May, 1992 LIDS-P 2110

Research Supported By:

ARO grant DAAL03-86-K-0171
AFOSR grant 92-J-0002
ONR grant N00014-91-J-1004

Multiresolution Stochastic Models, Data Fusion, and Wavelet Transforms*

Chou, K.C.
Golden, S.A.
Willsky, A.S.

May 1992 LIDS-P-2110

MULTIRESOLUTION STOCHASTIC MODELS, DATA FUSION, AND WAVELET TRANSFORMS [1]

Kenneth C. Chou [2], S.A. Golden [3], and Alan S. Willsky[4]

Abstract

In this paper we describe and analyze a class of multiscale stochastic processes which are modeled using dynamic representations evolving in scale based on the wavelet transform. The statistical structure of these models is Markovian in scale, and in addition the eigenstructure of these models is given by the wavelet transform. The implication of this is that by using the wavelet transform we can convert the apparently complicated problem of fusing noisy measurements of our process at several different resolutions into a set of decoupled, standard recursive estimation problems in which *scale* plays the role of the time–like variable. In addition we show how the wavelet transform, which is defined for signals that extend from $-\infty$ to $+\infty$, can be adapted to yield a modified transform matched to the eigenstructure of our multiscale stochastic models over finite intervals. Finally, we illustrate the promise of this methodology by applying it to estimation problems, involving single and multi-scale data, for a first–order Gauss–Markov process. As we show, while this process is *not* precisely in the class we define, it can be well–approximated by our models, leading to new, highly parallel, and scale–recursive estimation algorithms for multi–scale data fusion. In addition our framework extends immediately to 2D signals where the computational benefits are even more significant.

[1]This work was performed at the M.I.T. Laboratory for Information and Decision Systems with partial support from the Army Research Office under Grant DAAL03-86-K-0171, from the Air Force Office of Scientific Research under Grant AFOSR-92-J-0002, and from the Office of Naval Research under Grant N00014-91-J-1004.

[2]SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025.

[3]Sparta, Inc., 23041 Avenida de la Carlota, Suite 400, Laguna Hills, CA 92653.

[4]Laboratory for Information and Decision Systems and Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge,MA 02139, USA.

1 Introduction and Background

Multiresolution methods in signal and image processing have experienced a surge of activity in recent years, inspired primarily by the emerging theory of multiscale representations of signals and wavelet transforms [1, 7, 8, 9, 12, 15, 16, 31, 25]. One of the lines of investigation that has been sparked by these developments is that of the role of wavelets and multiresolution representations in statistical signal processing [5, 6, 10, 11, 14, 26, 27, 28, 29, 30]. In some of this work (e.g. [10, 11, 14, 24]) the focus is on showing that wavelet transforms simplify the statistical description of frequently used models for stochastic processes, while in other papers (e.g. [28, 29, 30, 5, 6, 26, 27]) the focus is on using wavelets and multiscale signal representations to construct new types of stochastic processes which not only can be used to model rich classes of phenomena but also lead to extremely efficient optimal processing algorithms using the processes' natural multiscale structure. The contributions of this paper lie in both of these arenas, as we both construct a new class of multiscale stochastic models (for which we also derive new and efficient algorithms) and demonstrate that these algorithms are extremely effective for the processing of signals corresponding to more traditional statistical models.

In [26, 27] a new class of fractal, *1/f*-like stochastic processes is constructed by synthesizing signals using wavelet representations with coefficients that are uncorrelated random variables with variances that decrease geometrically as one goes from coarse to fine scales. The wavelet transform, then, whitens such signals, leading to efficient signal processing algorithms. The model class we describe here not only includes these processes as a special case but also captures a variety of other stochastic phenomena and signal processing problems of considerable interest. In particular by taking advantage of the time-like nature of scale, we construct a class of processes that are Markov in scale rather than in time. The fact that scale is time-like for our models allows us to draw from the theories of dynamic systems and recursive estimation in developing efficient, highly parallelizable algorithms for performing optimal estimation. For our models we develop a smoothing algorithm, an algorithm which computes estimates of a multiscale process based on multiscale data, which uses the wavelet transform to transform the overall smoothing problem into a set of independently computable, small 1D standard smoothing problems.

If we consider smoothing problems for the case in which we have measurements of the full signal at the finest scale alone, this algorithmic structure reduces to a modest generalization of that in [27]–i.e., the wavelet transform whitens the measurements, allowing extremely efficient optimal signal processing. What makes even this modest contribution of some significance is the richness of the class of processes to which it can be applied, a fact we demonstrate in this paper. Moreover the methodology we describe directly yields efficient scale-recursive algorithms for optimal processing and fusion of measurements at *several* scales with only minimal increase in complexity as compared to the single scale case. This contribution should be of considerable value in applications such as remote sensing, medical imaging, and geophysical exploration, in which one often encounters data sets of different modalities (e.g. infrared and radar data) and resolutions. Furthermore, although we focus on 1D signals in this paper, the fact that scale is a time-like variable is true as well in the case of 2D, where similar types of models lead to efficient recursive and iterative algorithms; the computational savings in this case are even more dramatic than in the case of 1D.

In order to define some of the notation we need and to motivate the form of our models, let us briefly recall the basic ideas concerning wavelet transforms. The multiscale representation of a continuous signal $f(x)$ consists of a sequence of approximations of that signal at finer and finer scales where the approximations of $f(x)$ at the mth scale consists of a weighted sum of shifted and compressed (or dilated) versions of a basic <u>scaling function</u> $\phi(x)$:

$$f_m(x) = \sum_{n=-\infty}^{+\infty} f(m,n) 2^{\frac{m}{2}} \phi(2^m x - n) \tag{1}$$

For the $(m+1)$st approximation to be a refinement of the mth, we require $\phi(x)$ to be representable at the next scale:

$$\phi(x) = \sum_n \sqrt{2} h(n) \phi(2x - n) \tag{2}$$

As shown in [8], $h(n)$ must satisfy several conditions for (1) to be an orthonormal series and for several other properties of the representation to hold. In particular $h(n)$ must be the impulse response of a quadrature mirror filter (QMF) [8, 23], where the condition for $h(n)$ to be a QMF is as follows.

$$\sum_k h(k) h(k - 2n) = \delta_n \tag{3}$$

By considering the incremental detail added in obtaining the $(m+1)$st scale approximation from the mth, we arrive at the wavelet transform based on a single function $\psi(x)$ that has the property that the full set of its scaled translates $\left\{2^{m/2}\psi(2^m x - n)\right\}$ form a complete orthonormal basis for L^2. In [8] it is shown that ϕ and ψ are related via an equation of the form

$$\psi(x) = \sum_n \sqrt{2} g(n) \phi(2x - n) \tag{4}$$

where $g(n)$ and $h(n)$ form a *conjugate mirror filter* pair [23], and that

$$f_{m+1}(x) = f_m(x) + \sum_n d(m,n) 2^{m/2} \psi(2^m x - n) \tag{5}$$

Note that $g(n)$ and $h(n)$ must obey the following algebraic relationships.

$$\begin{aligned} \sum_k g(k)h(k-2n) &= 0 && (6)\\ \sum_k g(k)g(k-2n) &= \delta_n && (7)\\ \sum_k h(n)h(n-2k) + \sum_k g(n)g(n-2k) &= \delta_n && (8) \end{aligned}$$

If we have the coefficients $\{f(m+1,\cdot)\}$ of the $(m+1)$st-scale representation we can "peel off" the wavelet coefficients at this scale and carry the recursion one complete step by calculating the coefficients $\{f(m,\cdot)\}$ at the next coarser scale. The resulting wavelet analysis equations are

$$f(m,n) = \sum_k h(2n-k) f(m+1,k) \triangleq (H_m f(m+1,\cdot))_n \tag{9}$$

$$d(m,n) = \sum_k g(2n-k) f(m+1,k) \triangleq (G_m f(m+1,\cdot))_n \tag{10}$$

where the operators H_m and G_m are indexed with the subscript m to denote that they map sequences at scale $m+1$ to sequences at scale m[1]. From

[1]Note that for the case of infinite sequences the operators as defined here are precisely equivalent for each scale; i.e. they are not a function of m. However, we adhere to this notation for the reasons that a) we may allow for the QMF filter to *differ* at each scale and b) for the case of finite-length sequences the operators *are* in fact different at every scale due to the fact that the the number of points differ from scale to scale.

(3,7,6) we have the following

$$H_m H_m^* = I \tag{11}$$
$$G_m G_m^* = I \tag{12}$$
$$H_m G_m^* = 0 \tag{13}$$

where "$*$" denotes the adjoint of the operator.

Reversing this process we obtain the synthesis form of the wavelet transform in which we build up finer and finer representations via a coarse-to-fine scale recursion:

$$f(m+1,n) = \sum_k h(2k-n)f(m,k) + \sum_k g(2k-n)d(m,k) \tag{14}$$

Expressed in terms of the operators H_m and G_m we have

$$f(m+1,n) = (H_m^* f(m,\cdot))_n + (G_m^* f(m,\cdot))_n \tag{15}$$

or

$$H_m^* H_m + G_m^* G_m = I \tag{16}$$

which is an expression of eq.(8) in operator form.

Thus, we see that the synthesis form of the wavelet transform defines a *dynamical* relationship between the coefficients $f(m,n)$ at one scale and those at the next, with $d(m,n)$ acting as the input. Indeed this relationship defines an infinite lattice on the points (m,n), where $(m+1,k)$ is connected to (m,n) if $f(m,n)$ influences $f(m+1,k)$. This structure is illustrated in Figure 1 for the case where $h(n)$ is a 4-tap filter, where each level of the lattice represents an approximation of our signal at some scale m. Note that the dynamics in (14) are now with respect to *scale* rather than time, and this provides us with a natural framework for the construction of multiscale stochastic models.

In particular if the input $d(m,n)$ is taken to be a white sequence, then $f(m,n)$ is naturally Markov in scale (and, in fact, is a Markov random field with respect to the neighborhood structure defined by the lattice). Indeed the class of 1/f-like processes considered in [26, 27] is exactly of this form with the additional specialization that the variance of $d(m,n)$ decreases geometrically as m increases. While allowing more general variance structures on $d(m,n)$ expands the set of processes we can construct somewhat, a bit more thought yields a substantially greater extension. First of all, with wavelet coefficients which are uncorrelated, (14) represents a first-order recursion in scale that is driven by white noise. However, as we know from time series analysis, white-noise-driven first-order systems yield a comparatively small class of processes which can be broadened considerably if we allow higher-order dynamics, which can either be captured as higher-order difference equations in scale, or, as we do here, as first-order *vector* difference equations. As further motivation for such a framework, note that in sensor fusion problems one wishes to consider collectively an entire set of signals or images from a suite of sensors. In this case one is immediately confronted with the need to use higher-order models in which the actual observed signals may represent samples from such a model at *several* scales, corresponding to the differing resolutions and modalities of individual sensors.

Thus the perspective we adopt here is to view multiscale representations more abstractly than in the wavelet transform, by using the notion of a state model in which the state at a particular node in our lattice captures the features of signals up to that scale that are relevant for the "prediction" of finer-scale approximations. As we will see, this leads us to a model class that includes the wavelet representation of (14) as a special case and that leads to extremely efficient processing algorithms. In the next section we introduce our framework for state space models evolving in scale, and we show that the wavelet transform plays a central role in the analysis of the eigenstructure of these processes. This fact is then used in Section 3 to construct scale-recursive, and highly parallel algorithms for optimal smoothing for such processes given data at possibly a number of different scales. In Section 4 we then investigate an important issue in the practical application of these ideas, namely the issue of applying the wavelet transform to finite-length data. The typical approach is to base the transform on cyclic convolutions rather than on linear convolutions and to perform the scale by scale recursion up to some specified coarse scale. We present a more general perspective on the problem of adapting the wavelet transform to finite-length data which includes as a *special case* the approach using cyclic convolutions as well as other approaches which provide modifications of the wavelet transform to provide Karhunen-Loeve expansions of windowed multiscale processes. In Section 5 we illustrate the promise of our multiscale estimation framework by applying it to smoothing problems for 1st-order Gauss-Markov processes, including problems involving the estimation of such processes based on multiresolution data. Additional experimental results for other processes, including 1/f-like fractal processes can be found in [5].

2 Multiscale Processes on Lattices

In this section we define our class of multiscale state space models and analyze their eigenstructure. We develop our ideas for the case of the infinite lattice, i.e., for signals and wavelet transforms of infinite extent. In Section 4 we discuss the issue of adapting the wavelet transform and our results to the case of finite-length data.

Consider an infinite lattice corresponding to a wavelet whose scaling filter, $h(n)$, is an FIR filter of length P. Recall that in the wavelet transform of a signal f each level of the lattice can be viewed as the domain of an l^2 sequence, namely $f(m,\cdot) \stackrel{\triangle}{=} f(m)$. In our generalization of the dynamic model (15) we think of each level of the lattice as corresponding to a vector l^2 sequence $x(m) = x(m,\cdot)$, where $x(m,n)$ can be thought of as the vector state of our multiscale model at lattice node (m,n) and $x(m)$ as the representation at scale m of the phenomenon under study.

To motivate our general model let us first consider the synthesis equation (14) driven by uncorrelated wavelet coefficients $d(m,n)$, where the variances are constant along scale but varying from scale to scale. In this case we obtain the following stochastic dynamic state model where we define the scale index m from an initial coarse scale L to a finest scale, M, and where we assume that the coarsest scaling coefficients $f(L,n)$ are uncorrelated. Thus, with $x(m)$ corresponding to $f(m,\cdot)$ and $w(m)$ to $d(m,\cdot)$ we have for $m = L, L+1, ..., M-1$

$$\begin{aligned} E[x(L)x(L)^T] &= \tilde{\Lambda}_L && (17) \\ &= \tilde{\lambda}_L I && \\ x(m+1) &= H_m^* x(m) + G_m^* w(m) && (18) \\ E[w(i)w(i)^T] &= \Lambda_i && (19) \\ &= \lambda_i I \ , \ i = L, L+1, ... M-1 && \end{aligned}$$

Note that if we let $\lambda_i = \sigma^2 2^{-\gamma i}$, this model is precisely the one considered in [26, 27] for modeling a $1/f$-type process with spectral parameter γ.

The model class we consider in this paper is a natural generalization of (17)-(19). In specifying our model we abuse notation and use the same notation H_m, G_m for the coarsening and differencing operators defined as in (9,10) where $f(m,n)$ is a vector. With this convention our model takes the following form for $m = L, L+1, ...M-1$

$$\begin{aligned} E[x(L)x(L)^T] &= \mathcal{P}_x(L) & (20)\\ x(m+1) &= H_m^* \mathcal{A}(m+1)x(m) + \mathcal{B}(m+1)w(m+1) & (21)\\ E[w(i)w(j)^T] &= \mathcal{Q}(i)\delta_{i-j}\ ,\ i = L+1, ...M & (22) \end{aligned}$$

where

$$\begin{aligned} \mathcal{A}(m) &\triangleq diag(..., A(m), ...A(m), ...) & (23)\\ \mathcal{B}(m) &\triangleq diag(..., B(m), ...B(m), ...) & (24)\\ \mathcal{Q}(m) &\triangleq diag(..., Q(m), ...Q(m), ...) & (25)\\ \mathcal{P}_x(L) &\triangleq diag(..., P_x(L), ...P_x(L), ...) & (26) \end{aligned}$$

and where $A(m), B(m), Q(m)$, and $P_x(L)$ are *finite-dimensional* matrices representing the system matrix, the process noise matrix, the process noise covariance matrix, and the initial state covariance matrix, respectively.

