1)^{\ell-1}(D^{m+\ell-1}h_i^k)(x)(D^{m-\ell}h_j^k)(x)\Big|_{x_{i-1}}^{b} + (-1)^{m-1}\big(h_j^k(b) - h_j^k(x_{i-1})\big), \tag{177}$$

where in the second line we used $(D^k h_i^k)(x) = 1\{x > x_{i-1}\}$ and the fundamental theorem of calculus. The result for the case $i \le 2m$ is similar, the only difference being that we apply integration by parts a total of $i-m-1$ (rather than $m-1$ times), giving

$$\int_a^b (D^m h_i^k)(x)(D^m h_j^k)(x)\,dx = \sum_{\ell=1}^{i-m-1} (-1)^{\ell-1}(D^{m+\ell-1}h_i^k)(x)(D^{m-\ell}h_j^k)(x)\Big|_a^b + (-1)^{i-m-1}\big(h_j^k(b) - h_j^k(a)\big). \tag{178}$$

Putting together (177), (178) establishes the desired result (112) (recalling that the latter is cast in terms of the lower-right $(n-m)\times(n-m)$ submatrix of $\mathbb{Q}$, and is hence given by replacing i,j with $i+m, j+m$, respectively).

B.8 Proof of Lemma 15

For $k=0$ or $k=1$, we can use elementary piecewise constant or continous piecewise linear interpolation. For $k=0$, we set g to be the piecewise constant function that has knots in $x_{1:(n-1)}$, and $g(x_i)=f(x_i)$, $i=1\dots,n$; note clearly, $\mathrm{TV}(g)\le \mathrm{TV}(f)$. For $k=1$, we again set g to be the continous piecewise linear function with knots in $x_{2:(n-1)}$, and $g(x_i)=f(x_i)$, $i=1\dots,n$; still clearly, $\mathrm{TV}(Dg)\le\mathrm{TV}(Df)$. This proves (118).

For $k\ge 2$, we can appeal to well-known approximation results for kth degree splines, for example, Theorem 6.20 of Schumaker (2007). First we construct a quasi-uniform partition from $x_{(k+1):(n-1)}$, call it $x^*_{1:r}\subseteq x_{(k+1):(n-1)}$, such that $\delta_n/2 \le \max_{i=1,\dots,r-1}\,(y_{i+1}-y_i)\le 3\delta_n/2$, and an extended partition $y_{1:(r+2k+2)}$,

$$y_1=\cdots=y_{k+1}=a,\quad y_{k+2}=x_1^*<\cdots<y_{r+k+1}=x_r^*,\quad y_{r+k+2}=\cdots=y_{r+2k+2}=b.$$

Now for each $\ell=k+1,\dots,r+k+1$, define $I_\ell=[y_\ell,y_{\ell+1}]$ and $\bar I_\ell=[y_{\ell-k},y_{\ell+k+1}]$. Then there exists a kth degree spline g with knots in $x^*_{1:r}$, such that, for any $d=0,\dots,k$, and a constant $b_k>0$ that depends only on k,

$$\|D^d(f-g)\|_{L_\infty(\bar I_\ell)} \le b_k\delta_n^{k-d}\omega(D^k f;\delta_n)_{L_\infty(\bar I_\ell)}, \tag{179}$$

Here $\|h\|_{L_\infty(I)}=\sup_{x\in I}\,|f(x)|$ denotes the L_∞ norm of a function h an interval I, and

$$\omega(h;v)_{L_\infty(I)} = \sup_{x,y\in I,\,|x-y|\le v} |h(x)-h(y)|$$

denotes the modulus of continuity of h on I. Note that $\omega(D^kf;\delta_n)_{L_\infty(\bar I_\ell)}\le \mathrm{TV}(D^kf)$. Thus setting $d=0$ in (179), and taking a maximum over $\ell=k+1,\dots,r+k+1$, we get $\|f-g\|_{L_\infty}\le b_k\delta_n^k\cdot\mathrm{TV}(D^kf)$. Further, the importance of the result in (179) is that it is *local* and hence allows us to make statements about total variation as well. Observe

$$\begin{aligned}\mathrm{TV}(D^kg) &= \sum_{i=k+2}^{r+k+2} |D^kg(y_i)-D^kg(y_{i-1})| \\ &\le \sum_{i=k+2}^{r+k+2}\Big(|D^kf(y_i)-D^kg(y_i)| + |D^kf(y_{i-1})-D^kg(y_{i-1})| + |D^kf(y_i)-D^kf(y_{i-1})|\Big) \\ &\le \underbrace{\big(2(k+2)b_k+1\big)}_{a_k}\cdot\mathrm{TV}(D^kf),\end{aligned}$$

In the last step above, we applied (179) with $d=k$, and the fact that each interval $\bar I_\ell$ can contain at most $k+2$ of the points y_i, $i=k+1,\dots,r+k+2$. This proves (119).

B.9 Proof of Lemma 17

Observe that, by adding and subtracting y and expanding,

$$\|\hat{\theta}_a - \hat{\theta}_b\|_2^2 = (y - \hat{\theta}_a)^\mathsf{T}(\hat{\theta}_b - \hat{\theta}_a) + (y - \hat{\theta}_b)^\mathsf{T}(\hat{\theta}_a - \hat{\theta}_b). \tag{180}$$

