

Redundant wavelet processing on the half-axis with applications to signal denoising with small delays: Theory and experiments

François Chaplais, Panagiotis Tsiotras, Dongwon Jung

▶ **To cite this version:**

François Chaplais, Panagiotis Tsiotras, Dongwon Jung. Redundant wavelet processing on the half-axis with applications to signal denoising with small delays: Theory and experiments. International Journal on Adaptive Control and Signal Processing, 2006, 20 (9), pp.447 - 474. 10.1002/acs.911 . hal-00501864

HAL Id: hal-00501864

https://minesparis-psl.hal.science/hal-00501864

Submitted on 12 Jul 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Redundant Wavelet Processing on the Half-Axis with Applications to Signal Denoising with Small Delays: Theory and Experiments

François Chaplais, Panagiotis Tsiotras, *Senior Member, IEEE,* and Dongwon Jung

Abstract

A wavelet transform on the negative half real axis is developed using an average-interpolation scheme. This transform is redundant and can be used to perform causal wavelet processing, such as on-line signal denoising, without delay. Nonetheless, in practice some boundary effects occur and thus a small amount of delay is required to reduce them. The effect of this delay is studied using a numerical example of a signal with large noise and sharp transients. It is shown that the delay required to obtain acceptable denoising levels is decreased by using the proposed redundant transform instead of a non-redundant one. We also present results from the experimental implementation of the proposed algorithm for the denoising of a feedback signal during the control of a three-phase permanent-magnet synchronous brushless DC motor.

I. Introduction

Wavelets have become very popular in signal processing during the last two decades. They allow compact representation of a signal with a small number of wavelet coefficients. Given a sequence of data $\{a_0[n]\}_{n\in\mathbb{Z}}$, the discrete wavelet transform generates for each scale depth $J \geq 1$ a sequence of scale coefficients $a_J := \{a_J[n]\}_{n\in\mathbb{Z}}$ which provide an approximation of the signal at the low resolution scale J, along with a family of detail coefficients $d_j := \{d_j[n]\}_{n\in\mathbb{Z}}$, $1 \leq j \leq J$, which contain the information lost when going from a finer resolution scale $j-1$ to a coarser resolution scale j. As a result, the data a_J contain the "salient" (low-resolution) features of the signal and d_j $(1 \leq j \leq J)$ contain the residual (high-resolution) features or "details" at this scale.

The simplest, and probably most common, method for signal denoising via wavelet decomposition is thresholding. In thresholding one keeps only the coefficients d_j which are larger than a prespecified tolerance. For instance, using hard thresholding one redefines the detail coefficients as follows

$$\hat{d}_j[n] := \begin{cases} d_j[n], & |d_j[n]| > \varepsilon, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

When using soft thresholding each coefficient is reduced by a small amount. For instance,

$$\hat{d}_j[n] := \begin{cases} \operatorname{sgn}(d_j[n])(|d_j[n]| - \varepsilon), & |d_j[n]| > \varepsilon, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Several methods have been proposed for choosing the threshold value ε. Donoho and Johnstone [1] propose, for instance, to use $\varepsilon = \sigma\sqrt{2\log_{\mathrm{e}} N}$, where σ^2 is the variance of the original data set of N samples. It can be shown that this choice of threshold minimizes (as the number of samples goes to infinity, $N \to \infty$) the thresholding risk amongst all diagonal estimators [2].

The work of the last two authors was supported in part by NSF Award No. CMS-0084954.

F. Chaplais is with the Centre Automatique et Systèmes, École Nationale Supérieure des Mines de Paris, 35 rue Saint Honoré, 77305 Fontainebleau, France (e-mail: francois.chaplais@ensmp.fr). F. Chaplais wishes to thank Albert Cohen for suggesting the algorithms of Sections II and III.

P. Tsiotras is with the School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0150, USA (e-mail: tsiotras@gatech.edu). Corresponding author. Part of this work was performed while P. Tsiotras was on a leave to CAS, ENSMP in the spring of 2003.

D. Jung is with the School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0150, USA (e-mail: dongwon_jung@ae.gatech.edu).

Thresholding applied to the coefficients of a wavelet transform is also known to be an especially efficient method for denoising signals with sharp transients [3], [4]. Standard thresholding however typically uses wavelets on the whole real line. This causes significant delays in the processing, because some of the filters involved in the composition/decomposition phases of are not causal; see Fig. 1. In addition, when operating on on-line data (such as for denoising signals within a feedback control loop) it is imperative to use wavelets on the negative half real axis (the half axis in this context representing past, known values of the signal). When operating on data within a feedback loop, any delays arising from the process of denoising may impair performance, or even cause instability. In order to minimize any delays arising from wavelet processing, in this paper we propose a method that operates only on past data, so as to minimize or eliminate any delays associated with traditional wavelet denoising algorithms.

The proposed method, in summary, works as follows: (i) at each time t (the "current time"), a wavelet transform is performed on the whole available past data from $-\infty$ to t; (ii) the wavelet coefficients from this wavelet transform are then processed/denoised, using thresholding; and (iii) the original signal is recovered by cycle spinning [4], that is, by averaging all different reconstructions obtained from the various shifts of the signal. After reconstruction, a value of the signal at time $t-\tau$ is extracted, where τ is a user-specified small delay which – ideally – is zero. In fact, for $\tau = 0$, the previous procedure provides a true online wavelet processing *without delay*. One should contrast this to the case of wavelet processing on the whole real axis, which introduces a delay of order 2^{J-1}, where J is the number of scales used in the transform. In practice, however, boundary effects occur and a non-zero τ must be chosen. Note that only a finite amount of past data is required for the computations in practice since the processing uses Finite Impulse Response (FIR) filters on a finite number of scales.

It should be mentioned that although the word "wavelet" will be used throughout in the sequel, the basic point of view taken in this paper is that of FIR filter banks [5], [2]. The various signal decomposition and reconstruction schemes described in this paper were inspired by the average-interpolation scheme of Donoho [6] and Sweldens [7]; see also [8]. As noted in [7], this scheme can be modified to process signals on the half-axis. We show explicitly how this can be done. In addition, the proposed scheme is modified to provide a redundant transform on the negative half-axis.

For simplicity of exposition, in this paper we will restrict ourselves to wavelets with three discrete vanishing moments. However, the results are easily extendable to wavelets with any number of moments. We remind the reader that a filter h has M discrete vanishing moments if and only if $\sum_{n\in\mathbb{Z}} n^\ell h[n] = 0$, with $\ell = 0, \ldots, M-1$.

The paper is organized as follows. In Section II we review the average-interpolation scheme, which is the main building block behind our algorithm. We use average-interpolation to construct the low-pass and high-pass decomposition and reconstruction filters. The results of Section II assume that data are available both for past and future values of the signal. In Section III we modify the algorithm so that it uses only past data. The modification consists essentially of a decentering scheme when the data are close to the boundary. It is well known that redundant algorithms perform better than non-redundant ones for denoising applications. Specifically, translation invariant transforms are redundant, in general; see, for instance, [5, p.132-133] and [4], [9]. The standard (non-redundant) wavelet transform, on the other hand, is not translation invariant: if the signal is shifted, the wavelet coefficients are not simply a shifted version of the original signal. In Section IV we show how the denoising algorithm of Section III can be modified to obtain a redundant version of wavelet denoising on the half-axis. The benefits brought about by the redundancy are demonstrated in Section V, where the proposed algorithm is applied to a rather challenging signal having sharp transients. Since the main motivation of on-line denoising has to do with processing signals within a feedback loop, in Section VI we provide experimental validation of the proposed algorithm from denoising the angular velocity signal of a three-phase permanent magnet synchronous DC motor of a reaction wheel assembly. The angular velocity signal is used as an input to a PI loop to provide tight torque control of the reaction wheel. Finally, Section VII summarizes the theoretical and experimental results of the paper.

II. The Average-Interpolation Scheme

A very simple, yet classic, scheme for building a discrete wavelet transform with discrete vanishing moments is to use average-interpolation. The method is based on the so-called "cell averages" [8], also known as "blocky kernels" [6]. The starting point of average-interpolation is the interpretation of the data samples as averages of a function over the sample intervals. Specifically, given a sequence a_j, average-interpolation assigns to each data

sample $a_j[n]$ the dyadic interval $\mathcal{I}_{j,n} = [2^j(n-1), 2^j n]$ of length[1] 2^j, and assumes that each sample $a_j[n]$ represents the average of some (unknown) function f on $\mathcal{I}_{j,n}$. In other words,

$$a_j[n] = 2^{-j} \int_{2^j(n-1)}^{2^j n} f(x)\,\mathrm{d}x. \tag{3}$$

The discrete wavelet transform using average-interpolation then proceeds as usual, and involves two parts: the *decomposition* phase, which yields the transformed signal in wavelet space, and the *reconstruction* phase, which retrieves a signal from its transform, that is, its wavelet coefficients. The decomposition and reconstruction can both be obtained using a pair of FIR filters, namely, the wavelet decomposition and wavelet reconstruction filters. If no processing is performed on the data after the decomposition phase, the approach results in perfect reconstruction: the original data sequence is recovered without any error.

For clarity, the whole process for a one-step decomposition/reconstruction is shown in Fig. 1. The filters $\bar{h}$ and $\bar{g}$ are the low- and high-pass (decomposition) filters and $\tilde{h}$ and $\tilde{g}$ are the low- and high-pass (reconstruction) filters. The details of the construction of these filters can be found in [6] and [7]. For filters with three vanishing moments, as considered in this paper, Donoho [6] has shown that these filters are actually identical to the $(1, 3)$ B-spline biorthogonal filters[2] of Daubechies [10]. Nonetheless, here we will use an alternative derivation of the transform starting from the average-interpolation framework, together with the usual FIR filter presentation, as illustrated in Fig. 1. The reason for doing this, is because average interpolation can be easily adapted to derive a similar scheme for processing signals over the half-axis, which is the main objective of this paper. In addition, as it will be shown in Section IV, one can use the average-interpolation framework in order to obtain a *redundant* transform on the half-axis.

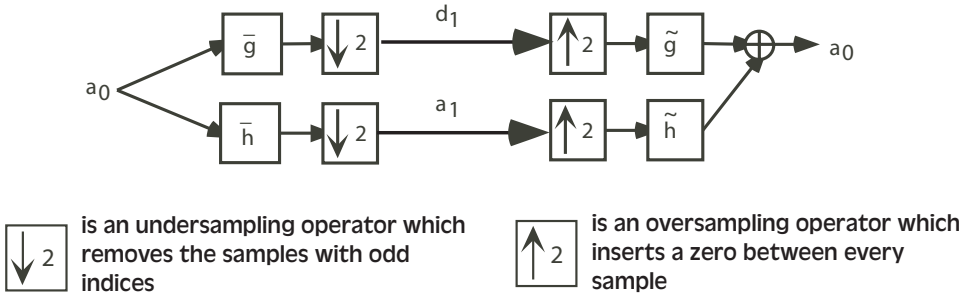


Fig. 1. Perfect reconstruction filter banks, used for the implementation of the wavelet transform on the real axis. One-step decomposition/reconstruction shown. The filters $\bar{h}$ and $\bar{g}$ are the low- and high-pass (decomposition) filters and $\tilde{h}$ and $\tilde{g}$ are the low- and high-pass (reconstruction) filters, respectively.

A. Decomposition

Similarly to standard wavelet processing, the data is first filtered by a low-pass filter $\bar{h}$ and a high-pass filter $\bar{g}$. This separates the signal to low-frequency and high-frequency components. The construction of these filters is given next.

1) Low-Pass Filter: Up to a normalizing scalar of $\sqrt{2}$, the proposed low-pass filter computes the average of two successive signal values as follows:

$$a_{j+1}[n] = \frac{a_j[2n] + a_j[2n-1]}{\sqrt{2}}. \tag{4}$$

In wavelet theory, it is convenient to write the decomposition filters as a convolution with a filter followed by subsampling, as in Fig. 1. This representation will be particularly useful for generating the redundant transforms of Section IV, where the subsampling operation is removed from the process. From (4) we have that

$$a_{j+1}[n] = \sum_{\ell=0}^{1} \bar{h}[\ell]\, a_j[2n-\ell] = \sum_{k=2n-1}^{2n} \bar{h}[2n-k]\, a_j[k]. \tag{5}$$

[1]Note that in the closed interval $\mathcal{I}_{j,n}$ we have $2^j + 1$ distinct integers.

[2]This indexing scheme implies that the scaling function is a first order cardinal B-spline and the wavelet has three vanishing moments.

Therefore the low-pass decomposition filter is the Haar decomposition filter [5, p. 60]

$$\bar{h} = \left[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right], \tag{6}$$

which has support $[0, 1]$. It is noted at this point that the use of a Haar filter in the low-pass decomposition phase is mandated by the restriction to have a polynomial identification that is performed on successive, non-overlapping intervals. The latter implies that averages of averages over these intervals are also averages over the union of intervals, something that is not true if the intervals are overlapping. This property makes the polynomial identification task (see next section as well as the Appendix) much easier.