If we let $x(m,n)$ and $w(m,n)$ denote the components of $x(m)$ and $w(m)$, respectively, the model (21) can be written in component form as

$$x(m+1,n) = \sum_k h(2k-n)A(m)x(m,k) + B(m)w(m,n) \qquad (27)$$

where the $w(m,n)$ are white with covariance $Q(m)$. Comparing (21), (27) to (14), (18) allows us to make several observations. In particular, (21) is indeed a generalization of (18). This might not appear to be obvious since the driving noise term in (21) does not involve the differencing operator G_m^*. However, if we examine (21) and define

$$\mu(m) = G_m^* w(m) \qquad (28)$$

then, thanks to (7) or (12) and the fact that the covariance of $w(m,n)$ varies with m but *not* n, we find that the $\mu(m,n)$ are white with covariance $\lambda_m I$, that varies with m but not n. That is, (18) is exactly of the form of (21), with $x(m,n)$, $w(m,n)$ scalar and $A(m) = B(m) = 1$. By augmenting our model class to include finite-dimensional state vectors defined on lattices, we allow for the possibility of higher-order models. This allows us to consider a considerably richer class of processes which is parametrized by the matrices corresponding to our state model. Note also that this model bears resemblance to Laplacian Pyramid schemes[4] where the added detail in going from one scale to the next is *not* constrained by the differencing operator G_m.

As mentioned in the introduction, the model (17)-(19) yields a covariance with eigenstructure given by the wavelet transform, and it is this fact that is used in [26, 27] to develop efficient processing algorithms for $1/f$-like processes. As we now state, the same is true for our more general model (21), where in this generalization we focus on what can be thought of as the "block" eigenstructure of the process $x(m)$. That is, if $x(m,n)$ is d-dimensional, the discrete wavelet transform of the signal $x(m,\cdot)$ for each m yields an uncorrelated set of random vectors. To see this, let us examine the covariance $R_{xx}(m)$ of $x(m)$, where, if we use the fact that the block-diagonal operators $\mathcal{A}(m)$, $\mathcal{B}(m)$, $\mathcal{Q}(m)$, $\mathcal{P}_x(L)$ and their adjoints commute with the operators H_m, H_m^*, we find that

$$\begin{aligned} R_{xx}(m) &\triangleq E[x(m)x(m)^T] \qquad (29)\\ &= (\overline{\Phi}(m-1,L)\mathcal{P}_x(L)\overline{\Phi}^*(m-1,L))(\prod_{i=L}^{m-1} H_i^*)(\prod_{m-1}^{i=L} H_i)\\ &+ \sum_{k=L+1}^{m-1} (\overline{\Phi}(m-1,k)\mathcal{B}(k)\mathcal{Q}(k)\mathcal{B}^*(k)\overline{\Phi}^*(m-1,k))(\prod_{i=k}^{m-1} H_i^*)(\prod_{i=m-1}^{k} H_i)\\ &+ \mathcal{B}(m)\mathcal{Q}(m)\mathcal{B}^*(m) \end{aligned}$$

where for $i \geq j$

$$\overline{\Phi}(i,j) = \begin{cases} I & i = j \\ \mathcal{A}(i)\overline{\Phi}(i-1,j) & i > j \end{cases} \qquad (30)$$

Let us next define a sequence of block unit vectors as follows.

$$\delta_i^j \triangleq [\ldots, 0_d, \ldots, 0_d, \underbrace{I_d}_{i\text{th}}, 0_d, \ldots 0_d, \ldots]^T \qquad (31)$$

where the superscript j is used to denote that the vector (in $(l^2)^d$) corresponds to the jth scale of the lattice and where I_d is the $d \times d$ identity matrix (and

0_d the $d \times d$ zero matrix). Note that in the present context the superscript j is completely superfluous notation. However, in Section 4 we consider the extension of the ideas presented here and in the next section to the case of finite length signals. In this case the (finite) lengths of the vectors $x(m)$ vary with m, as will the dimensions of the block unit vectors analogous to (31). As we will see, with changes in definitions of quantities such as δ_i^j, the following result on the block-eigenstructure of $R_{xx}(m)$, as well as the results of Section 3, hold in essentially unchanged form for the finite length case.

Lemma 2.1 *The block vectors $\overline{v}_i^L(m)$, $v_n^l(m)$ for $l = L, ...m-1$ and for $i, n \in \mathcal{Z}$ are block-eigenvectors of the correlation matrix at scale m, $R_{xx}(m)$, where*

$$\overline{v}_i^L(m) \triangleq (\prod_{j=L}^{m-1} H_j^*)\delta_i^L \tag{32}$$

and

$$v_n^l(m) \triangleq (\prod_{i=l+1}^{m-1} H_i^*)G_l^*\delta_n^l \tag{33}$$

The following holds:

$$\begin{aligned} R_{xx}(m)\overline{v}_i^L(m) &= diag(..., \tilde{\lambda}_L(m), ...\tilde{\lambda}_L(m), ...)\overline{v}_i^L(m) & (34) \\ R_{xx}(m)v_n^l(m) &= diag(..., \lambda_l(m), ...\lambda_l(m), ...)v_n^l(m) & (35) \end{aligned}$$

for $l = L, ...m-1$, $i, n \in \mathcal{Z}$ where $\tilde{\lambda}_L(m), \lambda_l(m)$ are $d \times d$ matrices of the form

$$\tilde{\lambda}_L(m) = \sum_{k=L+1}^{m} (\Phi(k,L)B(k)Q(k)B^T(k)\Phi^T(k,L)) + \Phi(m-1,L)P_x(L)\Phi^T(m-1,L) \tag{36}$$

$$\lambda_l(m) = \sum_{k=l+1}^{m} (\Phi(k,l)B(k)Q(k)B^T(k)\Phi^T(k,l)) \tag{37}$$

where

$$\Phi(i,j) = \begin{cases} I & i = j \\ A(i)\Phi(i-1,j) & i > j \end{cases} \tag{38}$$

The details of the proof of this result can be found in [5]. Rather than present these details here, we provide a simple interpretation of them which will also be of value in understanding the optimal estimation algorithms presented in the next section. Specifically, for $m = L, ..., M-1$ define the following transformed quantities, where j in (39) runs from l through $m-1$ and k from $-\infty$ to $+\infty$ in (39), (40) :

$$z_{j,k}(m) \triangleq (v_k^j(m))^T x(m) \tag{39}$$

$$u_{L,k}(m) \triangleq (v_k^L(m))^T x(m) \tag{40}$$

From (15) and (32, 33) we see that what we have done is to take the partial discrete wavelet transform decomposition of each component of the d-dimensional $x(m,n)$ viewed, for m fixed, as a discrete-time signal with time index n. That is, starting with $x(m,\cdot)$ we have first peeled off the finest level of the wavelet transform $z_{m-1,k}(n)$, viewed as a discrete signal with index k, and then have computed successively coarser wavelet coefficients, through $z_{L,k}(m)$, also producing the coarse scaling coefficients $u_{L,k}(m)$.

What the lemma states is that all of these variables, i.e., the set of d-dimensional vectors $z_{j,k}(m)$ and $u_{L,k}(m)$ for all values of j and k are mutually uncorrelated. Indeed much more is true, in that these variables in fact evolve in a statistically decoupled manner as a function of m. Specifically, if we transform both sides of (21) as in (39), (40), and use (11)-(13) and the commutativity of the operators in (23)-(26) with H_m^* and G_m^*, we obtain the following transformed dynamics for $m = L, ..., M-1$. First, the coarsest signal components evolve according to

$$u_{L,k}(m+1) = A(m+1)u_{L,k}(m) + B(m+1)r_{L,k}(m+1) \tag{41}$$

where $r_{L,k}(m+1)$ is the coarse wavelet approximation of $w(m+1,\cdot)$, i.e., [2]

$$r_{L,k}(m+1) = (v_k^L(m+1))^T w(m+1) \tag{42}$$

and where the initial condition for (41) is simply the coarse scale signal itself:

$$u_{L,k}(L) = x(L,k) \tag{43}$$

[2]Note that we are again abusing notation since $w(m,n)$ may have dimension $q \neq d$. In this case the only needed modifications are to use q-dimensional identity and zero matrices in (31) and similar q-dimensional versions of the operators H_k and G_k.

Next, the wavelet coefficiets at different scales evolve according to

$$z_{j,k}(m+1) \;=\; A(m+1)z_{j,k}(m) + B(m+1)s_{j,k}(m+1) \tag{44}$$

for $j = L, ..., m-1$, and where

$$s_{j,k}(m+1) = (v_k^j(m+1))^T w(m+1) \tag{45}$$

are the corresponding wavelet coefficients of $w(m+1,\cdot)$. Finally, as we move to scale $m+1$ from scale m we must initialize one additional finer level of wavelet coefficients:

$$z_{m,k}(m+1) = B(m+1)s_{m,k}(m+1) \tag{46}$$

What we have determined via this transformation is a set of decoupled ordinary dynamic systems (41), (44) where this set is indexed by k in (41) and by $j = L, ...m-1$ and k in (44), where m plays the role of the "time" variable in each of these models, and where at each "time" $m+1$ we initialize a new set of models as in (46). More importantly, thanks to (11)–(13) and (25), (26) the initial conditions and driving noises, $x(L,k)$, $r_{L,k}(m+1)$, and $s_{j,k}(m+1)$ are mutually uncorrelated with covariances $P_x(L), Q(m+1)$, and $Q(m+1)$, respectively, so that these models are statistically decoupled as well. From this fact the lemma essentially follows immediately, with the identification of $\tilde{\lambda}_L(m)$ as the covariance of $u_{L,k}(m)$ (for any value of k), which evolves according to

$$\tilde{\lambda}_L(m+1) = A(m+1)\tilde{\lambda}_L(m)A^T(m+1) + B(m+1)Q(m+1)B^T(m+1) \tag{47}$$

with initial condition

$$\tilde{\lambda}_L(L) = P_x(L) \tag{48}$$

Similarly, $\lambda_l(m)$ is the covariance of $z_{l,k}(m)$ which also evolves according to a Lyapunov equation exactly as in (47), but from initial condition at the $(l+1)$st scale:

$$\lambda_l(l+1) = B(l+1)Q(l+1)B^T(l+1) \tag{49}$$

3 Wavelet-Based Multiscale Optimal Smoothing

In this section we consider the problem of optimally estimating one of our processes as in (21) given sensors of varying SNR's and differing resolutions.

An example where this might arise is in the case of fusing data form sensors which operate in different spectral bands. We formulate this sensor fusion problem as an optimal smoothing problem in which the optimally smoothed estimate is formed by combining noisy measurements of our lattice process at various scales. In other words each sensor is modeled as a noisy observation of our process at some scale of the lattice.

Consider the following multiscale measurements for $m = L, L+1, ...M$.

$$y(m) = \mathcal{C}(m)x(m) + v(m) \tag{50}$$

where

$$\begin{aligned}
\mathcal{C}(m) &\triangleq diag(..., C(m), ...C(m), ...) && (51)\\
\mathcal{R}(m) &\triangleq diag(..., R(m), ...R(m), ...) && (52)\\
E[v(i)v(j)^T] &= \mathcal{R}(i)\delta_{i-j} && (53)
\end{aligned}$$

and where $C(m)$ is a $b \times d$ matrix and $R(m)$ is a $b \times b$ matrix representing the covariance of the additive measurement noise.. Note that the number of sensors, the resolution of each sensor, and the precise spectral characteristics of each sensor are represented in the matrix $C(m)$. For example if there were simply one sensor at the finest scale, then $C(m) = 0$ for all m except $m = M$.

We define the **smoothed** estimate, denoted as $x^s(m)$, to be the expected value of $x(m)$ conditioned on $y(i)$ for $i = L, L+1, ...M$; i.e.

$$x^s(m) = E[x(m)|y(L), ...y(M)] \tag{54}$$

We define the **coarse-to-fine filtered** estimate, to be the expected value of $x(m)$ conditioned on $y(i)$ for $i = L, L+1, ...m$; i.e

$$\hat{x}(m|m) = E[x(m)|y(L), ...y(m)] \tag{55}$$

We define the **coarse-to-fine one-step predicted** estimate to be the expected value of $x(m)$ conditioned on $y(i)$ for $i = L, L+1, ...m-1$; i.e

$$\hat{x}(m|m-1) = E[x(m)|y(L), ...y(m-1)] \tag{56}$$

From standard Kalman filtering theory, we can derive a recursive filter with its associated Riccati equations, where the recursion in the case of our lattice models is in the scale index m. We choose to solve the smoothing problem via the Rauch-Tung-Striebel(RTS) algorithm[22]. This gives us a correction sweep that runs recursively from fine to coarse scales with the initial condition of the recursion being the final point of the Kalman filter. The following equations describe the "down" sweep, i.e. the filtering step from coarse to fine scales.

For $m = L, ..., M$:

$$\begin{aligned}
\hat{x}(m|m-1) &= H^*_{m-1}\mathcal{A}(m)\hat{x}(m-1|m-1) && (57)\\
\hat{x}(m|m) &= \hat{x}(m|m-1) + \mathcal{K}(m)[y(m) - \mathcal{C}(m)\hat{x}(m|m-1)] && (58)\\
\mathcal{K}(m) &= \mathcal{P}(m|m-1)\mathcal{C}^*(m)\mathcal{S}(m) && (59)\\
\mathcal{S}(m) &= (\mathcal{C}(m)\mathcal{P}(m|m-1)\mathcal{C}^*(m) + \mathcal{R}(m))^{-1} && (60)\\
\mathcal{P}(m|m-1) &= H^*_{m-1}\mathcal{A}(m)\mathcal{P}(m-1|m-1)\mathcal{A}^*(m)H_{m-1} && \\
&+ \mathcal{B}(m)\mathcal{Q}(m)\mathcal{B}^*(m) && (61)\\
\mathcal{P}^{-1}(m|m) &= \mathcal{P}^{-1}(m|m-1) + \mathcal{C}^*(m)\mathcal{R}^{-1}(m)\mathcal{C}(m) && (62)
\end{aligned}$$

with initial conditions

$$\begin{aligned}
\hat{x}(L|L-1) &= 0 && (63)\\
\mathcal{P}(L|L-1) &= \mathcal{P}_x(L) && (64)
\end{aligned}$$

We also have the following equations for the correction sweep of the Rauch-Tung-Striebel algorithm, i.e. the "up" sweep from fine to coarse scales.

For $m = M-1, M-2, ...L+1, L$:

$$\begin{aligned}
x^s(m) &= \hat{x}(m|m) + \mathcal{P}(m|m)\mathcal{A}^*(m+1)H_m\mathcal{P}^{-1}(m+1|m)[x^s(m+1) - \hat{x}(m+1|m)] && (65)\\
\mathcal{P}^s(m) &= \mathcal{P}(m|m) + E(m)\left[\mathcal{P}^s(m+1) - \mathcal{P}(m+1|m)\right]E^*(m) && (66)\\
E(m) &= \mathcal{P}(m|m)\mathcal{A}^*(m+1)H_m\mathcal{P}^{-1}(m+1|m) && (67)
\end{aligned}$$

with initial conditions

$$\begin{aligned}
x^s(M) &= \hat{x}(M|M) && (68)\\
\mathcal{P}^s(M) &= \mathcal{P}(M|M) && (69)
\end{aligned}$$

Note that we could equally have chosen to start the RTS algorithm going from fine to coarse scales followed by a correction sweep from coarse to fine, i.e.

an up-down rather than the down-up algorithm just described. This involves defining the filtered and one-step predicted estimates in the direction from fine to coarse rather than coarse to fine. Similarly we could also construct a so-called two-filter algorithm [22] consisting of parallel upward and downward filtering step. Details on these variations are found in [5].

