March, 1991 LIDS- P 2027

Research Supported By:

AFOSR grant 88-0032
ARO contract DAAL03-86-K-0171
ONR grant N00014-91-J-1004

Parallel Smoothing Algorithms for Causal and Acausal Systems*

Taylor, D.
Willsky, A.S.

LIDS-P-2027

Parallel Smoothing Algorithms for Causal and Acausal Systems[1]

Darrin Taylor and Alan S. Willsky

Abstract

In this paper we describe parallel processing algorithms for optimal smoothing for discrete time linear systems described by two point boundary value difference equations. These algorithms involve the partitioning of the data interval with one processor for each subinterval. The processing structures considered consists of independent parallel processing on each subinterval, followed by an information exchange between processors and then a final sweep of independent subinterval processing. The local processing procedures that we describe produce maximum likelihood (ML) estimates in which dynamics and a priori conditions play the same role as measurements, i.e. they are all noisy constraints. Consideration of such ML procedures for descriptor systems requires that we develop a general procedure for recursive estimation in situations in which neither the error covariance nor its inverse is well defined. This leads among other things to a generalization of the well known Mayne-Fraser two filter algorithm in which the two directions of processing are treated symmetrically,. Furthermore using an ML procedure for the local processing step leads to considerable simplification of the subsequent interprocessor information exchange step. We present both a two filter implementation of this step as well as a highly parallel implementation exactly matched to the hypercube computer architecture. This algorithm by itself yields a new parallel smoothing algorithm and also, significantly, is extendible to higher dimension offering the promise of even more significant computational savings for applications involving the estimation of random fields.

[1]The authors are with the Dept. of Electrical Engineering and Computer Science and the Laboratory for Information and Decision Systems, MIT Cambridge, MA 02139. This work was supported in part by the Air Force Office of Scientific Research under Grant AFOSR-88-0032, and in part by the US Army Research Office under Contract DAAL03-86-K- 0171, and by the Office of Naval Research under Grant N00014-91-J-1004.

1 Introduction

In this paper we describe algorithms for parallel optimal smoothing for systems described by two point boundary value descriptor systems (TPBVDS's), a class that includes both the standard causal model as well as a rich class of acausal models. There are several reasons for interest in parallel smoothing algorithms. First, the processing environment has changed substantially over the years to allow for multiprocessor computational environments. The popular recursive estimation algorithms which were developed based on the Kalman filter, were designed to function in a single processor environment. In addition estimation of multi-dimensional processes nearly necessitates a multiprocessor environment. Specifically, while the boundary of a one dimensional process does not increase when the interval of consideration increases, a two dimensional system has a boundary that grows at a rate no smaller than the square root of the size of the region being considered. This is significant since the size of the boundary gives an indication of the complexity of the system. In particular, if we think of the dimension of the 'state' of a system as both a measure of the complexity of the system (e.g. in terms of required storage), and as a set of required boundary conditions needed for further computation, we see that there is a dramatic difference between the 1-D and the 2-D cases. Thus, for large regions partitioning the data and processing it separately makes sense in order to reduce the complexity of the entire algorithm. This not only yields time savings, in 1-D as well as in 2D, but also may be essential in 2-D in order to keep the computational burden on individual processors within reason.

Although this paper considers only the one dimensional problem, multidimensional estimation considerations are a guide to judge which approaches to consider for parallel estimation. In particular we wish to develop algorithms motivated by and (hopefully) generalizable to higher dimensions and which demonstrate the promise of efficient multidimensional estimation algorithms. Also, another criteria by which to guide us in consideration of parallel estimation algorithms is the notion of fault tolerance. In the event that a processor fails, do the remaining processors produce useful information? It is not enough to take an algorithm and parallelize it. It is important that the local processing involves computing meaningful information so that alternate strategies can be employed to recover useful information in the event of processor or communication failure. In particular we seek algorithms in which each processing step has a precise interpretation as producing an optimal estimate in some sense. Furthermore, we also wish to obtain algorithms that are capable of providing estimation error covariances. Such information not only allows performance assessment but also is essential for fault tolerant operation in which the absence of one or more information source must be accounted for in a statistically optimal fashion.

The two parallel estimation algorithms described here have common characteristics. First, the data is partitioned among the processors. Local calculations are performed by processors on their own sets of data. Local information is then exchanged between processors and this is followed by a parallel post processing step in which each processor updates the estimates on its subinterval to produce the final globally optimal estimate over the entire data interval. While a variety of approaches have been developed for various optimal estimation problems [1-7], only two of these [6,7] employ a similar data partitioning structure for parallel filtering and smoothing for causal systems. In [6] a square root algorithm is used for parallel filtering on the subintervals assuming perfect knowledge of the state at one endpoint. This is followed by an interprocessor information exchange and computation step. This step which is based on a change of initial condition formula in order to correct for imperfect endpoint knowledge, is similar in structure to the Mayne-Fraser two filter smoother in order to obtain optimal smoothed estimates at the boundaries of the data intervals and to allow subsequent parallel computation of smoothed estimates within each subinterval. A somewhat more efficient algorithm, with a similar structure, is described in [7]. This procedure deals symmetrically with the two endpoints of each subinterval by initially processing data outward toward and in the final step inward from the boundary points (essentially using in each interval a joint model for $x(t)$ and $x(-t)$, with $t = 0$ corresponding to the center of the interval). The interprocessor exchange steps makes use of the so called partition theorem [10], resulting again in a two-filter sweep from processor to processor in both directions to produce optimal estimates at all boundary points.

The algorithms we describe here bear some similarity to these approaches (in particular, and as we have indicated, they also use the same data partitioning and three step structure), but they also have some significant differences. First of all we deal here with the more general class of TPBVDS's. Also at each stage of our processing we compute maximum likelihood (ML) estimates based on the available information. Here as in [12], we essentially adopt the perspective that a priori statistics, dynamical relationships, and actual observations all play the same role, namely as noisy constraints. The use of this formalism has several important implications, perhaps most notably in the simplification and greatly enhanced flexibility it provides us in the interprocessor exchange step. However, let us first comment on some of the implications for the local processing step.

Specifically as discussed in [11] we can without any loss of generality restrict our attention to so called separable TPBVDS's (STPBVDS's) in which independent boundary conditions are specified at two ends of the interval, say , $t = \pm T$. In particular, if we do not have such separable conditions on $x(t)$, we can obtain them by considering the evolution of $z(t) = [x^T(t), x^T(-t)]^T$, so that the original boundary condition is now a condition on $z(t)$, and we acquire a separate condition on $z(0)$, namely that its two components must be exactly equal. Obviously this construction points to a connection with the model in [7]. Note also that as is clear from this construction and as must be true for any well posed TPBVDS, only partial boundary conditions are available at each end. Viewing these as initial measurements for recursive ML estimation procedures starting at either end of the data interval, we see that at least initially only incomplete information is available, which would seem to imply that we should use the information form in any recursive procedure, i.e to propagate P^{-1}, and $P^{-1}\hat{x}$. This could be carried out at least until P^{-1} is invertible so that P is well defined. However as discussed in [13], the consideration of descriptor dynamics with possibly singular dynamic matrices, also means that we may have some noiseless constraints or as in the boundary constraints on the two components of $z(0)$, implying that P^{-1} is not well defined either!

