Local cubic splines on non-uniform grids and real-time implementation of wavelet transform

Amir Averbuch[1] Pekka Neittaanmäki[3] Etay Shefi[2] Valery Zheludev[1,3]
[1]School of Computer Science
Tel Aviv University, Tel Aviv 69978, Israel

[2]School of Electrical Engineering
Tel Aviv University, Tel Aviv 69978, Israel

[3]Department of Mathematical Information Technology
P.O. Box 35 (Agora), University of Jyväskylä, Finland

Abstract

In this paper, local cubic quasi-interpolating splines on non-uniform grids are described. The splines are computed by fast computational algorithms that utilize the relation between splines and cubic interpolation polynomials. These splines provide an efficient tool for real-time signal processing. As an input, they use either clean or noised arbitrarily-spaced samples. Exact estimations of the approximation errors are established. The capability to adapt the grid to the structure of an object and to have minimal requirements to the operating memory are of great advantages for offline processing of signals and multidimensional data arrays. The designed splines serve as a source for generating real-time wavelet transforms for signals in scenarios where signal's samples subsequently arrive one after another at randomized times. The wavelet transforms are executed without delay. On arrival of samples, only a couple of adjacent wavelet transform coefficients are updated.

1 Introduction

Since their introduction in [13], splines have become one of the powerful tools in mathematics and computer aided geometric design. In recent decades, splines have served as a source for the constructions of wavelets, wavelet packets and wavelet frame. Splines and spline-based wavelets, wavelet packets and frames are extensively used in signal and image processing applications (see, for example, [3, 4]).

Interpolating splines possess exclusive approximation properties. In particular, an interpolating spline of order p, which consists of pieces of polynomials of degree $p-1$, restores polynomials of the same degree. Due to this property, these splines generated biorthogonal wavelets with p and, in some cases, $p+1$ vanishing moments ([3, 4]). However a drawback in the design and manipulation of interpolating splines is that, for their computation, a system of equations that contain all the available grid samples has to be solved. This fact prevents the usage of these splines in real-time processing when samples arrive dynamically and sequentially.

Therefore, the idea to have splines that can be designed and manipulated directly without resorting to systems of equations, while their approximation accuracy is close to that of the interpolating splines, is very attractive. A method for the design of such splines on the uniform grid was presented even in the pioneering spline paper [13]. When the grid is non-uniform, the design and estimation of the approximation properties of such splines is more complicated especially for higher-order splines. A number of investigations in this field were carried out in the 70 (see, for example, [12]). These splines are called local because the computation of a spline's value at a fixed point requires to use only a few adjacent grid samples. Nevertheless, there exist local splines which provide the same approximation order as the interpolating splines. That is, there exist local splines of any order p which restore polynomials of degree $p-1$. Such local splines are referred to as quasi-interpolating splines.

In this paper, we describe a procedure to design and analyze local cubic quasi-interpolating splines on arbitrary grids. Typically, local splines are designed via their B-splines representation. The presented approach, which is based on the relation between quasi-interpolating splines and cubic interpolating polynomials, results in a "simple" algorithm for spline computation . When samples are given on a limited interval, which is a typical situation, the spline is extended to the boundaries of the interval without loss of the approximation accuracy. In addition, a method for accurate extrapolation of the spline beyond the sampling interval is presented. Only six adjacent samples are needed to compute the spline's value at a certain point. Therefore, the design of the spline can be implemented in a real-time mode when samples of a signal arrive dynamically and sequentially at randomized times. Due to the extension formulas, the design can be carried out without delay up to the latest sample arrival. In addition, the extrapolation algorithm can be applied to a prediction-correction processing of signals evolving in time.

Similarly to cubic interpolating splines, the described splines restore cubic polynomials at inner parts of the sampling interval also near the boundaries and in the extrapolation process. Moreover, spline representation via interpolating polynomials provides exact estimations for the approximation errors. The described splines serve as a source for the construction of wavelet transforms of arbitrarily sampled signals.

Most of the existing wavelet transforms operate on uniformly sampled signals. A few works that describe wavelets on non-uniform grids appeared in recent years. Equidistant wavelets, which are utilized for denoising of non-uniformly sampled signals, appear in [6]. Scaling functions on different levels of enclosed irregular grids, which appear in [7], are designed as limit functions of subdivision and wavelets are designed as their linear combinations. A general method for constructing wavelets on irregular lattices in $\mathbb{R}^d$ and a general atomic frame decomposition of the space $L^2(\mathbb{R}^d)$ are described in [2].

A natural way to design and implement wavelet transforms of signals is the Lifting Scheme introduced in [18], which consists of subsequent applications of *predict* and *update* operators to signal's samples. The scheme was extended in [19] to non-uniformly sampled signals. The lifting wavelet transform on a non-uniform grid, where wavelets have two vanishing moments, is applied in [20] to signal denoising. Our design scheme is based on a quasi-interpolating local splines. The idea, in the case of uniform sampling, is explored in [4] to predict the odd samples of the signal to be transformed by values of the spline constructed on the even samples of the signal. Then, the detail coefficients are derived by subtraction of the predicted odd samples from the original ones. The next step consists of updating the even samples by values of the spline designed on the detail coefficients. The wavelets have four vanishing moments. Typically, wavelet transforms of signals

(images) defined on limited intervals (areas) require extension of the object beyond its boundaries ([5]). However, by using the definition of splines near the boundaries of of the sampling interval, the wavelet transforms are implemented without this type of extrapolation. The design of splines make them useful for real-time (with no delay) implementation of wavelet transforms in the situation when samples of a signal subsequently arrive at randomized times.

A real-time denoising method, which is based on the lifting wavelet transform on a uniform grid, is presented in [11]. The online wavelet transform is implemented using a mowing window -see also [**?**], which produces some delay with respect to the sample acquisition and requires an artificial extension of the data beyond the moving window. Our algorithm produces wavelet coefficients with no delay. Arrival of a new sample leads to the production of new coefficients and to updating of a couple of already produced coefficients.

The paper is organized as follows. Section 2 introduces the necessary notation and recalls definitions of the divided differences, interpolating polynomials and the B-splines. Section 3 introduces local cubic quasi-interpolating splines, describes algorithms for splines computation on unlimited and finite intervals and an extrapolation method. Approximation properties of the splines are investigated and numerical examples are provided. The Lifting Scheme of the wavelet transforms is presented in Section 4. In Section 5, spline-based predict and update operators are explicitly described. A scheme of real-time wavelet transforms is outlined and numerical examples are provided.

In order for the paper to be self-contained, some results about local quasi-interpolat splines, which appear in [22, 17], are included.

2 Preliminaries

General notations The following notations are used throughout the paper:

The grid on the real axis is denoted by $\mathbf{g} \stackrel{\text{def}}{=} \{t[k]\}$. The steps of the grid are denoted by $h[k] \stackrel{\text{def}}{=} t[k+1] - t[k]$.

If $t \in [t[k], t[k+1]]$ then a local variable is $\tau \stackrel{\text{def}}{=} (t - t[k])/h[k] \in [0, 1]$.

$$\omega_m[k](t) \stackrel{\text{def}}{=} (t - t[k])\,(t - t[k+1])\,...\,(t - t[k+m])\,. \tag{1}$$

For $t = t[k+l],\ 0 \le l \le m$, the derivative is

$$\omega'_m[k](t[k+l]) = \prod_{\nu=(0,...,m)\setminus l} (t[k+l] - t[k+\nu]). \tag{2}$$

$P^n(t)$ denotes a polynomial of of degree n. When $n = 3$ (cubic polynomial), we drop the degree index such that $P(t) \stackrel{\text{def}}{=} P^3(t)$. $P(t)[k]$ is used for a cubic polynomial that interpolates a function $f(t)$ at the grid points $\{t[k-1], t[k], t[k+1], t[k+2]\}$.

$C^n[a, b]$ is the space of functions that are continuous on the interval $[a, b]$ together with their derivatives up to the order n. $C^n \stackrel{\text{def}}{=} C^n(-\infty, \infty)$.

Divided and finite differences ([1, 9, 16]):

The first-order divided difference (DD) of a sequence $\mathbf{f} = \{f[k]\}$ with respect to the grid $\mathbf{g}$ is

$$f\left[k, k+1\right] \stackrel{\text{def}}{=} \frac{f[k+1] - f[k]}{t[k+1] - t[k]} = \frac{f[k]}{t[k] - t[k+1]} + \frac{f[k+1]}{t[k+1] - t[k]}.$$

The second-order DD is

$$f\left[k, k+1, k+2\right] \stackrel{\text{def}}{=} \frac{f\left[k+1, k+2\right] - f\left[k, k+1\right]}{t[k+2] - t[k]} = \frac{f[k]}{(t[k] - t[k+1])(t[k] - t[k+2])}$$
$$+ \frac{f[k+1]}{(t[k+1] - t[k])(t[k+1] - t[k+2])} + \frac{f[k+2]}{(t[k+2] - t[k])(t[k+2] - t[k+1])}.$$

Higher-orders DD are defined iteratively:

$$f\left[k, k+1, k+2, ..., k+n\right] \stackrel{\text{def}}{=} \frac{f\left[k+1, k+2, ..., k+n\right] - f\left[k, k+1, ..., k+n-1\right]}{t[k+n] - t[k]}$$
$$= \sum_{l=k}^{n} \frac{f[k+l]}{\omega_n'[k](t[k+l])}. \tag{3}$$

Proposition 2.1. *If a function $f(t)$ belongs to $C^n\left[t[k], t[k+n]\right]$, then the DD of the sequence $\mathbf{f} \stackrel{\text{def}}{=} \left\{f[k+\nu] = f(t[k+\nu])\right\},\ \nu = 0, ..., n,$ is*

$$f\left[k, k+1, k+2, ..., k+n\right] = \frac{f^{(n)}(\theta)}{n!}, \quad \theta \in \left(t[k], t[k+n]\right).$$

Consequently, if $f(t) = P^{n-1}(t)$ at the interval $[t[k], t[k+n]]$, then $f\left[k, k+1, k+2, ..., k+n\right] = 0$.

If the grid $\mathbf{g}$ is uniform, that is $h[k] = h,\ k \in \mathbb{Z}$, then

$$f\left[k, k+1, k+2, ..., k+n\right] = \frac{\Delta_h^n f[k]}{h^n\, n!},$$

where $\Delta_h^n f[k] \stackrel{\text{def}}{=} \sum_{\nu=0}^{n} (-1)^{n-\nu} \binom{n}{\nu} f[k+\nu]$ is the n-th order finite difference of the sequence $\mathbf{f}$ on the grid $\{hk\}$.

Interpolating polynomials ([1, 16, 9]):

The Newton form of the n-degree polynomial $P^n(t)$, which interpolates the samples $\{f[\nu]\}$ at the grid points $\left\{t[\nu]\right\},\ \nu = k, ..., k+n$, is represented by

$$P^n(t) = f[k] + f[k, k+1](t - t[k]) + \ldots + f[k, k+1, ..., k+n]\, \omega_{n-1}[k](t), \tag{4}$$

where the function $\omega_n[k](t)$ is defined in Eq. (1).

Denote by $f[t, k, k+1, ..., k+n]$ the DD of order $n+1$ of the set $\{f(t), f[k], f[k+1], ..., f[k+n]\}$ with respect to the grid $\mathbf{g}(t) \stackrel{\text{def}}{=} \{t, t[k], t[k+1], ..., t[k+n]\}$. Then, the remainder term of the interpolation can be explicitly expressed by

$$R^n(t) \stackrel{\text{def}}{=} f(t) - P^n(t) = f[t, k, k+1, ..., k+n]\, \omega_n[k](t). \tag{5}$$

B-splines ([8, 15]) The non-centered B-spline of order p (degree $p-1$) on the uniform grid $\{k\}$ is represented by

$$b^p(t) = \frac{1}{(p-1)!}\sum_{\nu=0}^{p}(-1)^p\binom{p}{\nu}\,(t-\nu)_+^{p-1} = \Delta_1^p\left[\frac{t_+^{p-1}}{(p-1)!}\right], \tag{6}$$

where $t_+ \stackrel{\text{def}}{=} (t+|t|)/2$ and $\Delta_1^p[f]$ denotes the order p finite difference of a function $f(t)$ with step 1. The B-spline $b^p(t)$ is supported on the interval $(0,p)$ and its nodes are located at the points $0,1,\ldots,p$.

When the grid in non-uniform, the finite difference is replaced by the DD. The B-splines of order p on the grid $\mathbf{g}$ are defined as

$$b^p(t)[k] \stackrel{\text{def}}{=} (-1)^p(t[k+p]-t[k])\sum_{\nu=0}^{p}\frac{(t-t[k+\nu])_+^{p-1}}{w_p'[k](t[k+\nu])}, \tag{7}$$

where the function $\omega_n[k](t)$ is defined in Eq. (1). The B-spline $b^p(t)[k]$ is supported on the interval $(t[k],t[k+p])$, where it is positive. It is a function from C^{p-2} such that at the intervals $(t[l],t[l+1])$ it coincides with polynomials of degree $p-1$. The grid points $\{t[k],...,t[k+p]\}$ are referred to as the nodes of the B-spline $b^p(t)[k]$.

Generally, a function $s^p(t)$, which belongs to C^{p-2} and at the intervals $(t[l],t[l+1])$ coincides with polynomials of degree $p-1$, is referred to as a spline of order p. The grid points $\{t[k]\}$ are nodes of the spline $s^p(t)$. Figure 1 displays the B-splines of different orders on a non-uniform grid.