The smoothing algorithm described to this point involves a single smoother for the entire state sequence $x(m)$ at each scale. However, if we take advantage of the eigenstructure of our process and, more specifically, the decoupled dynamics developed at the end of the preceding section, we can transform our smoothing problem into a set of decoupled 1D RTS smoothing problems which can be computed in parallel. Specifically, define the following transformed quantities.

$$\hat{z}_{j,k}(m|m-1) \triangleq (v^j_k(m))^T \hat{x}(m|m-1) \quad (70)$$

$$\overline{P}_{j,k}(m|m-1) \triangleq (v^j_k(m))^T \mathcal{P}(m|m-1) v^j_k(m) \quad (71)$$

$$\hat{z}_{j,k}(m|m) \triangleq (v^j_k(m))^T \hat{x}(m|m) \quad (72)$$

$$\overline{P}_{j,k}(m|m) \triangleq (v^j_k(m))^T \mathcal{P}(m|m) v^j_k(m) \quad (73)$$

$$\hat{u}_{L,k}(m|m-1) \triangleq (\overline{v}^L_k(m))^T \hat{x}(m|m-1) \quad (74)$$

$$\tilde{P}_{L,k}(m|m-1) \triangleq (\overline{v}^L_k(m))^T \mathcal{P}(m|m-1) \overline{v}^L_k(m) \quad (75)$$

$$\hat{u}_{L,k}(m|m) \triangleq (\overline{v}^L_k(m))^T \hat{x}(m|m) \quad (76)$$

$$\tilde{P}_{L,k}(m|m) \triangleq (\overline{v}^L_k(m))^T \mathcal{P}(m|m) \overline{v}^L_k(m) \quad (77)$$

$$z^s_{j,k}(m) \triangleq (v^j_k(m))^T x^s(m) \quad (78)$$

$$P^s_{j,k}(m) \triangleq (v^j_k(m))^T \mathcal{P}^s(m) v^j_k(m) \quad (79)$$

$$\tilde{z}^s_{L,k}(m) \triangleq (v^L_k(m))^T x^s(m) \quad (80)$$

$$\tilde{P}^s_{L,k}(m) \triangleq (v^L_k(m))^T \mathcal{P}^s(m) v^L_k(m) \quad (81)$$

These quantities represent the transformed versions of the predicted, filtered, and smoothed estimates in the Rauch-Tung-Striebel algorithm, along with their respective error covariances, in the transform domain. We also need to represent the transformed data, where the data at each scale, $y(m)$ has components which are finite-dimensional vectors of dimension $b \times 1$. We represent these vectors using eigenvectors as in (32), (33), where in this case the blocks in (31) are $b \times b$:

$$\overline{y}_{j,k}(m) \triangleq (v_k^j(m))^T y(m) \quad (82)$$

$$\tilde{y}_{L,k}(m) \triangleq (\overline{v}_k^L(m))^T y(m) \quad (83)$$

As before for each scale m, where $m = L+1, ...M$, the index ranges are $j = L, ..., m-1$ and $-\infty < k < \infty$. That is, for each m other than at the coarsest scale, L, we transform our quantities so that they involve eigenvectors whose coarsest scale is L.

We now can state the following, which follows directly from the analysis in the preceding section :

Algorithm 3.1 *Consider the smoothing problem for a lattice defined over a finite number of scales, labeled from coarse to fine as $m = L, L+1, ...M$. The following set of equations describes the solution to the smoothing problem, transformed onto the space spanned by the eigenvectors of $R_{xx}(m)$, in terms of independent standard Rauch-Tung Striebel smoothing algorithms.*

DOWN SWEEP:

For $j = L, L+1, ...M-2$ and $k \in \mathcal{Z}$:

$$\hat{z}_{j,k}(m|m-1) = A(m)\hat{z}_{j,k}(m-1|m-1) \quad (84)$$

$$\overline{P}_{j,k}(m|m-1) = A(m)\overline{P}_{j,k}(m-1|m-1)A^T(m) + B(m)Q(m)B^T(m) \quad (85)$$

$$m = j+2, j+3, ...M$$

with the initial conditions for $j = L, L+1, ...M-1$ and $k \in \mathcal{Z}$

$$\hat{z}_{j,k}(j+1|j) = 0 \quad (86)$$

$$\overline{P}_{j,k}(j+1|j) = B(j+1)Q(j+1)B^T(j+1) \quad (87)$$

For $j = L, L+1, ...M-1$ and $k \in \mathcal{Z}$:

$$\hat{z}_{j,k}(m|m) = \hat{z}_{j,k}(m|m-1) + \overline{K}_{j,k}(m)(\overline{y}_{j,k}(m) - C(m)\hat{z}_{j,k}(m|m-1)) \quad (88)$$

$$\overline{P}_{j,k}^{-1}(m|m) = \overline{P}_{j,k}^{-1}(m|m-1) + C^T(m)R^{-1}(m)C(m) \quad (89)$$

$$m = j+1, j+2, ...M$$

$$\overline{K}_{j,k}(m) \triangleq (v_k^j(m))^T \mathcal{K}(m)\mathcal{V}_k^j(m) \quad (90)$$

For $k \in \mathcal{Z}$ we have

$$\begin{aligned}
\hat{u}_{L,k}(m|m-1) &= A(m)\hat{u}_{L,k}(m-1|m-1) && (91)\\
\tilde{P}_{L,k}(m|m-1) &= A(m)\tilde{P}_{L,k}(m-1|m-1)A^T(m) + B(m)Q(m)B^T(m) && (92)\\
m &= L+1, L+2, \ldots M
\end{aligned}$$

with the initial conditions

$$\begin{aligned}
\hat{u}_{L,k}(L|L-1) &= 0 && (93)\\
\tilde{P}_{L,k}(L|L-1) &= P_x(L) && (94)
\end{aligned}$$

For $k \in \mathcal{Z}$ we have

$$\begin{aligned}
\hat{u}_{L,k}(m|m) &= \hat{u}_{L,k}(m|m-1) + \tilde{K}_{L,k}(m)(\tilde{y}_{L,k}(m) - C(m)\hat{u}_{L,k}(m|m-1)) && (95)\\
\tilde{P}^{-1}_{L,k}(m|m) &= \tilde{P}^{-1}_{L,k}(m|m-1) + C^T(m)R^{-1}(m)C(m) && (96)\\
m &= L, L+1, \ldots M\\
\tilde{K}_{L,k}(m) &\triangleq (\overline{v}^j_k(m))^T \mathcal{K}(m)\overline{\mathcal{V}}^j_k(m) && (97)
\end{aligned}$$

UP SWEEP:
For $j = L, L+1, \ldots M-1$ and $k \in \mathcal{Z}$

$$\begin{aligned}
z^s_{j,k}(m) &= \hat{z}_{j,k}(m|m) + \overline{P}_{j,k}(m|m)A^T(m+1)\overline{P}^{-1}_{j,k}(m+1|m)[z^s_{j,k}(m+1) - \hat{z}_{j,k}(m+1|m)] && (98)\\
P^s_{j,k}(m) &= \overline{P}_{j,k}(m|m) + \overline{E}_{j,k}(m)\left[P^s_{j,k}(m+1) - \overline{P}_{j,k}(m+1|m)\right]\overline{E}^T_{j,k}(m) && (99)\\
\overline{E}_{j,k}(m) &= \overline{P}_{j,k}(m|m)A^T(m+1)\overline{P}^{-1}_{j,k}(m+1|m) && (100)\\
m &= M-1, M-2, \ldots j+2, j+1 && (101)
\end{aligned}$$

with initial condition

$$\begin{aligned}
z^s_{j,k}(M) &= \hat{z}_{j,k}(M|M) && (102)\\
P^s_{j,k}(M) &= \overline{P}_{j,k}(M|M) && (103)
\end{aligned}$$

desired at some scale $L \leq m \leq\lt M$, we use the wavelet synthesis equation (15) to construct it from its coarse scale approximation $\hat{z}^s_{L,k}(m)$ and its finer scale wavelet coefficients $z^s_{j,k}$ for scales $j = L, ..., m-1$. Thus, the overall procedure is of complexity $O(lN)$.

Let us make several closing comments. First, as the preceding complexity analysis implies, the algorithm we have described can be adapted to finite length data, with appropriate changes in the eigenvectors/wavelet transform to account for edge effects. This extension is discussed in the next section. Secondly, note that if only finest-scale data are available (i.e. only $C(M) \neq 0$), our smoothers degenerate to coefficient-by-coefficient static estimators (i.e. each wavelet coefficient in (82) , (83), at scale $m = M$ is used separately to estimate the corresponding component of $x(M)$), which is an algorithm of exactly the same structure as that in [27] for the particular choice of parameters in the scalar version of our model corresponding to $1/f$-like processing.

Finally, it is important to note that the transform method of parallelizing the smoothing problem, used here and in [27], requires the matrix $\mathcal{C}(m)$ in (50) to have constants along the diagonal for all m, i.e. that the same measurements are made at all points at any particular scale. The case of missing data at a given scale is an example in which this structure is violated. This is relevant to situations in which one might want to use coarse data to interpolate sparsely distributed fine data. This problem can be handled via an alternate set of efficient algorithms using models based on homogeneous trees. We refer the reader to [5, 30] for details.

4 Finite Length Wavelet Transforms

In this section we discuss the problem of adapting the wavelet transform, thus far defined only for infinite sequences, to the case of finite-length sequences, i.e. producing a transform that maps finite-length sequences into finite-length sequences. This is a topic of considerable current interest in the wavelets literature, [31], as the effects of windowing in wavelet transforms are not as well-understood as those for Fourier analysis. To begin, note that both the analysis and synthesis equations, (9,10,14), for computing the wavelet and scaling coefficients are defined as operations on infinite length sequences. Adapting these equations to the case of finite length se-

For $k \in \mathcal{Z}$

$$\tilde{z}^s_{L,k}(m) = \hat{u}_{j,k}(m|m) + \tilde{P}_{j,k}(m|m)A^T(m+1)\tilde{P}^{-1}_{j,k}(m+1|m)[\tilde{z}^s_{j,k}(m+1) - \hat{u}_{j,k}(m+1|m)] \tag{104}$$

$$\tilde{P}^s_{L,k}(m) = \tilde{P}_{j,k}(m|m) + \overline{E}'_{j,k}(m)\left[\tilde{P}^s_{j,k}(m+1) - \tilde{P}_{j,k}(m+1|m)\right](\overline{E}'_{j,k})^T(m) \tag{105}$$

$$\overline{E}'_{j,k}(m) = \tilde{P}_{j,k}(m|m)A^T(m+1)\tilde{P}^{-1}_{j,k}(m+1|m) \tag{106}$$

$$m = M-1, M-2, \ldots L+1, L \tag{107}$$

with initial condition

$$\tilde{z}^s_{L,k}(M) = \hat{u}_{L,k}(M|M) \tag{108}$$

$$\tilde{P}^s_{L,k}(M) = \tilde{P}_{L,k}(M|M) \tag{109}$$

Note that our algorithm is highly efficient in that we have transformed the problem of smoothing what are, in principle, infinite dimensional or, in the case of windowed data, very high-dimensional vectors, to one of smoothing in parallel a set of *finite* dimensional vectors. Also, the smoothing procedure takes place in *scale* rather than in time, and for finite data of length N this interval is at most of order $logN$, since each successively coarser scale involves a decimation by a factor of 2. Note also that as we move to finer scales we pick up additional levels of detail corresponding to the new scale components (46) introduced at each scale. This implies that the smoothers in our algorithm smooth data over scale intervals of differing lengths: of length roughly $logN$ for the coarsest components (since data at all scales provide useful information about coarse scale features) to shorter length scale intervals for the finer scale detail (since data at any scale is of use only for estimating detail at that scale or at *coarser* scales, but *not* at finer scales). This structure is illustrated in Figure 2.

Let us next analyze the complexity of our overall algorithm for smoothing our lattice processes. We first transform our data using the wavelet transform which is fast: $O(lN)$ where N is the number of points at the finest scale and l is the length of the QMF filter. We then perform in parallel our 1D smoothers. Even if these smoothers are computed serially the total computation is $O(lN)$. After performing the parallel 1D smoothers on these transformed variables an additional inverse transformation is required, which is performed again using the inverse wavelet transform. That is if $\hat{x}^s(m)$ is quences *while preserving both the orthogonality and the invertibility* of the transformation proves to be non-trivial for the following reason. Take a 10-point sequence, $x(n)$, and consider performing its wavelet transform using a QMF filter, $h(n)$, of length four. To compute the scaling coefficients at the next coarsest scale we apply (9) to $x(n)$, resulting in a scaling coefficient sequence, $c(n)$, which is of length 6 (the linear convolution results in a 13-point sequence, while the downsampling by a factor of two reduces this to a 6-point sequence). Similarly, by applying (10) to $x(n)$ we get a wavelet coefficient sequence, $d(n)$, which is also of length 6. Thus, the overall transformation from the nonzero portion of $\{x(n)\}$ to the nonzero portions of $\{c(n), d(n)\}$ in this case is a map from $\mathcal{R}^{10}$ to $\mathcal{R}^{12}$, which makes it impossible for it to be invertible. This example is illustrated Figure 3, where $x(n)$ is defined as indicated on the first level of a truncated lattice and $\{c(n), d(n)\}$ are mapped into the second level where the lattice branches are illustrated for the case where the operators H_i, G_i correspond to a QMF filter, $h(n)$, of length four and only branches connecting to points in the nonzero portion of $x(n)$ are shown.

Thus, we can already see the fundamental problem in trying to develop an orthonormal matrix transformation based on the wavelet transform. At each scale we must have a well-defined orthonormal transformation from our approximation at that scale into its scaling coefficients and its wavelet coefficients at the next coarsest scale. To see how this can be done it is sufficient to focus on our previous example involving the map from $x(n)$ into $\{c(n), d(n)\}$. We can write the transformation in our example explicitly as follows. We denote our 4-tap QMF filter, h, as a row vector $[\ h_0\ \ h_1\ \ h_2\ \ h_3\ ]$. Similarly, our filter, g, is denoted as $[\ g_0\ \ g_1\ \ g_2\ \ g_3\ ]$ where from (6) - (8) a valid choice of g is

$$[\ g_0\ \ g_1\ \ g_2\ \ g_3\ ] = [\ h_3\ \ -h_2\ \ h_1\ \ -h_0\ ] \tag{110}$$

If we think of the non-zero portion of our sequence $x(n)$ as a vector, x, in $\mathcal{R}^{10}$ and the non-zero portions of $c(n), d(n)$ as vectors, c and d, in $\mathcal{R}^6$, our maps $x(n) \mapsto c(n)$ and $x(n) \mapsto d(n)$ can be thought of as the following 6×10

matrices.

$$H \triangleq \begin{bmatrix} h_2 & h_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 \end{bmatrix} \tag{111}$$

$$G \triangleq \begin{bmatrix} g_2 & g_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ g_0 & g_1 & g_2 & g_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & g_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & g_0 & g_1 \end{bmatrix} \tag{112}$$

where

$$c = Hx \tag{113}$$

$$d = Gx \tag{114}$$

Note that c and d are *precisely* the non-zero portions of the sequences one obtains by applying the operators H_i, G_i to $x(n)$. Thus, we can in fact reconstruct $x(n)$ from c, d using our synthesis equation, eq.(16). In matrix notation

$$x = H^T c + G^T d \tag{115}$$

If we denote our overall map $x \mapsto c, d$ as the 12×10 matrix

$$U \triangleq \begin{bmatrix} H \\ G \end{bmatrix} \tag{116}$$

then (115) says that $U^T U = I$. Note, however, that it is *not* the case that $UU^T = I$, since U is not even square. That is, while it is true that the finite dimensional version of (16), namely $H^T H + G^T G = I$, holds, the following conditions do *not* hold:

$$HH^T = I \tag{117}$$

$$GG^T = I \tag{118}$$

$$GH^T = 0 \tag{119}$$

The failure of these conditions to hold is due primarily to the first and last rows of H and G. In Figure 3 these correspond to the averaging performed at the edges of both ends of the lattice. Note that the rows of H are mutually orthogonal and the rows of G are mutually orthogonal. The reason for (117,118) is simply the fact that the edge rows of H and G are not normalized so that their inner products equal one. The reason for (119) is the fact that the edge rows of G are not orthogonal to the edge rows of H.

If we want our overall transformation, U, to be orthonormal, we must somehow eliminate two of its rows. Note that if we eliminate the first and last rows of the matrix H we get

$$\tilde{H} \triangleq \begin{bmatrix} h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 \end{bmatrix} \tag{120}$$

In this case (117) and (119) *do* hold with $\tilde{H}$ replacing H, but (118) does not quite hold due to the fact that the the first and last rows of G are not properly normalized. Examining G in detail and using the QMF property in (7) we see that

$$GG^T = \begin{bmatrix} a & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & b \end{bmatrix} \tag{121}$$

where

$$a = g_2^2 + g_3^2 \tag{122}$$
$$b = g_0^2 + g_1^2 \tag{123}$$

Thus, we can satisfy (118) simply by normalizing the first and last rows of G by a and b, respectively.