By the stationarity condition for problem (139), we have $y - \hat{\theta}_a = \lambda_a \mathbb{A}\hat{\theta}_a$, so that

$$\begin{aligned}(y - \hat{\theta}_a)^\mathsf{T}(\hat{\theta}_b - \hat{\theta}_a) &\le \lambda_a \hat{\theta}_a^\mathsf{T} \mathbb{A} \hat{\theta}_b - \lambda_a \hat{\theta}_a^\mathsf{T} \mathbb{A} \hat{\theta}_a \\ &\le \frac{1}{2}\lambda_a \hat{\theta}_b^\mathsf{T} \mathbb{A} \hat{\theta}_b - \frac{1}{2}\lambda_a \hat{\theta}_a^\mathsf{T} \mathbb{A} \hat{\theta}_a,\end{aligned}$$

where in the second line we used the inequality $u^\mathsf{T}\mathbb{A}v \le u^\mathsf{T}\mathbb{A}u/2 + v^\mathsf{T}\mathbb{A}v/2$. By the same logic,

$$(y - \hat{\theta}_b)^\mathsf{T}(\hat{\theta}_a - \hat{\theta}_b) \le \frac{1}{2}\lambda_b \hat{\theta}_a^\mathsf{T} \mathbb{B} \hat{\theta}_a - \frac{1}{2}\lambda_b \hat{\theta}_b^\mathsf{T} \mathbb{B} \hat{\theta}_b.$$

Applying the conclusion in the last two displays to (180),

$$\begin{aligned}\|\hat{\theta}_a - \hat{\theta}_b\|_2^2 &\le \frac{1}{2}\lambda_a \hat{\theta}_b^\mathsf{T} \mathbb{A} \hat{\theta}_b - \frac{1}{2}\lambda_a \hat{\theta}_a^\mathsf{T} \mathbb{A} \hat{\theta}_a + \frac{1}{2}\lambda_b \hat{\theta}_a^\mathsf{T} \mathbb{B} \hat{\theta}_a - \frac{1}{2}\lambda_b \hat{\theta}_b^\mathsf{T} \mathbb{B} \hat{\theta}_b \\ &\le \frac{1}{2}\sigma\lambda_a \hat{\theta}_b^\mathsf{T} \mathbb{B} \hat{\theta}_b - \frac{1}{2}\lambda_a \hat{\theta}_a^\mathsf{T} \mathbb{A} \hat{\theta}_a + \frac{1}{2}(\lambda_b/\tau) \hat{\theta}_a^\mathsf{T} \mathbb{A} \hat{\theta}_a - \frac{1}{2}\lambda_b \hat{\theta}_b^\mathsf{T} \mathbb{B} \hat{\theta}_b,\end{aligned}$$

where in the second line we twice used the spectral similarity property (138). The desired result follows by grouping terms.

B.10 Proof of Theorem 7

Note that

$$\begin{aligned}\mathbb{K}_n^2, \mathbb{W}_n^2 \text{ are } (\sigma,\tau)\text{-spectrally-similar} &\iff (\mathbb{K}_n^2)^{-1}, (\mathbb{W}_n^2)^{-1} \text{ are } (1/\sigma, 1/\tau)\text{-spectrally-similar} \\ &\iff \mathbb{W}_n^2(\mathbb{K}_n^2)^{-1}\mathbb{W}_n^2, \mathbb{W}_n^2 \text{ are } (1/\sigma, 1/\tau)\text{-spectrally-similar.}\end{aligned}$$

Set $\mathbb{A} = \mathbb{W}_n^2(\mathbb{K}_n^2)^{-1}\mathbb{W}_n^2$. From (21), we can see that

$$\mathbb{A}_{ij} = \begin{cases} \dfrac{x_{i+2} - x_i}{3} & \text{if } i = j \\ \dfrac{x_{i+1} - x_i}{6} & \text{if } i = j + 1. \end{cases}$$

Now define $a_i = (x_{i+2} - x_i)/3$ and $b_i = (x_{i+2} - x_{i+1})/6$, for $i = 1, \dots, n-2$. Also denote $q_i = (x_{i+2} - x_i)/2$, for $i = 1, \dots, n-2$. Fix $u \in \mathbb{R}^n$. For notational convenience, set $b_0 = u_0 = 0$ and $u_{n-1} = 0$. Then

$$\begin{aligned}u^\mathsf{T}\mathbb{A}u &= \sum_{i=1}^{n-2}\Big(a_i u_i^2 + b_{i-1}u_{i-1}u_i + b_i u_i u_{i+1}\Big) \\ &\le \sum_{i=1}^{n-2}\Big(a_i u_i^2 + \frac{b_{i-1}}{2}(u_{i-1}^2 + u_i^2) + \frac{b_i}{2}(u_i^2 + u_{i+1}^2)\Big) \\ &= \sum_{i=1}^{n-2}(a_i + b_{i-1} + b_i)u_i^2 \\ &= \sum_{i=1}^{n-2} q_i u_i^2 - \frac{x_2 - x_1}{6}u_1^2 - \frac{x_{n-1} - x_{n-2}}{6}u_{n-2}^2 \\ &\le \sum_{i=1}^{n-2} q_i u_i^2.\end{aligned}$$

In the second line above, we used $2st \le s^2 + t^2$, and in the fourth we used $a_i + b_{i-1} + b_i = q_i$, for $i = 1, \dots, n-2$. This shows that we can take $1/\tau = 1$, that is, $\tau = 1$.

As for the other direction, using $2st \ge -s^2 - t^2$, we have

$$\begin{aligned}
u^\mathsf{T} W u &\ge \sum_{i=1}^{n} \bigg(a_i u_i^2 - \frac{b_{i-1}}{2}(u_{i-1}^2 + u_i^2) - \frac{b_i}{2}(u_i^2 + u_{i+1}^2) \bigg) \\
&= \sum_{i=1}^{n-2} (a_i - b_{i-1} - b_i) u_i^2 \\
&= \frac{1}{2} \sum_{i=1}^{n-2} q_i u_i^2 + \frac{x_2 - x_1}{6} u_1^2 + \frac{x_{n-1} - x_{n-2}}{6} u_{n-2}^2 \\
&\ge \frac{1}{3} \sum_{i=1}^{n-2} q_i u_i^2,
\end{aligned}$$

where in the third line we used the fact that $a_i - b_{i-1} - b_i = q_i/3$, for $i = 1, \dots, n-2$. This shows that we can take $1/\sigma = 1/3$, that is, $\sigma = 3$, which completes the proof.