2) High-Pass Filter: We first introduce the high-pass filter $\bar{g}$ using the average-interpolation scheme. Later we will explicitly compute the values of its coefficients. To begin, we impose the restriction that the high-pass decomposition filter has three discrete vanishing moments. That is, the output through the filter of all discrete polynomial sequences of the form $a[n] = \sum_{k=0}^{k=2} a_k n^k$, of degree less than or equal to two, is zero

$$\sum_{n\in\mathbb{Z}} \bar{g}[\ell - n]\, a[n] = 0. \tag{7}$$

This vanishing moments property of the high-pass decomposition filter ensures that three moments of the original data will be retained during signal decomposition, and the corresponding low-pass reconstruction filter ($\tilde{h}$ in Fig. 1) will efficiently approximate sufficiently "regular" signals up to order two.

Recall next that, given a regularly spaced grid on $\mathbb{R}$ with step 2^j, for instance, $x_{j,n} := 2^j n$, $n \in \mathbb{Z}$, we say that a discrete signal p_j is a polynomial if and only if it is a sequence of the averages of a continuous polynomial $p(x)$ over the intervals determined by the grid, that is, if and only if

$$p_j[n] = 2^{-j} \int_{2^j(n-1)}^{2^j n} p(x)\, \mathrm{d}x. \tag{8}$$

The proposed high-pass filter therefore considers the averages

$$\frac{a_j[2n] + a_j[2n-1]}{2} = \frac{a_{j+1}[n]}{\sqrt{2}} \tag{9}$$

from (4) and identifies the unique polynomial $p(x)$ of degree less than or equal to two, whose averages on the coarser grid with step 2^{j+1} coincides with three successive values of $a_{j+1}[n]/\sqrt{2}$. It then computes the averages of $p(x)$ on the fine grid with step 2^j, subtracts the actual values of the signal from these fine averages, and normalizes the result by a factor $\sqrt{2}$. This sequence of operations is illustrated in Fig. 2. In Fig. 2 the solid piecewise constant line represents the original discrete time signal $a_j[-5], a_j[-4], \ldots, a_j[0]$. The dotted line depicts the averages of two successive discrete values at scale j. The result is a signal sampled at half the frequency of the original signal. The solid curve is the unique polynomial of degree less than or equal to two such that its averages on the three coarse scale intervals $[-6, -4], [-4, -2]$ and $[-2, 0]$ defined by the dotted lines are precisely equal to the three values given as $a_{j+1}[-2]/\sqrt{2}, a_{j+1}[-1]/\sqrt{2}$ and $a_{j+1}[0]/\sqrt{2}$, respectively. The two dashed piecewise constant lines in the middle interval $[-4, -2]$ are the averages of the polynomial on the two middle fine intervals defined by the original signal.

Observe that the difference between the value $\tilde{p}_{j+1}[-1]$ on the left subinterval and the original value $a_j[-3]$ is the opposite of the similar difference computed on the right interval, that is, $a_j[-3] - \tilde{p}_{j+1}[-1] = p_{j+1}[-1] - a_j[-2]$. Hence, only one of these differences needs to be retained during the algorithm.

The steps of the construction of the high-pass decomposition filter $\bar{g}$ can now be summarized as follows:

(i) At scale j and index $2n$, the three low-pass outputs $a_{j+1}[n-1]$, $a_{j+1}[n]$ and $a_{j+1}[n+1]$ are computed from the six data points $a_j[2n-3], a_j[2n-2], a_j[2n-1], a_j[2n], a_j[2n+1], a_j[2n+2]$ at the previous (finer) scale according to (4). That is,

$$a_{j+1}[k] = \frac{a_j[2k-1] + a_j[2k]}{\sqrt{2}}, \qquad k = n-1, n, n+1 \tag{10}$$

This is shown in Fig. 3.

(ii) Let $p(x)$ be the unique polynomial of degree less than or equal to two such that the discrete averages $a_{j+1}[n-1]/\sqrt{2}$, $a_{j+1}[n]/\sqrt{2}$ and $a_{j+1}[n+1]/\sqrt{2}$ are the averages of $p(x)$ on the intervals $[2^{j+1}(n -$

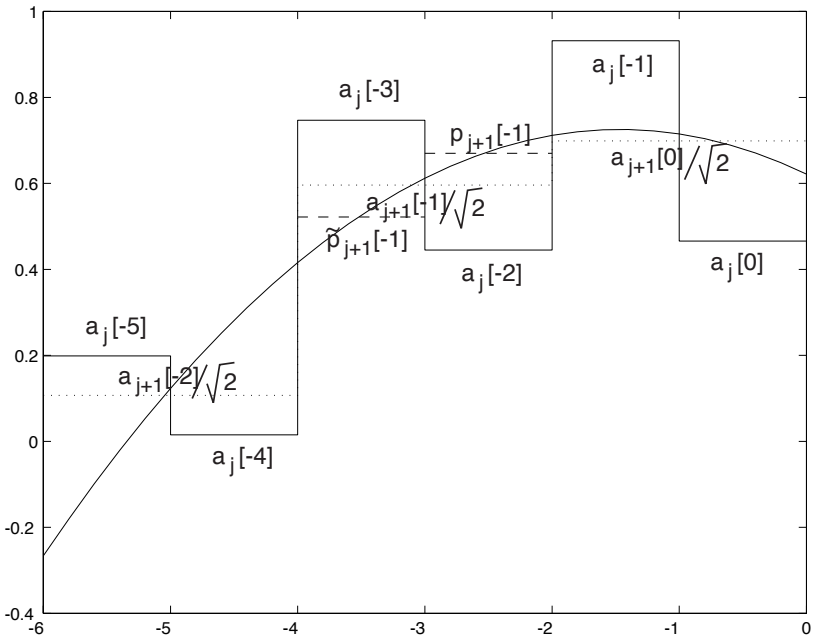


Fig. 2. Average-interpolation using a polynomial of degree two.

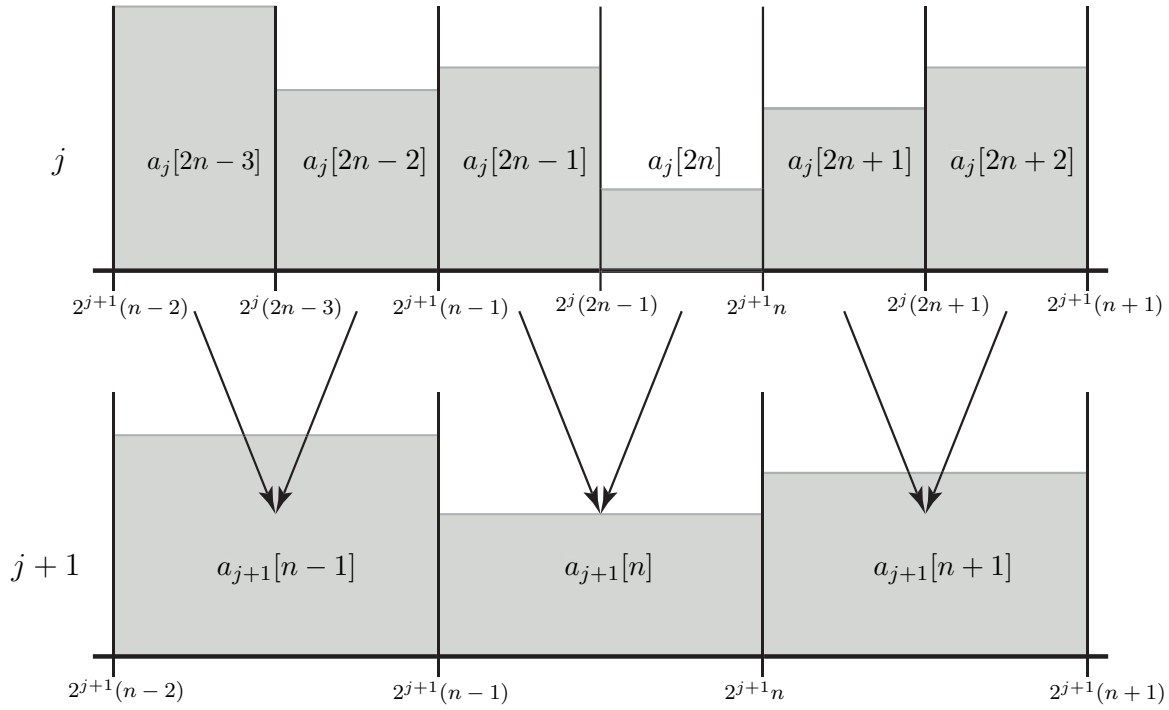


Fig. 3. Decomposition phase. Going from the fine to the coarse scale.

.

$2), 2^{j+1}(n-1)]$, $[2^{j+1}(n-1), 2^{j+1}n]$ and $[2^{j+1}n, 2^{j+1}(n+1)]$, respectively. Denote by $p_{j+1}[n]$ the average of $p(x)$ on $[2^j(2n-1), 2^{j+1}n]$, that is, let

$$p_{j+1}[n] := 2^{-j} \int_{2^j(2n-1)}^{2^{j+1}n} p(x)\,\mathrm{d}x \tag{11}$$

Then $p_{j+1}[n]$ is the *prediction* of $a_j[2n]$ obtained from $a_{j+1}[n-1]$, $a_{j+1}[n]$ and $a_{j+1}[n+1]$.

(iii) The output of the high pass filter (or detail) is the difference

$$d_{j+1}[n] = \sqrt{2}\,\big(a_j[2n] - p_{j+1}[n]\big). \tag{12}$$

Up to $\sqrt{2}$, this is the difference between the actual value of the signal and its prediction from the average-interpolating polynomial $p(x)$.

Similarly to the low-pass case, we can construct a filter to be used before subsampling. This filter can be computed from (12) after writing down explicitly the expression for $p_{j+1}[n]$. A simple calculation shows that

$$p_{j+1}[n] = \sqrt{2}\left(-\frac{1}{16}a_{j+1}[n-1] + \frac{1}{2}a_{j+1}[n] + \frac{1}{16}a_{j+1}[n+1]\right). \tag{13}$$

Using (12) and (10) this yields,

$$\begin{aligned} d_{j+1}[n] \;=\; & \sqrt{2}\left(\frac{1}{16}a_j[2n-3] + \frac{1}{16}a_j[2n-2] - \frac{1}{2}a_j[2n-1] + \frac{1}{2}a_j[2n]\right. \\ & \left. -\frac{1}{16}a_j[2n+1] - \frac{1}{16}a_j[2n+2]\right). \end{aligned} \tag{14}$$

Therefore the high-pass decomposition filter is given by

$$\bar{g} = \left[-\frac{\sqrt{2}}{16}\,, -\frac{\sqrt{2}}{16}, \frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{16}, \frac{\sqrt{2}}{16}\right] \tag{15}$$

which has support $[-2, 3]$. One can then write

$$d_{j+1}[n] = \sum_{\ell=-2}^{3} \bar{g}[\ell]\, a_j[2n-\ell] = \sum_{k=2n-3}^{2n+2} \bar{g}[2n-k]\, a_j[k]. \tag{16}$$

Notice that if the original discrete signal a_j is a sequence of averages of a polynomial $q(x)$ on the fine grid, the averages on the coarse grid of the averages of $q(x)$ are still averages of $q(x)$ (on the coarse grid). Hence the uniquely identified polynomial $p(x)$ whose averages on the coarse scale are identical to the signal values a_{j+1} on the coarse scale coincides with $q(x)$. Moreover, the averages of $p(x)$ on the fine grid coincide with the discrete values of the signal a_j at the fine scale (which are the averages of $q(x)$). As a result, the output of any polynomial of degree less than or equal to two through this high-pass filter will be zero. We use this result to show in the Appendix that the proposed high-pass decomposition filter satisfies the vanishing moments property. In the Appendix it is also shown that the details are oscillating, as required by the definition of a wavelet function.

After processing, the signal is reconstructed from the coarse approximation and the detail coefficients. During reconstruction we use the coarse scale data a_{j+1} and detail d_{j+1} to obtain the data a_j at the finer scale. The process is essentially the inverse of the decomposition phase. Specifically, at the reconstruction stage, the low-pass outputs a_{j+1} are available, and thus the average-interpolated polynomial $p(x)$ can be computed from these averages. Hence $p_{j+1}[n]$ and $\tilde{p}_{j+1}[n]$ can also be computed, for instance from (13) and (A.3). Since $d_{j+1}[n]$ is also available, $a_j[2n]$ and $a_j[2n-1]$ can be recovered using (12) and (A.2). In short, the sequence a_j can be reconstructed from the sequences a_{j+1} and d_{j+1}. The details of the reconstruction phase are given next.

B. Reconstruction

The reconstruction stage is based of the same ideas as those presented in the section devoted to the high-pass decomposition filter. The reconstruction basically involves reversing the decomposition steps. Specifically, first a polynomial $p(x)$ of degree less than or equal to two is identified as in step (ii) of the high-pass decomposition filter, using the outputs of the decomposition low-pass filter a_{j+1}. The averages $p_{j+1}[n]$ and $\tilde{p}_{j+1}[n]$ of this polynomial on the sub-intervals $[2^j(2n-1), 2^{j+1}n]$ and $[2^{j+1}(n-1), 2^j(2n-1)]$ are then computed using (13) and (A.3). The part of the signal a_j which has even indices is then recovered from the prediction and details using (12), whereas the part with odd indices uses equation (A.2). If the decomposition has not been modified, the reconstruction restores the signal exactly.