The resulting transformation maps our length 10 signal x into scaling coefficients c of length 4 and wavelet coefficients d of length 6. This has the following interpretation. While $\tilde{U}$ maps the nonzero portion of $x(n)$ into the nonzero portion of its wavelet coefficients, $d(n)$, at the next coarsest scale, normalizing the coefficients at the edges, it maps the nonzero portion of $x(n)$ into the nonzero portion of its scaling coefficients, $c(n)$, while *zeroing* the two scaling coefficients at the edges. Note that if we perform our transformation recursively in scale, at scale each scale we end up with scaling coefficients which are zeroed at the edges, leaving us with fewer and fewer scaling coefficients as we go to coarser scales. If we take our example one step coarser in scale, i.e. we apply the same idea used to create $\tilde{U}$ on the scaling coefficients c, we end up mapping c into one scaling coefficient and three wavelet coefficients at the next coarsest scale. The overall two scale decomposition results in scaling coefficients defined on the lattice in Figure 4. The resulting wavelet coefficients reside on the lattice in Figure 5, where the dotted lines represent averaging at the edges due to the normalization of the g_i's.

Note that if we consider a QMF pair of length greater than 4, there are more edge rows of G, and the required modification to these is more than simple normalization. For example if the filter is of length 6, then the corresponding H operator, with the edge rows removed has the form

$$H = \left[\begin{array}{cccccccccccc} h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & 0 & 0 & & \cdots & & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & 0 & & \cdots & 0 \\ \vdots & & & & & & & & & & & \vdots \\ 0 & \cdots & 0 & 0 & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & 0 & 0 \\ 0 & \cdots & 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 \end{array}\right] \tag{124}$$

and the corresponding G matrix, including the edge rows, is

$$G = \left|\begin{array}{cccccccccccc} g_4 & g_5 & 0 & 0 & 0 & 0 & 0 & 0 & & \cdots & & 0 \\ g_2 & g_3 & g_4 & g_5 & 0 & 0 & 0 & 0 & & \cdots & & 0 \\ \hline g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & 0 & 0 & & \cdots & & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & 0 & & \cdots & 0 \\ \vdots & & & & & & & & & & & \vdots \\ 0 & \cdots & 0 & 0 & g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & 0 & 0 \\ 0 & \cdots & 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 & g_4 & g_5 \\ \hline 0 & \cdots & 0 & 0 & 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 \\ 0 & \cdots & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & g_0 & g_1 \end{array}\right| \tag{125}$$

The point now is that each of the two pairs of edge-rows in (125) is not only not normalized but also not an orthogonal pair. Consequently we must

Gram–Schmidt orthonormalize each of these pairs. This changes the values of the nonzero coefficients in the edge rows but does *not* introduce additional nonzero entries, so that the local nature of the wavelet calculations is still preserved. More generally, if we have a QMF of length P (which, to satisfy the QMF conditions, must be even), we must perform two Gram–Schmidt orthonormalizations of sets of $P/2$ vectors.

Note that the coefficients $d(n)$ and $c(n)$ play a symmetric role in our procedure, and thus we could equally well have zeroed the edges of of our wavelet coefficients $d(n)$ rather than our scaling coefficients $c(n)$ or could have zeroed out the scaling coefficients at one end of the signal and the wavelet coefficients at the other. In addition there are other possible ways in which to modify the edge rows of H and G to achieve orthogonality, the most common being cyclic wrap-around. We refer the reader to [31] and [5] for further discussion of these variations, as we focus here on the one we have just described, as it is this form that yields the correct eigendecomposition for a windowed version of the state model described in the preceding section.

In particular, we specify our model on a finite lattice as follows for $m = L, L+1, ...M-1$,

$$\begin{aligned} E[x(L)x(L)^T] &= \mathcal{P}_x(L) && (126)\\ x(m+1) &= \tilde{H}_m^T \mathcal{A}(m+1)x(m) + \mathcal{B}(m+1)w(m+1) && (127)\\ E[w(i)w(j)^T] &= \mathcal{Q}(i)\delta_{i-j} \ , \ i = L+1, ...M && (128)\end{aligned}$$

where

$$\begin{aligned} \mathcal{A}(m) &\triangleq diag(A(m), ...A(m)) && (129)\\ \mathcal{B}(m) &\triangleq diag(B(m), ...B(m)) && (130)\\ \mathcal{Q}(m) &\triangleq diag(Q(m), ...Q(m)) && (131)\\ \mathcal{P}_x(L) &\triangleq diag(P_x(L), ...P_x(L)) && (132)\end{aligned}$$

Here $A(m), B(m), Q(m)$, and $P_x(L)$ are as before, $x(m)$ and $w(m)$ represent the finite vectors of variables $x(m,n)$ and $w(m,n)$, respectively, at the finite set of nodes at the mth scale, and $\tilde{H}_m$ and $\tilde{G}_m$ are the counterparts of the operators H_m and G_m adapted to the case of finite intervals by removing edge rows of H_m and orthonormalizing those of G_m (note that here we again allow these wavelet operators to act on vector signals component-by-component).

Note that the dynamics (127) are *not* square, since over a finite interval we increase the number of scaling coefficients as we move from coarse to fine scales. For example if scale L consists of a single root node and if we use QMF's of length 4, our dynamics evolve on the finite lattice of Figure 4 from coarse to fine scales, yielding a stochastic process at a sequence of resolutions on a finite interval. As we have indicated, the block-eigenstructure of our finite-lattice process is precisely as we derived in the previous section, except that we now must use the modified wavelet transform on a finite interval, as determined by the sequence of operators $\tilde{H}_m, \tilde{G}_m$. To make this precise, let $f(m)$ denote the number of nodes on our finite lattice at scale $m = L, ..., M$, where for a length P QMF we can easily check that

$$f(i+1) = 2f(i) + P - 2 \tag{133}$$

Define the block unit vectors

$$\delta_i^j \triangleq [0_d, ..., 0_d, \underbrace{I_d}_{i\text{th}}, 0_d, ...0_d]^T \tag{134}$$

where the superscript j is again used to denote that the vector (now in $\mathcal{R}^{f(j) \times d}$) corresponds to the jth scale of the lattice and where I_d is the $d \times d$ identity matrix (and 0_d the $d \times d$ zero matrix). The block vectors $\overline{v}_i^L(m)$, $v_n^l(m)$ for $l = L, ...m-1$ and for $i = 0, 1, 2...f(L)-1$ and $n = 0, 1, 2...f(l)-1$ are block-eigenvectors of the correlation matrix of the process at scale m, $R_{xx}(m)$, where

$$\overline{v}_i^L(m) \triangleq (\prod_{j=L}^{m-1} \tilde{H}_j^T)\delta_i^L \tag{135}$$

and

$$v_n^l(m) \triangleq (\prod_{i=l+1}^{m-1} \tilde{H}_i^T)\tilde{G}_l^T \delta_n^l \tag{136}$$

As we did for the infinite case we can now transform the smoothing problem using a wavelet basis composed of the block vectors $\overline{v}_i^L(m)$ and $v_n^l(m)$. Our transformed variables are formed as in eq.'s(70-81), except that now we have a finite number of variables to estimate. In particular for each scale index, j, the translation index k ranges from 0 to $f(j) - 1$. The wavelet transform smoothing algorithm developed in the preceding section then applies.

5 Numerical Examples

In this section we illustrate the use of our multiscale estimation framework for solving estimation problems involving both single scale as well as multiscale data. We do this by focusing on problems involving estimation of first-order Gauss–Markov processes. We have chosen this process as it is a frequently-used and well-understood and accepted model and it *cannot* be exactly modeled using the multiresolution framework. Thus we can demonstrate the richness of our models in approximating well-known processes by comparing the performance of our smoother, using model parameters chosen so as to well-approximate the Gauss-Markov process, with the performance of standard smoothers. Our examples indicate that our multiscale models do rather well both in modeling important classes of processes and as the basis for constructing computationally efficient algorithms. For first-order Gauss-Markov processes there, of course, already exist efficient estimation algorithms (Wiener and Kalman filters). However, these algorithms apply only in the case of pointwise measurement data. On the other hand, our multiscale modeling framework allows us to incorporate data at a *set* of resolutions *with no increase in algorithmic complexity.* We demonstrate the potential of this capability by fusing multiscale data for the estimation of a Gauss-Markov process, illustrating how the use of coarse-scale data can aid in estimating features which are not discernible using fine-scale data of poor quality. We refer the reader to [5] for other examples of the application of our framework, including the fusion of multiscale data for the $1/f$-processes introduced in [26, 27]

5.1 Processes and Multiscale Models

Consider the following stationary 1st-order Gauss-Markov process.

$$\dot{x}(t) = -\beta x(t) + w(t) \tag{137}$$

$$E[x^2(t)] = 1 \tag{138}$$

This process has the following correlation function and associated power spectral density function.

$$\phi_{xx}(\tau) = e^{-\beta|\tau|} \tag{139}$$

$$S_{xx}(\omega) = \frac{2\beta}{\omega^2 + \beta^2} \tag{140}$$

In the numerical examples that follow we use a discretized version of (137). In particular we use a sampled version of (137) in which the sampling interval is small enough to minimize any aliasing effects. We choose $\beta = 1$ and take the sampling rate to be twice ω_0 where $S_{xx}(\omega_0) = .002$, $S_{xx}(\omega)$ being the power spectral density function of $x(t)$. This yields a sampling interval of $\Delta = \pi/\omega_0$ where $\omega_0 = 30$. Our discretized model is as follows.

$$x(t+1) = \alpha x(t) + w(t) \tag{141}$$
$$E[x^2(t)] = 1 \tag{142}$$
$$\alpha = e^{-\beta\Delta} \simeq .9006 \tag{143}$$

We consider the following measurements of $x(t)$.

$$y(t) = x(t) + v(t) \tag{144}$$
$$E[v^2(t)] = R \tag{145}$$
$$Y = \{y(t) | t = 0, ... N-1\} \tag{146}$$

In the examples that follow we take the interval length $N = 128$.

Figure 6 is a gray-scale image of the covariance matrix of our stationary first order Gauss-Markov process defined on a finite interval, corresponding to the model in (141). Note that thanks to the normalization (138), what is displayed here is the array of correlation coefficients of the process, i.e. the covariance between two points normalized by the product of the standard deviation at each point. The diagonal of the matrix thus is unity, and the off-diagonal terms decay exponentially away from the diagonal. In [11] this correlation coefficient matrix is transformed using various wavelet bases, i.e. the matrix undergoes a similarity transformation with respect to the basis representing the wavelet transform based on a variety of QMF filters, $h(n)$. This transformation corresponds essentially to the separable form of the 2D wavelet transform[16]. Figures 7,8 are the images of the correlation coefficient matrix in Figure 6 transformed using QMF filters of length 2 and 8, respectively. That is, these are the correlation coefficient matrices for the multiscale wavelet coefficients of a finite length segment of a 1st-order Gauss-Markov process, where the finest level wavelet coefficients are located in the bottom half of the coefficient vector, the next coarser level coefficients comprise the next fourth of the vector, the next set fills the next eighth, etc. Note that aside from the finger-like patterns in these images, the off-diagonal elements are essentially zeroed. The finger patterns correspond to correlations between wavelet coefficients at different scales which share the same location in the interval. Note that even these correlations are weak. Furthermore, since the variances of many of the wavelet coefficients are actually quite small, the normalization we have introduced by displaying correlation coefficients actually *boosts* the magnitude of many of the off-diagonal terms, so that the approximate whitening of this process performed by wavelet transforms is even better than these figures would suggest. Note that analogous observations have been made for other processes, such as fractional Brownian motions [14, 24], suggesting a rather broad applicability of the methods described here.

To continue, the low level of inter-scale correlation in the wavelet representation of the Gauss-Markov process as illustrated in Figures 7 and 8 motivates the approximation of the wavelet coefficients of this process as uncorrelated. This results in a lattice model precisely as defined in (17-19). We use this model as an approximation to the Gauss-Markov process in order to do fixed interval smoothing. In particular, the class of models which we consider as approximations to Gauss-Markov processes is obtained precisely in the manner just described. That is, we construct models as in eq.'s(17-19) where the wavelet coefficients are assumed to be mutually uncorrelated. In this case the variances of the wavelet coefficients, $w(m)$ in eq.'s(17-19), are determined by doing a similarity transform on the covariance matrix of the process under investigation using a wavelet transform based on the Daubechies FIR filters[8]. In particular if P_x denotes the true covariance matrix of the process, V the diagonal matrix of wavelet coefficient variances, and W is the wavelet transform matrix, then

$$\Lambda = W P_x W^T \tag{147}$$
$$V = W P_{approx} W^T \tag{148}$$

Thus, this approximate model corresponds to assuming that Λ is diagonal (i.e. to neglecting its off-diagonal elements).

In our examples we use the 2-tap Haar QMF filter as well as the 4-tap, 6-tap, and 8-tap Daubechies QMF filters[8]. Note that in adapting the wavelet transform to the finite interval we have, for simplicity, used cyclic wrap-around in our wavelet transforms rather than the exact finite interval wavelet eigenvectors described in the preceding section. In this case the number of points at each scale is half the number of points at the next finer scale.

5.2 Smoothing Processes Using Multiscale Models

In this section we present examples in which we compare the performance of the optimal estimator for a 1st-order Gauss-Markov process with that of the suboptimal estimator based on our multiscale approximate model.

Let $x(t), t = 0, ..., N-1$ denote a finite window of our Gauss-Markov process, and consider the white-noise-corrupted observations.

$$\begin{aligned} y(t) &= x(t) + v(t) && (149) \\ E[v^2(t)] &= R && (150) \\ Y &= \{y(t) | t = 0, ...N-1\} && (151) \end{aligned}$$

Let the optimal smoothed estimate (implemented using the correct Gauss-Markov model) be denoted as

$$\hat{x}_s(t) \triangleq E[x(t)|Y] \tag{152}$$

Letting x and $\hat{x}_s$ denote the vectors of samples of $x(t)$ and $\hat{x}_s(t)$, respectively, we can define the optimal smoothing error covariance

$$\Sigma_{opt} \triangleq E[(x - \hat{x}_s)(x - \hat{x}_s)^T] \tag{153}$$

Also if P_x denotes the covariance of x the optimal estimate is given by

$$\hat{x}_s = L_x Y \tag{154}$$

with

$$L_x = P_x(P_x + RI)^{-1} \tag{155}$$

and

$$\Sigma_{opt} = P_x - P_x(P_x + RI)^{-1}P_x \tag{156}$$

More generally if we consider any estimator of the form of (154) (such as the one we will consider where L_x corresponds to the optimal smoother for our multiscale approximate model for the Gauss-Markov process), then the corresponding error covariance is given by

$$\begin{aligned}\Sigma_{sub} &\stackrel{\triangle}{=} E[(x - \hat{x}_{sub})(x - \hat{x}_{sub})^T] && (157)\\ &= (I - L_z)P_x(I - L_z)^T + L_zRL_z^T && (158)\end{aligned}$$

We now give several examples demonstrating the performance of our multiscale models in smoothing Gauss-Markov processes. We focus in this subsection on the case of a single scale of data at the finest scale. In Fig.'s 9-13 we compare the performance of the optimal estimator, with the performance of our suboptimal estimator based on lattice models for both 2-tap and 8-tap Daubechies filters. In these examples the measurement noise variance $R = .5$; i.e. the data is of $SNR = 1.4142$.

Note the strikingly similar performances of the optimal and suboptimal smoothers, as illustrated in Figure 12 for the case of the 2-tap lattice smoother. From visual inspection of the results of the two smoothers it is difficult to say which does a better job of smoothing the data; it seems one could make a case equally in favor of the standard smoother and the lattice-model smoother. The similarity in performance of the optimal smoother and our lattice smoothers is even more dramatic for the case of the 8-tap smoother as illustrated in Figure 13.