The preceding discussion makes clear that in considering recursive ML estimation for TPBVDS's we must directly confront the problem of estimation in the face of degeneracy, where the linear equations yielding the ML estimate need not have a unique solution (so that at least some part of $x(t)$ is unconstrained by available information) but may yield perfect estimates of other parts of $x(t)$. The generalized framework of such generalized estimation in the static case is developed in [9] (see also [8]). In [12] the results of [9] are used to develop recursive filtering procedures for TPBVDS in the case when all variables are estimable, (so that P is well defined). What we describe in the next section are algorithms for optimal STPBVDS smoothing in the general case. In particular we describe generalizations of the well known Mayne-Fraser and Rauch-Tung-Striebel algorithms and in fact provide a completely symmetric version of the first of these in which each of the two filters is initialized with the independent boundary information available to it. The algorithms described in Section 2, in addition to being of interest in their own right, also provide us with the first and third local processing steps for our data partitioned parallel processing procedure. Two new algorithms for the second, interprocessor data step are described in Section 3. As in [6,7], we can view the output of the first step as producing 'measurements' of $x(t)$ at the boundaries. However in the Bayesian approaches of [6,7], the errors in these 'measurements' are correlated since each local processor makes use of common prior information. This leads to the comparatively involved two filter procedure in [6,7] for exchanging and fusing endpoint information among processors. In contrast, by adopting the ML formalism we guarantee that the result of out first local processing step produces independent 'measurements' of boundary points. This leads to an algorithm, similar in structure but far simpler than the approach in [7],or[6].

However, it is the second algorithmic structure described in Section 3 that we feel is most novel and noteworthy. First of all, unlike the data exchange steps in [6,7] and our first algorithm, the data exchange structure of our second algorithm is itself highly parallel in nature and is in fact perfectly matched to the hypercube architecture. Secondly this algorithm can be applied to the original discrete data without local processing steps before and after it, yielding by itself a new highly parallel smoothing algorithm matched to a very different computer architecture than that in [6,7] Finally, the basic structure of this algorithm can be extended to multiple dimensions, offering the promise of achieving the needed efficiencies mentioned previously. We comment on this a bit more as we conclude the paper in Section 4.

2 Maximum Likelihood Recursive Estimation for TPBVDS's

2.1 Maximum Likelihood Estimation

In this section we describe algorithms for recursive filtering and smoothing for TPBVDS's As we indicated in the introduction, we adopt an ML perspective in part with an eye toward the parallel processing procedures of Section 3 and in part because such a formalism is particularly natural for descriptor systems in which the dynamics are more appropriately thought of as constraints rather than the basis of recursion. Also, as we have indicated, the problems of interest to us require that we examine, situations in which neither the estimation error covariance nor its inverse are well defined. To this end, let us begin with a brief look at a static ML estimation problem. Specifically consider the problem of estimating an unknown vector x based on the observations

$$y = Hx + v \tag{1}$$

where v is a zero mean Gaussian random variable, with possibly singular covariance R, and where H need not have full column rank, so that some part of x may be perfectly reconstructed while another part remains completely unknown. What we mean by an optimal estimate $\hat{x}_{ML}$ in this case is a linear function of y so that if $c^T x$ is estimable (i.e if a finite variance estimate of it can be constructed), then $c^T \hat{x}_{ML}$ is a minimum variance estimate of $c^T x$. Note that in general $\hat{x}_{ML}$ is not unique, as no constraint is placed on the non-estimable portion of x. The solution we choose is the minimum norm solution given by

$$\hat{x}_{ML} = \begin{bmatrix} 0 \\ I \end{bmatrix}^T \begin{bmatrix} R & H \\ H^T & 0 \end{bmatrix}^{\#} \begin{bmatrix} y \\ 0 \end{bmatrix} \tag{2}$$

where # denotes the Moore-Penrose pseudoinverse. As developed thoroughly in [9], (see also [12]) other generalized inverses can be used to obtain other valid choices for the ML estimate. However, this is the one we require for purposes we now outline, without proof (see [13] for details)

Note that the range space of the symmetric projection matrix

$$P_x = (H^T H)^{\#}(H^T H) \tag{3}$$

determines the estimable subspace of x. Then $P_x \hat{x}_{ML} = \hat{x}_{ML}$ and furthermore

$$Cov(P_x x - \hat{x}_{ML}) = \begin{bmatrix} 0 \\ I \end{bmatrix}^T \begin{bmatrix} R & H \\ H^T & 0 \end{bmatrix}^{\#} \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{4}$$

Note also that

$$P_x Cov(P_x x - \hat{x}_{ML}) P_x = Cov(P_x x - \hat{x}_{ML}) \tag{5}$$

Furthermore the following properties of $\hat{x}_{ML}$ are critical in deriving the recursive structure described later in this section:

a) Consider the problem of ML estimation of x and z based on the data in (3) together with the measurements

$$w = Gx + Jz + u \tag{6}$$

where u is a zero mean random vector uncorrelated with v . Then the optimal ML estimate for this problem is the same as that based on (6) and the observation

$$\hat{x}_{ML} = P_x x + \tilde{x}_{ML} \tag{7}$$

where $\tilde{x}_{ML}$ is a zero mean Gaussian random vector independent of u with covariance given by (4)

b) Suppose that

$$z = Ax \tag{8}$$

The ML estimate of z based on (3) is given by

$$\hat{z}_{ML} = P_z A \hat{x}_{ML} \tag{9}$$

$$Cov(P_z z - \hat{z}_{ML}) = P_z A \; Cov(P_x x - \hat{x}_{ML}) A^T P_z \tag{10}$$

where P_z is the largest rank symmetric projection matrix such that

$$P_z A(I - P_x) = 0 \tag{11}$$

and is given by the following

$$P_z = I - (A(I - P_x)A^T)^{\#}(A(I - P_x)A^T) \tag{12}$$

Note that one implication of (a) is that we can use our formalism for the recursive incorporation of information (so if in particular $J = 0$ in (6), and x is estimable based on (3), and (6), this procedure yields the unique optimal estimate of all of x). Also if A is invertible in (8), then P_z has the same rank as P_x. However, if A is singular, it is possible that P_z will have larger rank because A may kill some of the non-estimable portions of x.