It is common to express the reconstruction process as the sum of the outputs of a low-pass and a high-pass reconstruction filter. Then the filters are expressed as the cascade of an oversampling of the signal by zero insertion[3], followed by the filtering itself, as in Fig. 1. These filters will be useful in the next section for defining the reconstruction filters from redundant transforms.

1) Low-pass filter: The low-pass filter is constructed by writing down explicitly the two predictions $p_{j+1}[n]$ and $\tilde{p}_{j+1}[n]$ as the outputs $\pi_j[2n]$ and $\pi_j[2n-1]$ of the low-pass reconstruction filter after a zero insertion on a_{j+1}. That is,

$$\pi_j[2n] \;= p_{j+1}[n] = \sum_{p=n-1}^{n+1} \tilde{h}[2n-2p]\, a_{j+1}[p] = \sum_{p=-1}^{1} \tilde{h}[2p]\, a_{j+1}[n-p] \tag{17a}$$

$$\pi_j[2n-1] \;= \tilde{p}_{j+1}[n] = \sum_{p=n-1}^{n+1} \tilde{h}[2n-1-2p]\, a_{j+1}[p] = \sum_{p=-1}^{1} \tilde{h}[2p-1]\, a_{j+1}[n-p] \tag{17b}$$

Using the expressions (13) and (A.3) of $p_{j+1}[n]$ and $\tilde{p}_{j+1}[n]$ yields the following value for the low-pass reconstruction filter $\tilde{h}$

$$\tilde{h} = \frac{1}{\sqrt{2}}\left[-\frac{1}{8}, \frac{1}{8}, 1, 1, \frac{1}{8}, -\frac{1}{8}\right]$$

which has support $[-3, 2]$.

2) High-pass filter: Up to $\sqrt{2}$, the high-pass filter outputs the detail for the even indices and the opposite of the detail for the odd indices. If $\delta_j[2n]$ and $\delta_j[2n-1]$ are the outputs for the even and odd indices, we have

$$\delta_j[2n] = \frac{1}{\sqrt{2}} d_{j+1}[n] \;= \sum_{p=n}^{n} \tilde{g}[2n-2p] d_{j+1}[p] = \sum_{p=0}^{0} \tilde{g}[2p] d_{j+1}[n-p] \tag{18a}$$

$$\delta_j[2n-1] = -\frac{1}{\sqrt{2}} d_{j+1}[n] \;= \sum_{p=n}^{n} \tilde{g}[2n-1-2p] d_{j+1}[p] = \sum_{p=0}^{0} \tilde{g}[2p-1] d_{j+1}[n-p] \tag{18b}$$

which leads to the Haar reconstruction filter [5],

$$\tilde{g} = \left[-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right], \tag{19}$$

with support $[-1, 0]$.

Given the expressions (17) and (18) we can finally reconstruct the signal at the fine scale as follows:

$$a_j[2n] \;= \sum_{p=-1}^{1} \tilde{h}[2p]\, a_{j+1}[n-p] + \sum_{p=0}^{0} \tilde{g}[2p]\, d_{j+1}[n-p], \tag{20a}$$

$$a_j[2n-1] \;= \sum_{p=-1}^{1} \tilde{h}[2p-1]\, a_{j+1}[n-p] + \sum_{p=0}^{0} \tilde{g}[2p-1]\, d_{j+1}[n-p]. \tag{20b}$$

[3]A zero value is inserted between each successive values of the discrete signal.

And explicitly,

$$a_j[2n] = \sqrt{2}\left(\frac{1}{16}a_{j+1}[n+1] + \frac{1}{2}a_{j+1}[n] - \frac{1}{16}a_{j+1}[n-1] + \frac{1}{2}d_{j+1}[n]\right), \tag{21a}$$

$$a_j[2n-1] = \sqrt{2}\left(-\frac{1}{16}a_{j+1}[n+1] + \frac{1}{2}a_{j+1}[n] + \frac{1}{16}a_{j+1}[n-1] - \frac{1}{2}d_{j+1}[n]\right). \tag{21b}$$

III. Adaptation to the Half-Axis

The discussion thus far has assumed that a sufficient amount of data is available to process the signal. At each sample point there are enough samples to its left and to its right to construct the average-interpolating polynomial. In particular, it has been assumed that in order to estimate the denoised signal at the current instant, data past this instant is available. This may not always be possible. For instance, for on-line applications we have only access to past data. If t is the present time corresponding to time index $n = 0$ and the data are ordered such that $\ldots, a_j[-3], a_j[-2], a_j[-1], a_j[0]$, it is clear that the construction described in the previous section is not possible for sample $a_j[0]$ since $a_j[1]$ is not yet available. In this section we modify the previous scheme in order to take into account the case when only past values are available for processing. We achieve this by designing a wavelet-like scheme that only processes the data $a_j[n]$ with $-\infty < n \leq 0$. The idea we use is quite simple: The scheme of Section II will be modified at the right boundary so that it uses only the available, past values of the signal. This modification essentially decenters the prediction scheme at the boundary to account for the absence of future sample values.

A. Decomposition

The low-pass averaging filter does not require any modification since it uses only past values; see equation (4). For $n < 0$ the output of the high-pass filter $d_{j+1}[n]$ uses only data with negative or zero input indices. This is because the prediction at index n is at the center of the average interpolating polynomial and therefore it does not use any input data past the index $2n + 2$ (see equation (14)). In particular, $d_{j+1}[-1]$ is computed from $a_j[-5], a_j[-4], \ldots, a_j[0]$. Use of (14) to compute $d_{j+1}[0]$ would require knowledge of $a_j[1]$ and $a_j[2]$. Hence, the high-pass filter needs to be modified for the prediction at output index $n = 0$, since input data is not available at the indices $n = 1, 2$. In order to do so, the interpolating-polynomial is identified based on the three most recent averages, that is, $a_{j+1}[-2], a_{j+1}[-1]$ and $a_{j+1}[0]$, and the prediction is now computed at the index $n = 0$ instead of index $n = -1$. This is illustrated in Fig. 4.

The details of the construction are given below:

- (i) At scale j and index $n = 0$, the three low-pass outputs $a_{j+1}[-2]$, $a_{j+1}[-1]$ and $a_{j+1}[0]$ are computed, as usual, from $a_{j+1}[-5], a_{j+1}[-4], \ldots, a_{j+1}[-1], a_{j+1}[0]$ using (4).
- (ii) Let $p(x)$ be the unique polynomial of degree less than or equal to two, such that $a_{j+1}[-2]/\sqrt{2}$, $a_{j+1}[-1]/\sqrt{2}$ and $a_{j+1}[0]/\sqrt{2}$ are the averages of $p(x)$ on the intervals $[-3 \cdot 2^{j+1}, -2 \cdot 2^{j+1}]$, $[-2 \cdot 2^{j+1}, -2^{j+1}]$ and $[-2^{j+1}, 0]$, respectively. Denote by $p_{j+1}[0]$ the average of $p(x)$ on $[-2^j, 0]$, that is, the rightmost half interval.
- (iii) The output of the high-pass (or detail) filter is the difference

$$d_{j+1}[0] = \sqrt{2}(a_j[0] - p_{j+1}[0]). \tag{22}$$

One can compute the prediction $p_{j+1}[0]$ explicitly as follows

$$p_{j+1}[0] = \sqrt{2}\left(\frac{1}{16}a_{j+1}[-2] - \frac{1}{4}a_{j+1}[-1] + \frac{11}{16}a_{j+1}[0]\right). \tag{23}$$

It follows that the high-pass output $d_{j+1}[0]$ at the boundary is given by

$$d_{j+1}[0] = \sqrt{2}\left(\frac{5}{16}a_j[0] - \frac{11}{16}a_j[-1] + \frac{1}{4}a_j[-2] + \frac{1}{4}a_j[-3] - \frac{1}{16}a_j[-4] - \frac{1}{16}a_j[-5]\right).$$

The prediction has now been decentered. Nonetheless, it is still based on average interpolation. In particular, if the input data represents a sequence of averages of a polynomial $q(x)$ on the intervals $[n2^j, (n+1)2^j], n = -6, \ldots, -1$, the identified polynomial $p(x)$ will be equal to $q(x)$ and the detail $d_{j+1}[0]$ will be zero. Overall, the high-pass filter

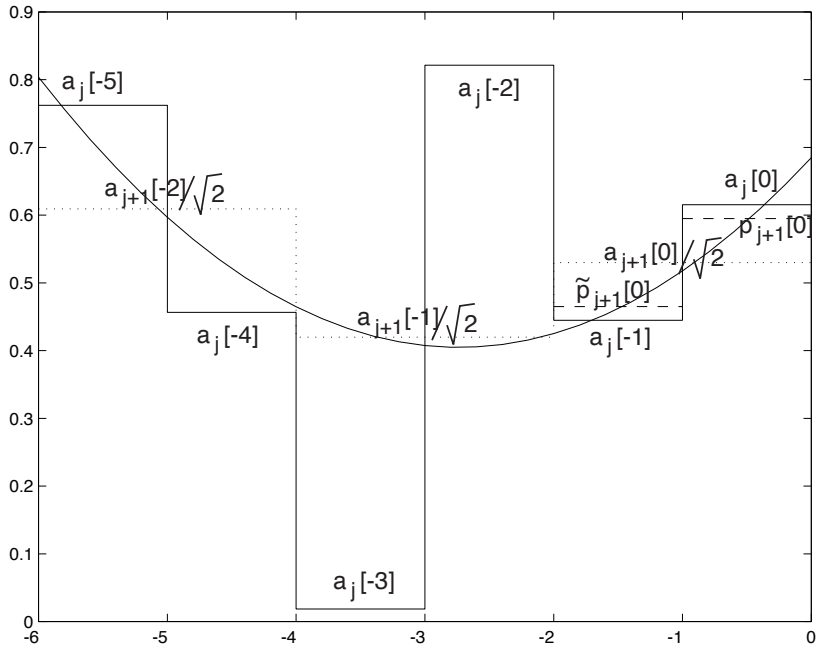


Fig. 4. The average interpolation scheme is decentered to produce a prediction at the boundary.

still has three discrete vanishing moments. Using the same arguments as in the centered scheme, one can show that the details are also oscillating at $n = 0$, that is,

$$\frac{1}{\sqrt{2}} d_{j+1}[0] = a_j[0] - p_{j+1}[0] = \tilde{p}_{j+1}[0] - a_j[-1], \tag{24}$$

where $\tilde{p}_{j+1}[0]$ is the prediction on the interval $[-2^{j+1}, -2^j]$ given by

$$\tilde{p}_{j+1}[0] = \sqrt{2}\left(-\frac{1}{16}a_{j+1}[-2] + \frac{1}{4}a_{j+1}[-1] + \frac{5}{16}a_{j+1}[0]\right). \tag{25}$$

B. Reconstruction

Except for the indices $n = -1$ and $n = 0$, the reconstruction is performed exactly as in Section II-B. The value $a_j[0]$ at the right boundary is recovered using equations (22) and (23), while $a_j[-1]$ is recovered from the formula $a_j[-1] = \tilde{p}_{j+1}[0] - d_{j+1}[0]/\sqrt{2}$ (see also equation (24)), where $\tilde{p}_{j+1}[0]$ from (25).

Numerically, the reconstruction is performed at the right boundary using the following equations

$$a_j[0] = \frac{1}{\sqrt{2}}\left(\frac{1}{8}a_{j+1}[-2] - \frac{1}{2}a_{j+1}[-1] + \frac{11}{8}a_{j+1}[0] + d_{j+1}[0]\right), \tag{26}$$

$$a_j[-1] = \frac{1}{\sqrt{2}}\left(-\frac{1}{8}a_{j+1}[-2] + \frac{1}{2}a_{j+1}[-1] + \frac{5}{8}a_{j+1}[0] - d_{j+1}[0]\right). \tag{27}$$

IV. Redundancy on the Half-Axis

The transforms presented in the previous sections are nonredundant, that is, any pair of sequences a_{j+1} and d_{j+1} can be interpreted as the (unique) decomposition of a signal a_j. Redundant transforms which provide more data per time step than their nonredundant counterpart can also be designed. These redundant transforms provide reconstructions which are more robust to errors and noise. The price to pay is that not all pairs of sequences a_{j+1} and d_{j+1} can be interpreted as a decomposition of the signal a_j. One must therefore be careful how to choose the pairs a_{j+1} and d_{j+1} to recover the original signal.

Redundant transforms are known to improve signal processing using wavelets, especially for denoising [4]. Ideally, the redundant transform should be shift-invariant. When wavelets on the whole real line are used, this is

achieved by removing the subsampling and zero insertion operations in the filter banks of Fig. 1 and by using instead filters *which depend on the scale*. More precisely, at input scale $j \geq 1$, $2^j - 1$ zeros are inserted between each coefficient of the original *filter*. The corresponding algorithm is known as the *algorithme à trous* [11]. This scheme cannot be used directly for transforms on the half-axis however, because boundary effects are not properly handled by the zero insertion trick.