Note that although the standard smoother results in a smaller average smoothing error (the trace of Σ_{opt} divided by the number of points in the interval), it seems the average error of our lattice-model smoothers is not that much larger. To quantify these observations let us define the *variance reduction* of a smoother as follows.

$$\begin{aligned}\rho &\stackrel{\triangle}{=} \text{variance reduction} && (159)\\ &= \frac{p_0 - p_s}{p_0} \\ p_0 &= \text{average process variance} && (160)\\ p_s &= \text{average smoothing error variance} && (161)\end{aligned}$$

	2-tap	4-tap	6-tap	8-tap
$SNR = 2.8284$	1.07 %	.550 %	.402 %	.334 %
$SNR = 1.412$	3.27 %	1.77 %	1.24 %	1.04 %
$SNR = .7071$	6.71 %	4.13 %	2.70 %	2.33 %
$SNR = .5$	9.58 %	6.14 %	3.87 %	3.27 %

Table 1: Performance Degradation Comparison of Lattice-Model Smoothers - 2-tap, 4-tap, 6-tap and 8-tap

We also define the *performance degradation* resulting from using a lattice smoother as compared with using the standard smoother as follows.

$$\begin{aligned} \Delta_{\text{perf}} &\triangleq \text{performance degradation} && (162) \\ &= \frac{\rho_{\text{standard}} - \rho_{\text{lattice}}}{\rho_{\text{standard}}} \\ \rho_{\text{standard}} &= \text{variance reduction of standard smoother} && (163) \\ \rho_{\text{lattice}} &= \text{variance reduction of lattice-model smoother} && (164) \end{aligned}$$

Table 1 shows the performance degradation of the lattice-model smoother relative to the standard smoother for filter tap orders 2, 4, 6, and 8 and for four different noise scenarios: 1) SNR = 2.8284 2) SNR = 1.412 3) SNR = .7071 4) SNR = .5. The variance reductions are computed using smoothing errors averaged over the entire interval. While the degradation in performance lessens as the order of the filter increases, a great deal of the variance reduction occurs just using a 2-tap filter. For example for the case of $SNR = 1.412$ the standard smoother yields a variance reduction of 85 percent. It is arguable whether there is much to be gained in using an 8-tap filter when its relative decrease in performance degradation is only 2.23 percent over the 2-tap smoother; i.e. the variance reduction of the 8-tap smoother is 83.8 percent while the variance reduction of the 2-tap smoother is already 81.9 percent.

The performance degradation numbers for the lower SNR case (SNR = .7071) seem to suggest that the effect of raising the noise is to decrease the performance of the lattice-model smoothers. But one should keep in mind that this decrease is at most only marginal. Consider the case where the

SNR = .5. In this case the data is extremely noisy, the noise power is *double* that of the case where SNR = .7071, and yet the performance degradation in using the 2-tap smoother compared with the standard smoother is 9.58 percent, up only 2.87 percent from the case of SNR = .7071. Furthermore, if one examines a plot (Fig 14) of the results of applying the two smoothers on such extremely noisy data, the performance of the two smoothers is as before quite comparable.

We emphasize that the average performance degradation is a *scalar* quantity, and at best gives only a rough measure of estimation performance. From this quantity it is difficult to get any idea of the qualitative features of the estimate. The plots of the sample path and its various smoothed estimates over the entire interval offer the reader much richer evidence to judge for himself what the relative differences are in the outputs of the various smoothers.

The preceding analysis indicates that multiscale models can well-approximate the statistical characteristics of 1st-order Gauss-Markov processes in that nearly equivalent smoothing performance can be obtained with such models. Further corroboration of this can be found in [5] where Bhattacharya distance is used to bound the probability of error in deciding, based on noisy observations as in (149), if a given stochastic process $x(t)$ is either a 1st-order Gauss-Markov process or the corresponding multiscale process obtained by ignoring interscale correlations. An important point here is that the 1st-order Gauss-Markov model is itself an idealization, and we would argue that our multiscale models are an equally good idealization. Indeed if one takes as an informal definition of a "useful" model class that (a) it should be rich enough to capture, with reasonable accuracy, important classes of physically meaningful stochastic processes; and (b) it should be amenable to detailed analysis and lead to efficient and effective algorithms, then we would argue that our multiscale models appear to have some decided advantages as compared to standard models. In particular not only do we obtain efficient, highly parallel algorithms for the smoothing problems considered in this section but we also obtain-equally efficient algorithms for problems such as multiscale data fusion, which we discuss next.

5.3 Sensor Fusion

In this section we provide examples that show how easily and effectively our framework handles the problem of fusing multiscale data to form optimal smoothed estimates. In our framework not only is there no added algorithmic complexity to the addition of multiscale measurements but it is also easy for us to *evaluate the performance* of our smoothers in using multiscale data.

For simplicity we focus here on the problem of fusing data at two scales. Consider the Gauss-Markov process used in our previous examples as defined in (141). We assume that we have fine-scale, noisy measurements as in (149) together with one coarser level of measurements. In particular, as before, the length of our interval is taken to be 128 points. Thus we assume that we have $2^M = 128$ measurements of the finest scale version of our signal as well as 2^K measurements of the coarser approximation of our signal at scale K. [3]

Consider the case where our fine scale measurements are of extremely poor quality. In particular we take the case where our data is of SNR = .3536 (the noise power is eight times the signal power). Figure 15 compares the result of using the standard smoother on these data with the result of using a 4-tap lattice model smoother on the same data. The performance of the two smoothers is comparable as we would expect from our results in the previous section.

Now we use the same data and consider fusing a higher quality coarse scale data set to form a smoothed estimate. In particular we take our coarse data to reside at the scale one level coarser than the original data (scale at which there are 64 points) and the coarsening operator, H_i, corresponds to a 4-tap filter. The SNR of this coarse data is equal to 2. Figure 16 compares the result of using the standard smoother on the low quality fine scale data alone with the result of using our 4-tap lattice smoother to fuse this low quality data with high quality coarse data.

Note that the coarse measurement aids dramatically in improving the quality of the estimate over the use of just fine-scale data alone. To quantify this recall that our smoother computes the smoothing error at each scale. We use these errors as *approximations* to the actual suboptimal errors (note that the computation of the actual error covariance from multiscale data is appreciably more complex than for the case of single scale measurements; the same is *not* true for our tree models, where the complexity of the two cases is essentially the *same*). The variance reduction in the case of fusing

[3]Note that as mentioned previously, the lattice models used in this section correspond *exactly* to the wavelet transform, i.e. to (17) - (19), so that the signal $x(K)$ is precisely the vector of scaling coefficients at scale K of the fine scale signal $x(M)$.

the two measurement sets is 97 percent versus only 36 percent for the case of using only the poor quality fine-scale data.

To explore even further the idea of fusing coarse measurements with poor quality fine measurements we compare the results of using coarse measurements of various degrees of coarseness in order to determine how the scale of the coarse data affects the resolution of the smoothed estimate. In particular, we take our fine scale data to be the same as that in Figure 16. However, we supplement this data with coarse measurements of extremely high quality (SNR = 31.6) and consider several cases: 1) the coarse data is at a scale at which there are 64 points 2) the coarse data is at a scale at which there are 32 points 3) the coarse data is at a scale at which there are 16 points. Figures 17-19 compare the original signal with its smoothed estimates using coarse data at the three different scales. Note how the estimates in these figures adapt automatically to the quality and resolution of the data used to produce them.

6 Conclusions

In this paper we have described a class of multiscale, stochastic models motivated by the scale-to-scale recursive structure of the wavelet transform. As we have described, the eigenstructure of these models is such that the wavelet transform can be used to convert the dynamics to a set of simple, decoupled dynamic models in which *scale* plays the role of the time-like variable. This structure then led us directly to extremely efficient, scale-recursive algorithms for optimal estimation based on noisy data. A most significant aspect of this approach is that it directly applies in cases in which data of differing resolutions are to be fused, yielding computationally efficient solutions to new and important classes of data fusion problems.

In addition we have shown that this modeling framework can produce effective models for important classes of processes not captured exactly by the framework. In particular we have illustrated the potential of our approach by constructing and analyzing the performance of multiscale estimation algorithms for Gauss-Markov processes. Furthermore we have shown how the problem of windowing — i.e. the availability of only a finite window of data - can be dealt with by a slight modification of the wavelet transform. Finally, while what we have presented here certainly holds considerable promise for 1-D signal processing problems, the payoffs for multidimensional signals should be even greater. In particular the identification of scale as a time-like variable holds in several dimensions as well, so that our scale-recursive algorithms provide potentially substantial computational savings in contexts in which the natural multidimensional index variable (e.g. space) does not admit natural "directions" for recursion.

References

[1] G. Beylkin, R. Coifman, and V. Rokhlin, "Fast Wavelet Transforms and Numerical Algorithms I", to appear in *Comm. Pure and Appl. Math.*

[2] A. Brandt, "Multi-level adaptive solutions to boundary value problems," Math. Comp. Vol. 13,1977, pp.333-390.

[3] W. Briggs, "*A Multigrid Tutorial*, SIAM, Philadelphia, PA, 1987.

[4] P. Burt and E. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Trans. Comm.*, vol. 31, pp. 482–540, 1983.

[5] K.C. Chou, "A Statistical Modeling Approach to Multiscale Signal Processing," PhD Thesis, Mass. Institute of Technology, Dept. of Electrical Engineering, May, 1991.

[6] K.C. Chou, S. Golden and A.S. Willsky, "Modeling and Estimation of Multiscale Stochastic Processes", *Int'l Conference on Acoustics, Speech, and Signal Processing"*, Toronto, April 1991.

[7] R.R. Coifman, Y. Meyer, S. Quake and M.V. Wickehauser, "Signal Processing and Compression with Wave Packets", preprint, April 1990.

[8] I. Daubechies,"Orthonormal bases of compactly supported wavelets", Comm. on Pure and Applied Math. 91, 1988, pp. 909-996.

[9] I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis," *IEEE Trans. on Information Theory*, 36, 1990, pp. 961-1005.

[10] P. Flandrin, "On the Spectrum of Fractional Brownian Motions", *IEEE Transactions on Information Theory*, Vol. 35, 1989, pp. 197-199.

[11] S. Golden, *Identifying Multiscale Statistical Models Using the Wavelet Transform*, S.M. Thesis, M.I.T. Dept. of EECS, May 1991.

[12] A. Grossman and J. Morlet, "Decomposition of Hardy functions into square integrable wavelets of constant shape", SIAM J. Math. Anal. 15, 1984, pp. 723-736.

[13] W. Hackbusch and U. Trottenberg, Eds., *Multigrid Methods and Applications*, Springer-Verlag, N.Y., N.Y., 1982.

[14] M. Kim and A.H. Tewfik, "Fast Multiscale Detection in the Presence of Fractional Brownian Motions", *Proceedings of SPIE Conference on Advanced Algorithms and Architecture for Signal Processing V*, San Diego, CA, July 1990.

[15] S.G. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation", *IEEE Transactions on Pattern Anal. and Mach. Intel.*, Vol. PAMI-11, July 1989, pp. 674-693.

[16] S.G. Mallat, "Multifrequency Channel Decompositions of Images and Wavelet Models", *IEEE Transactions on ASSP*, Vol. 37, December 1989, pp. 2091-2110.

[17] B. Mandelbrot, *The Fractal Geometry of Nature,* Freeman, New York, 1982.

[18] B. B. Mandelbrot and H.W. Van Ness, "Fractional Brownian Motions, Fractional Noises and Applications", *SIAM Review*, Vol. 10, October 1968, pp. 422-436

[19] S. McCormick, *Multigrid Methods*, Vol. 3 of the SIAM Frontiers Series, SIAM, Philadelphia, 1987.

[20] Y. Meyer, "L'analyse par ondelettes", Pour la Science, Sept. 1987.

[21] A. P. Pentland, "Fractal-Based Description of Natural Scenes", *IEEE Transactions on Patt. Anal. and Mach. Intel.,* Vol. PAMI-6, November 1989, 661-674.

[22] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum Likelihood Estimates of Linear Dynamic Systems," AIAA Journal, Vol. 3, No. 8, Aug. 1965, pp. 1445-1450.

[23] M.J. Smith and T.P. Barnwell, "Exact reconstruction techniques for tree-structured subband coders", IEEE Trans. on ASSP 34, 1986, pp. 434-441.

[24] A.H. Tewfik and M. Kim, "Correlation Structure of the Discrete Wavelet Coefficients of Fractional Brownian Motions", submitted to *IEEE Transactions on Information Theory.*

[25] M. Vetterli, and C. Herley, "Wavelet and Filter Banks: Relationships and New Results", *Proceedings of the ICASSP*, Albuquerque, NM, 1990.

[26] G.W. Wornell, "A Karhunen-Loeve-Like Expansion for 1/f Processes via Wavelets", *IEEE Transactions on Information Theory*, Vol. 36, No. 9, July 1990, pp. 859-861.

[27] G.W. Wornell and A.V. Oppenheim, "Estimation of Fractal Signals from Noisy Measurements Using Wavelets", submitted to *IEEE Transactions on ASSP.*

[28] M. Basseville, A. Benveniste, and A. S. Willsky, "Multiscale Autoregressive Processes, Parts I and II", *IEEE Transactions on Signal Proc.*, to appear.

[29] K.C. Chou, A.S. Willsky, A. Benveniste, and M. Basseville, "Recursive and Iterative Estimation Algorithms for Multiresolution Stochastic Processes," Proc. of IEEE Conference on Decision and Control, Tampa, Florida, Dec. 1989.

[30] M. Basseville, A. Benveniste, K.C. Chou, S.A. Golden, R. Nikoukhah, and A. S. Willsky, "Modeling and Estimation of Multiresolution Stochastic Processes," *IEEE Transactions on Information Theory*, Vol. 38, No. 2, March 1992.

[31] Y. Meyer, " Ondelettes sur l'Intervalle", CEREMADE Rept., No. 9020, Univ. de Paris- Dauphine, Paris, France, 1991.