2.2 Two Point Boundary Value Descriptor Systems

A general TPBVDS has the following form:

$$E(t+1)x(t+1) = A(t)x(t) + B_t u(t) \quad -T \leq t \leq T-1 \tag{13}$$

$$y(t) = C(t)x(t) + v(t) \quad -T \leq t \leq T \tag{14}$$

$$E(-T)x(-T) = A(T)x(T) + B_T u(T) \tag{15}$$

where $[u^T(t), v^T(t)]^T]$ is a white noise process with

$$Cov \begin{bmatrix} u(t) \\ v(t) \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & R(t+1) \end{bmatrix} \tag{16}$$

Note that written in this form, we have made little distinction between the boundary condition (15) on the process and the dynamics (13). We assume for simplicity, that the system (13), (15) is well-posed, i.e. that (13), (15) admits a unique solution for any choice of u(t), although, as in [12], the results here can be extend to a more general setting in which (13), (15) simply provide either an over- or under-constrained set of noisy constraints (and where, in fact x(t) may vary in dimension).

In general the boundary conditions (15) couple the values of $x(t)$ at the two endpoints $t = \pm T$. An important subclass of TPBVDS's are those that are separable in which (15) specifies independent constraints of $x(-T)$ and $x(T)$. (In the well-posed case (15) is a set of constraints of dimension equal to that of $x(t)$, so that in the separable case (15) provides incomplete constraints of at least one of the two boundary points). As shown in [11] it is always possible to transform a TPBVDS to one that is separable by considering the joint evolution of $x(t)$ and $x(-t)$:

$$\begin{bmatrix} E(t+1) & 0 \\ 0 & A(-t-1) \end{bmatrix} \begin{bmatrix} x(t+1) \\ x(-t-1) \end{bmatrix} = \tag{17}$$

$$\begin{bmatrix} A(t) & 0 \\ 0 & E(-t) \end{bmatrix} \begin{bmatrix} x(t) \\ x(-t) \end{bmatrix} + \begin{bmatrix} B_t & 0 \\ 0 & -B_{-t-1} \end{bmatrix} \begin{bmatrix} u(t) \\ u(-t-1) \end{bmatrix} \tag{18}$$

$$\begin{bmatrix} I & -I \end{bmatrix} \begin{bmatrix} x(0) \\ x(-0) \end{bmatrix} = 0 \tag{19}$$

$$\begin{bmatrix} -A(T) & E(-T) \end{bmatrix} \begin{bmatrix} x(T) \\ x(-T) \end{bmatrix} = B_T u(T) \tag{20}$$

$$\begin{bmatrix} y(t) \\ y(-t) \end{bmatrix} = \begin{bmatrix} C(t) & 0 \\ 0 & C(-t) \end{bmatrix} \begin{bmatrix} x(t) \\ x(-t) \end{bmatrix} + \begin{bmatrix} r(t) \\ r(-t) \end{bmatrix} \tag{21}$$

where (17) is defined for $0 \le t \le T-1$, (21) for $0 \le t \le T$, and the boundary conditions (19) and (20) are indeed separable. Note that (17), (19),and (20) represent the original system starting from the center, then moving outward. Thus any smoothing algorithm based on this model will involve processing outward toward and inward from the boundaries. Note also that the boundary conditions (19),and (20) provide only partial information about the states $[x^T(t), x^T(-t)]$ at $t = 0$, and $t = T$ and furthermore the boundary condition at $t = 0$ provides perfect information about part of the state $[x^T(0), x^T(-0)]$.

To continue, let us revert to a simpler notation for a general STPBVDS:

$$E(t+1)x(t+1) = A(t)x(t) + B_t u(t) \quad 0 \le t \le T-1 \tag{22}$$

$$y(t) = C(t)x(t) + v(t) \quad 0 \le t \le T \tag{23}$$

$$Cov \begin{bmatrix} u(t) \\ v(t+1) \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & R_{t+1} \end{bmatrix} \tag{24}$$

Here, consistent with our ML perspective, we have incorporated the independent boundary conditions on $x(0)$ and $x(T)$ into the measurements (23) at $t = 0$, and $t = T$. Viewing (22) through (24) as providing a set of noisy constraints, we can apply the ML estimation results of Subsection 2.1 to obtain recursive estimation algorithms. In presenting these algorithms it is useful to define two auxiliary (forward and backward prediction) variables.

$$z^f(t) = E(t)x(t) \tag{25}$$

$$z^b(t) = A(t)x(t) \tag{26}$$

Let $\hat{x}_{ML}[s|0,t]$ denote the ML estimate of $x(s)$ based on (22) for $0 \le \tau \le t-1$,and (23) for $0 \le \tau \le t$, and $\hat{z}^f_{ML}[s|0,t]$ denote the ML estimate of $z(s)$ based on (22) for $0 \le \tau \le t$,and (23) for $0 \le \tau \le t$. We then obtain the following forward ML filter (FMLF) equations:

$$\hat{x}_{ML}[t|0,t] = \begin{bmatrix} 0 \\ 0 \\ 0 \\ I \end{bmatrix}^T \begin{bmatrix} \Sigma^f_z[t|0,t-1] & 0 & I - P_{zf}(t) & E(t) \\ 0 & R(t) & 0 & C(t) \\ I - P_{zf}(t) & 0 & 0 & 0 \\ E^T(t) & C^T(t) & 0 & 0 \end{bmatrix}^{\#} \begin{bmatrix} \hat{z}^f_{ML}[t|0,t-1] \\ y(t) \\ 0 \\ 0 \end{bmatrix} \tag{27}$$

where

$$\Sigma^f_x[t|0,t] = \begin{bmatrix} 0 \\ 0 \\ 0 \\ I \end{bmatrix}^T \begin{bmatrix} \Sigma^f_z[t|0,t-1] & 0 & (I - P_{zf}(t)) & E(t) \\ 0 & R(t) & 0 & C(t) \\ (I - P_{zf}(t)) & 0 & 0 & 0 \\ E^T(t) & C^T(t) & 0 & 0 \end{bmatrix}^{\#} \begin{bmatrix} 0 \\ 0 \\ 0 \\ I \end{bmatrix} \tag{28}$$

$$P_x^f(t) = (E^T(t)P_{z^f}(t)E(t))^{\#}(E^T(t)P_{z^f}(t)E(t)) \tag{29}$$
$$\hat{z}_{ML}^f[t+1|0,t] = P_{z^f}(t+1)A(t)\hat{x}_{ML}[t|0,t] \tag{30}$$
$$\Sigma_z^f[t+1|0,t] = P_{z^f}(t+1)(A(t)\Sigma_x^f[t|0,t]A^T(t) + B(t)B^T(t))P_{z^f}(t+1) \tag{31}$$
$$P_{z^f}(t+1) = I - (A(t)(I - P_x^f(t))A^T(t))^{\#}(A(t)(I - P_x^f(t))A^T(t)) \tag{32}$$

where $P_x^f(t)$ indicates the symmetric projection matrix which defines the estimable part of $x(t)$ based on data through time t, and $P_{z^f}(t)$ is the symmetric projection matrix which defines the estimable part of $z^f(t)$. Equations (27) through (31) represent the generalization of [12] to allow for the possibility that $x(t)$ and / or $z^f(t)$ are not completely estimable. Also $\Sigma_x[t|0,t]$ can be thought of as the error covariance in the estimate of $\hat{x}_{ML}[t|0,t]$ in the sense of (4). $\Sigma_x[t|0,t]$ represents the corresponding error covariance for the estimable part of $x(t)$ The matrix $\Sigma_z^f[t+1|0,t]$ has a similar interpretation for $\hat{z}_{ML}^f[t+1|0,t]$.