In this section we develop a redundant transform on the negative half-axis. This will be achieved in two steps: first, we will extend the transform of Section III into a redundant transform at the original (fine) scale $j = 0$; second, by interpreting the zero insertion on filters as filtering of multiplexed signals, we extend the method to higher scales. The idea is that we now work with time-varying filters, and this point of view allows us to define the boundary filters for all scales $j > 0$.

A. Redundancy at the first scale

The task of this section is to modify the transform of Section III to obtain a redundant transform. This is achieved by keeping the values with even indices in the transform of Section III (without subsampling). Note that keeping the odd indices also provides perfect reconstruction, provided one also knows the last decomposition output $a_1[0]$.

The simplest idea for generating the values of the transform for the odd indices is to shift the signal one step to the right, drop the boundary value $a_0[0]$, use the transform of Section III, and then shift back the result to the left. This method has two flaws that we wish to remedy: first, $a_0[0]$ cannot be reconstructed with this approach; second, the average-interpolation scheme at the last index is still decentered, meaning that an extrapolation is performed in the prediction step instead of an interpolation. If we use the extra value $a_1[0]$ in addition to the low-pass output of the shifted transform instead, the boundary high-pass filter can be modified to use interpolation instead of extrapolation. Moreover, $a_0[0]$ can be recovered at the reconstruction. This method is described in detail next.

1) Decomposition: The shift-invariant filters of Section II-A are first applied to the signal, and the output is now subsampled at the odd indices, the last output index being $n = -1$ for the low-pass filter, and $n = -3$ for the high-pass filter. That is,

$$a_1[n] = \frac{a_0[n] + a_0[n-1]}{\sqrt{2}}, \qquad n = \ldots, -3, -1, \tag{28}$$

and

$$\begin{aligned} d_1[n] &= \sqrt{2}\left(\frac{1}{16}a_0[n-3] + \frac{1}{16}a_0[n-2] - \frac{1}{2}a_0[n-1] + \frac{1}{2}a_0[n]\right. \\ &\quad \left. -\frac{1}{16}a_0[n+1] - \frac{1}{16}a_0[n+2]\right), \qquad n = \ldots, -5, -3. \end{aligned} \tag{29}$$

For the index $n = -1$, the detail $d_1[-1]$ is computed as follows:

(i) Let $p(x)$ be the unique polynomial of degree less than or equal to two such that $a_1[-3]/\sqrt{2}$, $a_1[-1]/\sqrt{2}$ and $a_1[0]/\sqrt{2}$ are the averages of $p(x)$ on the intervals $[-5, -3]$, $[-3, -1]$ and $[-2, 0]$ respectively. Denote by $p_1[-1]$ the average of $p(x)$ on $[-2, -1]$. The difference with the boundary scheme of the previous section is that the average interpolating polynomial is not drawn from disjoint intervals, so that the prediction is computed from an interpolation, thanks to the use of $a_1[0]$. This prediction scheme is illustrated in Fig. 5.

(ii) The detail is now defined by

$$d_1[-1] = \sqrt{2}(a_0[-1] - p_1[-1]). \tag{30}$$

The detail $d_1[-1]$ can be computed by writing down the value of the prediction $p_1[-1]$

$$p_1[-1] = \sqrt{2}\left(-\frac{1}{24}a_1[-3] + \frac{3}{8}a_1[-1] + \frac{1}{6}a_1[0]\right), \tag{31}$$

which yields, via (28),

$$d_1[-1] = \sqrt{2}\left(\frac{1}{24}a_0[-4] + \frac{1}{24}a_0[-3] - \frac{3}{8}a_0[-2] + \frac{11}{24}a_0[-1] - \frac{1}{6}a_0[0]\right). \tag{32}$$

Since the outputs of the decomposition of Section II-A are to be now the even subsamples of a redundant transform, we shall denote by $a_1[2n]$ and $d_1[2n]$ their values, with $n \leq 0$. We have just defined the outputs of

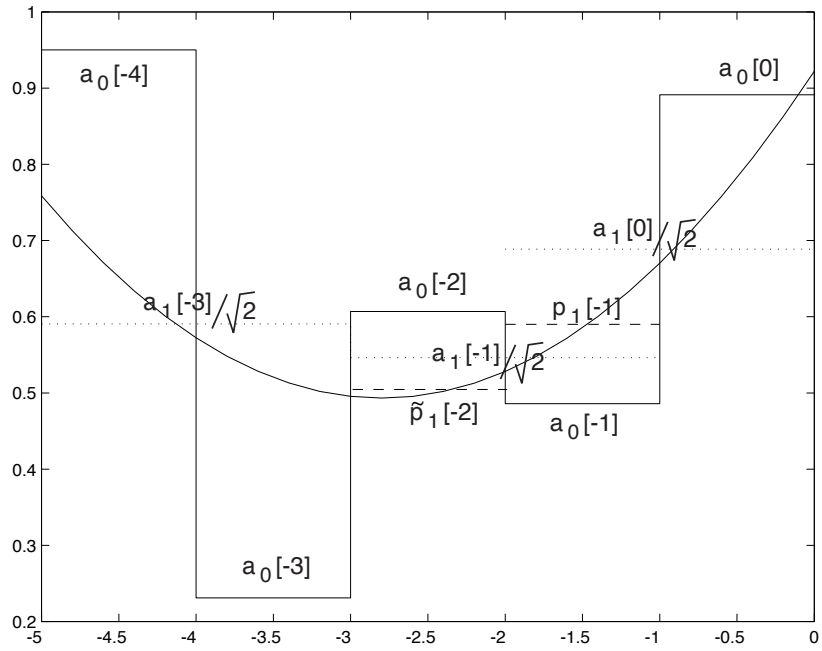


Fig. 5. Computation of the prediction for the index $n = -1$. The polynomial $p(x)$ is fitted on overlapping intervals, using the values $a_1[-3]$, $a_1[-1]$ and $a_1[0]$.

the decomposition for all odd indices $n = \ldots, -5, -3, -1$. This complements the sequences $a_1[2n]$, $d_1[2n]$ into a redundant transform with values for all negative or zero integer indices. In the Appendix it is shown that the previous modification at the boundary preserves the vanishing moments property. Similarly, therein we also show that this modification does not destroy the property of oscillation of the details.

2) Perfect reconstruction from the odd samples only: The signal can be entirely recovered, provided the last even low-pass output $a_1[0]$ is added to the previous transform. Specifically, for all indices less than or equal to $n = -3$, the signal is recovered as in Section II-B, since there is no interference with the boundary during decomposition. For the indices $n = -2$ and $n = -1$, the predictions $p_1[-1]$ and $\tilde{p}_1[-2]$ can be computed from the interpolating polynomial $p(x)$ via (31) and (A.5). Together with $d_1[-1]$, equations (30) and (A.4) provide the reconstruction of $a_0[-1]$ and $a_0[-2]$. Finally, $a_0[0]$ is restored through the equation

$$a_0[0] = \sqrt{2}a_1[0] - a_0[-1]. \tag{33}$$

Summarizing, the reconstruction of $a_0[-2]$, $a_0[-1]$ and $a_0[0]$ is given by

$$a_0[-2] = \frac{1}{\sqrt{2}}\left(\frac{1}{12}a_1[-3] + \frac{5}{4}a_1[-1] - \frac{1}{3}a_1[0] - d_1[-1]\right), \tag{34a}$$

$$a_0[-1] = \frac{1}{\sqrt{2}}\left(-\frac{1}{12}a_1[-3] + \frac{3}{4}a_1[-1] + \frac{1}{3}a_1[0] + d_1[-1]\right), \tag{34b}$$

$$a_0[0] = \frac{1}{\sqrt{2}}\left(\frac{1}{12}a_1[-3] - \frac{3}{4}a_1[-1] + \frac{5}{3}a_1[0] - d_1[-1]\right). \tag{34c}$$

This proves that the signal can be recovered from the odd outputs of the redundant transform, to which $a_1[0]$ has been added.

3) Perfect reconstruction from the redundant transform: For shift-invariant transforms on the real line, the signal is reconstructed by computing the average of the two possible reconstructions, e.g., from the even and odd indexed subsamples of the transform. We do the same thing here, and restore the signal by computing the average of the reconstructions of Sections III-B and IV-A.2. For indices smaller than $n = -2$, the reconstruction is performed as for the regular shift-invariant redundant transforms on the whole real line. For the indices $n = -2$, $n = -1$ and $n = 0$, the boundary filters are computed by averaging the reconstruction formulas from the even and odd

subsamples. Using (21) and equations (33) and (34) yields,

$$\begin{aligned}
a_0[-2] &= \frac{1}{\sqrt{2}}\left(-\frac{1}{16}a_1[-4]+\frac{1}{24}a_1[-3]+\frac{1}{2}a_1[-2]+\frac{5}{8}a_1[-1]-\frac{5}{48}a_1[0]\right.\\
&\quad \left.+\frac{1}{2}d_1[-2]-\frac{1}{2}d_1[-1]\right),\\
a_0[-1] &= \frac{1}{\sqrt{2}}\left(-\frac{1}{16}a_1[-4]-\frac{1}{24}a_1[-3]+\frac{1}{4}a_1[-2]+\frac{3}{8}a_1[-1]+\frac{23}{48}a_1[0]\right.\\
&\quad \left.+\frac{1}{2}d_1[-1]-\frac{1}{2}d_1[0]\right),\\
a_0[0] &= \frac{1}{\sqrt{2}}\left(\frac{1}{16}a_1[-4]+\frac{1}{24}a_1[-3]-\frac{1}{4}a_1[-2]-\frac{3}{8}a_1[-1]+\frac{73}{48}a_1[0]\right.\\
&\quad \left.-\frac{1}{2}d_1[-1]+\frac{1}{2}d_1[0]\right).
\end{aligned}$$

B. Redundancy at coarser scales

When using wavelets on the whole real axis, the redundant transforms at the coarse scales are obtained via the standard method, that is, by inserting zeros into the original filters. We give here an interpretation of this zero insertion as an operation of the original filters on a multiplexed signal. This interpretation can be used to extend the zero insertion to time-varying filters, specifically, filters whose coefficients take different values close to the boundary.

At the decomposition from scale $j \geq 1$ to scale $j+1$, we consider a signal x to be a combination of 2^j signals $\{x_\ell\}_{0\leq\ell<2^j}$, with $x_\ell[n] = x[2^j n - \ell]$. Let us recall that, at the scale j, the zero insertion operation on the filter h results in the filter defined by

$$h_j[n] := \begin{cases} h[p], & \text{if } n = 2^j p,\\ 0, & \text{otherwise.}\end{cases} \tag{35}$$

The redundant decomposition from scale j to scale $j+1$ is obtained by convolving the input with the zero insertion filters at the scale j. Then the convolution y of the signal x with the filter h_j can be split into the series of 2^j convolutions

$$\begin{aligned}
y_\ell[n] &:= y[2^j n - \ell]\\
&= \sum_{k\in\mathbb{Z}} h_j[k]\, x[2^j n - \ell - k]\\
&= \sum_{p\in\mathbb{Z}} h_j[2^j p]\, x[2^j n - \ell - 2^j p]\\
&= \sum_{p\in\mathbb{Z}} h[p]\, x_\ell[n-p]\\
&= \sum_{p\in\mathbb{Z}} h[n-p]\, x_\ell[p]\\
&= h * x_\ell[n], \qquad 0 \leq \ell < 2^j.
\end{aligned}$$

Hence the convolution with zero inserted filters can be viewed as the merging of 2^j parallel convolutions y_ℓ of the filter h with the 2^j signals x_ℓ. This can be extended to time varying filters $h_n[p]$ by setting

$$y_\ell[n] = \sum_{p\in\mathbb{Z}} h_n[p]\, x_\ell[p]. \tag{36}$$

In practice, zero insertion is used when the filtering coincides with the time-invariant processing of Section II. When close to the boundary, the multiplexed formula (36) is used on the explicit expressions of the boundary

filters. Similarly, during reconstruction, zero insertion is used when wavelets on the whole real axis are used. When boundary filters are used, the multiplexed formula (36) is used instead, as in the decomposition stage. Finally, the reconstructed signal is obtained, as usual, that is, by adding the outputs of the low-pass and high-pass filters and dividing the sum by two at each scale [11].

V. Numerical Example

In this section we test and compare the denoising transforms developed in the previous sections. To this end, a noisy reference signal with sharp transients is introduced, which is denoised using both the non-redundant and redundant transforms on the half-axis proposed in this paper. The results obtained from denoising using a wavelet transform on the interval, taken from Ref. [12], [13], are also presented for comparison purposes.

Figure 6 shows the original signal. The steep jump after sample number 2000 makes this signal particularly relevant to wavelet-based denoising methods. This is because it is known that wavelet thresholding can be used to efficiently denoise piecewise regular signals [3], [4].