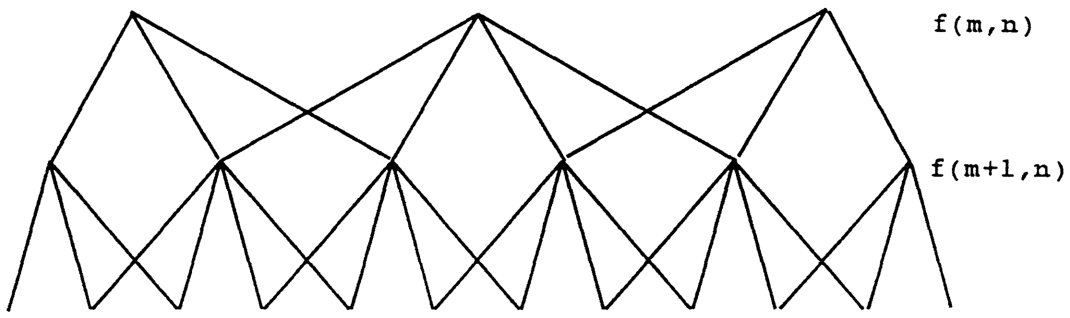

Figure 1: Infinite Lattice Representing Domain of Scaling Coefficients

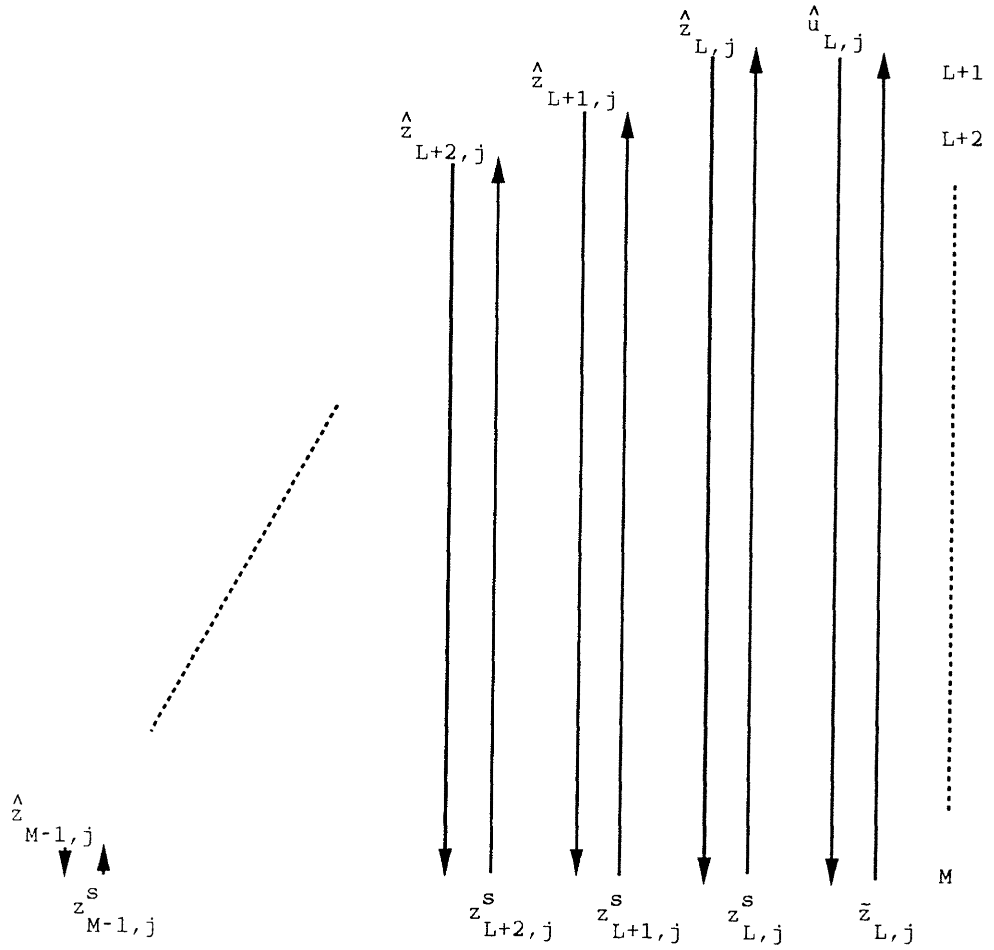

Figure 2: Parallel 1D Smoothing - Down-up

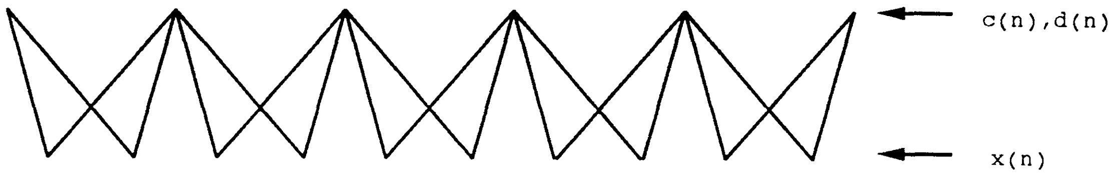

Figure 3: Transformation of a 10-pt. Sequence $x(n)$ into its 6-pt. Scaling Coefficients $c(n)$ and its 6-pt. Wavelet Coefficients $d(n)$

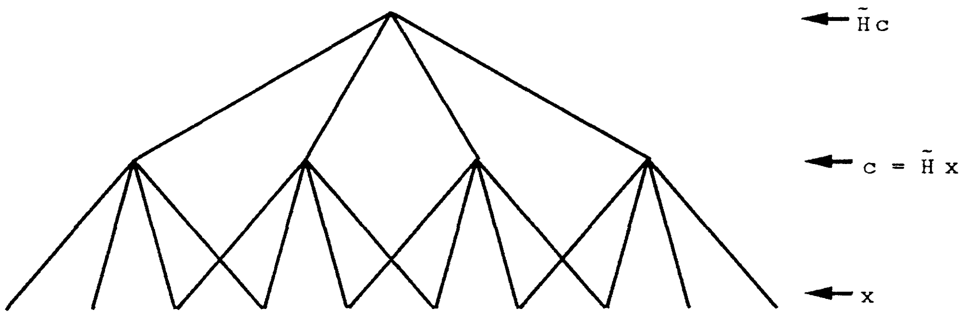

Figure 4: Lattice Representing Domain of Scaling Coefficients for 2-scale Decomposition Based on Zeroing Edge Scaling Coefficients

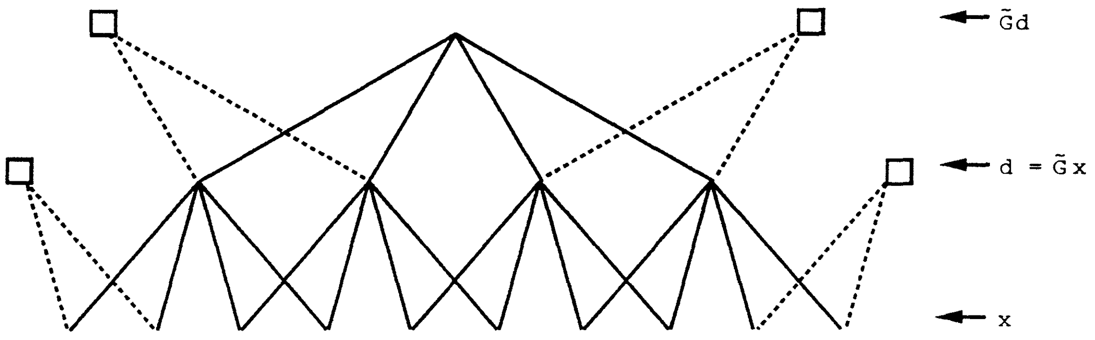

Figure 5: Lattice Representing Domain of the Wavelet Coefficients for 2-scale Decomposition Based on Zeroing Edge Scaling Coefficients

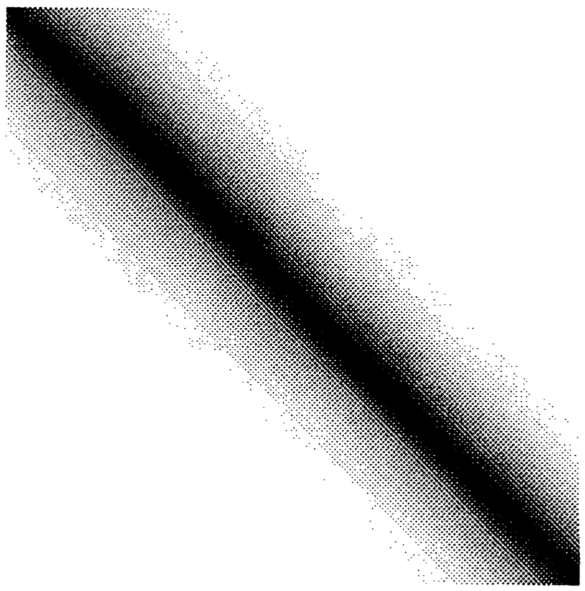

Figure 6: Covariance Matrix of a Stationary Gauss-Markov Process

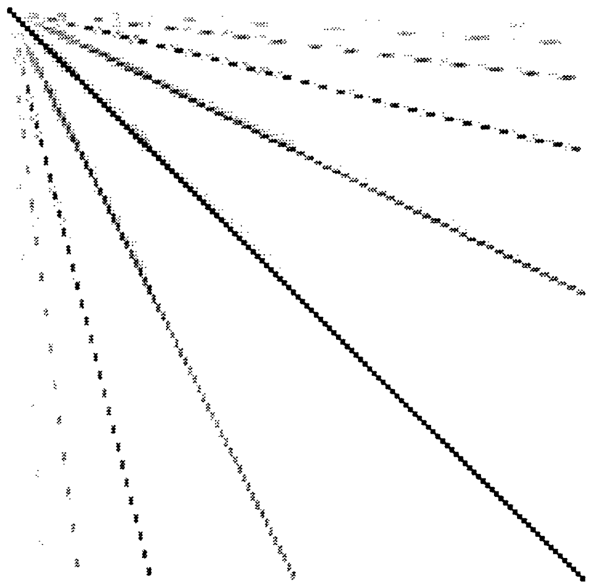

Figure 7: Representation of the Stationary Gauss-Markov Process in a Wavelet Basis using a 2-Tap QMF filter

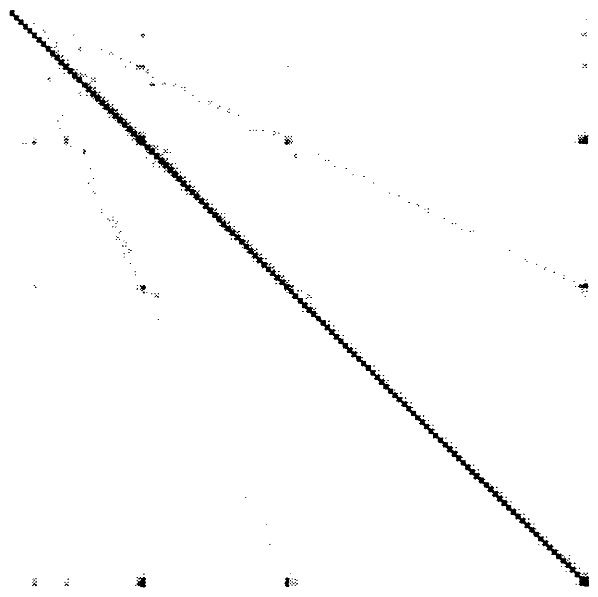

Figure 8: Representation of the Stationary Gauss-Markov Process in a Wavelet Basis using an 8-Tap QMF filter

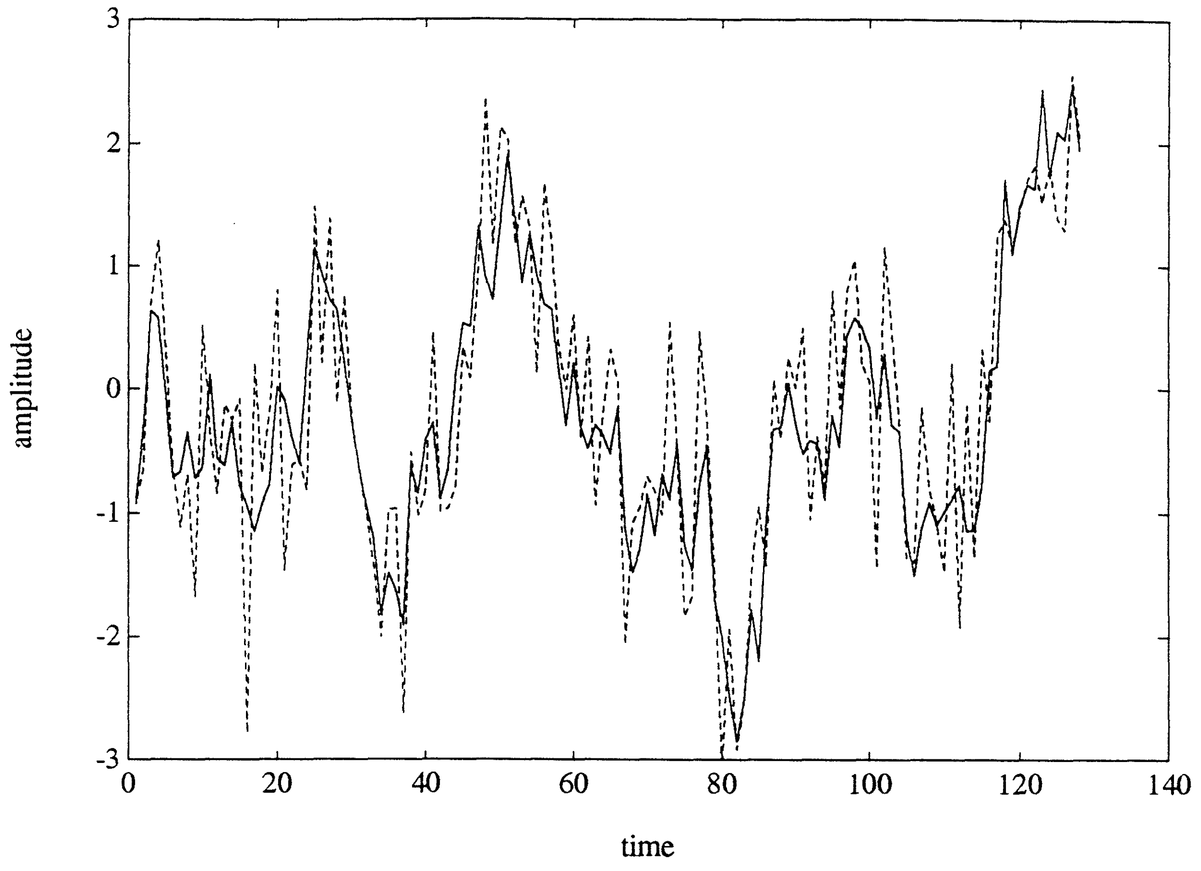

Figure 9: Sample Path of a Stationary Gauss-Markov Process (solid) and Its Noisy Version with SNR=1.4142 (dashed)

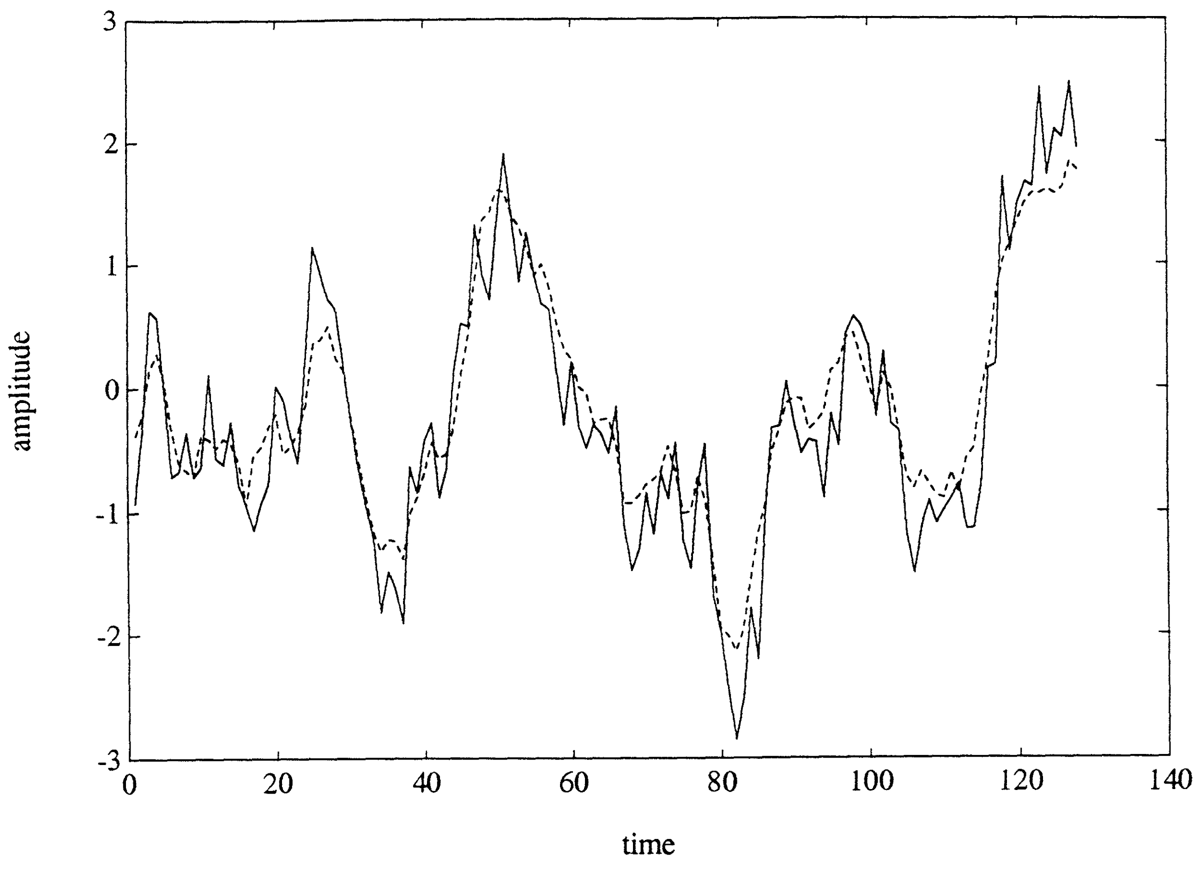

Figure 10: Stationary Gauss-Markov Process (solid) and Its Smoothed Version (dashed) Using Standard Minimum Mean-Square Error Smoother (Data of SNR=1.4142)