Similarly we can define the backward ML filter (BMLF) where $\hat{x}_{ML}[s|t,T]$ denotes the ML estimate of $x(s)$ based on (22), for $t \le \tau \le T$ and (23) for $t \le \tau \le T$, and $\hat{z}_{ML}^b[s|t,T]$ denotes the ML estimate of $z^b(s)$ based on (22), for $t-1 \le \tau \le T$ and (23) for $t \le \tau \le T$.The BMLF is then given by

$$\hat{x}_{ML}[t|t,T] = \begin{bmatrix} 0 \\ 0 \\ 0 \\ I \end{bmatrix}^T \begin{bmatrix} \Sigma_z^b[t|t+1,T] & 0 & I - P_{z^b}(t) & A(t) \\ 0 & R(t) & 0 & C(t) \\ I - P_{z^b}(t) & 0 & 0 & 0 \\ A^T(t) & C^T(t) & 0 & 0 \end{bmatrix}^{\#} \begin{bmatrix} \hat{z}_{ML}^b[t|t+1,T] \\ y(t) \\ 0 \\ 0 \end{bmatrix} \tag{33}$$

$$\Sigma_x^f[t|t,T] = \begin{bmatrix} 0 \\ 0 \\ 0 \\ I \end{bmatrix}^T \begin{bmatrix} \Sigma_z^b[t|t+1,T] & 0 & (I - P_{z^b}(t)) & A(t) \\ 0 & R(t) & 0 & C(t) \\ (I - P_{z^b}(t)) & 0 & 0 & 0 \\ A^T(t) & C^T(t) & 0 & 0 \end{bmatrix}^{\#} \begin{bmatrix} 0 \\ 0 \\ 0 \\ I \end{bmatrix}$$
$$P_x^b(t) = (A^T(t)P_{z^f}(t)A(t))^{\#}(A^T(t)P_{z^f}(t)A(t)) \tag{34}$$
$$\hat{z}_{ML}^b[t-1|t,T] = P_{z^b}(t)E(t)\hat{x}_{ML}[t|t,T] \tag{35}$$
$$\Sigma_{z^b}[t|t+1,T] = P_{z^b}(t)(E(t+1)\Sigma_x^b[t+1|t+1,T]E(t+1)^T + B(t)B^T(t))P_{z^b}(t) \tag{36}$$
$$P_{z^b}(t+1) = I - (E(t)(I - P_x(t))E(t)^T)^{\#}(E(t)(I - P_x(t))E(t)^T) \tag{37}$$

Also the FMLF and the BMLF can be combined to produce the optimal smoothed estimate using one of two forms. The first combines forward filtered with backward predicted estimates, and the second combines forward predicted with backward filtered estimates.

$$\hat{x}_{ML}[t|0,T] = \tag{38}$$
$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ I \end{bmatrix}^T \begin{bmatrix} \Sigma_x^f[t|0,t] & 0 & I - P_x^f(t) & 0 & I \\ 0 & \Sigma_{z^b}[t|t+1,T] & 0 & I - P_{z^b}(t) & A(t) \\ I - P_x^f(t) & 0 & 0 & 0 & 0 \\ 0 & I - P_{z^b}(t) & 0 & 0 & 0 \\ I & A^T(t) & 0 & 0 & 0 \end{bmatrix}^{\#} \begin{bmatrix} \hat{x}_{ML}[t|0,t] \\ \hat{z}_{ML}^b[t|t+1,T]) \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\hat{x}_{ML}[t|0,T] = \tag{39}$$

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ I \end{bmatrix}^T \begin{bmatrix} \Sigma_x^b[t|t,T] & 0 & I - P_x^b(t) & 0 & I \\ 0 & \Sigma_{zf}[t|0,t-1] & 0 & I - P_{zf}(t) & E(t) \\ I - P_x^b(t) & 0 & 0 & 0 & 0 \\ 0 & I - P_{zf}(t) & 0 & 0 & 0 \\ I & E^T(t) & 0 & 0 & 0 \end{bmatrix}^{\#} \begin{bmatrix} \hat{x}_{ML}[t|t,T] \\ \hat{z}^f_{ML}[t|0,t-1]) \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

The smoothed error covariances are given by the following

$$\Sigma_x[t|0,T] = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ I \end{bmatrix}^T \begin{bmatrix} \Sigma_x^f[t|0,t] & 0 & P_x(t) & 0 & I \\ 0 & \Sigma_{zb}[t|t+1,T] & 0 & P_{zb}(t) & A(t) \\ P_x(t) & 0 & 0 & 0 & 0 \\ 0 & P_{zb}(t) & 0 & 0 & 0 \\ I & A^T(t) & 0 & 0 & 0 \end{bmatrix}^{\#} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ I \end{bmatrix} \qquad (40)$$

$$\Sigma_x[t|0,T] = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ I \end{bmatrix}^T \begin{bmatrix} \Sigma_x^b[t|t,T] & 0 & P_x(t) & 0 & I \\ 0 & \Sigma_{zf}[t|0,t-1] & 0 & P_{zf}(t) & E(t) \\ P_x(t) & 0 & 0 & 0 & 0 \\ 0 & P_{zf}(t) & 0 & 0 & 0 \\ I & E^T(t) & 0 & 0 & 0 \end{bmatrix}^{\#} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ I \end{bmatrix} \qquad (41)$$

The FMLF and the BMLF together with either (38), or (39) form a generalization of the Mayne-Fraser two filter formulas for optimal smoothing on STPBVDS's in the case where $x(t)$ may not be estimable, while portions of it are specified perfectly. Specifically if E=I and only initial conditions are specified (making the system well posed), the FMLF and the BMLF and (39) reduce to the usual Mayne-Fraser equations. As a result, the generalization to STPBVDS deals in a symmetric way with information available at the two ends of the interval.