Intermediate value theorem

Theorem 2.2. *Assume that a function $f(t)$ is continuous on the interval $[a,b]$ and α and β are real numbers with the same sign. Then, there exists a point $c\in[a,b]$ such that $\alpha f(a)+\beta f(b) = (\alpha+\beta)f(c)$.*

Proof: Denote $F(t) \stackrel{\text{def}}{=} \alpha f(a)+\beta f(b)-(\alpha+\beta)f(t)$. The function $F(t)$ is continuous on the interval $[a,b]$ and $F(a) = \beta(f(b)-f(a)),\quad F(b) = \alpha(f(a)-f(b))$. The continuous function $F(t)$ has different signs at the points a and b. Therefore, there exists a point $c\in[a,b]$ such that $F(c)=0$.■

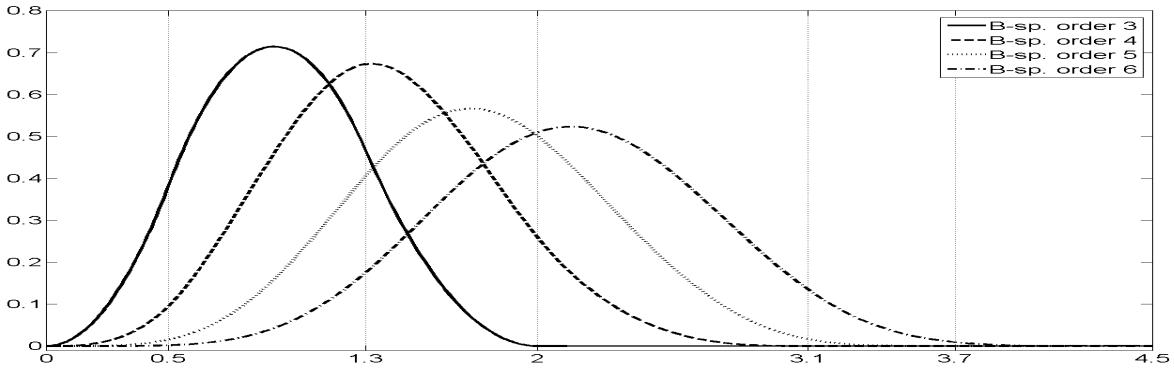

Figure 1: B-splines of order 3, 4, 5, 6 on the grid $\mathbf{g} = \{0, 0.5, 1.3, 2, 3.1, 3.7, 4.5\}$

3 Local cubic splines

In the rest of the paper, we deal with splines of order 4 (cubic splines). For such splines we drop the order index such that $b^4(t)[k] \stackrel{\text{def}}{=} b(t)[k]$ and $s(t) \stackrel{\text{def}}{=} s^4(t)$. The cubic B-spline is

$$b(t)[k] = (t[k+4] - t[k]) \sum_{\nu=0}^{4} \frac{(t - t[k+\nu])_+^3}{w_4'[k](t[k+\nu])}. \tag{8}$$

It is supported on the interval $(t[k], t[k+4])$. The following observation will be used for further design,

Proposition 3.1.

1. *The values of the B-spline $b(t)[k-3]$ at the interval $[t[k], t[k+1]]$ do not depend on the location of the grid point $t[k-3]$.*
2. *The values of the B-spline $b(t)[k]$ at the interval $[t[k], t[k+1]]$ do not depend on the location of the grid point $t[k+3]$.*

Proof: When $t \in [t[k], t[k+1]]$, the B-spline $b(t)[k-3]$ is

$$b(t)[k-3] = (t[k+1] - t[k-3]) \sum_{\nu=-3}^{0} \frac{(t - t[k+\nu])^3}{w_4'[k-3](t[k+\nu])}.$$

Adding one term to the sum

$$b(t)[k-3] + (t[k+1] - t[k-3]) \frac{(t - t[k+1])^3}{w_4'[k-3](t[k+1])} = (t[k+1] - t[k-3]) \sum_{\nu=-3}^{1} \frac{(t - t[k+\nu])^3}{w_4'[k-3](t[k+\nu])}$$

produces, due to Eq. (3), the fourth-order DD of the cubic polynomial $(t - t[k])^3$ and, as such, it is identical zero (see Proposition 2.1). Therefore, when $t = t[k] + h[k]\tau,\ \tau \in [0,1]$, the B-spline is

$$b(t)[k-3] = (t[k+1] - t[k-3]) \frac{(t[k+1] - t)^3}{w_4'[k-3](t[k+1])} = \frac{(h[k])^2 (1-\tau)^3}{(t[k+1] - t[k-2])(t[k+1] - t[k-1])}. \tag{9}$$

Item 2 is straightforward. ■

Cubic splines on the grid $\mathbf{g} = \{t[k]\}$ are represented via B-splines ([8, 15]). For $t \in [t[k], t[k+1]]$, a cubic spline $s(t)$ is given by

$$s(t) = \sum_{\nu \in \mathbb{Z}} q[\nu+2]\, b(t)[\nu] = \sum_{\nu=k-3}^{k} q[\nu+2]\, b(t)[\nu], \tag{10}$$

where $\mathbf{q} = \{q[\nu]\}$ is a sequence of real numbers, which determines the spline's properties.

Typically, for a spline that approximates a function $f(t)$, the coefficients $q[\nu]$ are derived from the samples $\left\{ f[k] \stackrel{\text{def}}{=} f(t[k]) \right\}$ of $f(t)$ on the grid $\mathbf{g}$. In that case, the spline is denoted by $s[f](t)$. For interpolating splines, the coefficients $q[\nu]$ are obtained by solving of a three-diagonal system of equations, which involves all the available samples and some boundary conditions. An alternative is provided by the so-called local splines which are defined in 3.2 and 3.3.

Definition 3.2. *If the coefficients $q[\nu]$ in Eq. (10) are finite linear combinations of grid samples $\{f[k]\}$ then the spline $s(t)$ is referred to as local spline.*

Definition 3.3. *If for any polynomial $f(t) = P^n(t)$ of degree n, a spline $s[f](t) \equiv f(t)$, then it is said that the spline $s[f](t)$ restores polynomials of degree n and its approximation order is n.*

3.1 Simplest cubic splines

In order to obtain the simplest cubic spline $s_0[f](t)$ on the grid $\mathbf{g}$, which approximates the function $f(t)$ from the samples $\left\{f[k] \stackrel{\text{def}}{=} f(t[k])\right\}$, the coefficients in Eq. (10) should be chosen to be $q_0[\nu] \stackrel{\text{def}}{=} f[\nu]$. Thus, the spline, which is denoted as $s_0[f](t)$, is expressed by

$$s_0[f](t) = \sum_{\nu=k-3}^{k} f[\nu+2]\, b(t)[\nu]. \tag{11}$$

The following important theorem was proved by I. J. Schoenberg in [14]:

Theorem 3.4 ([14])**.** *The spline $s_0[f](t)$ restores linear functions and it is variation-diminishing. It means that for $t \in [t[k], t[k+1]]$,*

$$s_0[1](t) = \sum_{\nu=k-3}^{k} b(t)[\nu] \equiv 1, \quad s_0[t](t) = \sum_{\nu=k-3}^{k} (\nu+2)\, b(t)[\nu] = t$$

and for every linear function $L(t) = ct + d$ and interval interval $[\alpha, \beta]$, the difference $s_0[f](t) - L(t)$ has no more sign variations on the interval $[\alpha, \beta]$ than what the difference $f(t) - L(t)$ has.

In addition, the shape-preserving property is inherent to splines $s_0[f](t)$ [8][1]. Due to these properties, the splines $s_0[f](t)$ have a smoothing capability. For computation of the spline's values $s_0[f](t)$ at the interval $[t[k], t[k+1]]$, four grid samples $f[k-1]$, $f[k]$, $f[k+1]$ and $f[k+2]$ are needed. Therefore, the spline can be computed in real time with a delay of one sample. Consequently, these splines can be utilized for real-time approximation and smoothing of signals from non-uniformly-spaced samples.

3.2 Quasi-interpolating cubic splines

3.2.1 B-spline representation of quasi-interpolating cubic splines

Cubic splines are able to restore polynomials of degree not exceeding three. This approximation order is provided by splines that interpolate the signal $\mathbf{f}$ on the grid $\mathbf{g} = \{t[k]\}$ ([10]). To derive the coefficients $q[\nu]$ in Eq. (10) for the spline which interpolates $\mathbf{f}$ at some interval, a system of linear equations with a three-diagonal matrix should be solved (see [8, 16], for example).

However, this approximation order can be achieved by a local spline which uses six grid samples $f[k-2], ..., f[k+3]$ for the computation of the spline's values at the interval $(t[k], t[k+1])$. Denote by

$$\begin{aligned} \beta_{-1}[k] &\stackrel{\text{def}}{=} \frac{-(h[k])^2}{3\,h[k-1]\,(h[k-1]+h[k])}, \quad \beta_1[k] \stackrel{\text{def}}{=} \frac{-(h[k-1])^2}{3\,h[k]\,(h[k-1]+h[k])}, \\ \beta_0[k] &\stackrel{\text{def}}{=} 1 - \beta_{-1}[k] - \beta_1[k], \end{aligned} \tag{12}$$

[1]This means preservation of monotonicity and convexity of a given data

where $h[k]$ is the grid step.

Proposition 3.5 ([21]). *If the coefficients in Eq. (10) are chosen as*

$$q_1[\nu] \stackrel{\text{def}}{=} \beta_{-1}[\nu]\, f[\nu-1] + \beta_0[\nu]\, f[\nu] + \beta_1[\nu]\, f[\nu+1],$$

then the spline

$$s_1[f](t) = \sum_{\nu\in\mathbb{Z}} q_1[\nu+2]\, b(t)[\nu] \tag{13}$$

restores cubic polynomials. For the computation of the spline's values $s_1[f](t)$, $t \in [t[k], t[k+1]]$, six grid samples $f[k-2], ..., f[k+3]$ are needed.

Proof: Proved by direct computation of the spline's values of the functions $f(t) \equiv 1$, $f(t) \stackrel{\text{def}}{=} t^r$, $r = 1, 2, 3$.■

Remark 3.1. *In the rest of the paper, we deal exclusively with the quasi-interpolating splines $s_1[f](t)$. Therefore, we drop the index $\cdot_1$. Thus, in the sequel $s[f](t) \stackrel{\text{def}}{=} s_1[f](t)$.*

3.2.2 Computation of quasi-interpolating cubic splines

We present a simple algorithm that computes the spline's values at the interval $(t[k], t[k+1])$, provided that the samples $f[k-2], ..., f[k+3]$ are available. Originally, it is introduced in [22]. The algorithm is based on the relation between the splines $s[f](t)$ and cubic interpolation polynomials.

We denote by $P(t)[k]$ the cubic polynomial which interpolates the function $f(t)$ at the grid points $\{t[k-1], t[k], ..., t[k+2]\}$. It is represented by

$$\begin{aligned} P(t)[k] &= f[k-1] + f[k-1,k](t - t[k-1]) + f[k-1,k,k+1]\,(t - t[k-1])\,(t - t[k]) \\ &+ f[k-1,k,k+1,k+2]\,(t - t[k-1])\,(t - t[k])\,(t - t[k+1]), \end{aligned} \tag{14}$$

where $f[k] = f(t[k])$.

Theorem 3.6. *For $t = t[k] + h[k]\,\tau$, $\tau \in [0,1]$, the cubic spline $s[f](t)$, which restores cubic polynomials, is expressed by*

$$s[f](t) = P(t)[k] + G[k]\,(1-\tau)^3 + F[k]\,\tau^3, \tag{15}$$

where the interpolating polynomial $P(t)[k]$ is presented in Eq. (14) and the coefficients $G[k]$ and $F[k]$ are

$$\begin{aligned} F[k] &\stackrel{\text{def}}{=} -f[k-1,k,k+1,k+2,k+3]\,\frac{(h[k])^2\,(h[k+1])^2\,(t[k+3]-t[k-1])}{3\,(t[k+2]-t[k])}, \\ G[k] &\stackrel{\text{def}}{=} -f[k-2,k-1,k,k+1,k+2]\,\frac{(h[k])^2\,(h[k-1])^2\,(t[k+2]-t[k-2])}{3\,(t[k+1]-t[k-1])} = F[k-1]. \end{aligned} \tag{16}$$

Proof: Due to Eq. (5), the reminder term of the interpolation is

$$R(t) \stackrel{\text{def}}{=} f(t) - P(t)[k] = f[t, k-1, k, k+1, k+2]\,(t - t[k-1])\,(t - t[k])\,(t - t[k+1])\,(t - t[k+2]).$$

Grid samples are $R(t[\nu]) = 0$ for $\nu = k-1, k, k+1, k+2$. For $\nu = k-2, k+3$, we have

$$\begin{aligned} R(t[k-2]) &= h[k-2]\,T[k]\,(t[k] - t[k-2])\,(t[k+1] - t[k-2]), \\ R(t[k+3]) &= h[k+2]\,T[k+1]\,(t[k+3] - t[k])\,(t[k+3] - t[k+1]), \\ T[k] &\stackrel{\text{def}}{=} f[k-2, k-1, k, k+1, k+2]\,(t[k+2] - t[k-2]). \end{aligned} \tag{17}$$