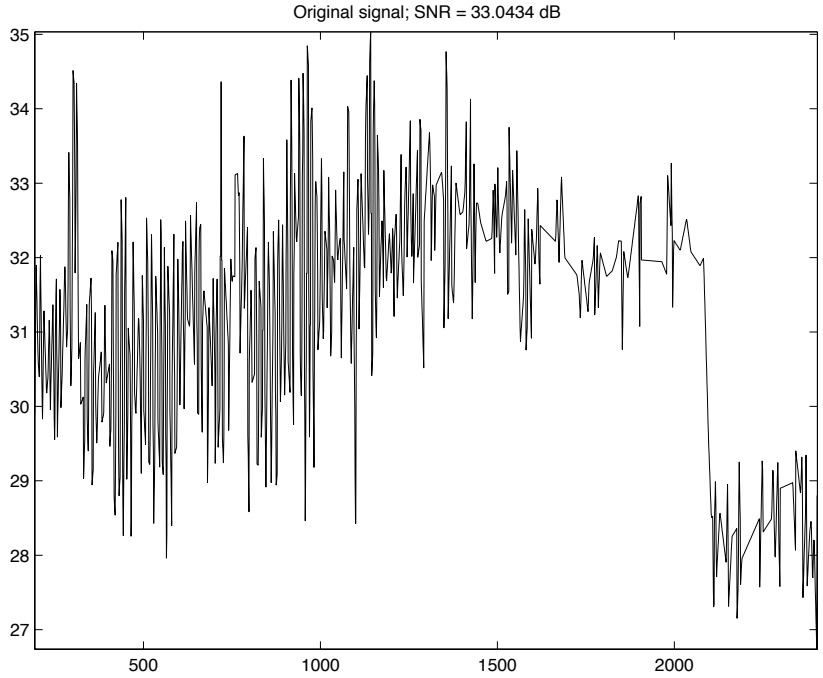
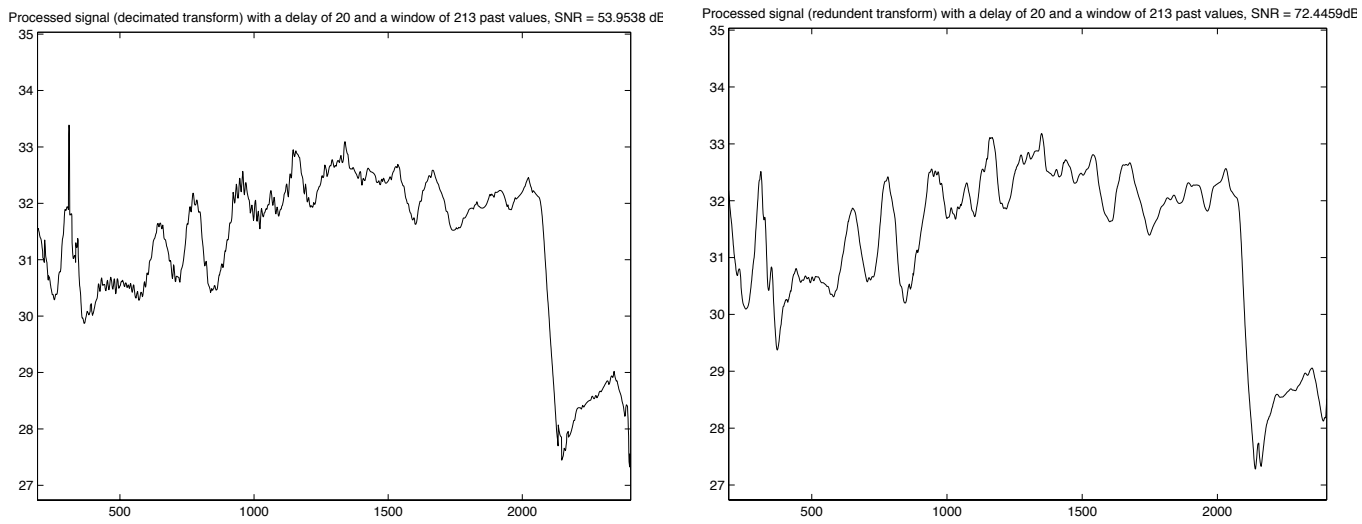


Fig. 6. Original signal. The horizontal axis indicates sample number. The sharp jump of the signal after 2000 has a duration of 60 samples.

The denoising procedure is performed as follows: at each time t, a time window $[t-T, t]$ of length T is used to select the data on which the transform is computed. The window length T is essentially determined by the number of scales used in the transform. It is equal to the length of the filter which is the longest at the fine scale, multiplied by 2^{j-1}, $j = 1$ being the finest scale. For this problem, with a choice of $j = 6$ scales, a maximum filter length of six and a delay of $\tau = 20$ samples, gives a window of $T = 213$ samples. Hard thresholding is then performed on the wavelet coefficients[4], that is, all $d_j[n]$ which are below a given threshold ε are set to zero. The signal is then reconstructed using the techniques of Section IV-A.3. Ideally, sampling the reconstructed signal at time t provides an estimate of the denoised signal without any delay. In practice, however, it was observed that the estimate is degraded at instants close to the present time. For this reason, a user-defined delay τ is introduced and the value of the reconstructed signal at time $t-\tau$ is taken as an estimate of the denoised signal at time $t-\tau$. The main purpose of the numerical experiments in this section is to find a delay τ which is smaller than the duration of any notable sharp transients of the signal, while still providing acceptable denoising levels. This scheme suggests a procedure of how online denoising could be performed in practice.

Figure 7(a) shows the effect of thresholding using the nonredundant transform of Section III. Figure 7(b) shows

[4]The number of scales and the threshold have been determined empirically. They are the same for all transforms.



(a) Nonredundant transform

(b) Redundant transform

Fig. 7. Denoising with wavelet transform over 6 scales, a hard threshold of 8 and a delay of $\tau = 20$ samples.

the results using the redundant transform with the same threshold value. One can see – at least visually – the benefits brought about by the redundancy. The signal to noise ratio (SNR) has been increased from 53.9 dB to 72.4 dB via the use of the redundant transform. Note that the SNR of the original signal is 33 dB. In Fig. 7 six scales have been used, with a hard threshold of 8 and a delay $\tau = 20$ samples. As a comparison, the large jump of the signal after 2000 has a duration of 60 samples.

For comparison, the Cohen-Daubechies-Vial-Jawerth (CDVJ) wavelets were used to denoise the signal of Fig. 6. Recall that the CDVJ wavelets [12], [13] are orthogonal wavelets on the interval and here have two vanishing moments. Nonetheless, the associated filters do not have vanishing moments, that is, polynomial sequences do not vanish through the high-pass filter. Figure 8 shows the result of denoising using hard thresholding with the CDVJ wavelet filters, and with the same threshold ($\varepsilon = 8$) and number of scales as before ($j = 6$). The delay is again taken to be $\tau = 20$ samples. It is evident that the performance is much poorer than the one obtained using the average interpolation scheme. In particular, the SNR is 36.34 dB versus a SNR of 72.4 dB for the redundant average interpolation scheme, and a SNR of 53.9 dB for the nonredundant scheme. We can improve the performance of the CDVJ filtering, and achieve a SNR closer to the ones of Fig. 7, by increasing the number of scales. Figure 9(a) shows that by increasing the number of scales from 6 to 8 we can increase the SNR to 52.2 dB, which is similar to the SNR achieved when using the nonredundant average interpolation algorithm. If, on the other hand, we want to achieve a SNR close to the SNR of the redundant transform, the number of scales to be processed increases to 10. The SNR for the latter case is 74.3 dB (see Fig. 9(b)), which is close to the SNR of 72.4 dB when using the redundant average interpolation scheme.

Since the computational complexity of any denoising algorithm increases with the number of scales involved in the processing, it is evident from these results that the proposed denoising algorithm outperforms CDVJ filtering since the quality of denoising is better when the same number of scales is used in both cases. Although one can, in principal, achieve SNRs similar to ones from average interpolation using CDVJ wavelet filtering, the required number of scales will most likely prohibit its implementation for on-line applications. It should be noted at this point that the number of computations does not increase merely because the computations are performed over a larger number of scales. Rather, the increased number of computations stems mainly from the increasingly larger windows of data to be processed as the number of scales increases. For instance, in Figure 9(b) only half the data are processed since the window is quite large (1024 samples from a total of 2424) in this case. A large window implies, in turn, a large delay before the filter can produce some output. It is well known that delays within a feedback loop are detrimental in terms of stability and performance of the closed-loop.

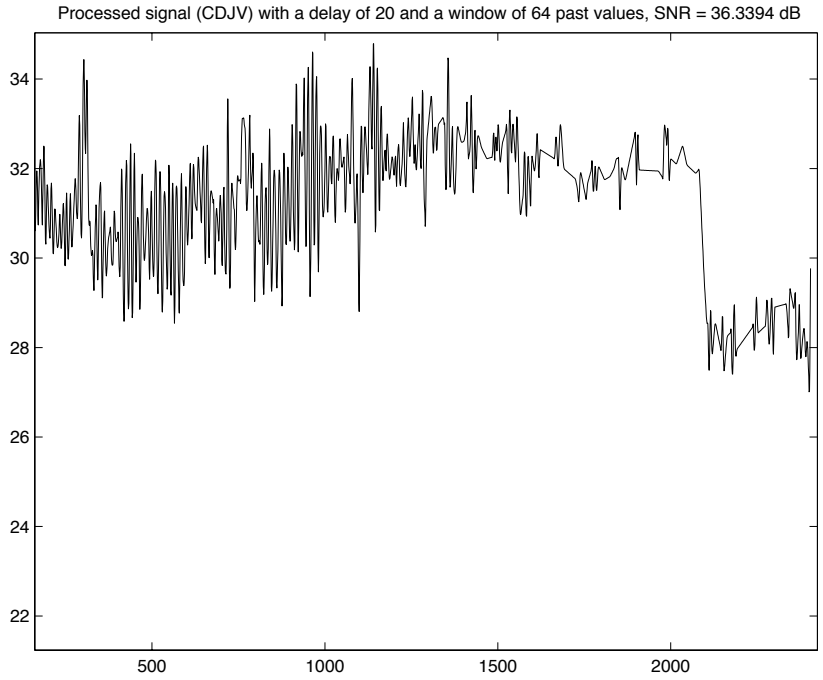
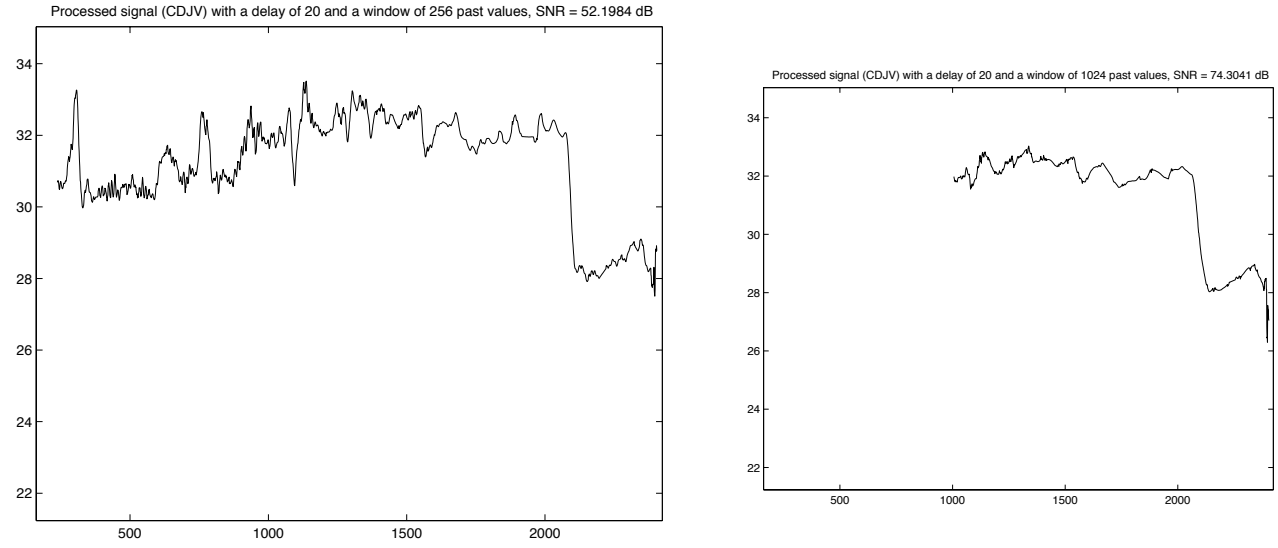


Fig. 8. Denoising with Cohen-Daubechies-Vial-Jawerth wavelets over 6 scales, a hard threshold of $\varepsilon = 8$ and a delay of $\tau = 20$ samples.



(a) CDVJ denoising with 8 scales

(b) CDVJ denoising with 10 scales

Fig. 9. Denoising with Cohen-Daubechies-Vial-Jawerth wavelets using 8 scales and 10 scales. A hard threshold of $\varepsilon = 8$ and a delay of $\tau = 20$ samples has been used in both cases. For the case of 10 scales only half the data is processed because of the very large window length.