B.11 Proof of Lemma 18

To keep indexing simple in the current case of $m = 1$, we will compute the entries of the matrix $\mathbb{Q}$ in (158), then carry out the recursion (165)–(168), and the desired matrix $\mathbb{V}_n$ will be given be reading off the lower-right $(n-1) \times (n-1)$ submatrix of the result. Consider $i \ge j$. For $i \ge 3$, observe that

$$\begin{aligned}
\mathbb{Q}_{ij} &= \int_a^b (Dh_i^1)(x)(Dh_j^1)(x)\, dx \\
&= \int_a^b 1\{x > x_{i-1}\}\, dx \\
&= b - x_{i-1}.
\end{aligned}$$

Meanwhile, for $i = 2$, by a similar calculation, $\mathbb{Q}_{ij} = b - a$. Therefore, introducing the convenient notation $\bar{x}_i = x_i$ for $i \ge 3$ and $\bar{x}_i = a$ for $i = 2$, we get

$$\mathbb{Q}_{ij} = b - \bar{x}_{i-1},$$

for all $i \ge 2$. We know that the result of the recursion in (165)–(168) will be diagonal. As $m = 1$, this recursion reduces to simply (166), (168), which together give

$$\begin{aligned}
\mathbb{U}_{ii}^{1,1} &= (\mathbb{Q}_{ii} - \mathbb{Q}_{i+1,i}) - (\mathbb{Q}_{i,i+1} - \mathbb{Q}_{i+1,i+1}) \\
&= \big((b - \bar{x}_{i-1}) - (b - \bar{x}_i)\big) - \big((b - \bar{x}_i) - (b - \bar{x}_i)\big) \\
&= \bar{x}_i - \bar{x}_{i-1}.
\end{aligned}$$

This proves (149) (recalling that this is written in terms of $\mathbb{V}_n = \mathbb{V}^{1,1}$, the lower-right $(n-1) \times (n-1)$ submatrix of $\mathbb{U}^{1,1}$, and so for (149) we simply replace i with $i+1$).

C B-splines and discrete B-splines

C.1 B-splines

Though the truncated power basis (14) is the simplest basis for splines, the *B-spline basis* is just as fundamental, as it was "there at the very beginning", appearing in Schoenberg's original paper on splines (Schoenberg, 1946a). Here we are quoting de Boor (1976), who gives a masterful survey of the history and properties of B-splines (and points out that the name "B-spline" is derived from Schoenberg's use of the term "basic spline", to further advocate for the idea that B-splines can be seen as *the* basis for splines). A key feature of B-splines is that they have local support, and are thus extremely useful for computational purposes.

Peano representation. There are different ways to construct B-splines; here we cover a construction based on what is called the *Peano representation* for B-splines (see, for example, Theorem 4.23 in Schumaker (2007)). If f is a $k+1$ times differentiable function f on an interval $[a,b]$ (and its $(k+1)$st derivative is integrable), then by Taylor expansion

$$f(z) = \sum_{i=0}^{k} \frac{1}{i!}(D^i f)(a)(z-a)^i + \int_a^z \frac{1}{k!}(D^{k+1} f)(x)(z-x)^k \, dx.$$

Note that we can rewrite this as

$$f(z) = \sum_{i=0}^{k} \frac{1}{i!}(D^i f)(a)(z-a)^i + \int_a^b \frac{1}{k!}(D^{k+1} f)(x)(z-x)_+^k \, dx.$$

Next we take a divided difference with respect to arbitrary centers $z_1, \dots, z_{k+2} \in [a,b]$, where we assume without a loss of generality that $z_1 < \cdots < z_{k+2}$. Then by linearity we can exchange divided differentiation with integration, yielding

$$k! \cdot f[z_1, \dots, z_{k+2}] = \int_a^b (D^{k+1} f)(x) \underbrace{(\cdot - x)_+^k [z_1, \dots, z_{k+2}]}_{P^k(x; z_{1:(k+2)})} \, dx, \tag{181}$$

where we have also used the fact that a $(k+1)$st order divided difference (with respect to any $k+2$ centers) of a kth degree polynomial is zero (for example, see (58)), and lastly, we multiplied both sides by $k!$. To be clear, the notation $(\cdot - x)_+^k [z_1, \dots, z_{k+2}]$ means that we are taking the divided difference of the function $z \mapsto (z-x)_+^k$ with respect to centers $z_1, \dots, z_{k+2}$.

B-spline definition. The result in (181) shows that the $(k+1)$st divided difference of any (smooth enough) function f can be written as a weighted average of its $(k+1)$st derivative, in a local neighborhood around the corresponding centers, where the weighting is given by a universal kernel $P^k(\cdot; z_{1:(k+2)})$ (that does not depend on f), which is called the *Peano kernel* formulation for the B-spline; to be explicit, this is

$$P^k(x; z_{1:(k+2)}) = (\cdot - x)_+^k [z_1, \dots, z_{k+2}]. \tag{182}$$

Since

$$(z-x)_+^k - (-1)^{k+1}(x-z)_+^k = (z-x)^k,$$

and any $(k+1)$st order divided difference of the kth degree polynomial $z \mapsto (z-x)^k$ is zero, we can rewrite the above (182) as:

$$P^k(x; z_{1:(k+2)}) = (-1)^{k+1}(x - \cdot)_+^k [z_1, \dots, z_{k+2}]. \tag{183}$$

The function $P^k(\cdot; z_{1:(k+2)})$ is called a kth degree *B-spline* with knots $z_{1:(k+2)}$. It is a linear combination of kth degree truncated power functions and is hence indeed a kth degree spline.