The spline $s[f](t)$ restores the polynomial $P(t)[k]$, therefore,

$$\begin{aligned} s[f](t) &= s[P](t) + s[R](t) = P(t)[k] + s[R](t) \\ &= P(t)[k] + \sum_{\nu=k-3}^{k} Q[\nu+2]\,b(t)[\nu], \\ Q[\nu+2] &= \beta_{-1}[\nu+1]\,R(t[\nu+1]) \\ &+ \beta_0[\nu+2]\,R(t[\nu+2]) + \beta_1[\nu+3]\,R(t[\nu+3]). \end{aligned}$$

Keeping in mind that $R(t[\nu]) = 0$ for $\nu = k-1, k, k+1, k+2$, we get

$$\begin{aligned} s[f](t) &= P(t)[k] + \beta_{-1}[k-1]\,R(t[k-2])\,b(t)[k-3] \\ &+ \beta_1[k+2]\,R(t[k+3])\,b(t)[k]. \end{aligned} \tag{18}$$

Equation (8) implies that for $t = t[k] + h[k]\,\tau,\ \tau \in [0,1]$,

$$\begin{aligned} b(t)[k] &= (t[k+4] - t[k])\,\frac{(t - t[k])^3}{w_4'[k](t[k])} = A_k\,\tau^3, \\ A_k &\stackrel{\text{def}}{=} \frac{(h[k])^2}{(t[k+2] - t[k])\,(t[k+3] - t[k])}. \end{aligned} \tag{19}$$

Equation (9) provides values of the B-spline $b(t)[k-3]$ for $t = t[k] + h[k]\,\tau,\ \tau \in [0,1]$. The explicit expression in Eq. (15) for the spline $s[f](t)$ is obtained by substitution of the B-splines samples from Eqs. (19) and (9), the coefficients β_j from Eq. (12) and the samples of $R(t)$ from Eq. (17) into Eq. (18). ■

When the grid $\mathbf{g}$ is uniform and $t[k] = hk,\ k \in \mathbb{Z}$, the finite differences are used instead of the divided ones to express the spline $s[f](t)$. In this case, when $t = h\,(k + \tau),\ \tau \in [0,1]$, the spline is represented by

$$\begin{aligned} s[f](t) &= f[k-1] + \Delta[\mathbf{f}][k-1]\,(1+\tau) + \frac{\Delta^2[\mathbf{f}][k-1]}{2}\,(1+\tau)\,\tau \\ &+ \frac{\Delta^3[\mathbf{f}][k-1]}{6}\,(1+\tau)\,\tau\,(\tau-1) - \frac{\Delta^4[\mathbf{f}][k-1]\,\tau^3 + \Delta^4[\mathbf{f}][k-2]\,(1-\tau)^3}{36}. \end{aligned} \tag{20}$$

For computation of the spline's values $s[f](t)$ at the interval $[t[k], t[k+1]]$, six grid samples $f[k-2], f[k-1], f[k], f[k+1], f[k+2]$ and $f[k+3]$, are needed. Thus, the spline can be computed in real time with a delay of two samples.

The representation of the spline $s[f](t)$ given in Eq. (15) is computationally efficient because it utilizes the well-established algorithms for the computation of interpolation polynomials. In addition, Eq. (15) provides a precise estimation for the approximation accuracy of the spline $s[f](t)$.

3.3 Approximation properties of quasi-interpolating cubic splines

Approximation properties of the spline $s[f](t)$ are close to the properties of the cubic spline $s_i[f](t)$ which interpolates the function $f(t)$ on the grid $\mathbf{g}$ such that $s_i[f](t[k]) = f(t[k]) = f[k]$. Like the interpolating splines, the splines $s[f](t)$ restore cubic polynomials.

Theorem 3.7 ([22]). *If the function $f(t)$ belongs to $C^4[I[k]]$, where $I[k] \stackrel{\text{def}}{=} [t[k-2], t[k+3]]$, then for $t \in [t[k], t[k+1]]\,, \quad t = t[k] + h[k]\,\tau$, the following error estimate is true:*

$$\max_{t\in[t[k],t[k+1]]} |f(t) - s[f](t)| \le \frac{43\,(\bar{h}[k])^4}{1152} \max_{t\in I[k]} \left|f^{(4)}(t)\right| \approx 0.0373\,(\bar{h}[k])^4 \max_{t\in I[k]} \left|f^{(4)}(t)\right|, \tag{21}$$

where $\bar{h}[k] \stackrel{\text{def}}{=} \max_{\nu=k-2,...,k+2} h[\nu]$. If the grid $\boldsymbol{g}$ is uniform, at least locally (that means that $h[\nu] = h$ for $\nu = k-2, ..., k+2$), then

$$\max_{t\in[t[k],t[k+1]]} |f(t) - s[f](t)| \le \frac{35\,h^4}{1152} \max_{t\in I[k]} \left|f^{(4)}(t)\right| \approx 0.0304\,h^4 \max_{t\in I[k]} \left|f^{(4)}(t)\right|. \tag{22}$$

35/1152 is the least possible constant in the inequality at Eq. (22).

Proof: In the Appendix.

Remark 3.2. *The estimation in Eq. (22) is exact in the sense that it becomes an identity for the function $f(t) = t^4$ when $t = h(k + 1/2)$.*

For comparison, a similar exact estimation for interpolating splines $s[f](t)$ defined on the interval $I \stackrel{\text{def}}{=} [a, b]$ is ([10]):

$$\max_{t\in I} |f(t) - s[f](t)| \le \frac{5\,\tilde{h}^4}{384} \max_{t\in I} \left|f^{(4)}(t)\right| \approx 0.0130\,\tilde{h}^4 \max_{t\in I} \left|f^{(4)}(t)\right|, \tag{23}$$

where $\tilde{h}$ is the maximal grid step at the interval I.

3.4 Quasi-interpolating cubic splines at finite intervals

Assume that only finite number of samples $f[\nu] = f(t[\nu])$, $\nu = 0, 1, ..., N$, of the function $f(t)$ are available. We construct the spline $\bar{s}[f](t)$ which quasi-interpolates the function $f(t)$ at the interval $[t[0], t[N]]$. As before, $P(t)[k]$ denotes the polynomial which interpolates the function $f(t)$ at the points $\{t[k-1], ..., t[k+2]\}$. At the inner interval $[t[2], t[N-2]]$, the representation of the spline $s[f](t)$ given in Eq. (15) is valid. We denote

$$\begin{aligned} A_0 &\stackrel{\text{def}}{=} -f[0,1,2,3,4]\,\frac{(h[2])^2\,(t[4]-t[0])}{3\,h[1]\,(t[3]-t[1])}, \\ A_N &\stackrel{\text{def}}{=} -f[N-4,N-3,N-2,N-1,N]\,\frac{(h[N-3])^2\,(t[N]-t[N-4])}{3\,h[N-2]\,(t[N-1]-t[N-3])}. \end{aligned} \tag{24}$$

Theorem 3.8. *The function*

$$\bar{s}[f](t) \stackrel{\text{def}}{=} \begin{cases} P(t)[1] + A_0\,(t - t[1])_+^3, & \text{as } t \in [t[0], t[2]]; \\ s[f](t), & \text{as } t \in [t[2], t[N-2]]; \\ P(t)[N-2] + A_N\,(t[N-1] - t)_+^3, & \text{as } t \in [t[N-2], t[N]]; \end{cases} \tag{25}$$

is a cubic spline that quasi-interpolates the function $f(t)$ on the finite grid $\mathbf{g}_N = \{t[\nu]\}$, $\nu = 0, 1, ..., N$.

Proof: If $t = t[2] + h[2]\,\tau,\ \tau \in [0.1]$, then $s[f](t) = P(t)[2] + G[2]\,(1-\tau)^3 + F[2]\,\tau^3$, where $G[k]$ and $F[k]$ are given in Eq. (16). However, for spline extension to $[t[0], t[2]]$, we utilize the polynomial $P(t)[1]$ and represent the spline by

$$s[f](t) = P(t)[1] + s[\tilde{R}](t), \quad \tilde{R}(t) \stackrel{\text{def}}{=} f(t) - P(t)[1]. \tag{26}$$

An additional grid point $t[-1] < t[0]$ is introduced. Then, the spline $s[\tilde{R}](t) = \sum_{\nu=-1}^{2} \tilde{q}[\nu + 2]\,b(t)[\nu]$. Due to Proposition 3.1, the location of the point $t[-1]$ does not affect the spline's values at the interval $[t[2], t[3]]$.

We need to know the behavior of the spline $s[\tilde{R}](t)$ as $t \to t[2] + 0$. The reminder term $\tilde{R}(t)$ vanishes at the points $\{t[0], ..., t[3]\}$. Therefore, the coefficients are

$$\tilde{q}[1] = \tilde{q}[2] = 0, \quad \tilde{q}[3] = \beta_1[3]\,\tilde{R}(t[4]), \quad \tilde{q}[4] = \beta_0[4]\,\tilde{R}(t[4]) + \beta_1[4]\,\tilde{R}(t[5]),$$

where $\beta_i[\nu]$ are defined in Eq. (12). When $t \in [t[2], t[3]]$, the B-splines $b(t)[1](t) = \alpha\,(t - t[1])^3 + \gamma\,(t - t[2])^3$ and $b(t)[2](t) = \delta\,(t - t[2])^3$, where α, γ and δ are constants. Consequently, at this interval, the spline is structured as $s[\tilde{R}](t) = A_0\,(t - t[1])^3 + B\,(t - t[2])^3$. When $t = t[2]$, we have

$$s[\tilde{R}](t[2]) = s[f](t[2]) - P(t[2])[1] = s[f](t[2]) - f[2] = A_0\,(h[1])^3. \tag{27}$$

Now, using Eqs. (15) and (16), we get

$$A_0 = \frac{s[f](t[2]) - f[2]}{(h[1])^3} = \frac{G[2]}{(h[1])^3} = -f[0,1,2,3,4]\,\frac{(h[2])^2\,(t[4] - t[0])}{3\,h[1]\,(t[3] - t[1])}. \tag{28}$$

In addition to Eq. (27), the following relations for the derivatives hold:

$$s'[\tilde{R}](t[2]) = 3A_0\,(h[1])^2, \quad s''[\tilde{R}](t[2]) = 6A_0\,h[1]. \tag{29}$$

We define the function $\varphi_0(t) \stackrel{\text{def}}{=} P(t)[1] + A_0\,(t - t[1])_+^3$. This function consists of two pieces of cubic polynomials, which are glued at the point $t[1]$ such that $\varphi_0(t) \in C^2[t[0], t[2]]$. On the other hand,

$$\begin{aligned} \varphi_0(t[2]) &= P(t[2])[1] + A_0\,(h[1])^3 = f[2] + A_0\,(h[1])^3 = s[f](t[2]), \\ \varphi_0^{'}(t[2]) &= P(t[2])^{'}[1] + 3A_0\,(h[1])^2 = s[f]^{'}(t[2]), \\ \varphi_0^{(2)}(t[2]) &= P(t[2])^{(2)}[1] + 6A_0\,(h[1])^2 = s[f]^{(2)}(t[2]). \end{aligned}$$

Therefore, the function

$$\hat{s}[f](t) \stackrel{\text{def}}{=} \begin{cases} \varphi_0(t), & \text{as } t \in [t[0], t[2]]; \\ s[f](t), & \text{as } t \in [t[2], t[N-2]] \end{cases}$$

is a cubic spline.

The design for the extension of the spline to $[t[N-2], t[N]]$ is similar to the design at the left hand side of the interval. ■

Remark 3.3. *The spline $\bar{s}[f](t)$ defined by Eq. (25) interpolates the function $f(t)$ at the points $t[0]$, $t[1]$ and $t[N-1]$, $t[N]$. At the intervals $[t[0],\, t[1]]$ and $[t[N-1],\, t[N]]$, the spline $\bar{s}[f](t)$ coincides with the interpolating polynomials $P(t)[1]$ and $P(t)[N-2]$, respectively.*

Theorem 3.9. *If a function $f(t) \in C^4[I_0]$, where $I_0 \stackrel{\text{def}}{=} [t[0], t[4]]$, then for $t \in [t[1], t[2]]$, the following error estimation is true:*

$$\max_{t\in[t[1],t[2]]} |f(t) - \bar{s}[f](t)| \le \frac{\bar{h}^4}{18} \max_{t\in I_0} \left|f^{(4)}(t)\right| \approx 0.0556\, \bar{h}^4 \max_{t\in I_0} \left|f^{(4)}(t)\right|, \tag{30}$$

where $\bar{h} \stackrel{\text{def}}{=} \max_{\nu=0,1,2,3} h[\nu]$. If the grid $\boldsymbol{g}$ is uniform, at least locally (it means that $h[\nu] = h$ for $\nu = 0, 1, 2, 3$), then

$$\max_{t\in[t[1],t[2]]} |f(t) - \bar{s}[f](t)| \le h^4\, \frac{16 - 3\sqrt{2}}{288\sqrt{2}} \max_{t\in I_0} |f^{(4)}(t)|. \tag{31}$$

The constant $(16 - 3\sqrt{2})/288\sqrt{2} \approx 0.0289$ is the least possible in the inequality at Eq. (31).

For $t \in [t[0], t[1]]$, the following error estimation is true:

$$\max_{t\in[t[0],t[1]]} |f(t) - \bar{s}[f](t)| \le \frac{\bar{h}^4}{24} \max_{t\in I_0} |f^{(4)}(t)|. \tag{32}$$

The estimation is exact even for the uniform grid. It becomes an identity for the function $f(t) = t^4$. Similar estimations hold at the interval $[t[N-2], t[N]]$.