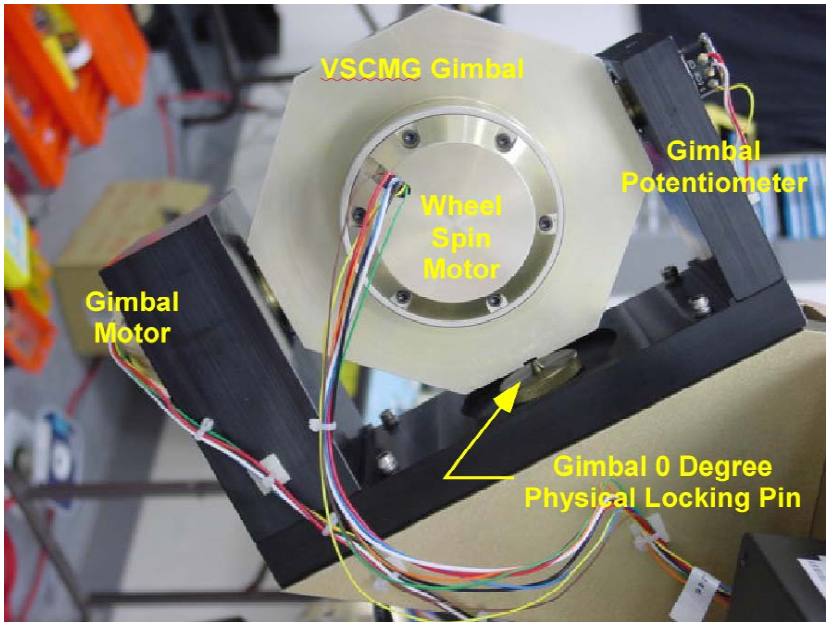


Fig. 10. Main components of the VSCMG assembly. In reaction wheel mode only the wheel DC motor is active while the gimbal remains fixed.

VI. Experimental Results

The greatest benefit of an *on-line* denoising algorithm, as the one proposed in this paper, is its potential for denoising signals in a feedback loop. Traditional orthogonal wavelet denoising algorithms introduce too much delay that may destabilize the closed-loop system. In general, a compromise must be reached between the level of acceptable delay and the requirement for noise removal.

In this section we report experimental results from the implementation of the algorithm of Section IV to denoise the angular velocity signal of a three-phase permanent magnet synchronous DC motor. The DC motor used in the experiments controls the wheel spin axis of a variable speed control moment gyro (VSCMG)[5]. A cluster of four VSCMGs is used to provide attitude control for the Integrated Attitude Control Simulator (IACS) located at the School of Aerospace Engineering at the Georgia Institute of Technology. IACS is a three-axial air-bearing experimental facility for simulating three-axis spacecraft attitude maneuvers. Four wheels are available for complete control about all three axis (with one redundant wheel). Figure 10 shows one of the four VSCMG modules used for the experiments. Details for the design, construction and other specifications for the IACS spacecraft simulator can be found in [14].

In reaction wheel mode, control is provided by accelerating or decelerating each of the four spin axis wheels. For accurate attitude control it is necessary that the motors promptly deliver the commanded control torques. Figures 11 and 12 show the open-loop motor responses to a square and a sinusoidal angular acceleration command, respectively. Clearly, the open-loop response of the spin DC motor is not satisfactory. This behavior is primarily due to bearing friction and other nonlinear effects.

Such discrepancies between commanded and delivered torque have deleterious effects in achieving tight pointing attitude requirements with the IACS. It was therefore deemed necessary to improve the torque command following performance for each motor by implementing a PI controller. The PI controller was designed to use the angular velocity error of the wheel (angular velocity measurements provided by a Hall sensor embedded in the DC motor) as an input in order to provide tight closed-loop torque (i.e., angular acceleration) control.

A C code of the algorithm of Section IV was written and implemented as an S-function in SIMULINK®. The Real-Time Workshop® toolbox was used to compile and generate the code from the complete SIMULINK® diagram and the xPCTarget® (with Embedded Option) toolbox was used to run the executable module in real-time.

[5]Each VSCMG actuator has two motors. One motor controls the wheel speed (reaction wheel mode) and the other controls the gimbal rate (CMG mode). Only the spin wheel axis was used in the experiments. The gimbal angle was kept constant by physically locking the gimbal.

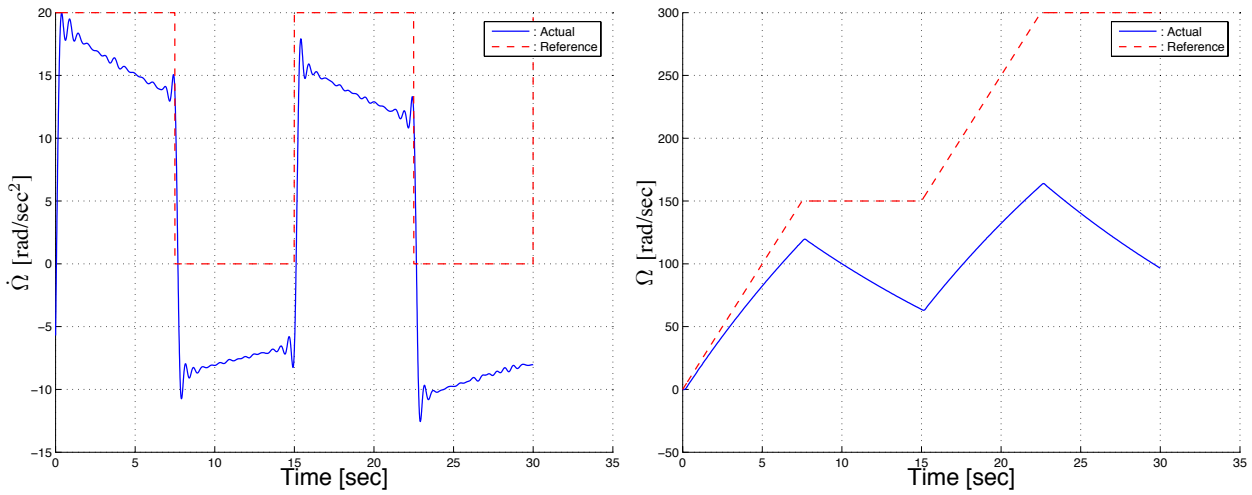


Fig. 11. Open-loop response of angular acceleration and velocity to a square-wave angular acceleration command.

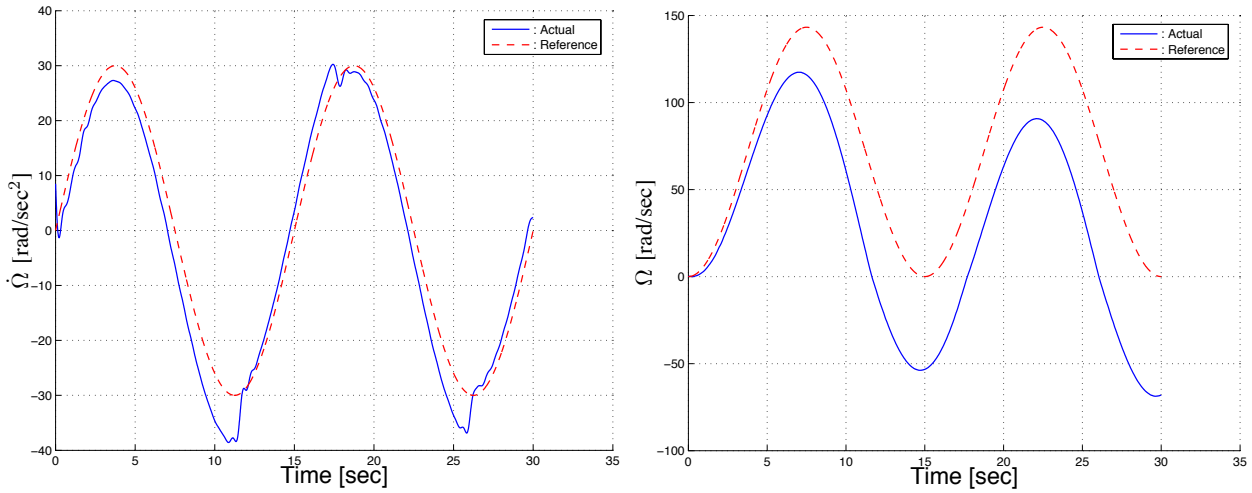


Fig. 12. Open-loop response of angular acceleration and velocity to a sinusoidal angular acceleration commands.

The results from three separate experiments are presented below.

During the first experiment, measurements of the angular velocity signal from the Hall sensor were processed off-line using the wavelet denoising algorithm. The purpose of those experiments was to estimate appropriate values for the number of scales, threshold and user-defined delay parameter τ for the wavelet filter.

Figure 13 shows typical angular velocity data from the Hall sensor. The results of denoising with a hard threshold of 50, a delay of 15 samples and for 4 and 6 scales are shown in Fig. 14.

During the second set of experiments, the denoised angular velocity signal was used as the input to a PI controller. The purpose of the PI controller is to achieve good tracking to torque (i.e., angular acceleration) commands.

Remark 1: Although in the presence of nonlinearities the denoising filter may introduce a small bias (hence the closed-loop system may not remain stable when denoised variables are used in lieu of their true values), nonetheless it was found in our experiments that such effects were negligible, and denoising did not alter the overall stability characteristics, except – of course – in cases when the delay induced by the denoising process was prohibitively large.

Figure 15 shows the simplified block diagram schematic of the PI/Motor interconnection. In the figure, the block labeled PWM contains the internal servo loop implemented by the motor amplifier. The measurements from the Hall sensor were noisy, with a very slow measurement update rate (about 3 Hz). This low update rate induces a significant delay (about 0.3 sec) in the overall closed-loop system that makes it rather challenging to control.

Figure 16 shows the results from the closed-loop control using the Hall sensor measurements. A wavelet filter

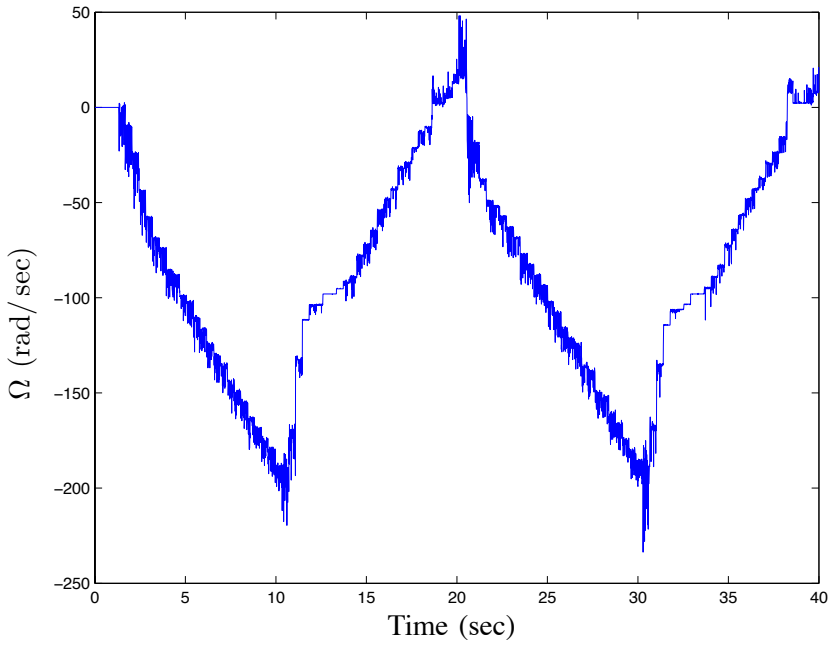
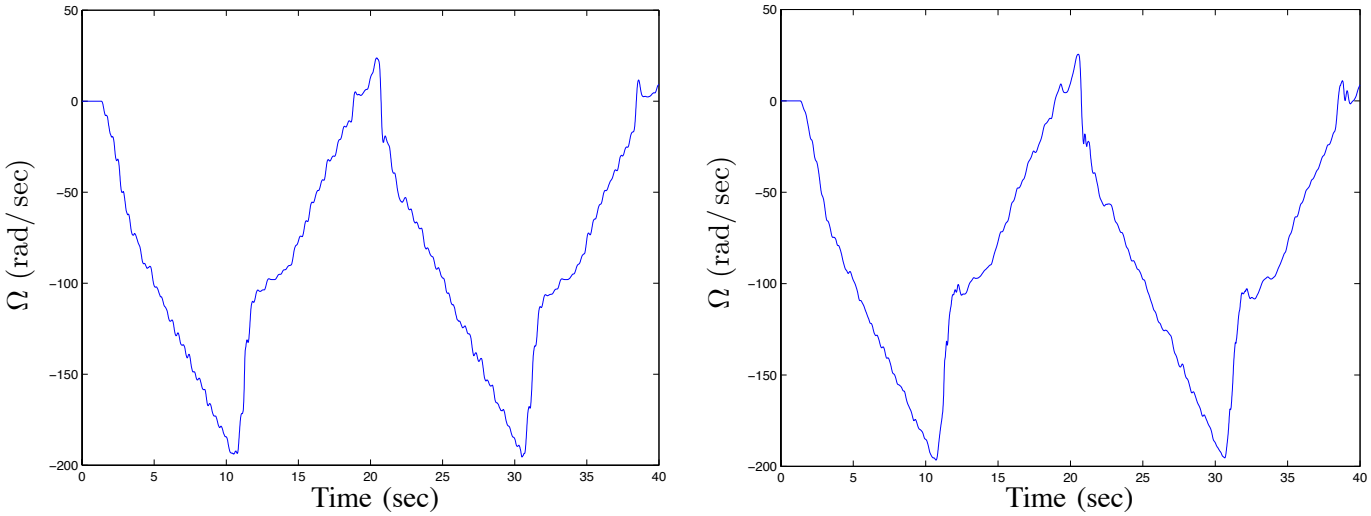


Fig. 13. Angular velocity measurements from Hall sensor.