It is also possible to generalize the Rauch-Tung-Striebel algorithm to STPBVDS's. This algorithm involves a forward sweep to compute $\hat{x}_{ML}[t|0,t]$ for each t producing the smoothed estimate $\hat{x}_s(T) = \hat{x}_{ML}[T|0,T]$ at one endpoint, which initiates a reverse sweep to compute $\hat{x}_s(t) = \hat{x}_{ML}[t|0,T]$ over the entire interval. The key to this backward sweep is again to interpret it as the computation of ML estimates based on an appropriate set of observations. In particular suppose that we have computed $\hat{x}_s(t+1)$ and its corresponding error covariance $\Sigma_x[t+1|0,T]$, where $\Sigma_x[t+1|0,T]$ is interpreted as in (4) if x(t) is not estimable. Then as shown in [13] the computation of $\hat{x}_s(t)$ and $\Sigma_x[t+1|0,T]$ can be obtained by solving the following ML estimation problem which captures all relevant information relating $x(t)$ to $x(t+1)$ and the available estimates of of each of these:

$$\begin{bmatrix} \hat{x}_{ML}[t|0,t] \\ \hat{z}^f_{ML}[t+1|0,t] \\ 0 \\ E(t+1)\hat{x}_s(t+1) \end{bmatrix} = \begin{bmatrix} P_x^f(t) & 0 \\ 0 & P_{zf}(t+1) \\ (I - P_{zf}(t+1))A(t) & (I - P_{zf}(t+1)) \\ 0 & I \end{bmatrix} \begin{bmatrix} x(t) \\ z^f(t+1) \end{bmatrix} + \nu(t) (42)$$

By choosing appropriate change of variables this can be partitioned into two independent observations.

$$\begin{bmatrix} \hat{x}_{ML}[t|0,t] - \{\Sigma_x[t|0,t]A^T(t)\Sigma^{\#}_{zf}[t+1|0,t]\}\hat{z}^f_{ML}[t+1|0,t] \\ \{(I - P_{zf}(t+1))B^T(t)B(t)\Sigma^{\#}_{zf}[t+1|0,t]\}\hat{z}^f_{ML}[t+1|0,t] \end{bmatrix} = \qquad (43)$$

$$\begin{bmatrix} P_x^f(t) & -\Sigma_x[t|0,t]A^T(t)\Sigma_{zf}^{\#}[t+1|0,t] \\ (P_{zf}(t+1)-I)A & (I-P_{zf}(t+1))(A(t)-B_T(t)B(t)\Sigma_{zf}^{\#}[t+1|0,t]) \end{bmatrix} \begin{bmatrix} x(t) \\ z^f(t+1) \end{bmatrix} + \nu_1(t)$$

$$\begin{bmatrix} \hat{z}_{ML}^f[t+1|0,t] \\ E(t+1)\hat{x}[t+1|0,T] \end{bmatrix} \begin{bmatrix} P_{zf}(t+1) \\ I \end{bmatrix} z^f(t+1) + \nu_2(t) \tag{44}$$

where $\nu_1(t)$ has a covariance given by

$$Cov[\nu_1(t)]_{11} = \Sigma_x^f[t|0,t] - \Sigma_x[t|0,t]A(t)^T\Sigma_{zf}^{\#}[t+1|0,t]A(t)\Sigma_x^f[t|0,t] \tag{45}$$

$$Cov[\nu_1(t)]_{12} = -\Sigma_x^f[t|0,t]A(t)^T\Sigma_{zf}^{\#}[t+1|0,t]B^T(t)B(t)(I-P_{zf}(t+1)) \tag{46}$$

$$Cov[\nu_1(t)]_{21} = -(I-P_{zf}(t+1))B^T(t)B(t)\Sigma_{zf}^{\#}[t+1|0,t]A(t)\Sigma_x^f[t|0,t]A^T(t) \tag{47}$$

$$Cov[\nu_1(t)]_{22} = (I-P_{zf}(t+1))B^T(t)B(t) - B^T(t)B(t)\Sigma_{zf}^{\#}[t+1|0,t]B^T(t)B(t)(I-P_{zf}) \tag{48}$$

In equation (44) two observations are provided of the estimate of $z^f(t+1)$ since one of the measurements is the smoothed estimate no additional information is contained in $\hat{z}_{ML}^f[t+1|0,t]$. The estimate is therefore equal to $E\hat{x}[t+1|0,T]$ with error covariance given by $E(t+1)\Sigma_x^f[t+1|0,T]E^T(t+1)$.

The resulting ML estimate of $x(t)$ is precisely $\hat{x}_s(t)$. In the causal case in which E=I and all covariances are well defined and invertible, this reduces to the usual Rauch-Tung-Striebel algorithm. We refer the reader to [13] for explicit computations in the more general case.

3 Parallel Smoothing Algorithms

In this section we describe two highly parallel algorithms for optimal smoothing for TPBVDS as in (13) - (15). Amplifying on our discussion in Section 1, our algorithms have the following structure. First the overall data interval of definition is partitioned into disjoint subintervals. In each such subinterval we define the STPBVDS model (17), (19),(21) with the time origin taken at the center of the subinterval and perform outward filtering using the FMLF described in the preceding section. At the end of this stage information must be exchanged among the subinterval processors. From the perspective of any individual subinterval, the relevant information from all other subintervals can be interpreted as providing additional measurements of $x(t)$ at the boundaries of this interval. Once this information is incorporated, each subinterval processor can proceed independently with either the BMLF / Mayne-Fraser procedure or the Rauch-Tung-Striebel algorithm described in the preceding section in order to produce optimal smoothed estimates across the entire subinterval. At this stage the advantage of the Rauch-Tung-Striebel algorithm is that the original data is not necessary to recover smoothed estimates.

The preceding description requires several additional comments. First since the most general boundary conditions for TPBVDS (15) couples $x(-T)$ with $x(T)$, our partitioning into subintervals must in essence view the points $-T$ and T as neighbors. Thus for example if we partition our data into two subintervals, the natural choice of partition is $[-T/2, T/2]$ and $[-T,(-T/2)-1] \bigcup [(T/2)+1, T]$. In this case the outward processing over the first subinterval

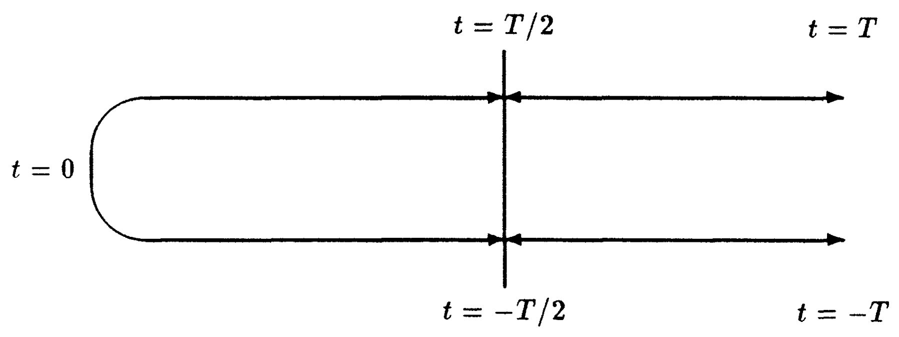

Figure 1: Combining data at $\pm T/2$

incorporates the boundary measurement (19) while that for the other interval has as its 'center' the pair of points $[x(-T), x(T)]$ and incorporates the boundary 'measurement' (20), In general this boundary condition couples the two components of the state $[x^T(t), x^T(-t]^T$ over the interval $t \in [T/2 + 1, T]$. However, if the original system is separable, these two components are completely decoupled implying as illustrated in Figure 1 that we in fact have a three interval decomposition with outward processing in the central interval and completely decoupled processing at the two ends. For simplicity in the subsequent discussion we will assume that the processing in the end intervals is also outward from their centers. If further subinterval decomposition is performed the additional intervals also employ outward processing.