It is often more convenient to deal with the *normalized B-spline*:

$$M^k(x; z_{1:(k+2)}) = (-1)^{k+1}(z_{k+2} - z_1)(x - \cdot)_+^k [z_1, \dots, z_{k+2}]. \tag{184}$$

It is easy to show that

$$M^k(\cdot; z_{1:(k+2)}) \text{ is supported on } [z_1, z_{k+2}], \text{ and } M^k(x; z_{1:(k+2)}) > 0 \text{ for } x \in (z_1, z_{k+2}). \tag{185}$$

To see the support result, note that for $x > z_{k+2}$, we are taking a divided difference of all zeros, which of course zero, and for $x < z_1$, we are taking a $(k+1)$st order divided difference of a polynomial of degree k, which is again zero. To see the positivity result, we can, for example, appeal to induction on k and the recursion to come later in (188).

B-spline basis. To build a local basis for $\mathcal{S}^k(t_{1:r}, [a,b])$, the space of kth degree splines with knots $t_{1:r}$, where we assume $a < t_1 < \cdots < t_r < b$, we first define boundary knots

$$t_{-k} < \cdots < t_{-1} < t_0 = a, \quad \text{and} \quad b = t_{r+1} < t_{r+2} < \cdots < t_{r+k+1}.$$

(Any such values for $t_{-k}, \ldots, t_0$ and $t_{r+1}, \ldots, t_{r+k+1}$ will suffice to produce a basis; in fact, setting $t_{-k} = \cdots = t_0$ and $t_{r+1} = \cdots = t_{r+k+1}$ would suffice, though this would require us to understand how to properly interpret divided differences with repeated centers; as in Definition 2.49 of Schumaker (2007).) We then define the normalized B-spline basis M_j^k, $j = 1, \ldots, r+k+1$ for $\mathcal{S}^k(t_{1:r}, [a,b])$ by

$$M_j^k = M^k(\cdot; t_{(j-k-1):j})\Big|_{[a,b]}, \quad j = 1, \ldots, r+k+1. \tag{186}$$

It is clear that each M_j^k, $j = 1, \ldots, r+k+1$ is a kth degree spline with knots in $t_{1:r}$; hence to verify that they are a basis for $\mathcal{S}^k(t_{1:r}, [a,b])$, we only need to show their linear independence, which is straightforward using the structure of their supports (for example, see Theorem 4.18 of Schumaker (2007)).

For concreteness, we note that the 0th degree normalized B-splines basis for $\mathcal{S}^0(t_{1:r}, [a,b])$ is simply

$$M_j^0 = 1_{I_j}, \quad j = 1, \ldots, r+1. \tag{187}$$

Here $I_0 = [t_0, t_1]$ and $I_i = (t_i, t_{i+1}]$, $i = 1, \ldots, r$, and we use $t_{r+1} = b$ for notational convenience. We note that this particular choice for the half-open intervals (left- versus right-side open) is arbitrary, but consistent with our definition of the truncated power basis (14) when $k = 0$. Figure 9 shows example normalized B-splines of degrees 0 through 3.

Recursive formulation. B-splines satisfy a recursion relation that can be seen directly from the recursive nature of divided differences: for any $k \geq 1$ and centers $z_1 < \cdots < z_{k+2}$,

$$\begin{aligned}
(x - \cdot)_+^k [z_1, \ldots, z_{k+2}] &= \frac{(x - \cdot)_+^k [z_2, \ldots, z_{k+2}] - (x - \cdot)_+^k [z_1, \ldots, z_{k+1}]}{z_{k+2} - z_1} \\
&= \frac{(x - z_{k+2})(x - \cdot)_+^{k-1} [z_2, \ldots, z_{k+2}] - (x - z_1)(x - \cdot)_+^{k-1} [z_1, \ldots, z_{k+1}]}{z_{k+2} - z_1},
\end{aligned}$$

where in the second line we applied the Leibniz rule for divided differences (for example, Theorem 2.52 of Schumaker (2007)), $fg[z_1, \ldots, z_{k+1}] = \sum_{i=1}^{k+1} f[z_1, \ldots, z_i] g[z_i, \ldots, z_{k+1}]$, to conclude that

$$\begin{aligned}
(x - \cdot)_+^k [z_1, \ldots, z_{k+1}] &= (x - z_1) \cdot (x - \cdot)_+^{k-1} [z_1, \ldots, z_{k+1}] \\
(x - \cdot)_+^k [z_2, \ldots, z_{k+2}] &= (x - \cdot)_+^{k-1} [z_2, \ldots, z_{k+2}] \cdot (x - z_{k+2}).
\end{aligned}$$

Translating the above recursion over to normalized B-splines, we get

$$M^k(x; z_{1:(k+2)}) = \frac{x - z_1}{z_{k+1} - z_1} \cdot M^{k-1}(x; z_{1:(k+1)}) + \frac{z_{k+2} - x}{z_{k+2} - z_2} \cdot M^{k-1}(x; z_{2:(k+2)}), \tag{188}$$

which means that for the normalized basis,

$$M_j^k(x) = \frac{x - t_{j-k-1}}{t_{j-1} - t_{j-k-1}} \cdot M_{j-1}^{k-1}(x) + \frac{t_j - x}{t_j - t_{j-k}} \cdot M_j^{k-1}(x), \quad j = 1, \ldots, r+k+1. \tag{189}$$

Above, we naturally interpret $M_0^{k-1} = M^{k-1}(\cdot; t_{-k:0})|_{[a,b]}$ and $M_{r+k+1}^{k-1} = M^{k-1}(\cdot; t_{(r+1):(r+k+1)})|_{[a,b]}$.

The above recursions are very important, both for verifying numerous properties of B-splines and for computational purposes. In fact, many authors prefer to use recursion to define a B-spline basis in the first place: they start with (187) for $k = 0$, and then invoke (189) for all $k \geq 1$.