Proof: In the Appendix.

3.5 Extrapolation of signals using cubic splines

Assume that a continuous function $f(t)$ exists on the interval $[t[0], t[N+1]]$ but only the samples $\{f[\nu] = f(t[\nu]\}$, $\nu = 0, ..., N$, are available. In order to approximate the function $f(t)$ on the interval $\vec{I} \stackrel{\text{def}}{=} [t[N], t[N+1]]$ and, in particular, to "predict" the sample $f[N+1]$, we extend the quasi-interpolating spline $\bar{s}[f](t)$ constructed in Theorem 3.8 to the interval $\vec{I}$.

Define the extended spline $\vec{s}[f](t)$ by

$$\vec{s}[f](t) \stackrel{\text{def}}{=} \begin{cases} \bar{s}[f](t), & \text{as } t \in [t[0], t[N]]\ ; \\ P(t)[N-2] + \vec{A}\,(t - t[N])^3, & \text{as } t \in \vec{I} \end{cases} \tag{33}$$

and choose the constant $\vec{A}$ such that, under some conditions, the difference $\vec{\Delta} \stackrel{\text{def}}{=} f(t[N+1]) - \vec{s}[f](t[N+1])$ is minimal. The function $\vec{s}[f](t)$ is a cubic spline because it is continuous together with its first and second derivatives at the point $t[N]$.

The difference $\vec{\Delta}$ is

$$\begin{aligned} \vec{\Delta} &= f(t[N+1]) - P(t[N+1])[N-2] - \vec{A}\,(h[N])^3 \\ &= \vec{C}\, f[N-3, N-2, N-1, N, N+1] - \vec{A}\,(h[N])^3, \\ \vec{C} &\stackrel{\text{def}}{=} (t[N+1] - t[N-3])\,(t[N+1] - t[N-2])\,(t[N+1] - t[N-1])\, h[N]. \end{aligned}$$

We choose $\vec{A} \stackrel{\text{def}}{=} \vec{C}\, f[N-4, N-3, N-2, N-1, N]/(h[N])^3$. Then, the difference becomes

$$\begin{aligned}\vec{\Delta} &= \vec{C}\,(f[N-3, N-2, N-1, N, N+1] - f[N-4, N-3, N-2, N-1, N]) \\ &= \vec{C}\, f[N-4, N-3, N-2, N-1, N, N+1]\,(t[N+1] - t[N-4]).\end{aligned}$$

Denote $\vec{h} \stackrel{\text{def}}{=} \max_{\nu=N-3,\ldots,N} h[\nu]$.

Proposition 3.10. *If the fifth-order difference satisfies* $|f[N-4, N-3, N-2, N-1, N, N+1]| \le F$, *then the extrapolation error at the point* $t[N+1]$ *is estimated by*

$$|f(t[N+1]) - \vec{s}[f](t[N+1])| \le 120F\,\vec{h}^5.$$

If the function $f(t)$ *has a continuous fifth-order derivative at the interval* $[t[N-4], t[N+1]]$, *then the estimation*

$$|f(t[N+1]) - \vec{s}[f](t[N+1])| \le \max_{t\in[t[N-4],t[N+1]]} |f^{(5)}(t)|\,\vec{h}^5 \tag{34}$$

is true. In particular, if $f(t)$ *coincides with a fourth-degree polynomial on the interval* $[t[N-4], t[N+1]]$, *then* $f(t[N+1]) = \vec{s}[f](t[N+1])$.

A similar design is carried out at the left hand side of the sampling interval, In order to extrapolate the spline $\bar{s}[f](t)$ defined on the grid $t[0], \ldots t[N]$ to a point $t[-1] < t[0]$, we define the new spline as follows:

$$\acute{s}[f](t) \stackrel{\text{def}}{=} \begin{cases} \bar{s}[f](t), & \text{as } t \in [t[0], t[N]]\ ; \\ P(t)[1] + \acute{A}\,(t - t[0])^3, & \text{as } t \in \acute{I}, \end{cases} \tag{35}$$

where the interval $\acute{I} \stackrel{\text{def}}{=} [t[-1], t[0]]$ and

$$\acute{A} = -f[0, 1, 2, 3, 4]\,(t[1] - t[-1])\,(t[2] - t[-1])\,(t[3] - t[-1])/h[-1]^2. \tag{36}$$

3.6 Remarks on the real-time spline implementation

Due to the fact that up to 6 adjacent grid samples are needed for the computation of the spline $s[f]$ at a point t, the spline can be computed in real time. It means that the spline follows the samples that arrive one after another at randomized times.

Assume that the samples of a function $f(t)$, $\{f[k]\}$, $k = 0, \ldots, N$, are available where $f[k] = f(t[k])$. Then, the spline $s[f](t)$ can be designed on the interval $[t[0], t[N]]$ in line with the scheme described in Sections 3.2 and 3.4 in the following way:

- At the inner subinterval $[t[2], t[N-2]]$, the spline is constructed by a regular 6-samples based algorithm from Section 3.2, while at the intervals $[t[0], t[2]]$ and $[t[N-2], t[N]]$ the 5-samples based extension formulas from Section 3.4 are utilized.

- When the sample $f[N+1] = f(t[N+1])$ arrives, the spline at the interval $[t[N-2], t[N-1]]$ is recomputed by utilizing 6 samples $\{f[k]\}$, $k = N-4, \ldots, N+1$. The spline at the interval $[t[N-1], t[N]]$, which was constructed by using 5 samples $\{f[k]\}$, $k = N-4, \ldots, N$, is recomputed with the new set of samples $\{f[k]\}$, $k = N-3, \ldots, N+1$. The spline is extended to the interval $[t[N], t[N+1]]$ by using the samples $\{f[k]\}$, $k = N-4, \ldots, N+1$. The spline remains unchanged at the interval $[t[0], t[N-2]]$.

- When the sample $f[N+2] = f(t[N+2])$ arrives, the spline at the interval $[t[N-1], t[N]]$ is recomputed by utilizing 6 samples $\{f[k]\}$, $k = N-3, ..., N+2$. The spline at the interval $[t[N], t[N+1]]$, which was constructed by using 5 samples $\{f[k]\}$, $k = N-3, ..., N+1$, is recomputed with the new set of samples $\{f[k]\}$, $k = N-2, ..., N+2$. The spline is extended to the interval $[t[N+1], t[N+2]]$ by using samples $\{f[k]\}$, $k = N-3, ..., N+2$. The spline remains unchange aAt the interval $[t[0], t[N-1]]$.

Remark 3.4. *Note that the arrival of two additional samples $f[N+1]$ and $f[N+2]$ leads to the re-computation at the interval $[t[N-2], t[N]]$ of the initial spline, which was defined at $[t[0], t[N]]$, while the spline remains unchanged at the interval $[t[0], t[N-2]]$.*

The above scheme is illustrated in Example 3 in Section 3.7.

3.7 Examples

Example 1: Restoration of the *chirp* function: In this section, we compare the restoration of the chirp function $\sin(1/t)$ by the cubic local spline $s[f](t)$ from uniformly spaced samples on the interval [0.013, 0.14] with the restoration from samples taken on a non-uniform grid. The chirp function is displayed in Fig. 2. The grid is adapted to the variable-frequency oscillations of the chirp function.

Figure 3 illustrates the difference between the performances of the local splines $s[f](t)$, which are constructed either on uniform or non-uniform grids. For the design of the splines, we use 50 samples $f[k] = f(t[k])$, $k = 0, ..., 49$, which are either equally spaced or their steps are growing quadratically such that $t[k] = t[0] + k^2\, h$.

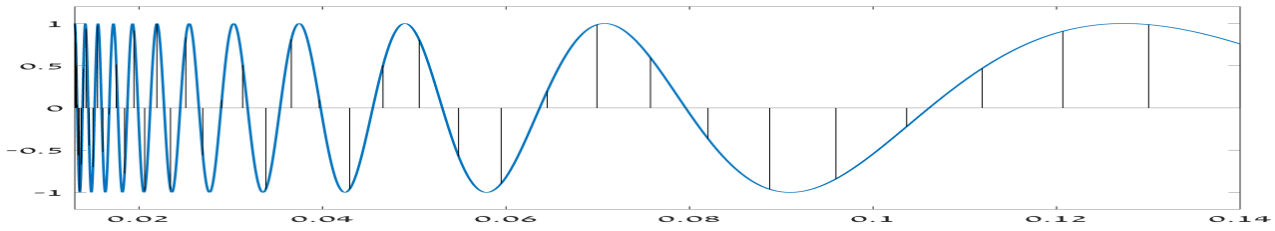

Figure 2: The chirp $\sin(1/t)$ on the interval [0.013, 0.14]

The upper plot in Fig. 3 displays the spline $s[f](t)$ designed on non-uniform grid. The bottom plot displays the spline $s[f](t)$ designed on a uniform grid.

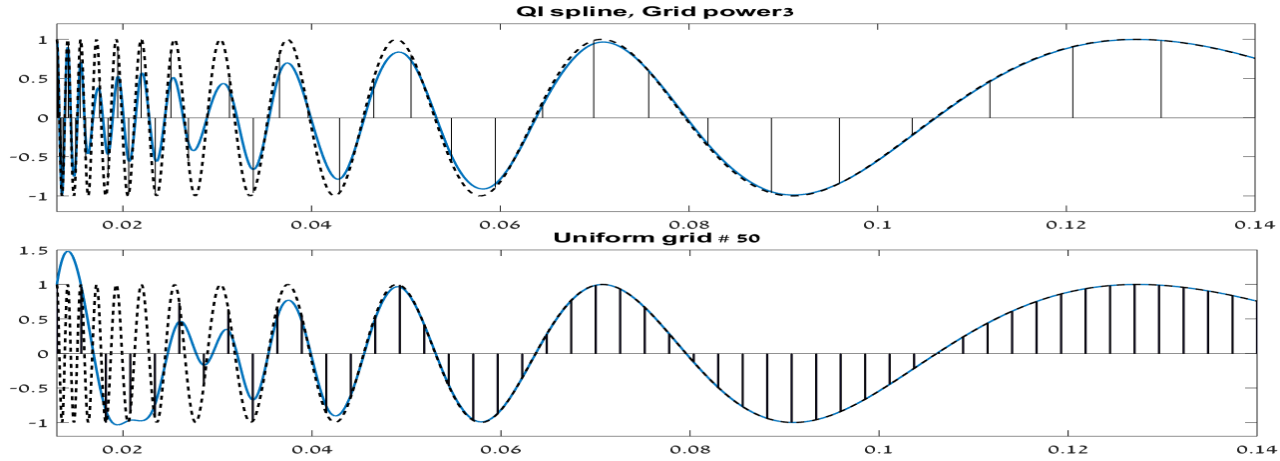

Figure 3: Top: restoration of the function $f(t) = \sin(1/t)$ from 50 non-equally spaced samples by the quasi-interpolating $s[f](t)$. Bottom: restoration by $s[f](t)$ from 50 equally spaced samples. Dashed line denotes the original function and bars denote the available samples. The splines $s[f](t)$ are displayed by solid lines

We observe that, when the grid is adapted to the function to be approximated, we get a near-perfect restoration of the function by the spline $s[f](t)$ (top picture). This is not the case for the spline designed on the uniform grid. It is failed to restore the high-frequency oscillations of the function, while sampling at the smooth part of the function is excessive. This advantage of the adapted grid over the uniform one is even more apparent when the restoration is derived from noised samples. The results of this restoration are shown in Fig. 4. Note that at the low-frequency intervals, the uniform-grid spline is not smooth unlike the adapted-grid spline.

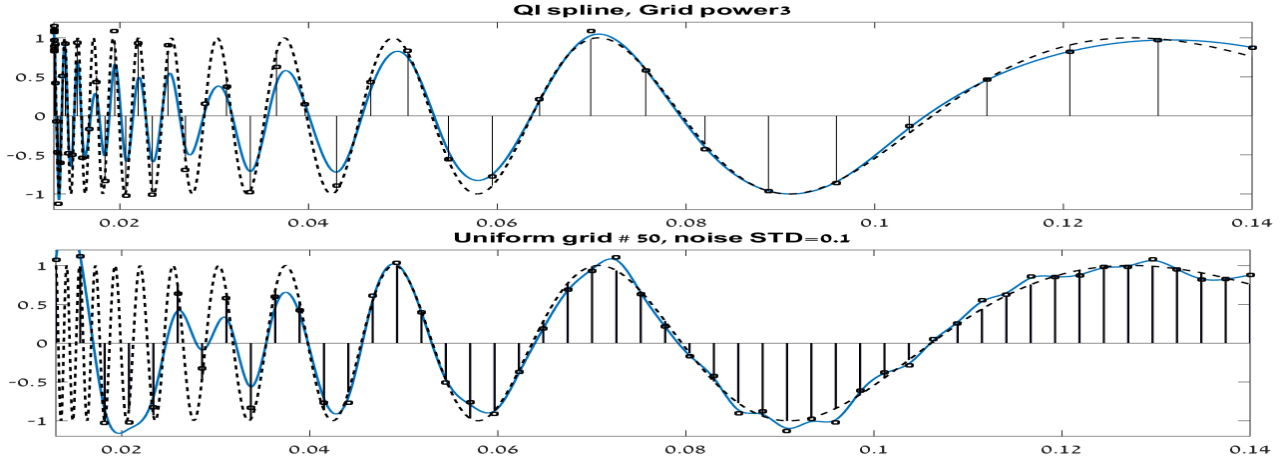

Figure 4: Top: restoration of the function $f(t) = \sin(1/t)$ from 50 non-equally spaced samples by the quasi-interpolating $s[f](t)$ affected by noise. Bottom: restoration by $s[f](t)$ from 50 equally spaced samples. Dashed line denotes the original function and bars denote samples of the original function. The splines $s[f](t)$ are displayed by solid lines, the squares denote the available samples

Figure 5 displays fragments from Fig. 4 that are related to high-frequency oscillations of the chirp function.