Because of the discrete nature of our time index, a general comment is required concerning the precise nature of the data exchange step. To illustrate this consider the case of an STPBVDS and the three interval decomposition described in the preceding paragraph. In this case one might expect the outward processing in the central interval to culminate with the filtered estimates $\hat{x}_{ML}[T/2| - T/2, T/2]$, and $\hat{x}_{ML}[-T/2| - T/2, T/2]$ while the two outer interval culminate in $\hat{x}_{ML}[(T/2) + 1|(T/2) + 1, T]$ and $\hat{x}_{ML}[-(T/2) - 1| - T, -(T/2) - 1]$. However if we wish to view neighboring intervals as providing 'measurements' at subinterval endpoints then we need to produce predicted estimates as well. Note in particular that these prediction steps in essence incorporate the final dynamic constraints not used in the first local processing step, namely those relating boundary points of the neighboring subintervals.

$$E((T/2) + 1)x((T/2) + 1) = A(T/2)x(T/2) + B(T/2)u(T/2) \quad (49)$$
$$E(-T/2)x(-T/2) = A(-T/2) - 1)x((-T/2) - 1) + B(-T/2) - 1)u((-T/2) - 1) \quad (50)$$

To simplify the discussion in the remainder of this section, we focus on the case of STPBVDS's (17). Furthermore we assume that the values of $x(t)$ at subinterval boundary endpoints are estimable based on the data in the subinterval. As a result it is no longer necessary to propagate pseudo-inverses in our subsequent discussion since all covariances are well defined. The results of Section 2 can of course be used in the more general case. By making the assumption of strong observability introduced in [12] we are able to guarantee the estimability of $x(t)$ at the end points. In the case of constant coefficients systems the assumption of strong observability implies that $x(t)$ and $x(-t)$ are jointly estimable based on data over the interval $[-t, t]$(so that the joint error covariance is well defined) as long as $t \geq n$ where n is the dimension of $x(t)$.[2]. In addition it is useful to adopt simplified notation describing only those variables of interest in the exchange step. Specifically, again using an ML perspective, we have the following unknowns which we wish to estimate.

$$\begin{bmatrix} x_0 \\ x_1 \end{bmatrix}, \begin{bmatrix} x_2 \\ x_3 \end{bmatrix}, \begin{bmatrix} x_4 \\ x_5 \end{bmatrix}, \ldots, \begin{bmatrix} x_{2m-4} \\ x_{2m-3} \end{bmatrix}, \begin{bmatrix} x_{2m-2} \\ x_{2m-1} \end{bmatrix} \tag{51}$$

where x_{2k-2} represents the left most internal boundary point of the $(k)^{th}$ subinterval, and x_{2k-1} represent the right most boundary point of the $(k)^{th}$ interval. The time indices indicate that the indicated quantities are appropriate samples of $x(t)$ at the various endpoints. In our three interval example, $[x_0^T, x_1^T]$ is given by $[x^T(0), x^T((-T/2) - 1)]$, $[x_2^T, x_3^T]$ is given by $[x^T(-T/2), x^T(T/2)]$ and finally $[x_{2m-2}^T, x_{2m-1}^T]$ is given by $[x^T((T/2)+1), x^T]$. Our estimates of these variables are based on the following 'measurements'

$$\begin{bmatrix} y_{2i-2} \\ y_{2i-1} \end{bmatrix} = \begin{bmatrix} x_{2i-2} \\ x_{2i-1} \end{bmatrix} + \begin{bmatrix} \epsilon_{2i-2} \\ \epsilon_{2i-1} \end{bmatrix} \tag{52}$$

$$\tag{53}$$

as well as the following additional noisy constraints.

$$E_{2i}x_{2i} = A_{2i-1}x_{2i-1} + B_{2i-1}u_{2i-1} \tag{54}$$

where ϵ_i, and u_i are independent Gaussian random variables with the following covariances.

$$Cov(u_i) = I \tag{55}$$

$$Cov \begin{bmatrix} \epsilon_{2i-2} \\ \epsilon_{2i-1} \end{bmatrix} = \begin{bmatrix} R_{2i-2} & R_{2i-2,2i-1} \\ R_{2i-1,2i-2} & R_{2i-1} \end{bmatrix} \tag{56}$$

Here the 'measurements' (52) correspond to the independent endpoint estimates produced by each subinterval processor during the first stage, while the constraints (54) correspond to the dynamics (22) across subinterval boundaries. Note that because of our adoption of an ML procedure for the first stage, the zero mean gaussian variables ϵ_i, and u_j are mutually independent. In contrast to the approaches in [6,7] this leads to dramatic simplifications in terms of interpretations of the result, computations, and preprocessing. Note also that (52), and (54) looks very much like our original STPBVDS. The only difference being the fact that the system (52), and (54) is trivially estimable since (54) provide complete measurements of each x_i and the special form of the dynamics linking the bottom half of one state to the top half of the next. This form allows us to describe two procedures for the data exchange step, one of which is a natural application of the methods in Section 2 and the other of which offers some new possibilities for parallel processing.

Algorithm # 1:

We use a Mayne-Fraser or a Rauch-Tung-Striebel procedure to exchange information between subintervals. In particular let us describe a version of the FMLF tailored to this

[2]Note that this implies that the projection matrices $P_x(t)$ and $P_z(t)$ need only be propagated over a limited interval of length of at most 2n before they are equal to the identity

model. In particular thanks to the form of this of the dynamics (54) our FMLF propagates estimates of the odd numbered $x(t)$'s , i.e. the bottom halves of each state vector. [3] Also let $\hat{x}_{i|j}$ denote the ML estimate of x_i based on y_k, $k \leq j$. Then

$$\hat{x}_{1|1} = y_1 \tag{57}$$

$$\Sigma_{x_{1|1}} = R_1 \tag{58}$$

We can then compute $\hat{x}_{2k-1|2k-1}$ the estimates of x_{2k-1} based on on y_1 through y_{2k-1}, and equations (54) for $i < k-1$. recursively as solutions to the ML estimation problems

$$\begin{bmatrix} \hat{x}_{2k-3|2k-3} \\ 0 \\ y_{2k-2} \\ y_{2k-1} \end{bmatrix} \begin{bmatrix} I & 0 & 0 \\ -A_{2k-3} & E_{2k-2} & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} x_{2k-3|2k-3} \\ x_{2k-2|2k-2} \\ x_{2k-1|2k-1} \end{bmatrix} + \begin{bmatrix} \tilde{x}_{2k-3|2k-3} \\ B_{2k-3}u_{2k-3} \\ \epsilon_{2k-2} \\ \epsilon_{2k-1} \end{bmatrix} \tag{59}$$

where $\tilde{x}_{2i-3|2i-3}$, the error in $\hat{x}_{2k-1|2k-3}$, is uncorrelated with u_{2k-3}, ϵ_{2k-2}, and ϵ_{2k-1} and has covariance $\Sigma_{x_{2i-3|2k-3}}$. Equation (59) is of the form (3), and the solution $\hat{x}_{2k-1|2k-1}$ is directly obtained in the form given by (2). Finally, at the last stage we compute the full smoothed estimate of x_{2m-1} will have been obtained. This FMLF can then be combined either with an analogous BMLF to yield a Mayne-Fraser procedure or with a backward Rauch-Tung-Striebel step. Note that the extension to the case of non-estimable systems can be readily accomplished using the formalism in Section 2. However, even in the case of estimable variables it is still necessary in general to use pseudo-inverses to solve the ML problems since the noise covariances needed to solve (59) are in general singular. [4]