C.2 Discrete B-splines

Here we will assume the design points are evenly-spaced, taking the form $[a,b]_v = \{a, a+v, \dots, b\}$ for $v > 0$ and $b = a + Nv$. As covered in Chapter 8.5 of Schumaker (2007), in this evenly-spaced case, discrete B-splines can be developed in a similar fashion to B-splines. Below we will jump directly into defining the discrete B-spline, which is at face value just a small variation on the definition of the usual B-spline given above. Chapter 8.5 of Schumaker (2007) develops several properties for discrete B-splines (for evenly-spaced design points)—such as a Peano kernel result for the discrete B-spline, with respect to a discrete integral—that we do not cover here, for simplicity.

Discrete B-spline definition. Let $z_{1:(k+2)} \subseteq [a,b]_v$. Assume without a loss of generality that $z_1 < \cdots < z_{k+2}$, and also $z_{k+2} \le b - kv$. We define the kth degree *discrete B-spline* or DB-spline with knots $z_1, \dots, z_{k+2}$ by

$$U^k(x; z_{1:(k+2)}) = \Big((\cdot - x)^{k,v} \cdot 1\{\cdot > x\} \Big)[z_1, \dots, z_{k+2}], \tag{190}$$

where now we denote by $(z)^{k,v} = z(z+v)\cdots(z+(k-1)v)$ the rising factorial polynomial of degree k with gap v, which we take to be equal to 1 when $k=0$. To be clear, the notation $((\cdot - x)^{k,v} \cdot 1\{\cdot > x\})[z_1, \dots, z_{k+2}]$ means that we are taking the divided difference of the function $z \mapsto (z-x)^{k,v} \cdot 1\{z > x\}$ with respect to the centers $z_1, \dots, z_{k+2}$. Since

$$(z-x)^{k,v} \cdot 1\{z > x\} - (-1)^{k+1}(x-z)_{k,v} \cdot 1\{x > z\} = (z-x)^{k,v},$$

and any $(k+1)$st order divided difference of the kth degree polynomial $z \mapsto (z-x)^{k,v}$ is zero, we can equivalently rewrite (190) as:

$$U^k(x; z_{1:(k+2)}) = (-1)^{k+1} \Big((x - \cdot)_{k,v} \cdot 1\{x > \cdot\} \Big)[z_1, \dots, z_{k+2}]. \tag{191}$$

We see (191) is just as in the usual B-spline definition (183), but with a truncated falling factorial polynomial instead of a truncated power function. Also, note $U^k(\cdot; z_{1:(k+2)})$ is a linear combination of kth degree truncated falling factorial polynomials and is hence a kth degree discrete spline.

As before, it is convenient to define the *normalized discrete B-spline* or normalized DB-spline:

$$V^k(x; z_{1:(k+2)}) = (-1)^{k+1}(z_{k+2} - z_1) \Big((x - \cdot)_{k,v} \cdot 1\{x > \cdot\} \Big)[z_1, \dots, z_{k+2}]. \tag{192}$$

We must emphasize that

$$V^k(x; z_{1:(k+2)}) = M^k(x; z_{1:(k+2)}) \quad \text{for } k=0 \text{ or } k=1$$

(and the same for the unnormalized versions). This should not be a surprise, as discrete splines are themselves exactly splines for degrees $k=0$ and $k=1$. Back to a general degree $k \ge 0$, it is easy to show that

$$V^k(\cdot; z_{1:(k+2)}) \text{ is supported on } [z_1, z_{k+2}]. \tag{193}$$

Curiously, $V^k(\cdot; z_{1:(k+2)})$ is no longer positive on the whole interval (z_1, z_{k+2}): for $k \ge 2$, it has a negative "ripple" close to the leftmost knot z_1. This is more pronounced when the knots are closer together (separated by fewer design points), see Figure 9.

Discrete B-spline basis. To develop a local basis for $\mathcal{DS}^k_v(t_{1:r}, [a,b]_v)$, the space of kth degree discrete splines with knots in $t_{1:r}$, where $a < t_1 < \cdots < t_r < b$, and also $t_{1:r} \subseteq [a,b]_v$ and $t_r \le b - kv$, we first define boundary knots

$$t_{-k} < \cdots < t_{-1} < t_0 = a, \quad \text{and} \quad b = t_{r+1} < t_{r+2} < \cdots < t_{r+k+1},$$

as before. We then define the normalized discrete B-spline basis V^k_j, $j = 1, \dots, r+k+1$ for $\mathcal{DS}^k_v(t_{1:r}, [a,b]_v)$ by

$$V^k_j = V^k(\cdot; t_{(j-k-1):j})\Big|_{[a,b]}, \quad j = 1, \dots, r+k+1. \tag{194}$$

It is clear that each V^k_j, $j = 1, \dots, r+k+1$ is a kth degree discrete spline with knots in $t_{1:r}$; hence to verify that they form a basis for $\mathcal{DS}^k_n(t_{1:r}, [a,b])$, we only need to show their linear independence, which follows from similar arguments to the result for the usual B-splines (see also Theorem 8.55 of Schumaker (2007)).