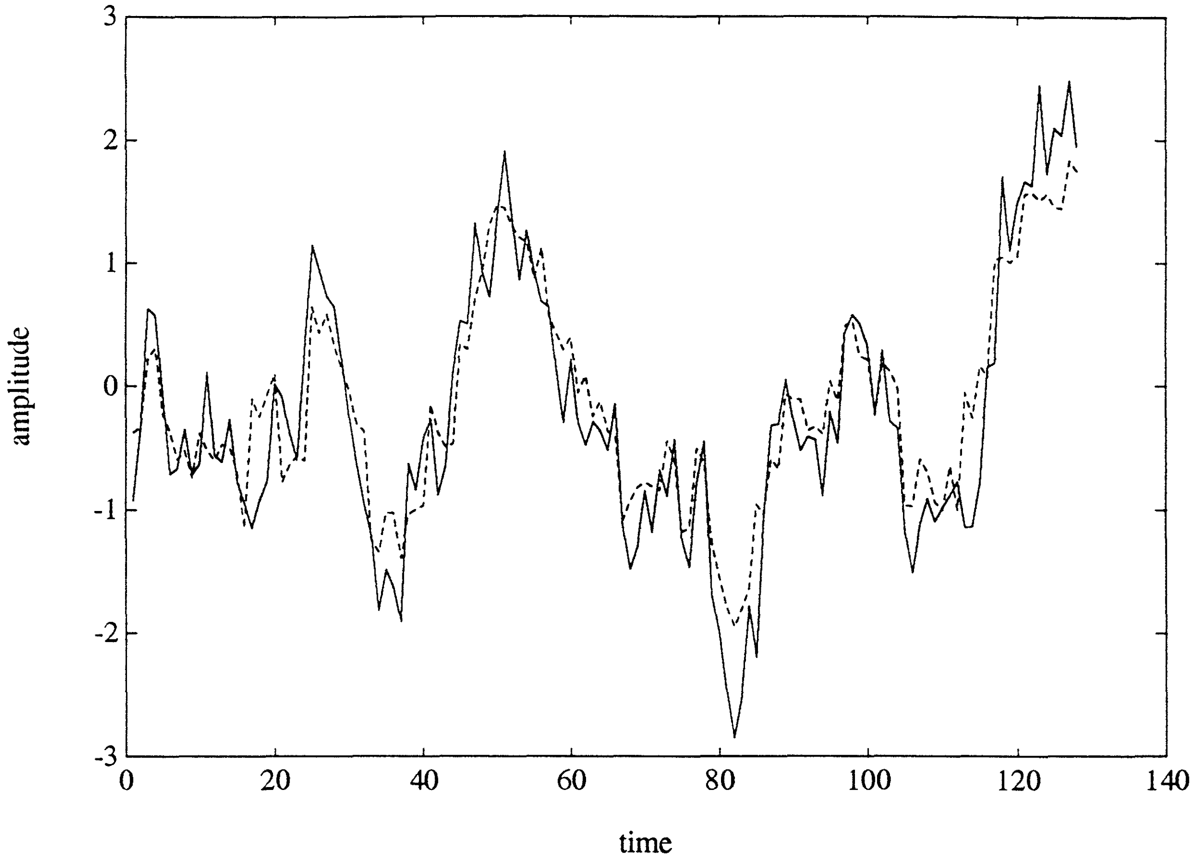

Figure 11: Stationary Gauss-Markov Process (solid) versus Multiscale Smoother Using 2-Tap (dashed) (Data of SNR=1.4142)

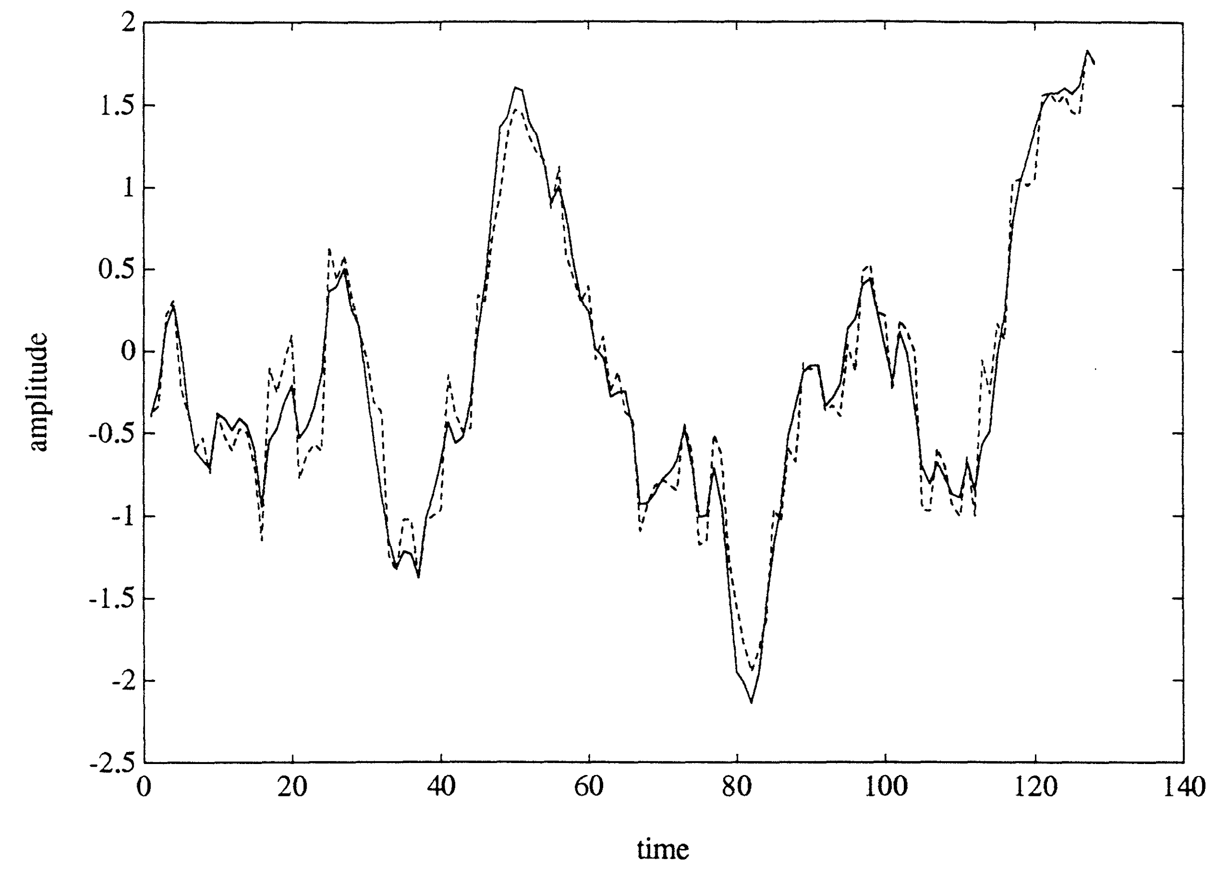

Figure 12: Standard Minimum Mean-Square Error Smoother (solid) versus Multiscale Smoother Using 2-Tap (dashed) (Data of SNR=1.4142)

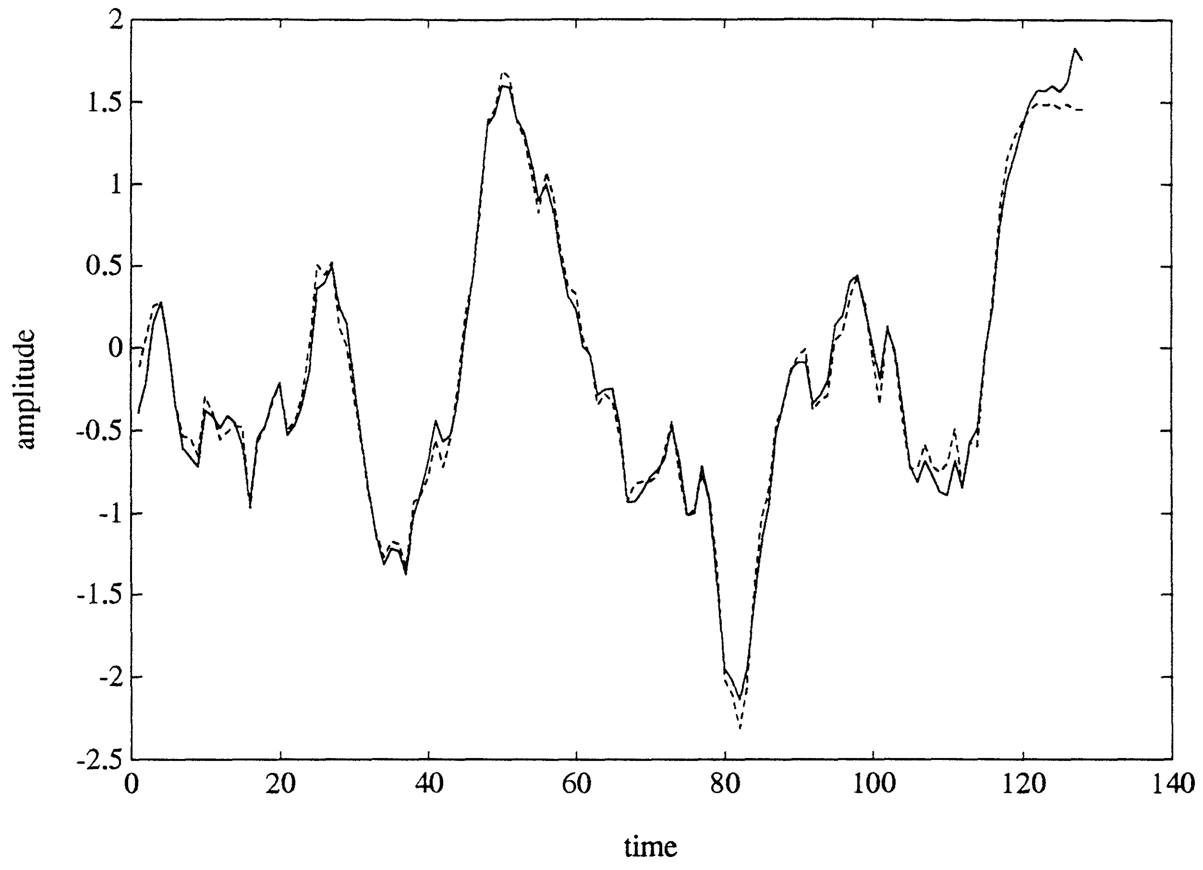

Figure 13: Standard Minimum Mean-Square Error Smoother (solid) versus Multiscale Smoother Using 8-Tap (dashed) (Data of SNR=1.4142)

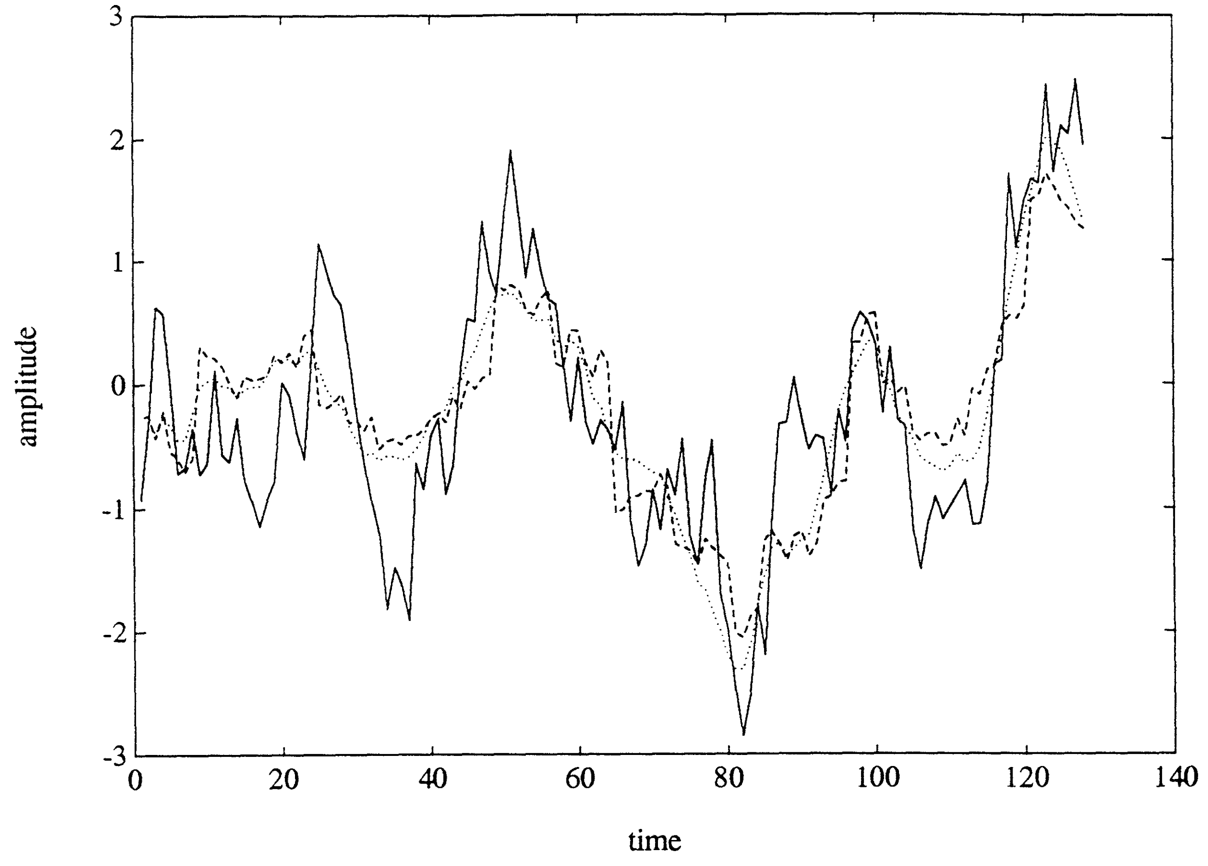

Figure 14: Sample Path of a Stationary Gauss-Markov Process (solid), Standard Smoother (dotted), 2-Tap Smoother (dashed) (Data of SNR=.5)

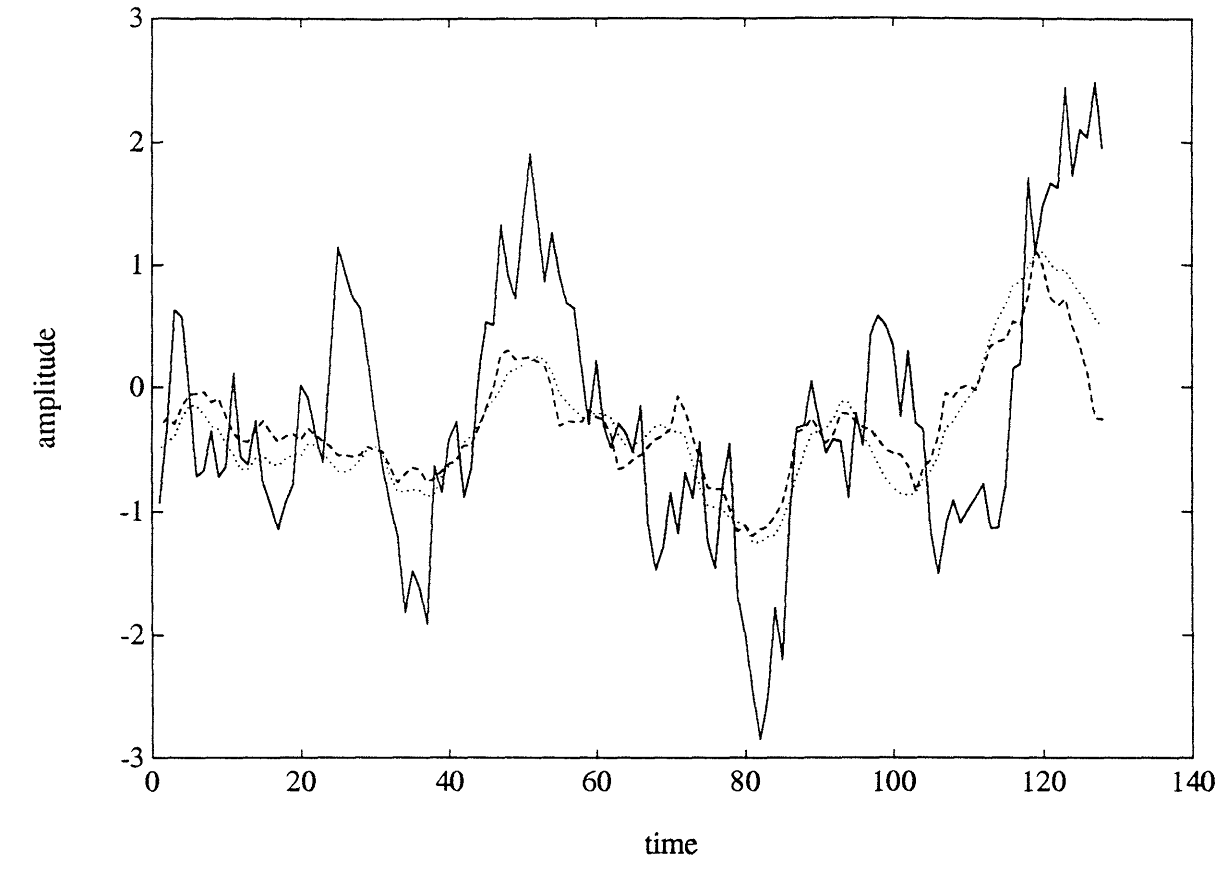

Figure 15: Sample Path of Stationary Gauss-Markov Process (solid), Result of Standard Smoother on Poor Data of SNR=.3536 (dotted), Result of 4-tap Lattice Smoother on Same Data (dashed)