The computational structure of Algorithm # 1 involves essentially serial processing from subinterval to subinterval and thus takes time proportional to m the number of subintervals and in fact does not make use of the parallel computing power of the array processors, although it does only require nearest neighbor connectivity for the subinterval processors. Also the use of a purely serial formalism indicates that this approach is naturally associated with 1D processes. In contrast our second approach is highly parallelizable, although it uses more dense interprocessor communication, corresponding most naturally to a hypercube architecture. Also this approach, which involves propagation from fine to coarse partitions(of the data), is naturally extended to higher dimensions. Furthermore since the subinterval interchange step is itself a TPBVDS smoothing problem, this second approach also provides an alternate parallel processing algorithm for our original TPBVDS smoothing problem.

The statistical interpretation of our second algorithm is best understood by contrasting it with that of Algorithm #1. Specifically in the FMLF step of Algorithm #1 we essentially use the Markovian nature of the TPBVDS to obtain a recursion for the best estimate $\hat{x}_{2i-1|2i-1}$ of the boundary point of a data interval of increasing size based on all of the data within the interval. However this same philosophy leads to the idea of simultaneously obtaining recursions for estimates over several disjoint data intervals of increasing size,

[3] The BMLF then would calculate estimates of the top halves leading to a slightly more involved form for the step for combining the FMLF and BMLF estimates. The Rauch-Tung-Striebel algorithm is similarly modified. See [13] for details.

[4] For example even in causal systems the dynamic constraints [] does not necessarily have a full rank noise process w.

where the estimates are merged as data intervals are joined. Taking this to it full limit, we obtain the following.

Algorithm# 2:

We suppose, for simplicity that $m = 2^K$ so that the number of vectors to be estimated in (51) is a power of 2. To initialize the algorithm we use the measurements y_i, in pairs as independent ML estimates of the following 2^K quantities.

$$\begin{bmatrix} \hat{x}_{2i-2|0} \\ \hat{x}_{2i-1|0} \end{bmatrix} = \begin{bmatrix} y_{2i-2} \\ y_{2i-1} \end{bmatrix}, i = 2, 4, ..., m \tag{60}$$

with the corresponding estimation error given by the appropriate ϵ_j's with the estimation error covariances given in (56). Then the first step of the algorithm merges the estimates in non-overlapping pairs together with the appropriate intervening dynamic constraint (54). Specifically, we can solve in parallel the following 2^K ML estimation problems:

$$\begin{bmatrix} \hat{x}_{2i-2|0} \\ \hat{x}_{2i-1|0} \\ \hat{x}_{2i|0} \\ \hat{x}_{2i+1|0} \end{bmatrix} = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \\ 0 & -A_{2i-1} & E_{2i} & 0 \end{bmatrix} \begin{bmatrix} x_{2i-2} \\ x_{2i-1} \\ x_{2i} \\ x_{2i+1} \end{bmatrix} + \begin{bmatrix} \epsilon_{2i-2} \\ \epsilon_{2i-1} \\ \epsilon_{2i} \\ \epsilon_{2i+1} B_{2i-1} u_{2i-1} \end{bmatrix} \tag{61}$$

where i is an odd integer. Because of the block diagonal nature of the noise covariances in these ML estimation problems, and the special structure of the measurement matrices (i.e. each consists of an identity block together with one dynamic coupling constraint), these ML problems can be solved efficiently (see [13]). Furthermore the resulting estimates, which we denote by $\hat{x}_{2i-2|1}$, $\hat{x}_{2i-1|1}$, $\hat{x}_{2i|1}$, and $\hat{x}_{2i+1|1}$, etc., have independent errors from ML problem to ML problem (e.g. the error in carrying out the estimation indicated in equation (61) for i=1 is uncorrelated with the error in from the same equation when i=3). Note also that we have used half of the dynamic constraints in this first step. To continue the process it is important to realize that we essentially have the same problem as we did at the first stage! To make this more explicit, consider the estimate resulting from (61), for $i = 3$, i.e.$\hat{x}_{4|1}$, $\hat{x}_{5|1}$, $\hat{x}_{6|1}$, and $\hat{x}_{7|1}$ which the best estimates of x_4, x_5, x_6, and x_7 based on the corresponding data from (52) and the intervening dynamic constraint from (54). However, thanks to the Markovian nature our system– or equivalently the local nature of the dynamic constraints– it is only the boundary elements of this set of estimates $\hat{x}_{4|1}$, and $\hat{x}_{7|1}$ that are relevant to the estimation of variables outside this data interval when the remaining dynamic constraints from (54) are taken into account. Thus for the next step of the problem we wish to estimate the variables,

$$\begin{bmatrix} x_0 \\ x_3 \end{bmatrix}, \begin{bmatrix} x_4 \\ x_7 \end{bmatrix}, \begin{bmatrix} x_8 \\ x_{11} \end{bmatrix}, \ldots, \begin{bmatrix} x_{2m-8} \\ x_{2m-5} \end{bmatrix}, \begin{bmatrix} x_{2m-4} \\ x_{2m-1} \end{bmatrix} \tag{62}$$

based on the measurements

$$\begin{bmatrix} x_{0|1} \\ x_{3|1} \end{bmatrix} = \begin{bmatrix} x_0 \\ x_{3|1} \end{bmatrix} + \begin{bmatrix} \tilde{x}_{0|1} \\ \tilde{x}_{3|1} \end{bmatrix} \tag{63}$$