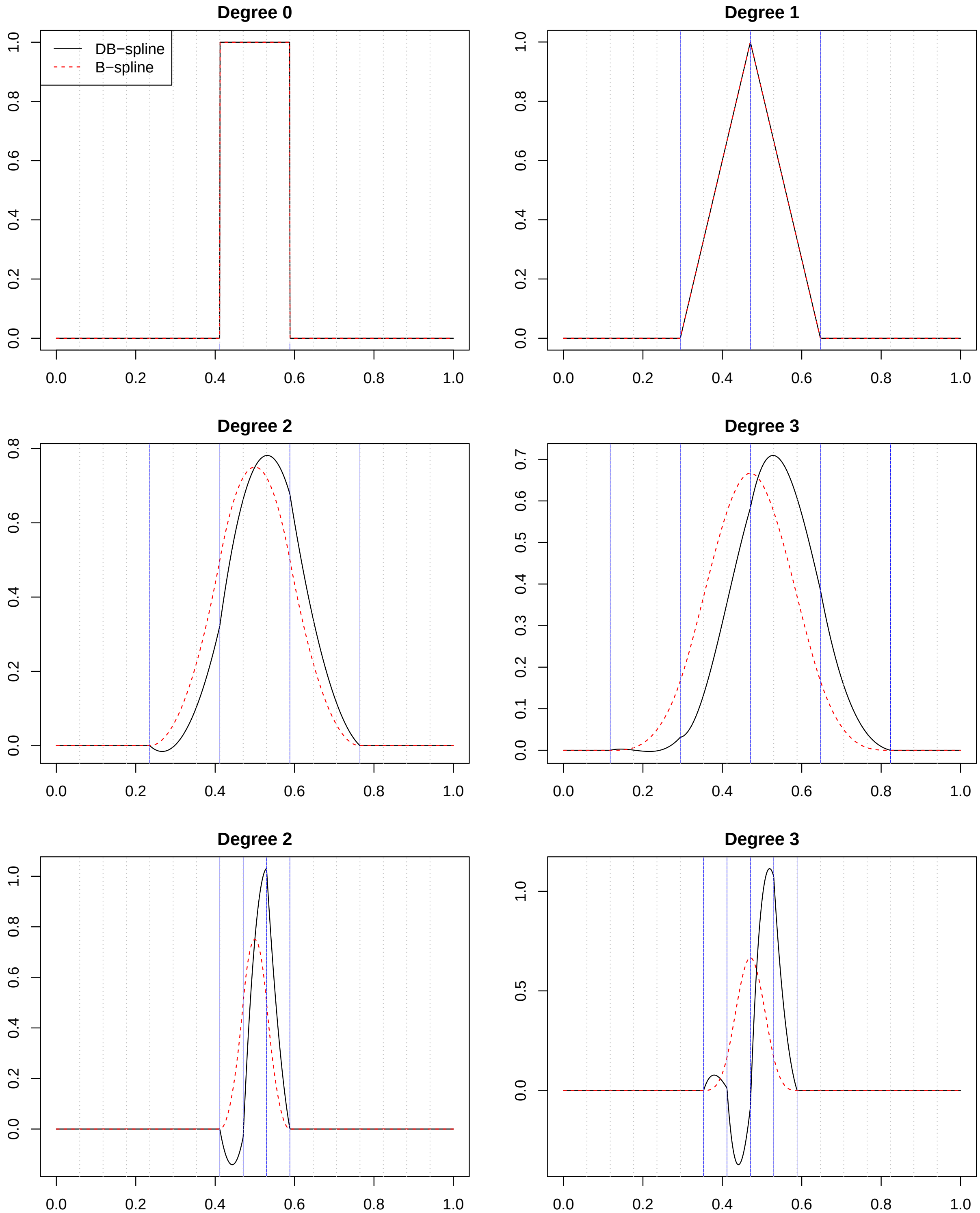

Figure 9: Normalized DB-splines in black, and normalized B-splines in dashed red, of degrees 0 through 3. In each example, the $n = 16$ design points are evenly-spaced between 0 and 1, and marked by dotted vertical lines. The knot points are marked by blue vertical lines (except for $k = 0$, as here these would obscure the B-splines, so in this case we use small blue ticks on the horizontal axis). In the bottom row, the knots are closer together; we can see that the DB-splines of degrees 2 and 3 have negative "ripples" near their leftmost knots, which is much more noticeable when the knot points are closer together.

Recursive formulation. To derive a recursion for discrete B-splines, we proceed as in the usual B-spline case, using the recursion that underlies divided differences: for any $k \geq 1$ and centers $z_1 < \cdots < z_{k+2}$ (such that $z_{1:(k+2)} \subseteq [a,b]_v$ and $z_{k+2} \leq b - kv$),

$$\begin{aligned}
&\Big((x-\cdot)_{k,v} \cdot 1\{x > \cdot\}\Big)[z_1, \ldots, z_{k+2}] \\
&\qquad = \frac{((x-\cdot)_{k,v} \cdot 1\{x > \cdot\})[z_2, \ldots, z_{k+2}] - ((x-\cdot)_{k,v} \cdot 1\{x > \cdot\})[z_1, \ldots, z_{k+1}]}{z_{k+2} - z_1} \\
&\qquad = \bigg((x - z_{k+2} - (k-1)v) \cdot ((x-\cdot)_{k-1,v} \cdot 1\{x > \cdot\})[z_2, \ldots, z_{k+2}] \\
&\qquad\quad - (x - z_1 - (k-1)v) \cdot ((x-\cdot)_{k-1,v} \cdot 1\{x > \cdot\})[z_1, \ldots, z_{k+1}]\bigg) \Big/ (z_{k+2} - z_1),
\end{aligned}$$

where as before, in the second line, we applied the Leibniz rule for divided differences to conclude

$$\begin{aligned}
\Big((x-\cdot)_{k,v} \cdot 1\{x > \cdot\}\Big)[z_1, \ldots, z_{k+1}] &= \big(x - z_1 - (k-1)v\big) \cdot \Big((x-\cdot)_{k-1,v} \cdot 1\{x > \cdot\}\Big)[z_1, \ldots, z_{k+1}] \\
\Big((x-\cdot)_{k,v} \cdot 1\{x > \cdot\}\Big)[z_2, \ldots, z_{k+2}] &= \Big((x-\cdot)_{k-1,v} \cdot 1\{x > \cdot\}\Big)[z_1, \ldots, z_{k+1}] \cdot \big(x - z_{k+2} - (k-1)v\big).
\end{aligned}$$