(a) scales = 4 (b) scales=6

Fig. 14. Wavelet denoising of angular velocity signal from Hall sensor using a threshold of 50 and a delay of 15 samples.

with $j = 5$, $\tau = 15$ and threshold of 30 was used to clean up the signal. Figures 16 and 17 show the closed-loop response to a square wave and a sinusoidal angular acceleration command, respectively.

As shown in these figures, the torque following performance has been significantly improved, although owing to the low update rate (3 Hz) of the Hall sensor, the response exhibits some large overshoot and appears to be a bit sluggish. In order to improve the performance it was necessary to obtain measurements at higher rates. Therefore, a set of angular encoders (US Digital E4) were installed on each wheel. Angular velocity was obtained from numerical differentiation of angular position data of the wheel spin axis provided via the encoders at a sampling rate of 100 Hz. Although having a much better performance than the Hall sensors, numerical differentiation still introduced noise in the angular velocity signal. In addition, the type of noise induced by differentiation is of a different character than the one from Hall sensor, which was mainly due to electromagnetic interference from surrounding electronics and motor. The noise due to differentiation is mainly due to quantization and can be seen in Figure 19. The purpose of denoising was to remove this noise from the signal before it is used by the PI controller.

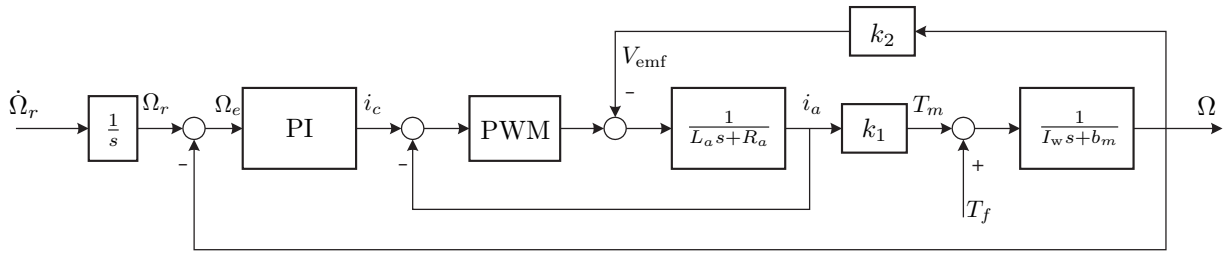


Fig. 15. Closed loop control for $\dot{\Omega}$ using velocity feedback.

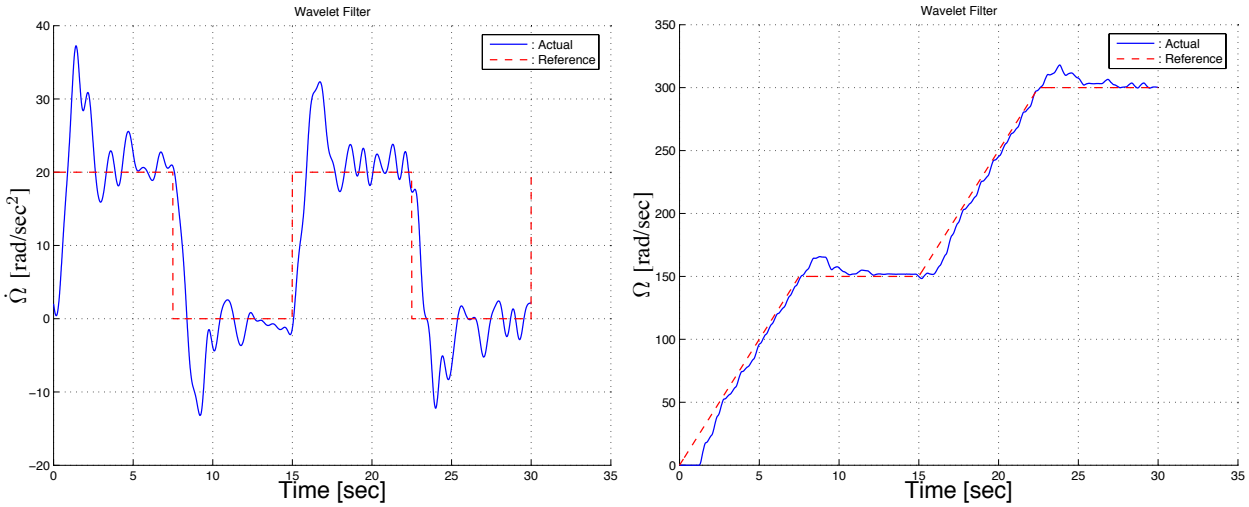


Fig. 16. Closed-loop control response of angular acceleration and velocity to square wave angular acceleration command. Hall sensor measurements.

The results of this experiment for a sawtooth angular velocity input are shown in Fig. 18. The actual and filtered signal in this case are very close to each other at this plot scale. Zooming in around $t = 15$ sec allows a more detailed examination of the results. Figure 19(a) shows the results of denoising over a 3 sec interval (300 samples). The dotted line shows the angular velocity data, and the solid line is the result after wavelet denoising using 4 scales, a threshold of 20 and a delay of 5 samples. The measured data has been smoothed out, and most of the noise has been removed. A small delay is evident because of the processing, and a small amount of noise has remained in the signal. This noise can be further removed at the expense of more delay. For instance, Fig. 19(b) shows the results of denoising using a delay of 15 samples or 0.15 sec.

The results from the feedback implementation of square-wave angular acceleration command of magnitude 20 rad/sec^2 are shown in Fig. 20, and from a sinusoidal angular acceleration command of amplitude 30 rad/sec^2 and frequency of 1/15 Hz are shown in Fig. 21. In Fig. 20(a) the command is shown by a dashed line and the actual response of the DC motor is shown by a solid line. The corresponding response of the angular velocity is shown in Fig. 20(b). Both Figs. 20 and 21 show very good tracking of the commanded signals.

Remark 2: Although the same, constant threshold has been used across all scales in all previous experiments, the proposed algorithm can be easily adapted to be used with a time-varying threshold. Several algorithms exist (e.g., the SURE algorithm [2, pp. 455-462]) which can adjust the threshold on-line based on an estimate of the noise variance. For the IACS the noise characteristics remain the same with time. Therefore the use of a constant value for the threshold was deemed sufficient. The reader should be aware of this additional flexibility of the proposed algorithm since in practice one may want to use an adaptive threshold, especially when the noise level is not known a priori or if it is changing.

VII. Conclusions

In this paper we propose a method for wavelet signal processing on the half-axis. The starting point for the development of the results in this paper is the method of average-interpolating polynomials. Using this method

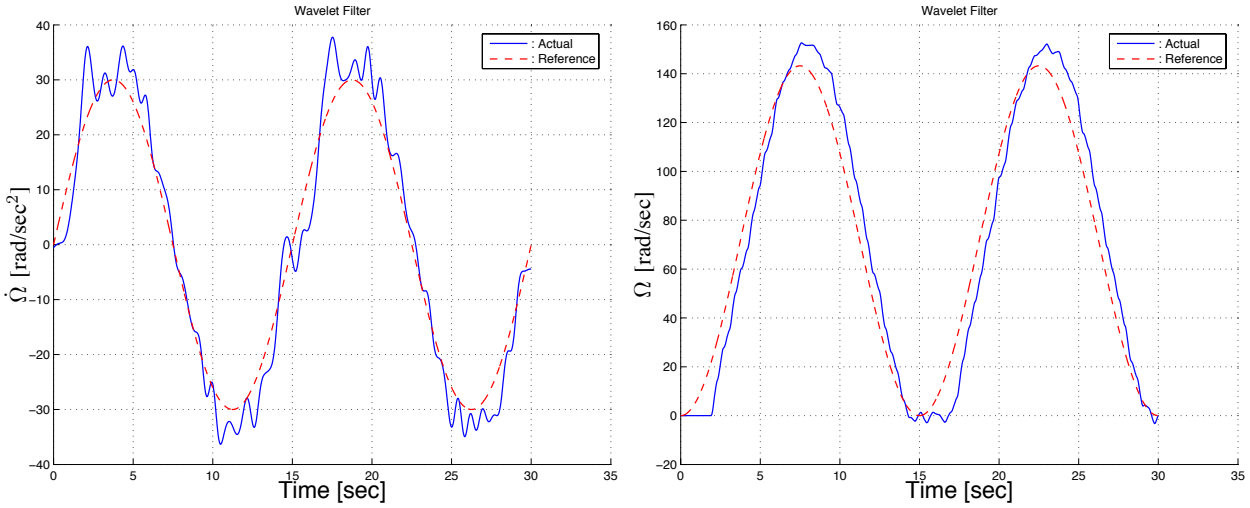


Fig. 17. Closed-loop control response of angular acceleration and velocity to sinusoidal angular acceleration command. Hall sensor measurements.

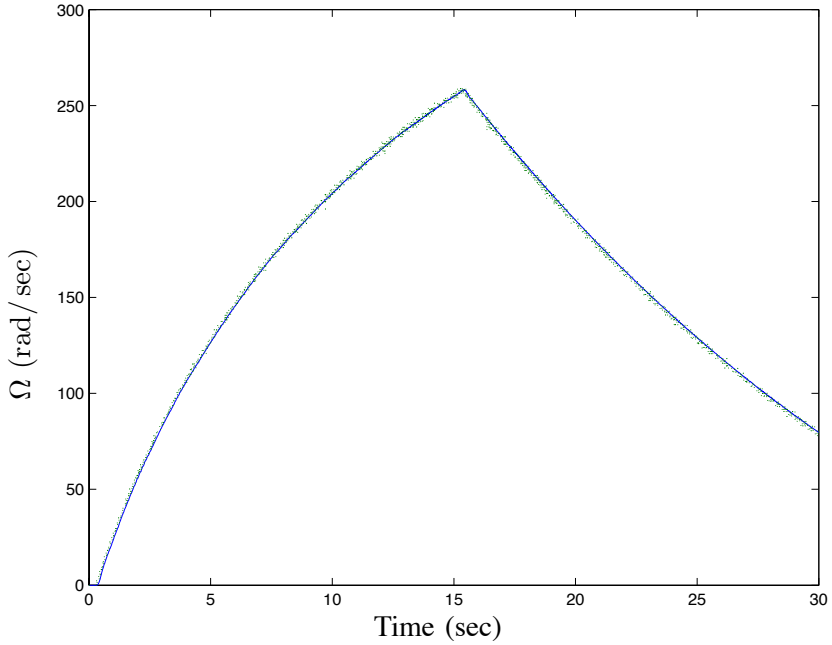
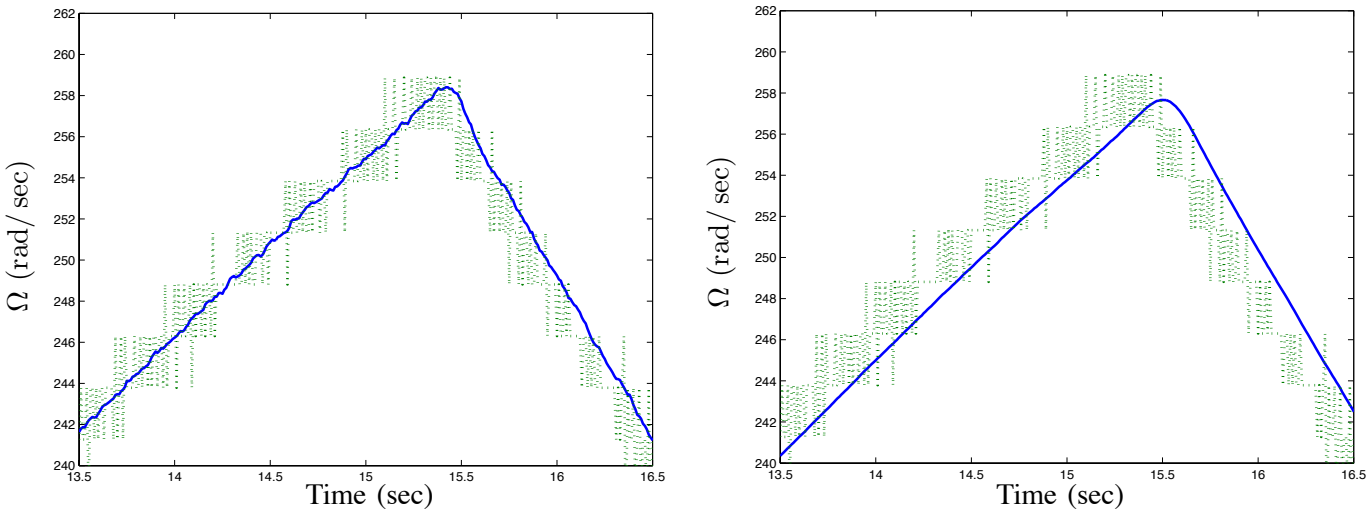


Fig. 18. Wavelet denoising of angular velocity signal using 4 scales and a threshold of 20. Encoder measurement feedback.

boundary effects arising from working in a semi-infinite interval can be handled is a straightforward manner. In addition, the algorithm can be easily implemented using wavelet filter banks. The motivation behind denoising on the half-axis stems from the need for on-line denoising for certain applications (e.g., within a feedback loop), where future values of the data are not available. It is well known that redundant transforms are typically superior to non-redundant ones for eliminating or reducing noise in a signal. We thus extend our results and we also provide a redundant wavelet transform on the half-axis. The superiority of the redundant transform is demonstrated via a numerical example of a very noisy signal with a sharp discontinuity. In addition, we provide experimental evidence from the use of the proposed wavelet method for on-line denoising of feedback signals. Specifically, experimental results from angular velocity and angular acceleration control of a brushless DC motor of a reaction wheel assembly using noisy angular velocity measurements are shown.