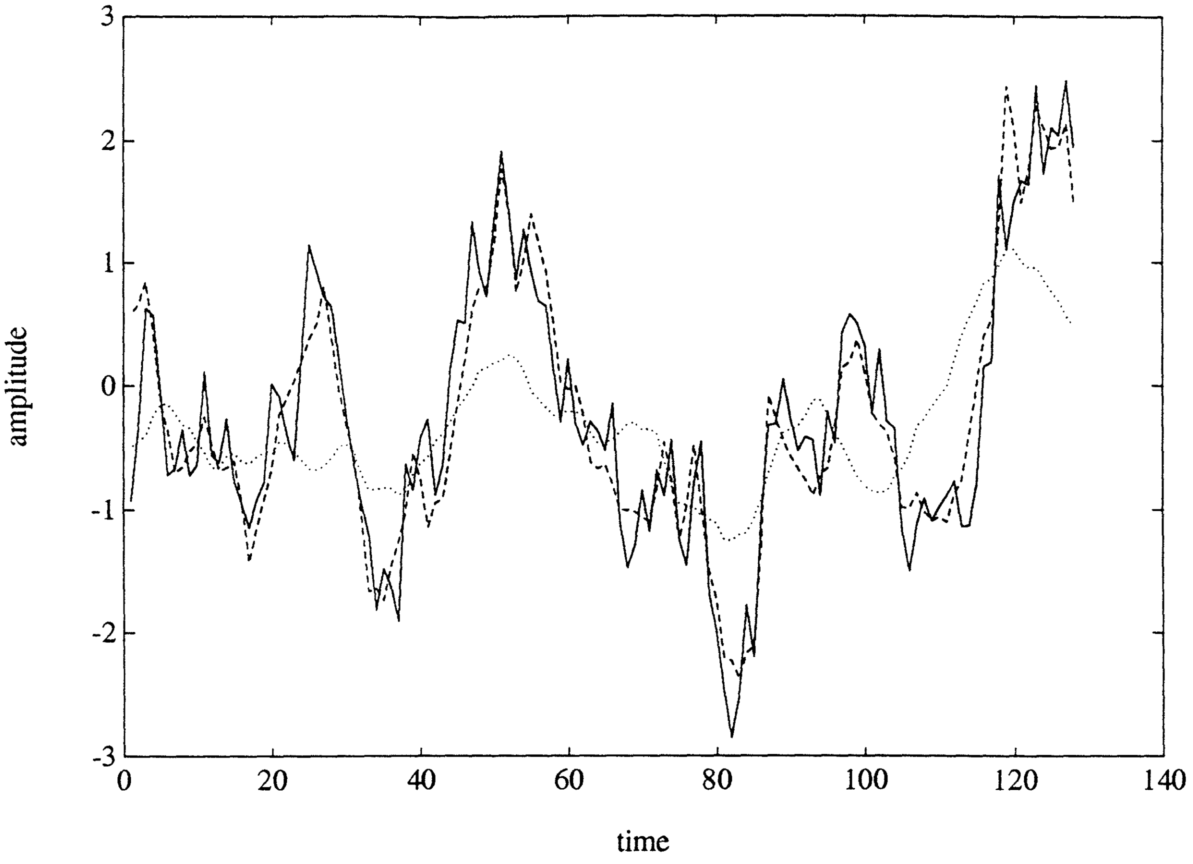

Figure 16: Sample Path of Stationary Gauss-Markov Process (solid), Result of Standard Smoother on Fine Data of SNR = .3536 (dotted), Result of 4-tap Lattice Smoother on Same Data Supplemented with Coarse Data of SNR = 2 (dashed)

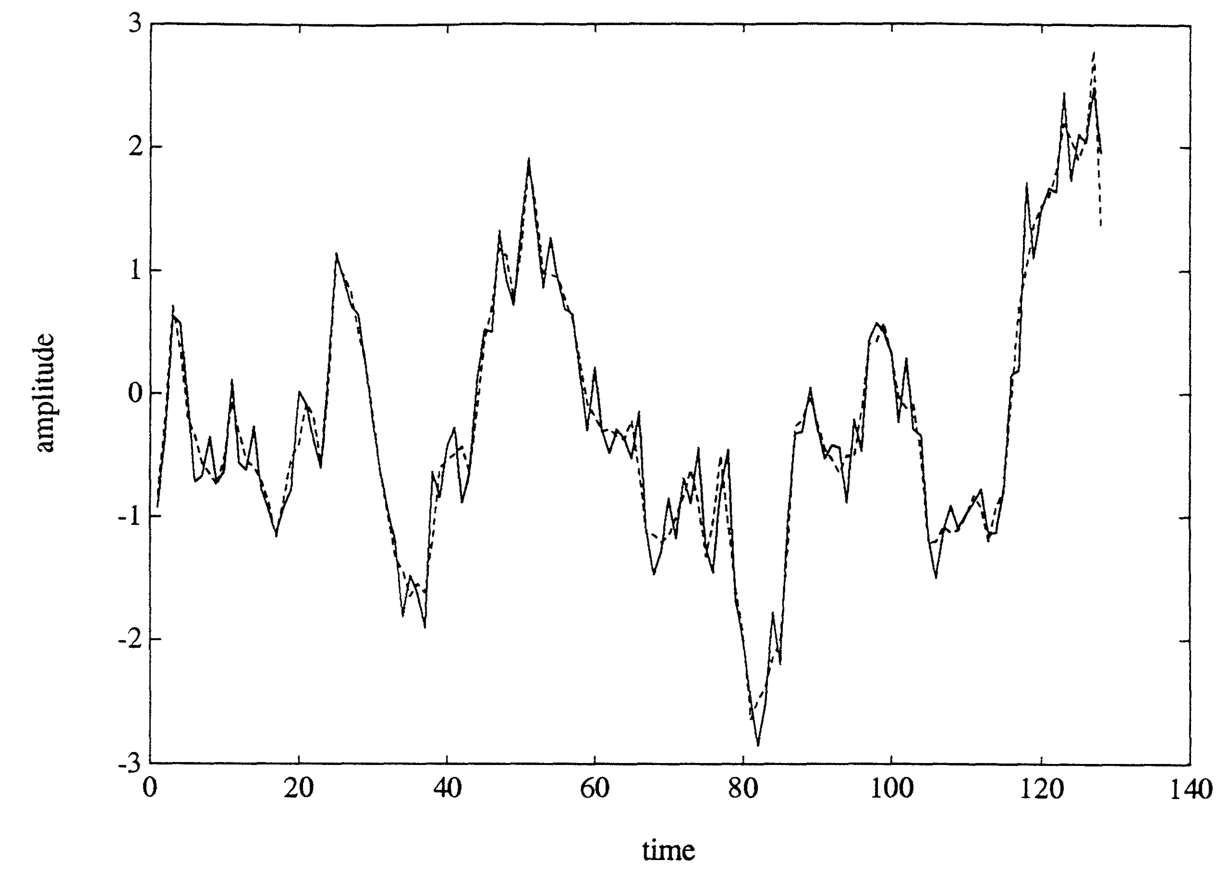

Figure 17: Sample Path of Stationary Gauss-Markov Process (solid), Results of 4-tap Lattice Smoother Using Fine Data of SNR = .3536 Supplemented with Coarse Data of SNR = 31.6: Coarse Data at 64 pt. Scale (dashed)

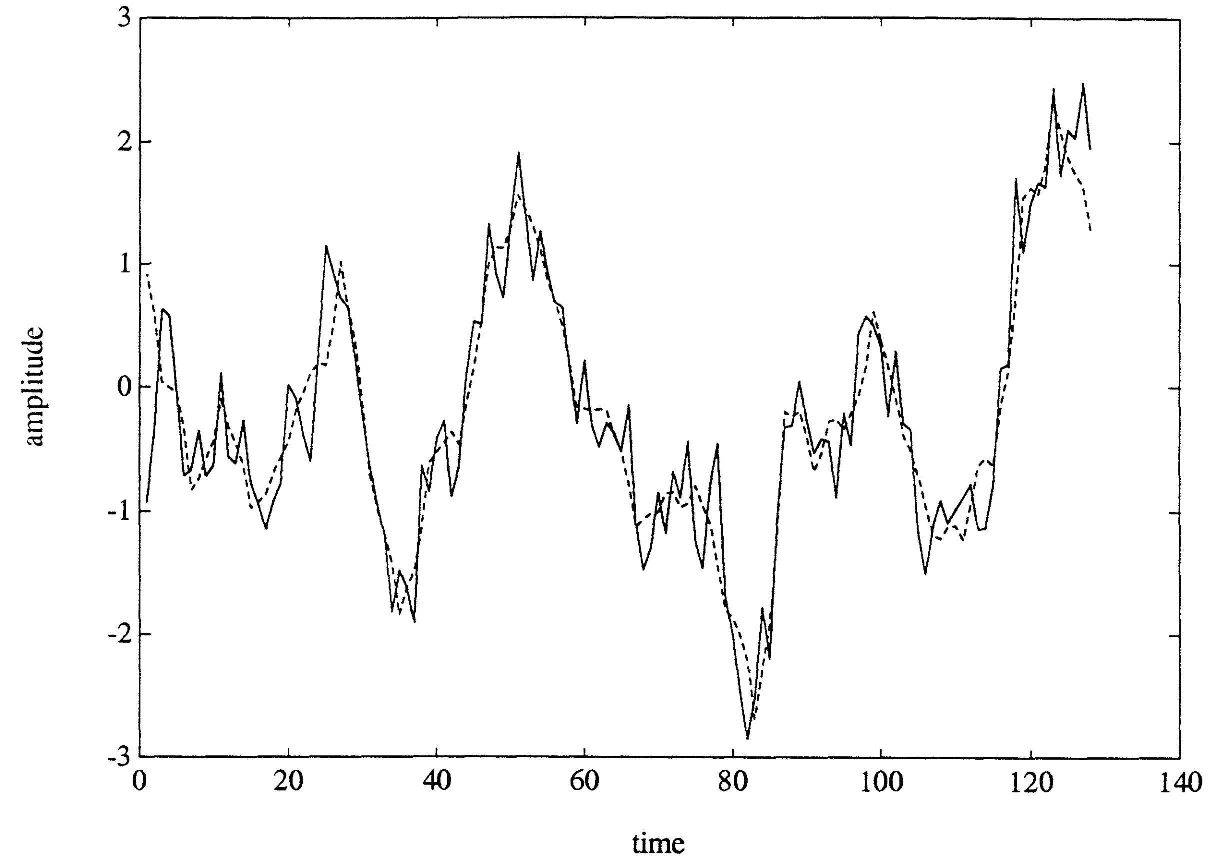

Figure 18: Sample Path of Stationary Gauss-Markov Process (solid), Results of 4-tap Lattice Smoother Using Fine Data of SNR = .3536 Supplemented with Coarse Data of SNR = 31.6: Coarse Data at 32 pt. Scale (dashed)

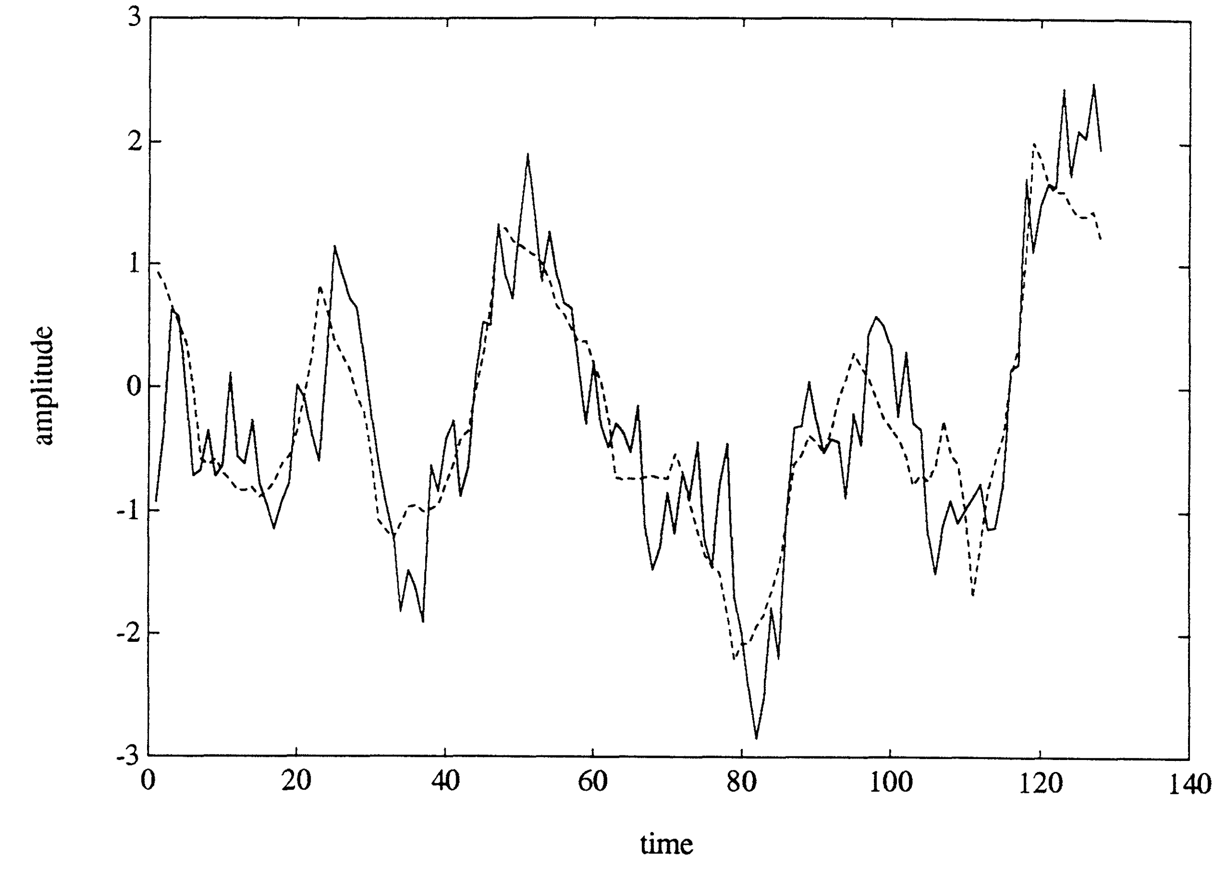

Figure 19: Sample Path of Stationary Gauss-Markov Process (solid), Results of 4-tap Lattice Smoother Using Fine Data of SNR = .3536 Supplemented with Coarse Data of SNR = 31.6: Coarse Data at 16 pt. Scale (dashed)