Translating the above recursion over normalized DB-splines, we get

$$V^k(x; z_{1:(k+2)}) = \frac{x - z_1 - (k-1)v}{z_{k+1} - z_1} \cdot V^{k-1}(x; z_{1:(k+1)}) + \frac{z_{k+2} + (k-1)v - x}{z_{k+2} - z_2} \cdot V^{k-1}(x; z_{2:(k+2)}), \tag{195}$$

which means that for the normalized basis,

$$V_j^k(x) = \frac{x - t_{j-k-1} - (k-1)v}{t_{j-1} - t_{j-k-1}} \cdot V_{j-1}^{k-1}(x) + \frac{t_j + (k-1)v - x}{t_j - t_{j-k}} \cdot V_j^{k-1}(x), \quad j = 1, \ldots, r+k+1. \tag{196}$$

Above, we naturally interpret $V_0^{k-1} = V^{k-1}(\cdot; t_{-k:0})|_{[a,b]}$ and $V_{r+k+1}^{k-1} = V^{k-1}(\cdot; t_{(r+1):(r+k+1)})|_{[a,b]}$.

D Fast matrix multiplication

We recall the details of the algorithms from Wang et al. (2014) for fast multiplication by $\mathbb{H}^k_n$, $(\mathbb{H}^k_n)^{-1}$, $(\mathbb{H}^k_n)^\mathsf{T}$, $(\mathbb{H}^k_n)^{-\mathsf{T}}$, in Algorithms 1–4. In each case, multiplication takes $O(nk)$ operations (at most $4nk$ operations), and is done in-place (no new memory required). We use cumsum to denote the cumulative sum operator, $\mathrm{cumsum}(v) = (v_1, v_1 + v_2, \ldots, v_1 + \cdots + v_n)$, for $v \in \mathbb{R}^n$, and diff for the pairwise difference operator, $\mathrm{diff}(v) = (v_2 - v_1, v_3 - v_2, \ldots, v_n - v_{n-1})$. We also use rev for the reverse operator, $\mathrm{rev}(v) = (v_n, \ldots, v_1)$, and $\odot$ for elementwise multiplication between vectors.

Algorithm 1 Multiplication by $\mathbb{H}^k_n$

Input: Integer degree $k \geq 0$, design points $x_{1:n}$ (assumed in sorted order), vector to be multiplied $v \in \mathbb{R}^n$.
Output: v is overwritten by $\mathbb{H}^k_n v$.
for $i = k$ to 0 **do**
  $v_{(i+1):n} = \mathrm{cumsum}(v_{(i+1):n})$
  if $i \neq 0$ **then**
    $v_{(i+1):n} = v_{(i+1):n} \odot \frac{x_{(i+1):n} - x_{1:(n-i)}}{i}$
  end if
end for
Return v.

Algorithm 2 Multiplication by $(\mathbb{H}^k_n)^{-1}$

Input: Integer degree $k \geq 0$, design points $x_{1:n}$ (assumed in sorted order), vector to be multiplied $v \in \mathbb{R}^n$.
Output: v is overwritten by $(\mathbb{H}^k_n)^{-1} v$.
for $i = 0$ to k **do**
  if $i \neq 0$ **then**
    $v_{(i+1):n} = v_{(i+1):n} \odot \frac{i}{x_{(i+1):n} - x_{1:(n-i)}}$
  end if
  $v_{(i+2):n} = \mathrm{diff}(v_{(i+1):n})$
end for
Return v.

Algorithm 3 Multiplication by $(\mathbb{H}^k_n)^\mathsf{T}$

Input: Integer degree $k \geq 0$, design points $x_{1:n}$ (assumed in sorted order), vector to be multiplied $v \in \mathbb{R}^n$.
Output: v is overwritten by $(\mathbb{H}^k_n)^{-1} v$.
for $i = 0$ to k **do**
  if $i \neq 0$ **then**
    $v_{(i+1):n} = v_{(i+1):n} \odot \frac{x_{(i+1):n} - x_{1:(n-i)}}{i}$
  end if
  $v_{(i+1):n} = \mathrm{rev}(\mathrm{cumsum}(\mathrm{rev}(v_{(i+1):n})))$
end for
Return v.

Algorithm 4 Multiplication by $(\mathbb{H}^k_n)^{-\mathsf{T}}$

Input: Integer degree $k \geq 0$, design points $x_{1:n}$ (assumed in sorted order), vector to be multiplied $v \in \mathbb{R}^n$.
Output: v is overwritten by $(\mathbb{H}^k_n)^{-\mathsf{T}} v$.
for $i = k$ to 0 **do**
  $v_{(i+1):n-1} = \mathrm{rev}(\mathrm{diff}(\mathrm{rev}(v_{(i+1):n})))$
  if $i \neq 0$ **then**
    $v_{(i+1):n} \odot \frac{i}{x_{(i+1):n} - x_{1:(n-i)}}$
  end if
end for
Return v.

References

Taylor Arnold and Ryan J. Tibshirani. Efficient implementations of the generalized lasso dual path algorithm. *Journal of Computational and Graphical Statistics*, 25(1):1–27, 2016.

Joshua Batson, Daniel Spielman, Nikhil Srivastava, and Shang-Hua Teng. Spectral sparsification of graphs: theory and algorithms. *Communications of the ACM*, 56(8):87–94, 2013.

Georg Bohlmann. Ein ausgleichungsproblem. *Nachrichten von der Gesellschaft der Wissenschaften zu Gottingen, Mathematisch-Physikalische Klasse*, pages 260–271, 1899.

Scott Chen, David L. Donoho, and Michael Saunders. Atomic decomposition for basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

Peter Craven and Grace Wahba. Smoothing noisy data with spline functions. *Numerische Mathematik*, 31(4):377–403, 1978.

Carl de Boor. Splines as linear combinations of B-splines. In Goerge G. Lorentz, Charles K. Chui, and Larry L. Schumaker, editors, *Approximation Theory II*, pages 1–47. Academic Press, 1976.