Acknowledgement: The second and third author acknowledge the support from NSF through awards CMS-0084954 and CMS-0510259.



(a) $\tau = 5$ (b) $\tau = 15$

Fig. 19. Wavelet denoising of angular velocity signal using 4 scales and a threshold of 20 (detailed view). Encoder measurement feedback.

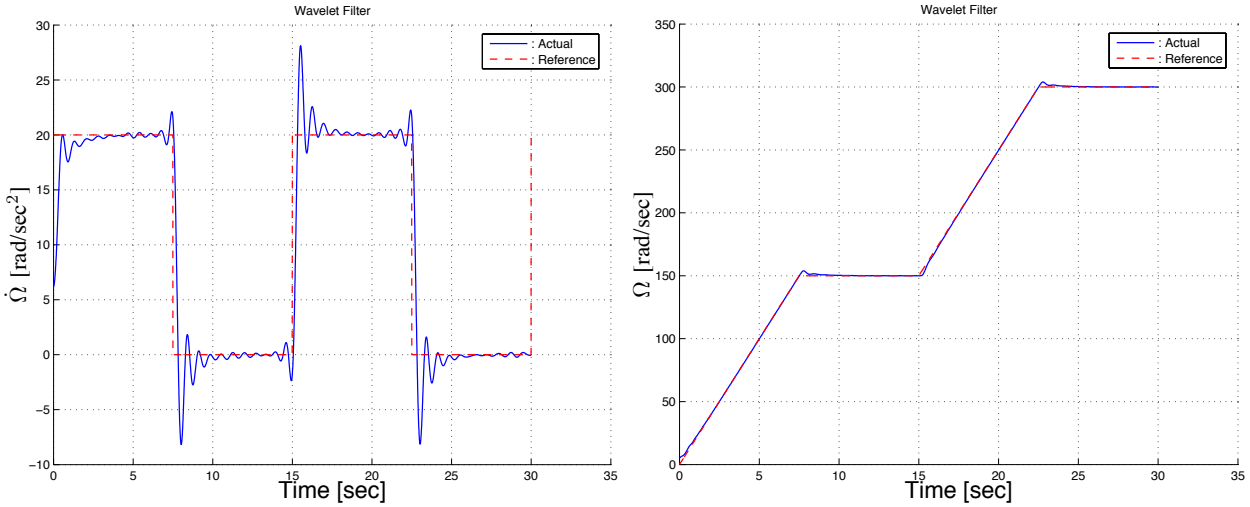


Fig. 20. Closed-loop response of angular acceleration and velocity to square-wave angular acceleration commands; on-line denoising of angular velocity signal using a wavelet filter with 4 scales, a threshold of 20 and a delay of 10

REFERENCES

[1] D. Donoho and I. Johnstone, "Ideal denoising in orthonormal basis chosen from a library of bases," *C. R. Academy of Science*, vol. 39, pp. 1317–1322, 1994. Serie I, Paris.

[2] S. Mallat, *A Wavelet Tour of Signal Processing*. New York: Academic Press, 1999.

[3] D. Donoho and I. Johnstone, "Ideal spatial adaptation via wavelet shrinkage," *Biometrika*, vol. 81, pp. 425–455, December 1994.

[4] R. Coifman and D. Donoho, "Translation invariant de-noising," in *Wavelets and Statistics* (A. Antoniadis and G. Oppenheim, eds.), Lecture Notes in Statistics, pp. 125–150, Berlin: Springer-Verlag, 1995.

[5] C. S. Burrus, R. A. Gopinath, and H. Guo, *Introduction to Wavelets and Wavelet Transforms*. New Jersey: Prentice-Hall, 1998.

[6] D. Donoho, "Smooth wavelet decompositions with blocky coefficient kernels," in *Recent Advances in Wavelet Analysis* (L. Schumaker and F. Ward, eds.), pp. 259–308, Academic Press, 1993.

[7] W. Sweldens and P. Schröder, *Building Your Own Wavelets at Home*, pp. 15–87. ACM, 1996. ACM SIGGRAPH Course Notes.

[8] A. Harten, "Multiresolution representation of cell-averaged data: A promotional review," in *Signal and Image Representation in Combined Spaces* (Zeevi and Coifman, eds.), Academic Press, 1997.

[9] M. Lang, H. Guo, J. E. Odegard, C. S. Burrus, and O. R. Wells, "Noise reduction using an undecimated discrete wavelet transform," *IEEE Signal Processing Letters*, vol. 3, no. 1, pp. 10–12, 1996.

[10] I. Daubechies, *Ten Lectures on Wavelets*. Philadelphia, PA: SIAM, 1992.

[11] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian, *Wavelets, Time-Frequency Methods and Phase Space*, ch. A Real-Time Algorithm for Signal Analysis with the Help of the Wavelet Transform, pp. 289–297. Berlin: Springer-Verlag, 1989.

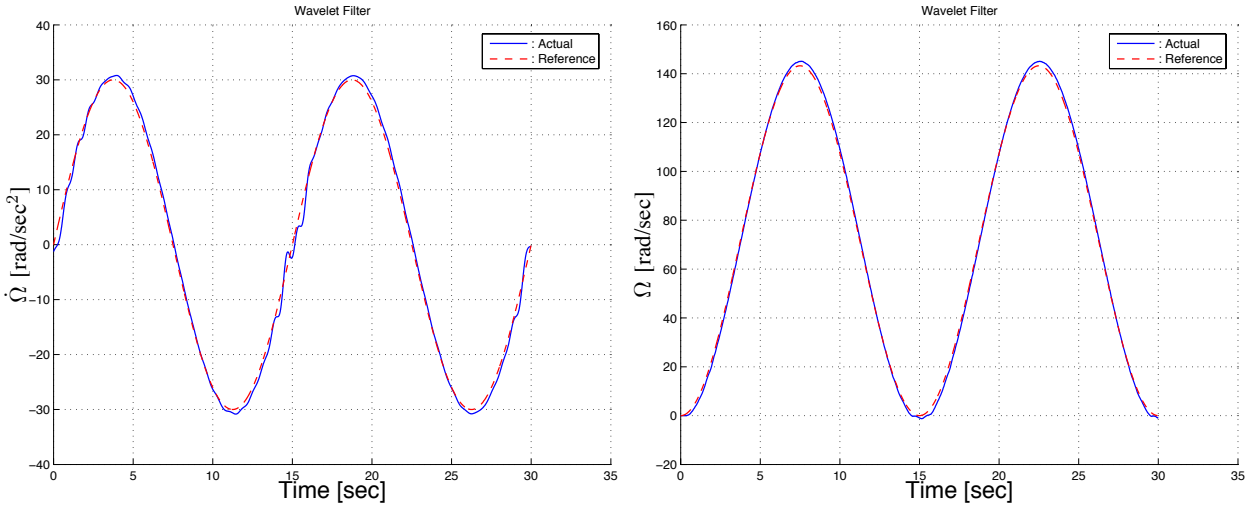


Fig. 21. Closed-loop response of angular acceleration and velocity to sinusoidal angular acceleration commands; on-line denoising of angular velocity signal using a wavelet filter with 4 scales, a threshold of 20 and a delay of 10

[12] A. Cohen, I. Daubechies, and P. Vial, "Wavelet bases on the interval and fast algorithms," *J. of Appl. and Comput. Harmonic Analysis*, vol. 1, pp. 54–81, 1993.

[13] A. Cohen, I. Daubechies, B. Jawerth, and P. Vial, "Multiresolution analysis, wavelets and fast algorithms on an interval," *Compt. Rend. Acad. Sci. Paris, A*, vol. 316, pp. 417–421, 1992.

[14] D. Jung and P. Tsiotras, "A 3-dof experimental test-bed for integrated attitude dynamics and control research," in *AIAA Guidance, Navigation and Control Conference*, (Austin, TX), 2003. AIAA Paper 03-5331.

APPENDIX

Below we show that the high-pass decomposition filter (15) satisfies a vanishing moments property and that the details oscillate.

a) Vanishing moments: It suffices to show that the detail coefficients are all zero when the input signal is a discrete polynomial of degree two. To this end, assume that $a_j[2n-3],\ldots,a_j[2n+2]$ is a polynomial sequence of degree less than or equal to two. That is, $a_j[2n-3],\ldots,a_j[2n+2]$ is a sequence of the averages of some polynomial $q(x)$ of degree less than or equal to two on the intervals $[k2^j,(k+1)2^j]$ for $k=2n-4,\ldots,2n+1$. Since the averages of averages on disjoint intervals are averages, the average-interpolating polynomial $p(x)$ at step (ii) of the high-pass filter construction necessarily satisfies $p(x)=q(x)$. Moreover, the average of $p(x)$ on $[2^j(2n-1),2^{j+1}n]$ is exactly equal to $a_j[2n]$, which proves that $d_{j+1}[n]=0$.

b) Oscillation of the details: Let us denote by $\tilde{p}_{j+1}[n]$ the average of the average-interpolating polynomial $p(x)$ on the interval $[2^{j+1}(n-1),2^j(2n-1)]$. Recall that, by construction, $a_{j+1}[n]/\sqrt{2}$ is both the average of $a_j[2n]$ and $a_j[2n-1]$ and the average of $p(x)$ on $[2^{j+1}(n-1),2^{j+1}n]$. Hence it is also the average of $p_{j+1}[n]$ and of $\tilde{p}_{j+1}[n]$, the latter two being the averages of $p(x)$ on the left and right half sub-intervals of $[2^{j+1}(n-1),2^{j+1}n]$; see Fig. 3. This yields

$$\frac{a_j[2n]+a_j[2n-1]}{2}=\frac{a_{j+1}[n]}{\sqrt{2}}=\frac{p_{j+1}[n]+\tilde{p}_{j+1}[n]}{2}. \tag{A.1}$$

This proves that the details are oscillating (compare with (12)), that is,

$$a_j[2n-1]-\tilde{p}_{j+1}[n]=-d_{j+1}[n]/\sqrt{2}. \tag{A.2}$$

Notice that (A.1) and (13) imply that

$$\tilde{p}_{j+1}[n]=\sqrt{2}\left(\frac{1}{16}a_{j+1}[n-1]+\frac{1}{2}a_{j+1}[n]-\frac{1}{16}a_{j+1}[n+1]\right). \tag{A.3}$$

Similarly, we next show that the high-pass decomposition filter of Section III has similar properties.

a) Vanishing moments: If $a_0[-4], \ldots, a_0[0]$ are the averages of a polynomial $q(x)$ of degree less than or equal to two on the intervals $[-5,-4], \ldots, [-1,0]$, then $a_1[-3]/\sqrt{2}$, $a_1[-1]/\sqrt{2}$ and $a_1[0]/\sqrt{2}$ are the averages of the interpolating polynomial $p(x)$ on the intervals $[-5,-3]$, $[-3,-1]$ and $[-2,0]$, respectively. Hence, by construction, $p(x) = q(x)$ and the detail is zero. This shows that the previous modification at the boundary preserves the vanishing moments property.

b) Oscillation of the details: Denote by $\tilde{p}_1[-2]$ the average of the interpolating polynomial $p(x)$ on $[-3,-2]$. Then

$$\frac{a_0[-2] + a_0[-1]}{2} = \frac{a_1[-1]}{\sqrt{2}} = \frac{p_1[-1] + \tilde{p}_1[-2]}{2}$$

since both quantities give the average of $p(x)$ over the interval $[-3,-1]$. This proves that

$$-d_1[-1] = \sqrt{2}(a_0[-2] - \tilde{p}_1[-2]) \tag{A.4}$$

and the details are oscillating. It can be easily verified that $\tilde{p}_1[-2]$ is given by

$$\tilde{p}_1[-2] = \sqrt{2}\left(\frac{1}{24}a_1[-3] + \frac{5}{8}a_1[-1] - \frac{1}{6}a_1[0]\right). \tag{A.5}$$