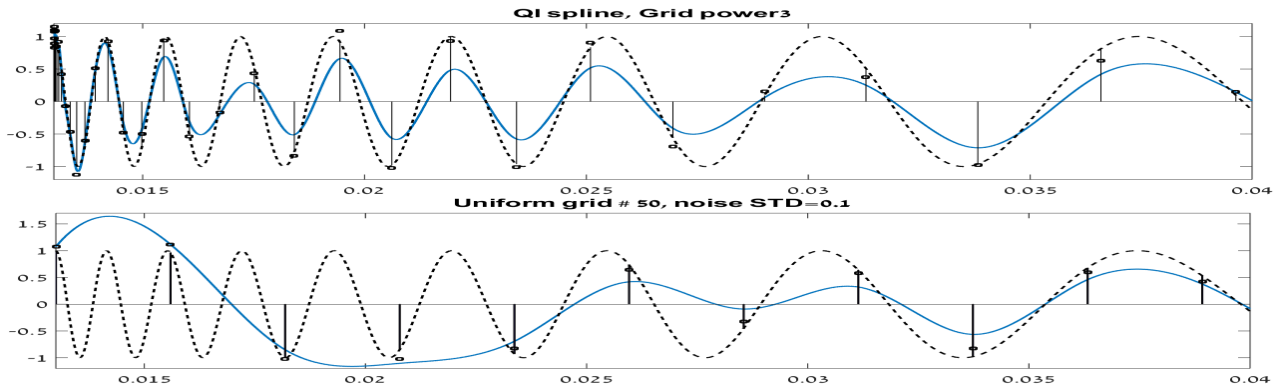

Figure 5: High-frequency fragments from Fig. 4

Example 2: Restoration of the *sine* function from randomly located samples: Figure 6 illustrates results from the restoration and extrapolation experiments of the function $\sin(t)+\sin 2t$ from 30 randomly spaced samples (top plot) and from uniformly spaced samples (bottom plot). In both cases, the first and the last samples were taken at $t[0] = 0.67$ and $t[N] = 19.23$ and the splines were extrapolated to $t[-1] = 0.3$ and $t[N+1] = 19.7$.

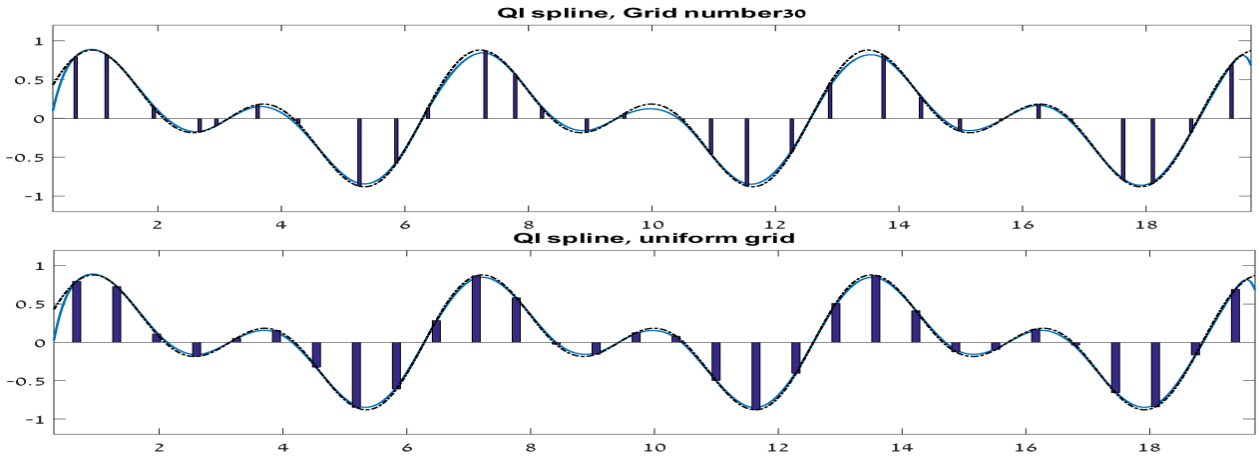

Figure 6: Top: restoration of the function $f(t) = \sin t + \sin 2t$ from 30 randomly spaced samples by the quasi-interpolating $s[f](t)$. Bottom: restoration by $s[f](t)$ from 30 equally spaced samples. Dash-dot lines denote the original function and bars denote the available samples. The splines $s[f](t)$ are displayed by solid lines

We observe that the splines at either grid restore the function near perfectly. However, the extrapolated samples differ from the original ones. It happens due to the fact that in both cases the discrepancy of the differences $f[N-3, N-2, N-1, N, N+1] - f[N-4, N-3, N-2, N-1, N]$

is essential. A similar situation exists at the left hand side of the grid. However, when the function $f(t)$ is a fourth-degree polynomial, the extrapolation is exact. It is displayed in Fig. 7. The function $f(t)$ is a fourth-degree polynomial. The spline is construcred using 6 randomly spaced grid points. The first and last samples were taken at $t[0] = 0.35$ and $t[N] = 9.9$ and the spline was extrapolated to $t[-1] = -3$ and $t[N+1] = 13$.

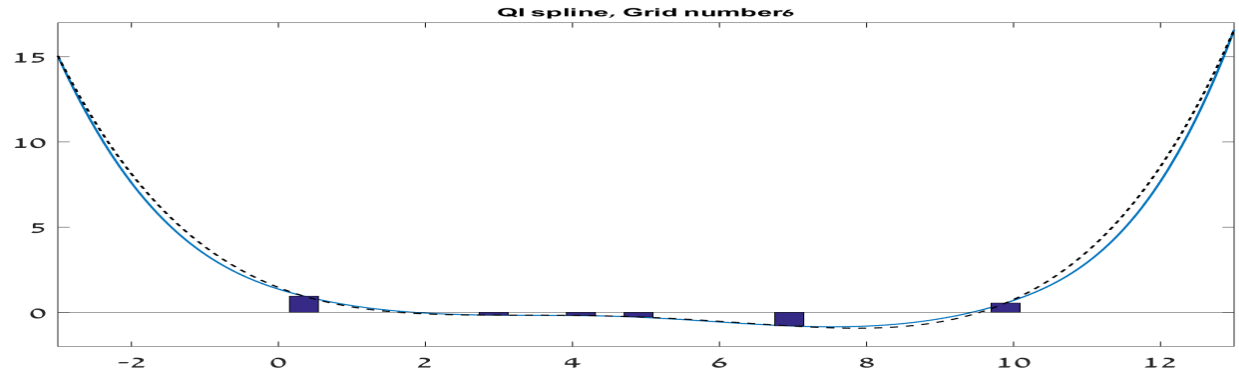

Figure 7: Extrapolation of the spline $s[f](t)$ from six grid points denoted by bars. Dashed line denotes the original function and solid line denotes the spline $s[f](t)$

Example 3: Illustration of the real-time spline implementation: Figure 8 illustrates the scheme for the real-time spline implementation when samples of a function to be approximated by the spline arrive sequentially at randomized times. The scheme was described in Section 3.6. Initially, the spline was implemented on the interval $[t[0], t[N]]$ where in this example $N = 9$. In the figure, it is shown by the dashed curve. On the arrival of the sample $f[N+1]$, the spline is extended to the interval $[t[N], t[N+1]]$. In the process, the spline is updated at the interval $[t[N-2], t[N-1]]$. The extended spline is denoted by a dash-dot curve. On the arrival of the sample $f[N+2]$, the spline is extended to the interval $[t[N+1], t[N+2]]$. In the process, the spline is updated at the interval $[t[N-1], t[N]]$. The extended spline is denoted by a solid line.

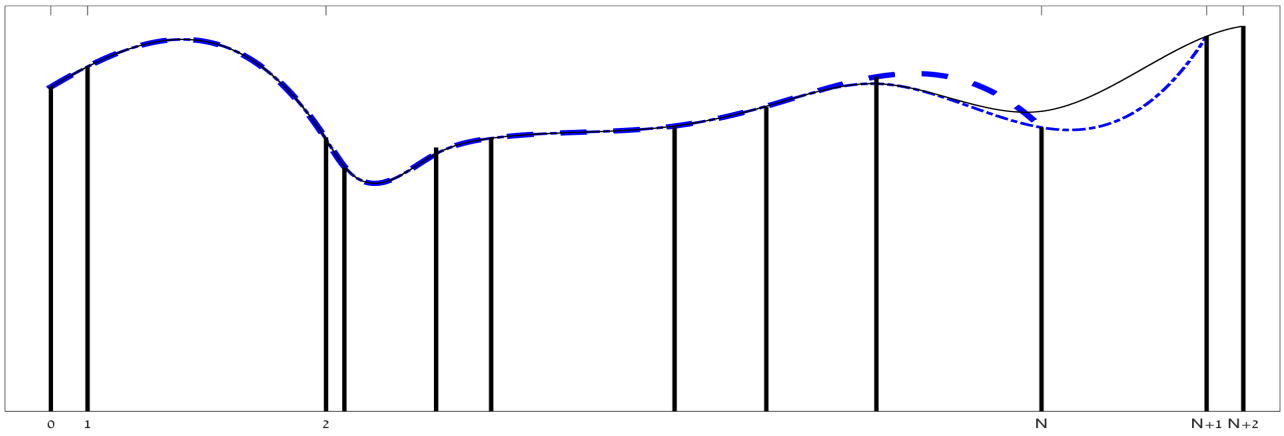

Figure 8: Dashed line: spline $s[f](t)$ is implemented on the interval $[t[0], t[N]]$. Dash-dot line: the spline is extended to the interval $[t[N], t[N+1]]$. Solid line: the spline is extended to the interval $[t[N+1], t[N+2]]$

4 Spline-based wavelet transform

The *Lifting Scheme*, introduced in [18, 19], is a method that constructs bi-orthogonal wavelet transforms and their efficient implementation. The main feature of the lifting scheme is that all the constructions are derived directly in the spatial domain and therefore can be custom designed to more general, irregular settings such as non-uniformly spaced data samples and bounded intervals. In this section, we describe the lifting scheme and how to use the quasi-interpolating local cubic splines, designed in Section 3, for the construction of wavelet transforms of non-equally sampled signals.

4.1 1-D discrete lifting wavelet transform

The 1D discrete lifting wavelet transform can be implemented either in a *primal* or in a *dual* mode. We discuss only the primal mode.

4.1.1 Primal decomposition

The lifting scheme for wavelet decomposition of signals from the space l_1 consists of four steps:

1. **Split**: The signal $\mathbf{f}$ is split into an even and odd sub-arrays such that $\mathbf{f} = \mathbf{e} \bigcup \mathbf{o}$ where $\mathbf{e} \stackrel{\text{def}}{=} \{e[k] = f[t[2k]] = f[2k]\}$ and $\mathbf{o} \stackrel{\text{def}}{=} \{o[k] = f[t[2k+1]] = f[2k+1]\}$.

2. **Predict**: The even array $\mathbf{e}$ is used to predict the odd array $\mathbf{o}$. Then, the array $\mathbf{o}$ is replaced by the array $\mathbf{d} = \mathbf{o} - \mathcal{P}_{[1]}\,\mathbf{e}$ where $\mathcal{P}_{[1]}$ is a linear $l_1 \to l_1$ *prediction* operator. If the predictor is correctly chosen, then this step decorrelates the signal and reveals its high-frequency components.

3. **Update(Lifting)**: The even array $\mathbf{e}$ is updated by using the new odd array $\mathbf{d}$. For this, we use a linear $l_1 \to l_1$ *update* operator $\mathcal{U}_{[1]}$, which is assumed to commute with $\mathcal{P}_{[1]}$ to get $\mathbf{a} = \mathbf{e} + \mathcal{U}_{[1]}\,\mathbf{d}$. Provided that the update operator is properly chosen, the even array $\mathbf{e}$ is transformed into downsampled and smoothed replica of $\mathbf{f}$.

4. **Normalization**: Finally, the smoothed $\mathbf{y}^0_{[1]}$ and the details $\mathbf{y}^1_{[1]}$ transform coefficient arrays, respectively, are obtained by the normalization $\mathbf{y}^0_{[1]} = \sqrt{2}\mathbf{a}, \quad \mathbf{y}^1_{[1]} = \mathbf{d}/\sqrt{2}$.

4.1.2 Primal reconstruction

Reconstruction of the signal $\mathbf{f}$ from the arrays $\mathbf{y}^0_{[1]}$ and $\mathbf{y}^1_{[1]}$ is implemented in a reverse order:

1. **Undo Normalization**: $\mathbf{a} = \mathbf{y}^0_{[1]}/\sqrt{2}, \quad \mathbf{d} = \sqrt{2}\mathbf{y}^1_{[1]}$.

2. **Undo Lifting**: The even array is restored by $\mathbf{e} = \mathbf{a} - \mathcal{U}_{[1]}\,\mathbf{d}$.

3. **Undo Predict**: The odd array component is restored by $\mathbf{o} = \mathbf{d} + \mathcal{P}_{[1]}\,\mathbf{e}$.