Carl de Boor. *A Practical Guide to Splines*. Springer, 1978.

Carl de Boor. Divided differences. arXiv: 0502036, 2005.

Ronald DeVore and George Lorentz. *Constructive Approximation*. Springer, 1993.

David L. Donoho and Iain M. Johnstone. Minimax estimation via wavelet shrinkage. *Annals of Statistics*, 26(8): 879–921, 1998.

Jerome Friedman, Trevor Hastie, Holger Hoefling, and Robert Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332, 2007.

Peter J. Green and Bernard W. Silverman. *Nonparametric Regression and Generalized Linear Models: A Roughness Penalty Approach*. Chapman & Hall/CRC Press, 1993.

Thomas N. E. Greville. The general theory of osculatory interpolation. *Transactions of the Acturial Society of America*, 45(112):202–265, 1944.

Robert Henderson. A new method of graduation. *Transactions of the Actuarial Society of America*, 25:29–53, 1924.

Robert J. Hodrick and Edward C. Prescott. Postwar U.S. business cycles: An empirical investigation. Working paper, Northwestern University, 1981.

Robert J. Hodrick and Edward C. Prescott. Postwar U.S. business cycles: An empirical investigation. *Journal of Money, Credit, and Banking*, 29(1):1–16, 1997.

Nicholas Johnson. A dynamic programming algorithm for the fused lasso and l_0-segmentation. *Journal of Computational and Graphical Statistics*, 22(2):246–260, 2013.

Seung-Jean Kim, Kwangmoo Koh, Stephen Boyd, and Dimitry Gorinevsky. ℓ_1 trend filtering. *SIAM Review*, 51(2): 339–360, 2009.

Roger Koenker, Pin Ng, and Stephen Portnoy. Quantile smoothing splines. *Biometrika*, 81(4):673–680, 1994.

Tom Lyche. Discrete polynomial spline approximation methods. In Klaus Bohmer, Gunter Meinardus, and Walter Schempp, editors, *Spline Functions*, pages 144–176. Springer, 1975.

Enno Mammen and Sara van de Geer. Locally apadtive regression splines. *Annals of Statistics*, 25(1):387–413, 1997.

Olvi L. Mangasarian and Larry L. Schumaker. Discrete splines via mathematical programming. *SIAM Journal on Control*, 9(2):174–183, 1971.

Olvi L. Mangasarian and Larry L. Schumaker. Best summation formulae and discrete splines. *SIAM Journal on Numerical Analysis*, 10(3):448–459, 1973.

Isaac Newton. *Philosophiae Naturalis Principia Mathematica*. 1687.

Isaac Newton. *Methodus Differentialis*. 1711.

Aaditya Ramdas and Ryan J. Tibshirani. Fast and flexible ADMM algorithms for trend filtering. *Journal of Computational and Graphical Statistics*, 25(3):839–858, 2016.

Christian H. Reinsch. Smoothing by spline functions. *Numerische Mathematik*, 10(3):177–183, 1967.

Leonid I. Rudin, Stanley Osher, and Emad Faterni. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268, 1992.

Veeranjaneyulu Sadhanala and Ryan J. Tibshirani. Additive models via trend filtering. *Annals of Statistics*, 47(6): 3032–3068, 2019.

Veeranjaneyulu Sadhanala, Yu-Xiang Wang, and Ryan J. Tibshirani. Graph sparsification approaches for laplacian smoothing. *International Conference on Artificial Intelligence and Statistics*, 19, 2016.

Isaac J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions, Part A: on the problem of smoothing of graduation, a first class of analytic approximation formulae. *Quarterly of Applied Mathematics*, 4(1):45–99, 1946a.

Isaac J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions, Part B: on the problem of smoothing of graduation, a second class of analytic approximation formulae. *Quarterly of Applied Mathematics*, 4(2):112–141, 1946b.

Isaac J. Schoenberg. Spline functions and the problem of graduation. *Proceeding of the National Academy of Sciences*, 52(4):947–950, 1964.

Donald R. Schuette. A linear programming approach to graduation. *Transactions of Society of Actuaries*, 30, 1978.

Larry L. Schumaker. Constructive aspects of discrete polynomial spline functions. In George G. Lorentz, editor, *Approximation Theory*, pages 469–476. Academic Press, 1973.

Larry L. Schumaker. *Spline Functions: Basic Theory*. Cambridge University Press, 2007. Third edition.

Gabriel Steidl, Stephan Didas, and Julia Neumann. Splines in higher order TV regularization. *International Journal of Computer Vision*, 70(3):214–255, 2006.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58(1):267–288, 1996.

Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B*, 67(1):91–108, 2005.

Ryan J. Tibshirani. Adaptive piecewise polynomial estimation via trend filtering. *Annals of Statistics*, 42(1):285–323, 2014.

Ryan J. Tibshirani and Jonathan Taylor. The solution path of the generalized lasso. *Annals of Statistics*, 39(3): 1335–1371, 2011.

Ryan J. Tibshirani and Jonathan Taylor. Degrees of freedom in lasso problems. *Annals of Statistics*, 40(2):1198–1232, 2012.

Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2009.

Sara van de Geer. *Empirical Processes in M-Estimation*. Cambdrige University Press, 2000.

Grace Wahba. *Spline Models for Observational Data*. SIAM, 1990.

Yu-Xiang Wang, Alexander Smola, and Ryan J. Tibshirani. The falling factorial basis and its statistical applications. *International Conference on Machine Learning*, 31, 2014.

Edmund T. Whittaker. On a new method of graduation. *Proceedings of the Edinburgh Mathematical Society*, 41:63–73, 1923.

Edmund T. Whittaker and George Robinson. *The Calculus of Observations: A Treatise on Numerical Mathematics*. Blackie and Son, Ltd., 1924.