4. **Undo Split**: Restoration of the signal from its even and odd arrays by $\mathbf{f} = \mathit{Merge}\{\mathbf{e}, \mathbf{o}\}$.

The lifting transform is perfectly invertible with any choice of the operators $\mathcal{P}_{[1]}$ and $\mathcal{U}_{[1]}$. The direct and inverse transforms can be represented in a matrix of the form:

$$\begin{pmatrix} \mathbf{y}^0_{[1]} \\ \mathbf{y}^1_{[1]} \end{pmatrix} = \begin{pmatrix} \sqrt{2}\mathcal{I} & 0 \\ 0 & \mathcal{I}/\sqrt{2} \end{pmatrix} \begin{pmatrix} \mathcal{I} & \mathcal{U}_{[1]} \\ 0 & \mathcal{I} \end{pmatrix} \begin{pmatrix} \mathcal{I} & 0 \\ -\mathcal{P}_{[1]} & \mathcal{I} \end{pmatrix} \begin{pmatrix} \mathbf{e} \\ \mathbf{o} \end{pmatrix} = \tilde{\mathbf{M}}_{[1]} \begin{pmatrix} \mathbf{e} \\ \mathbf{o} \end{pmatrix},$$

where $\mathcal{I}$ is the identical operator and the "polyphase matrix" $\tilde{\mathbf{M}}_{[1]}$ analysis of the first decomposition level is

$$\tilde{\mathbf{M}}_{[1]} \stackrel{\text{def}}{=} \begin{pmatrix} \sqrt{2}\,(\mathcal{I} - \mathcal{U}_{[1]}\,\mathcal{P}_{[1]}) & \sqrt{2}\mathcal{U}_{[1]} \\ -\mathcal{P}_{[1]}/\sqrt{2} & \mathcal{I}/\sqrt{2} \end{pmatrix}.$$

The inverse transform is represented by

$$\begin{pmatrix} \mathbf{e} \\ \mathbf{o} \end{pmatrix} = \begin{pmatrix} \mathcal{I} & 0 \\ \mathcal{P}_{[1]} & \mathcal{I} \end{pmatrix} \begin{pmatrix} \mathcal{I} & -\mathcal{U}_{[1]} \\ 0 & \mathcal{I} \end{pmatrix} \begin{pmatrix} \mathcal{I}/\sqrt{2} & 0 \\ 0 & \mathcal{I}\sqrt{2} \end{pmatrix} \begin{pmatrix} \mathbf{y}^0_{[1]} \\ \mathbf{y}^1_{[1]} \end{pmatrix} = \mathbf{M}_{[1]} \begin{pmatrix} \mathbf{y}^0_{[1]} \\ \mathbf{y}^1_{[1]} \end{pmatrix},$$

where the "polyphase matrix" $\mathbf{M}_{[1]}$ synthesis of the first decomposition level is

$$\mathbf{M}_{[1]} \stackrel{\text{def}}{=} \begin{pmatrix} \mathcal{I}/\sqrt{2} & -\sqrt{2}\mathcal{U}_{[1]} \\ \mathcal{P}_{[1]}/\sqrt{2} & \sqrt{2}\,(\mathcal{I} - \mathcal{P}_{[1]}\,\mathcal{U}_{[1]}) \end{pmatrix}.$$

It is readily seen that once the operators $\mathcal{P}_{[1]}$ and $\mathcal{U}_{[1]}$ commute with each other, we have $\mathbf{M}_{[1]}\,\tilde{\mathbf{M}}_{[1]} = \mathcal{I}$. Therefore, subsequent applications of the operators $\tilde{\mathbf{M}}_{[1]}$ and $\mathbf{M}_{[1]}$ to the vector-signal $(\mathbf{e}, \mathbf{o})^T$ restores this vector-signal.

4.2 1-D multilevel wavelet transform

The multilevel wavelet transform is achieved by an iterated application of the lifting operations to the smoothed coefficient arrays. The predict and update operators can be different for different decomposition levels. Figure 9 shows a diagram of the 2-level 1-D discrete wavelet transform.

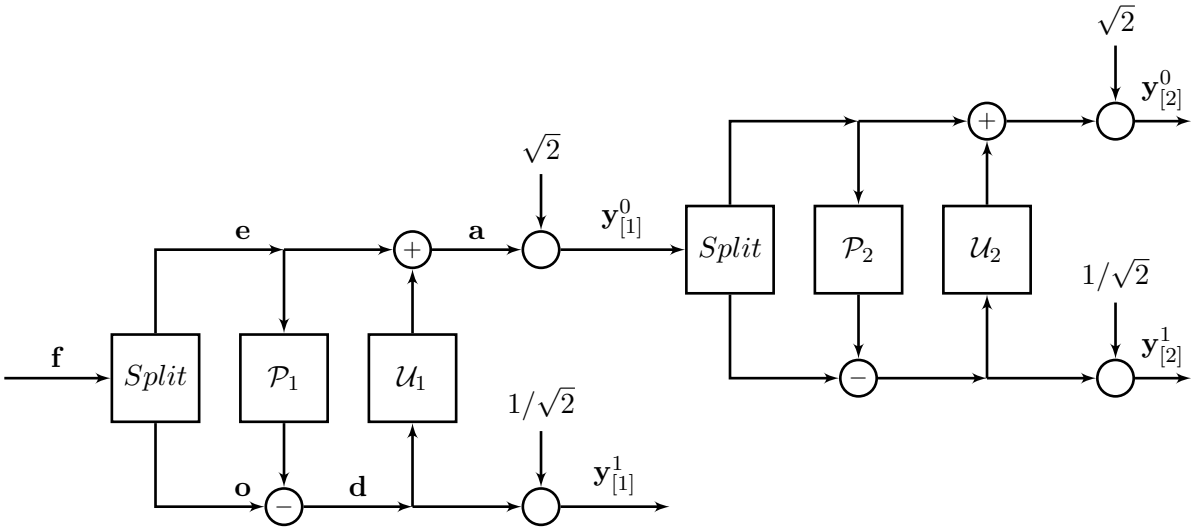

Figure 9: Two level Discrete Lifting Wavelet Transform.

Reconstruction of the signal $\mathbf{f}$ from the coefficient array $\{\mathbf{y}^0_{[m]} \bigcup \mathbf{y}^1_{[m]} \bigcup \cdots \mathbf{y}^1_{[1]}\}$ is implemented in a reverse order.

5 Spline-based predict and update operators

The wavelets generated by the lifting scheme are determined by the choice of the predict $\mathcal{P}$ and update $\mathcal{U}$ operators. Splines provide flexible tool for the design of such operators. The main idea is to construct a spline on the even grid $t[2k]$, which either interpolates or quasi-interpolates the samples of the signal $\mathbf{e}$ - see Fig. 9. Then, the values of this spline at the odd grid points $t[2k+1]$ are computed. These values are used for the prediction of the odd samples $\mathbf{o}$. The next step is to construct a spline on the odd grid points, which interpolates or quasi-interpolates the samples of the prediction error signal $\mathbf{d}$. Then the values of this splines at the even grid points are computed. These values are used for updating the even samples $\mathbf{e}$.

In the case of equally spaced samples, calculations are reduced to low-pass filtering of the corresponding arrays [4]. Application of wavelet transforms to signals sampled on finite grids and, in particular, to images, requires to extend the signals beyond their boundaries, otherwise distortion appears near the boundaries [5]. Various extension schemes have been developed to deal with the boundaries effects of finite length signals: zero padding, periodic extension and symmetric extension are basic extension methods. However, quasi-interpolating splines on finite intervals designed in Section 3.4, make it possible to implement wavelet transforms of signals sampled on bounded intervals without extending the signals beyond their boundaries.

5.1 Predict and update operators

The prediction and update operations are reduced to the design of the splines $s[f](t)$ on different grids and computation of their values at intermediate points. It can be done in a fast way using algorithms described in Section 3. However, in order to highlight the structure of the operators, we represent them in a matrix form.

5.1.1 Predict and update operators on unlimited grids

For this, we use a standard representation of the spline given in Eq. (13):

$$s[f](t) = \sum_{\nu\in\mathbb{Z}} q_1[\nu+2]\, b(t)[\nu] \tag{37}$$
$$= \sum_{\nu\in\mathbb{Z}} \left(\beta_{-1}[\nu+2]\, f[\nu+1] + \beta_0[\nu+2]\, f[\nu+2] + \beta_1[\nu+2]\, f[\nu+3]\right)\, b(t)[\nu] = \sum_{\nu\in\mathbb{Z}} f[\nu+2]\, B(t)[\nu],$$

where the coefficients $\beta_i[k]$ are defined by Eq. (12) and the cubic spline $B(t)[k]$ is

$$B(t)[k] \stackrel{\text{def}}{=} \beta_{-1}[k+3]\, b(t)[k+1] + \beta_0[k+2]\, b(t)[k] + \beta_1[k+1]\, b(t)[k-1]. \tag{38}$$

The spline $B(t)[k]$ is supported on the interval $(t[k-1], t[k+5])$. Therefore, if $t \in [t[k], t[k+1]]$, then the sum in Eq. (37) comprises of only 6 terms:

$$s[f](t) = \sum_{\nu=k-4}^{k+1} f[\nu+2]\, B(t)[\nu]. \tag{39}$$

As before, $\mathbf{e} = \{e[k] = f[2k]\}$ and $\mathbf{o} = \{o[k] = f[2k+1]\}$. Denote by $b_e(t)$ and $B_e(t)$ the B-spline $b(t)$ and the spline $B(t)$, respectively, defined on the even sub-grid $\mathbf{g}_e \stackrel{\text{def}}{=} \{t[2k]\}$ and by $b_o(t)$ and $B_o(t)$ the splines are defined on the odd sub-grid $\mathbf{g}_o \stackrel{\text{def}}{=} \{t[2k+1]\}$. Equation (39) implies that the values at odd grid points of the spline $s[e](t)$, which is constructed on the even sub-grid, are

$$s[e](t[2k+1]) = \sum_{\nu=k-4}^{k+1} f[2(\nu+2)]\, B_e(t[2k+1])[2\nu] = \sum_{\nu=k-4}^{k+1} P_{[1]}[k,\nu]\, e[\nu], \tag{40}$$
$$P_{[1]}[k,\nu] \stackrel{\text{def}}{=} \begin{cases} B_e(t[2k+1])[2\nu], & \text{for } \nu = k-4, ..., k+1\ ; \\ 0, & \text{otherwise.} \end{cases}$$

The coefficients $\left\{P_{[1]}[k,\nu]\right\}$ form a six-diagonal matrix $\mathbf{P}_{[1]}$ where in the case of uniform grid, is a Toeplitz matrix. Consequently, the prediction operation can be represented as the matrix multiplication by $\mathcal{P}_{[1]}\,\mathbf{e} = \mathbf{P}_{[1]}\,\mathbf{e}$.

Similarly, the update operation can be represented as the matrix multiplication: $\mathcal{U}_{[1]}\{\mathbf{d}\} = \mathbf{U}_{[1]}\,\mathbf{d}$, where the six-diagonal matrix $\mathbf{U}_{[1]} = \left\{U_{[1]}[k,\nu]\right\}$

$$s[d](t[2k]) = \sum_{\nu=k-5}^{k} \left(f[2(\nu+2)+1] - s[e](t[2(\nu+2)+1])\right) B_o(t[2k])[2\nu+1] \tag{41}$$
$$= \sum_{\nu=k-5}^{k} U_{[1]}[k,\nu]\, d[\nu], \quad U_{[1]}[k,\nu] \stackrel{\text{def}}{=} \begin{cases} B_o(t[2k])[2\nu+1], & \text{for } \nu = k-5, ..., k\ ; \\ 0, & \text{otherwise.} \end{cases}$$

5.1.2 Predict and update operators on limited grids

Assume that the function $f[k] = f(t[k]),\ k = 0, ..., N$ is sampled on a bounded interval.

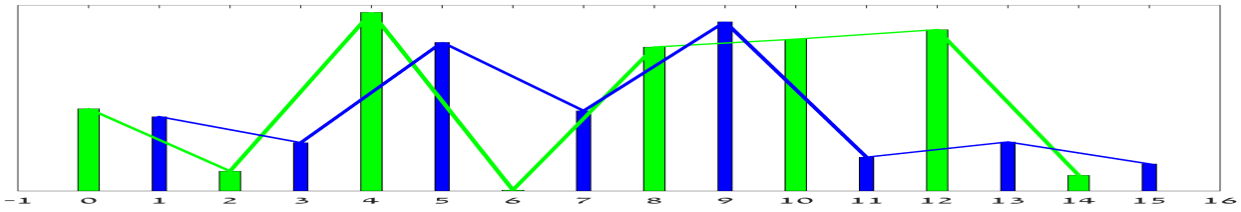

Figure 10: Random samples, N odd

N is an odd integer (Fig. 10):

Prediction: The odd samples $f[2k+1],\ k = 2, ..., (N-1)/2-3$, are predicted using the formulas in Eq. (40). The samples $f[1]$ and $f[3]$ as well as $f[N-4]$ and $f[N-2]$ are predicted by the values of the spline $s[e](t)$ at the respective grid points $t[1]$ and $t[3]$ as well as $t[N-4]$ and $t[N-2]$. These values are computed using the extension formulas in Eq. (25). The remaining sample $f[N]$ is predicted by the extrapolation of the spline $s[e](t)$ to the grid point $t[N]$ using the algorithm described in Section 3.5. Thus, we derive the differences $d[k] = f[2k+1] - s[e](t[2(k+2)+1]),\ k = 0, ..., (N-1)/2$.

Update: The even samples $f[2k],\ k = 3, ..., (N-1)/2-2$, are updated by the values of the spline $s[d](t[2k])$ using the formulas in Eq. (40). To update the samples $f[2]$, $f[4]$, $f[N-3]$, $f[N-1]$, the values of $s[d](t)$ are computed using the extension formulas in Eq. (25). The remaining sample $f[0]$ is predicted by the extrapolation of the spline $s[d](t)$ to the grid point $t[0]$ using the algorithm described in Section 3.5.

N is an even integer: (Fig. 11): At the left hand side of the grid, the operations are the same as before. The samples $f[N-3]$ and $f[N-1]$ are predicted by the values of the spline $s[e](t)$ that is computed by using the extension formulas in Eq. (25). The even samples $f[N-4]$ and $f[N-2]$ are updated by the values of the spline $s[d](t)$ using the formulas in Eq. (25), while the remaining sample $f[N]$ is updated by the extrapolation of the spline $s[d](t)$ to the grid point $t[N]$.

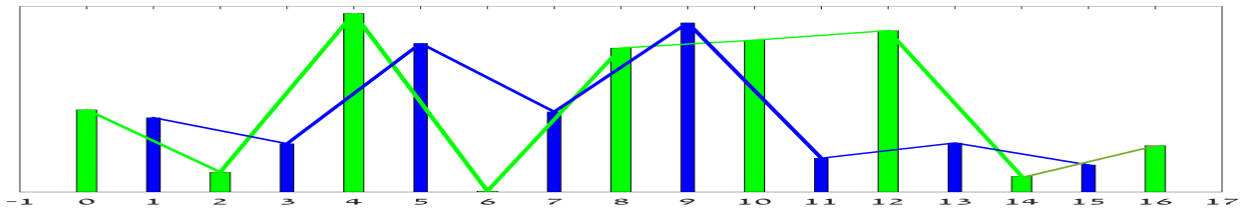

Figure 11: Random samples, N even

To implement the next step of the wavelet transform, the above operations are applied to the smooth array $\mathbf{y}^0_{[1]}$. As a result, the arrays $\mathbf{y}^0_{[2]}$ and $\mathbf{y}^1_{[2]}$ are produced. This process repeats itself.

The spline-based wavelet transform possesses the "vanishing moments" property, which is formulated next.

Proposition 5.1. *Assume that samples* $\{f[\nu]\}$, $\nu = 2k-4, ..., 2k+6$, *are samples of the cubic polynomial* $f[\nu] = P^3(t[\nu])$. *Then, the wavelet coefficient* $y^1_{[1]}[k] = 0$.

Proof: The first-level wavelet coefficients are $y^1_{[1]}[k] = (f[2k+1] - s[e](t[2k+1]))/\sqrt{2}$. Theorem 3.7 implies that this difference is zero. ■

Remark 5.1. *We emphasize that the described scheme of the wavelet transform does not require extension of the signal beyond the sampling boundaries. Theorem 3.9 and Proposition 3.10 imply that the "vanishing moments" property is valid near the boundaries of the interval.*

Figure 12 displays a randomly sampled signal and coefficients from its four-level wavelet transform. Note that the inverse wavelet transform from these coefficients provides a restoration of the signal where its maximal deviation, which is displayed in Fig. 13, is $1.5\,10^{-12}$.

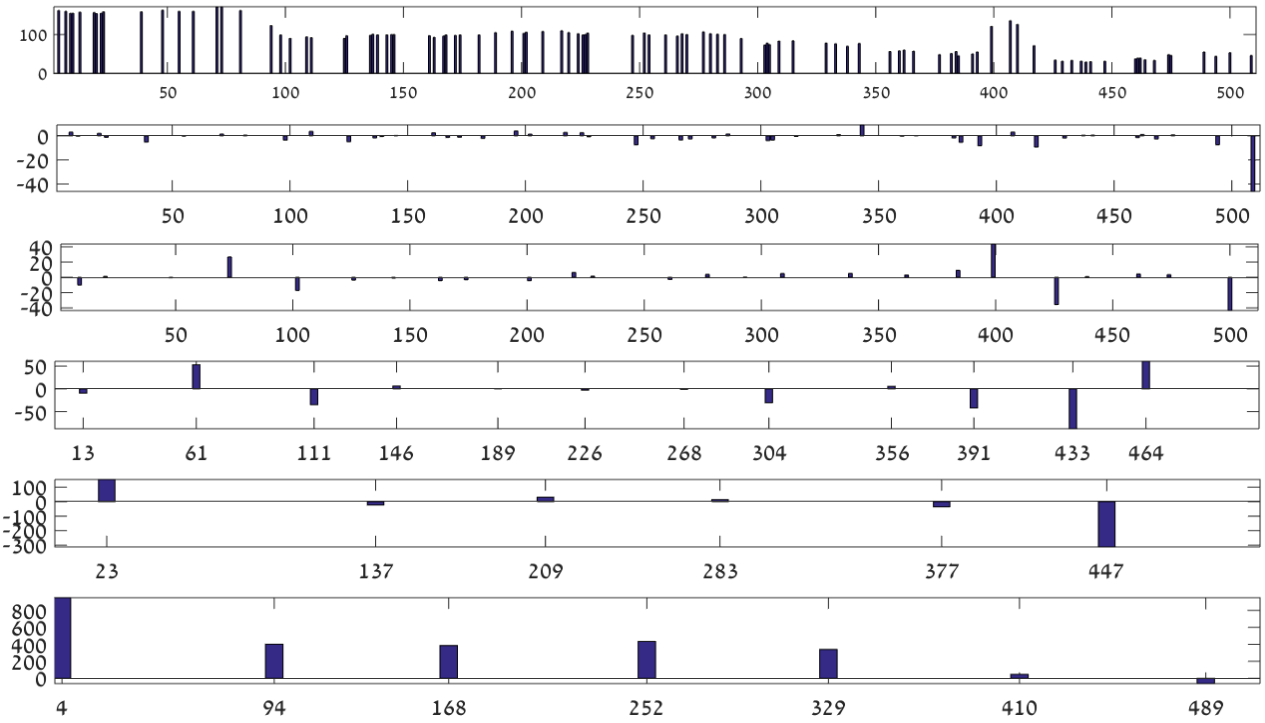

Figure 12: Wavelet transform of a randomly sampled signal. Top: the source signal. Bottom: smooth coefficients from the fourth decomposition level. The other 4 plots: the detail coefficients from first to fourth decomposition levels

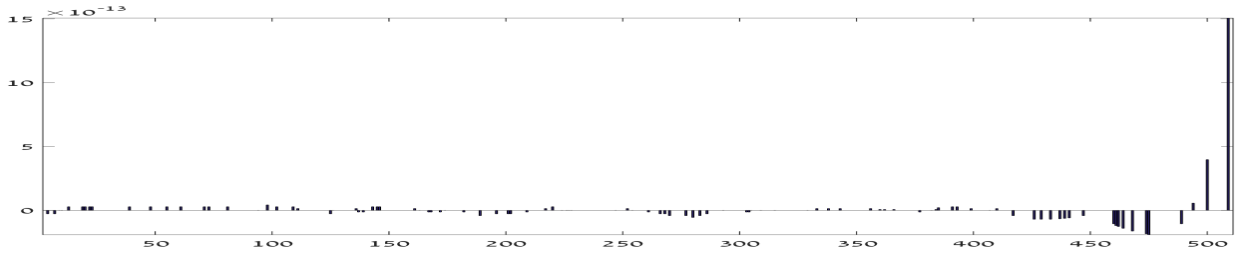

Figure 13: Discrepancy between the original signal and its restoration from the wavelet coefficients shown in Fig. 12. Maximal discrepancy is $1.5\,10^{-12}$

5.2 Real-time implementation of the wavelet transform

The transform scheme described in Section 5.1.2 makes it possible to implement the wavelet transform of a signal in real time. It means that the samples $f[\nu]$ of a signal arrive sequentially at randomized times $t[\nu]$, $\nu = 0, 1, ...$

- Assume that at time $t[N]$, the samples $f[\nu]$, $\nu = 0, ..., N$, arrived already and N is an odd number. Then, the one-level decomposition is implemented as was described in Section 5.1.2. Recall that a few of the initial coefficients $y^0_{[1]}[0]$, $y^0_{[1]}[1]$, $y^0_{[1]}[2]$ and $y^1_{[1]}[0]$, $y^1_{[1]}[1]$ are produced by using the extension formulas in Eq. (25) and the extrapolation algorithm. The coefficients $y^0_{[1]}[\nu]$, $\nu = 3, ..., (N-1)/2-2$, and $y^1_{[1]}[\nu]$, $\nu = 2, ..., (N-1)/2-3$, are computed by using the regular formulas in Eq. (40). The coefficients $y^0_{[1]}[(N-1)/2-1]$, $y^0_{[1]}[(N-1)/2]$, and $y^1_{[1]}[(N-1)/2-1]$, $y^1_{[1]}[(N-1)/2-2]$ are produced by using the extension formulas in Eq. (25). The coefficient $y^1_{[1]}[(N-1)/2]$ is derived by the extrapolation algorithm. The number $(N+1)/2$ of smooth coefficients $y^0_{[1]}[\nu]$ is the same as the number of the detail coefficients $y^1_{[1]}[\nu]$.

- When the sample $f[N+1]$ arrives, the new smooth coefficient $y^0_{[1]}[(N+1)/2]$ is derived by updating the even sample $f[N+1]$ using the extrapolation of the spline $s[d](t)$ to the grid point $t[N+1]$. In addition, the detail coefficient $y^1_{[1]}[(N-1)/2-2]$ is recomputed in a regular way by prediction from 6 even samples $f[2\nu]$, $\nu = (N-1)/2-4, ...(N+1)/2$. The rest of the detail and the smooth coefficients remain unchanged. The number of the smooth coefficients $y^0_{[1]}[\nu]$ is $(N+3)/2$ while the number of the detail coefficients $y^1_{[1]}[\nu]$ is $(N+1)/2$.

- When the sample $f[N+2]$ arrives, the new detail coefficient $y^1_{[1]}[(N+1)/2]$ is derived by predicting the odd sample $f[N+2]$ using the extrapolation of the spline $s[e](t)$ to the grid point $t[N+2]$. In addition, the smooth coefficient $y^0_{[1]}[(N-1)/2-1]$ is recomputed in a regular way by updating from 6 numbers $d[\nu]$, $\nu = (N-1)/2-4, ...(N+1)/2$. The rest of the detail and smooth coefficients remain unchanged. The number $(N+3)/2$ of smooth coefficients $y^0_{[1]}[\nu]$ is the same as the number of the detail coefficients $y^1_{[1]}[\nu]$.

At the same time, the above procedures are applied to the smooth coefficient array $\left\{y^0_{[1]}[\nu]\right\}$, $\nu = 0, ..., (N-1)/2$, to obtain the arrays $\left\{y^0_{[2]}[\nu]\right\}$ and $\left\{y^1_{[2]}[\nu]\right\}$, and so on.

Figure 14 illustrates the one-level wavelet transform of a signal consisting of ten randomly spaced samples when additional samples arrive.

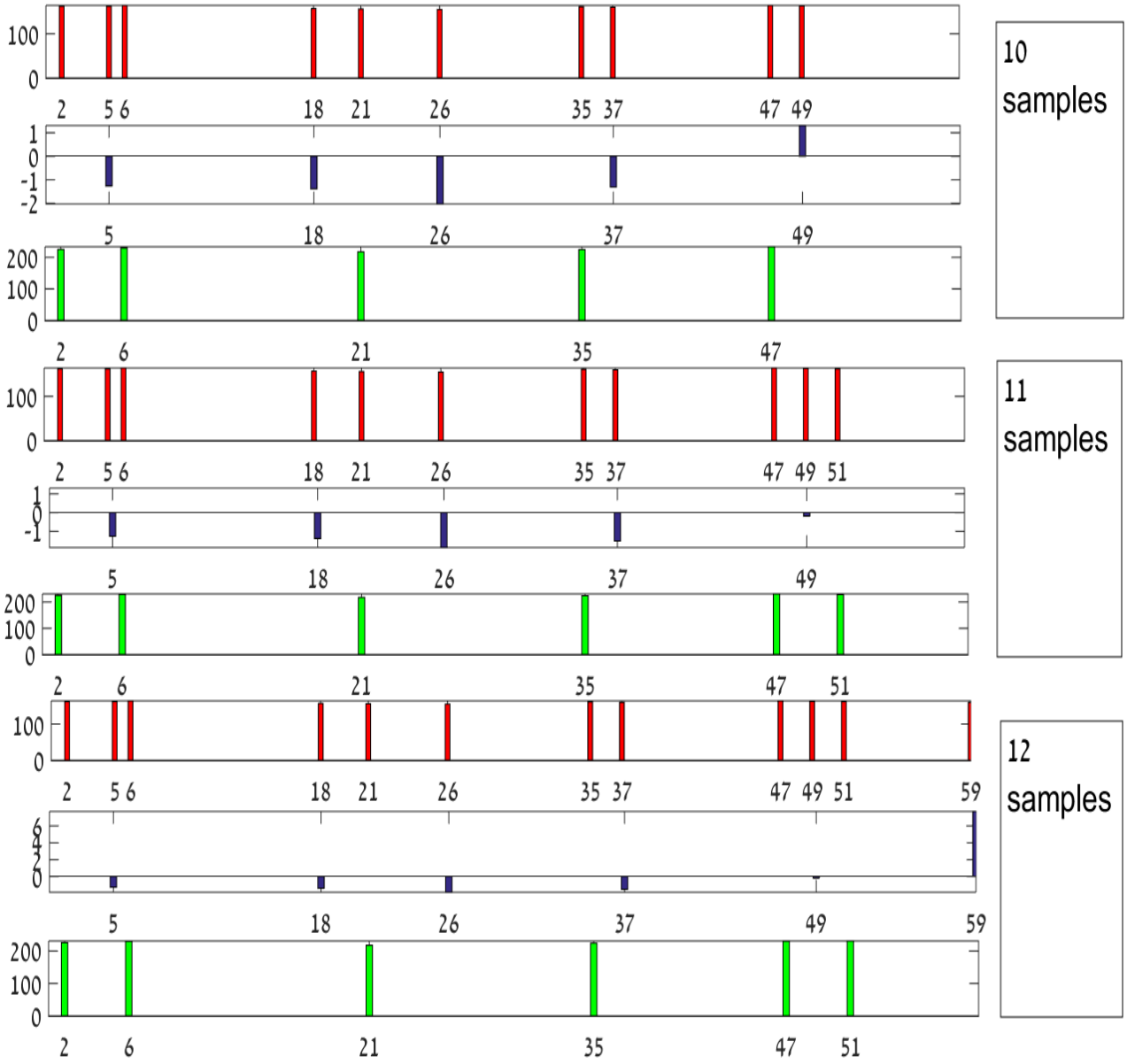

Figure 14: Wavelet transform of a randomly sampled signal. Top triple: red – 10 samples signal, blue – 5 detail coefficients $\left\{y^1_{[1]}[\nu]\right\}$, green – 5 smooth coefficients $\left\{y^0_{[1]}[\nu]\right\}$. Central triple: red – 11 samples signal, blue – 5 detail coefficients , green – 6 smooth coefficients.Bottom triple: red – 12 samples signal, blue – 6 detail coefficients , green – 6 smooth coefficients

Conclusion

The local cubic splines, which was designed in Section 3 and supplied with a simple fast computational algorithms, can serve as an efficient tool for the real-time signal processing. As an input, the splines use either clean or noised arbitrarily-spaced samples. An important application of the designed splines is its real-time wavelet analysis of non-uniformly sampled signals.

On the other hand, the capability to adapt the grid to the structure of an object with minimal requirements to the operating memory are great advantages for offline processing of signals and multidimensional data arrays. The above approach to splines' design enables us to smoothly extend the spline to the boundaries of the definition area and even to efficiently extrapolate the spline beyond this area.

In future work, the designed splines and the spline-based wavelets will be applied to online denoising and compression of practical signals such as, for example, ECG signals.

Appendix

Proof of Theorem 3.7: By using Eqs. (15) and (5), we can represent the remainder term of the approximation as follows:

$$\begin{aligned}
\rho(t) &\stackrel{\text{def}}{=} f(t) - s[f](t) = f(t) - P^3(t)[k] - G[k]\,(1-\tau)^3 + F[k]\,\tau^3 \\
&= f[t,k,k+1,...,k+3]\,\omega_3[k-1](t) \\
&+ f[k-1,k,k+1,k+2,k+3]\,\frac{(h[k])^2\,(h[k+1])^2\,(t[k+3]-t[k-1])}{3\,(t[k+2]-t[k])}\,\tau^3 \\
&+ f[k-2,k-1,k,k+1,k+2]\,\frac{(h[k])^2\,(h[k-1])^2\,(t[k+2]-t[k-2])}{3\,(t[k+1]-t[k-1])}\,(1-\tau)^3.
\end{aligned}$$

Due to Proposition 2.1, we have $\rho(t) = \frac{1}{24}\left(f^{(4)}(\xi)\alpha + f^{(4)}(\eta)\beta + f^{(4)}(\zeta)\gamma\right)$, where the numbers ξ, η and ζ belong to the interval $(t[k-2], t[k+3])$ and the coefficients are

$$\begin{aligned}
\alpha &= \omega_3[k-1](t) = (t-t[k-1])\,(t-t[k])\,(t-t[k+1])\,(t-t[k+2]) \geq 0, \\
\beta &= \frac{(h[k])^2\,(h[k+1])^2\,(t[k+3]-t[k-1])}{3\,(t[k+2]-t[k])}\,\tau^3 \geq 0, \\
\gamma &= \frac{(h[k])^2\,(h[k-1])^2\,(t[k+2]-t[k-2])}{3\,(t[k+1]-t[k-1])}\,(1-\tau)^3 \geq 0.
\end{aligned}$$

The intermediate value (Theorem 2.2) implies that

$$\begin{aligned}
\rho(t) &= \frac{1}{24} f^{(4)}(\vartheta)\,(\alpha+\beta+\gamma) = \frac{1}{24} f^{(4)}(\vartheta)\,H(\tau), \\
H(\tau) &= (h[k])^2\Big((h[k+1]+h[k](1-\tau))(h[k-1]+h[k]\tau)\,\tau\,(1-\tau) \\
&+ \frac{(h[k+1])^2\,(t[k+3]-t[k-1])}{3\,(h[k+1]+h[k])}\,\tau^3 + \frac{(h[k-1])^2\,(t[k+2]-t[k-2])}{3\,(h[k]+h[k-1])}\,(1-\tau)^3\Big),
\end{aligned}$$

where the number ϑ belongs to the interval $(t[k-2], t[k+3])$. Note that $h[k]+h[k-1] \geq h[k-1]$ and $h[k]+h[k+1] \geq h[k+1]$. Thus,

$$H(\tau) \leq \bar{h}^4\left((2-\tau)(1+\tau)\tau(1-\tau) + \frac{4}{3}(\tau^3 + (1-\tau)^3)\right) \leq \frac{43}{48}\,\bar{h}^4,$$

where $\bar{h} \stackrel{\text{def}}{=} \max_{\nu=k-2,\ldots,k+2} h[\nu]$. Hence, Eq. (21) follows. If $h[\nu] = h$ for $\nu = k-2, ..., k+2$ then we have the estimation

$$H(\tau) = h^4 \left((2-\tau)(1+\tau)\tau(1-\tau) + \frac{2}{3}(\tau^3 + (1-\tau)^3 \right) \leq \frac{35}{48}\, h^4, \tag{42}$$

which implies Eq. (22). We emphasize that when $\tau = 1/2$, the function $H(1/2) = 35/48\, h^4$. Therefore, 35/48 is the least possible constant in Eq. (42) and, consequently, 35/52 is the least possible constant in Eq. (22). ■

Proof of Theorem 3.9: When $t \in [t[1],\, t[2]],\ t = t[1] + h[1]\tau,\ \tau \in [0,1]$, the remainder term

$$\begin{aligned} \rho(t) &\stackrel{\text{def}}{=} f(t) - \bar{s}[f](t) = f(t) - P^3(t)[1] - A_0\,(t - t[1])^3 \\ &= f[0,1,2,3,t](t-t[0])(t-t[1])(t-t[2])(t-t[3]) \\ &+ f[0,1,2,3,4]\,\frac{(h[2])^2\,(t[4]-t[0])}{3\,h[1]\,(t[3]-t[1])}\,(t-t[1])^3. \end{aligned}$$

On the interval $[t[1],\, t[2]]$, both coefficients of the differences are positive and, from Theorem 2.2 we have

$$\begin{aligned} \rho(t) &= f[0,1,2,3,\xi](t-t[1])\left((t-t[0])(t-t[2])(t-t[3]) + \frac{(h[2])^2\,(t[4]-t[0])}{3\,h[1]\,(t[3]-t[1])}\,(t-t[1])^2\right) \\ &= \frac{f^{(4)}(\vartheta)}{24}\,(h[1])^2\tau\left((h[0]+h[1]\tau)(1-\tau)\,(h[2]+h[1](1-\tau)) + \frac{(h[2])^2\,(t[4]-t[0])}{3\,(h[2]+h[1])}\,\tau^2\right), \end{aligned}$$

where the points $\xi, \vartheta \in I_0$. Denote by $\bar{h} = \max_{k=0,1,2,3} h[k]$. Then, the following estimation holds

$$|\rho(t)| \leq \frac{\bar{h}^4}{72} \max_{t\in I_0} |f^{(4)}(t)|\,\tau\,\left(3\tau^3 - 2\tau^2 - 3\tau + 6\right) \leq \frac{\bar{h}^4}{18} \max_{t\in I_0} |f^{(4)}(t)|.$$

If $h[0] = h[1] = h[2] = h[3] = h$ then we have the exact estimation, which becomes an identity for the function $f(t) = t^4$:

$$|\rho(t)| \leq \frac{h^4}{72} \max_{t\in I_0} |f^{(4)}(t)|\,\tau\,\left(3\tau^3 - 4\tau^2 - 3\tau + 6\right) \leq h^4\,\frac{16 - 3\sqrt{2}}{288\sqrt{2}} \max_{t\in I_0} |f^{(4)}(t)|.$$

When $t \in [t[0],\, t[1]],\ t = t[0] + h[0]\tau,\ \tau \in [0,1]$, the remainder term is

$$\begin{aligned} \rho_0(t) &\stackrel{\text{def}}{=} f(t) - \bar{s}[f](t) = f(t) - P^3(t)[1] \\ &= -f[0,1,2,3,t](h[0])^2\tau(1-\tau)(h[0](1-\tau) + h[1])(h[0](1-\tau) + h[1] + h[2]). \end{aligned}$$

As above, $\bar{h} = \max_{k=0,1,2} h[k]$. Then, the estimation holds

$$|\rho(t)| \leq \frac{\bar{h}^4}{24} \max_{t\in I_0} |f^{(4)}(t)|\,\tau\,\left(\tau^3 - 6\tau^2 + 11\tau - 6\right) \leq \frac{\bar{h}^4}{24} \max_{t\in I_0} |f^{(4)}(t)|.$$

The estimation is exact even for the uniform grid. It becomes an identity for the function $f(t) = t^4$.■

References

[1] M. Abramowitz and I. A. Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables.* Dover, New York, 1972.

[2] A. Aldroubi, C. Cabrellib, and U. M. Molterb. Wavelets on irregular grids with arbitrary dilation matrices and frame atoms for $l^2(\mathbb{R}^d)$. *Applied and Comp. Harm. Analysis*, 17(2):119–140, 2004.

[3] A. Z. Averbuch, P. Neittaanmäki, and V. A. Zheludev. *Spline and spline wavelet methods with applications to signal and image processing, Volume I: Periodic splines.* Springer, 2014.

[4] A. Z. Averbuch, P. Neittaanmäki, and V. A. Zheludev. *Spline and spline wavelet methods with applications to signal and image processing, Volume II: Non-periodic splines.* Springer, 2015.

[5] C. M. Brislawn. Classification of nonexpansive symmetric extension transforms for multirate filter banks. *Appl. Comput. Harmon. Anal.*, 3(4):337–357, 1996.

[6] T. Cai and L. Brown. Wavelet shrinkage for nonequispaced samples. *Annals of Statistics*, 26(5):1783–1799, 1998.

[7] I. Daubechies, I. Guskov, P. Schroder, and W. Sweldens. Wavelets on irregular point sets. *Phil. Trans. R. Soc. Lond. A*, 357, 1999.

[8] C. de Boor. *A practical guide to splines.* Springer, New York, 1978.

[9] C. de Boor. Divided differences. *Surveys in Approximation Theory*, 1, 2005.

[10] C. A. Hall. On error bounds for spline interpolation. *J. Approx. Theory*, 1, 1968.

[11] Z. Liu, Y. Mi, and Y. Mao. An improved real-time denoising method based on lifting wavelet transform. *Measurement Science Review*, 14(3):152–159, 2014.

[12] T. Lyche and L.L. Schumaker. *Local Spline Approximation Methods.* MRC Technical Summary Report. Mathematics Research Center, University of Wisconsin, 1974.

[13] I. J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions. *Quart. Appl. Math.*, 4:45–99, 112–141, 1946. Parts A and B.

[14] I. J. Schoenberg. On variation diminishing approximation methods. In R. E. Langer, editor, *On Numerical Approximation, MRC Symposium*, pages 249–274, Madison, WI, 1959. University of Wisconsin Press.

[15] L. L. Schumaker. *Spline functions: Basic theory.* John Wiley & Sons, New York, 1981.

[16] J Stoer and R. Burlich. *Introduction to numerical analysis, Second edition.* Springer–Verlag, New York, 1993.

[17] M. G. Suturin and V. Zheludev. On the approximation on finite intervals and local spline extrapolation. *Russian J. Numer. Anal. Math. Modelling*, 9(1):75–89, 1994.

[18] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Appl. Comput. Harmon. Anal.*, 3(2):186–200, 1996.

[19] W. Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM J. Math. Anal.*, 29(2):511–546, 1997.

[20] E. Vanraesa, M. Jansenb, and A. Bultheela. Stabilised wavelet transforms for non-equispaced data smoothing. *Signal Processing*, 82(12):1979–1990, 2002.

[21] Yu. S. Zav'yalov, B. I. Kvasov, and V. L. Miroshnichenko. *Methods of spline-functions*. Nauka, Moscow, 1980. (In Russian).

[22] V. Zheludev. Local spline approximation on arbitrary meshes. *Sov. Math.*, 8:16–21, 1987.