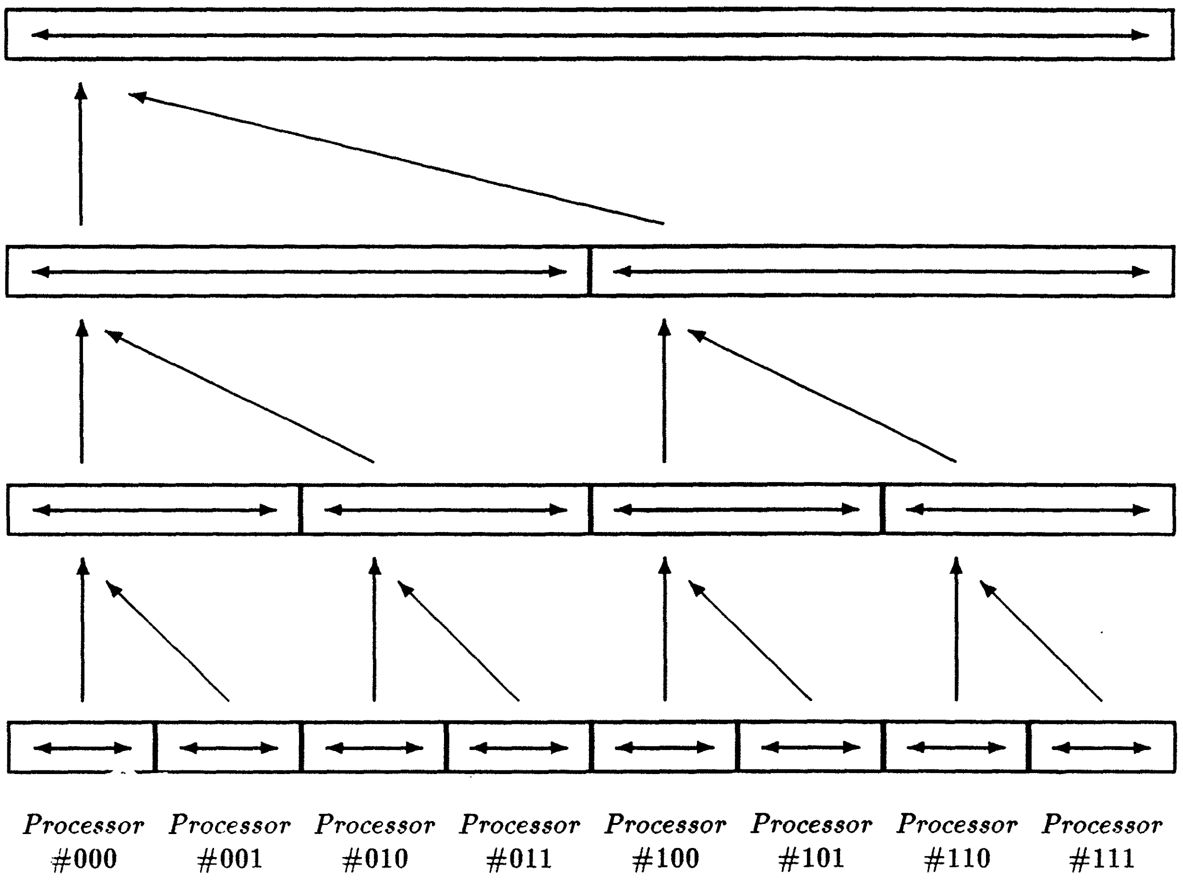

Figure 2: Combining Independent Boundary Data in a Tree Structure

$$\begin{bmatrix} \hat{x}_{4|1} \\ \hat{x}_{7|1} \end{bmatrix} = \begin{bmatrix} x_{4|1} \\ x_{7|1} \end{bmatrix} + \begin{bmatrix} \tilde{x}_{4|1} \\ \tilde{x}_{7|1} \end{bmatrix} \tag{64}$$

$$\vdots \tag{65}$$

and the remaining dynamic constraints (i.e (54) for i=2,4, m) Thus we have half as many variables to estimate based an half as many independent dynamic constraints, and half as many 'measurements' representing the accumulated information over intervals of twice the length as before. The complete processing structure is as depicted in Figure 2. Specifically, we have a tree of computations producing estimates at boundary points of merged intervals that double in size as we move up the tree indicating a coarsening of the data partitioning and a concomitant thinning of the required estimates. All of the computations in going from one level to the next can be calculated in parallel. Since the number of such computations is halved at each level, we have a natural pyramidal structure for the computations. Such a structure is perfectly well-suited to a hypercube architecture in which processors are placed at the vertices of a unit cube in an L-dimensional space and are directly connected to processors at nodes connected by edges. For our problem ideally we would like to use a K-dimensional hypercube so that no processing steps required communication with any latency (i.e., communication between non-adjacent nodes). For example as illustrated in Figure 2, for the case of $m = 3$, the initialization step (corresponding to the initial local processing within each data subinterval) is carried out in parallel on all $8=2^3$ processors. The next step involves pairings of processors that differ in only one bit (i.e. (000, 001), (010, 011), (100, 101), (100,111)) with processing accomplished in the first element of each pair incorporating the data for each of the two processors as well as the intervening dynamic constraint (which is then removed at the next step). At the next level, the remaining active processors are again paired so that there is only one bit difference ((000, 010) and (100, 110)), etc.

Note that when we have reached the top of the tree, we have computed the full optimal estimate at only a pair of the x_i boundary points. However, the procedure we have described is exactly the same in structure as the recursive method outlined in Section 2, except that here the recursion is indexed by the resolution of the data partitioning – i.e., we have described a fine-to-coarse recursion. It is not difficult to see then that what remains is the Rauch-Tung-Striebel back-substitution step, proceeding back down the tree,in parallel at each level, until at the end we have distributed appropriately the optimal smoothed estimates based on all data of the end points of each subinterval, which is exactly the same as the result of Algorithm # 1, although in this case the time required to do this is proportional to $\log(m)$ since we have been able to use parallel rather than serial operations.

4 Conclusion

In this paper we have described new parallel algorithms for optimal smoothing for the class of two-point boundary value descriptor systems, which includes not only standard causal linear state models but also a rich class of noncausal models. Our approach is based on a partitioning of the data into subintervals with parallel local processing, followed by an interprocessor exchange step, and a subsequent parallel local processing step. The desire to simplify the problem of merging estimates and the nature of TPBVDS's led us to adopt an ML philosophy throughout our development, necessitating a generalization of recursive ML estimation procedures to allow for the possibility that neither the estimation error covariance nor its inverse may be well defined. This led us to a generalization of the Mayne-Fraser two-filter smoothing algorithm, in which the two ends of the data interval are treated symmetrically, and of the Rauch-Tung-Striebel algorithm.

As we have shown, the data interchange step of our parallel processing algorithms can itself be viewed as a smoothing problem for a TPBVDS whose "state" represents the boundaries of the subintervals used in the first, local processing stage. This led naturally to one class of algorithmic structures using the ML version of Mayne-Fraser or Rauch-Tung-Striebel that we have derived. An important point to note here that in this structure the estimates produced by the first local processing stage play the role of "measurements", with the dynamic relationships between subinterval boundaries playing the role of dynamics. Note that the interval dynamic relationships have in a sense been absorbed into the "measurements". This emphasizes not only the fact that measurements and dynamics play essentially identical roles as noisy constraints in the ML formalism but also that the key to essentially all efficient (and in our case parallel) estimation algorithms is the judicious choice of the order in which these constraints are applied and in which variables are eliminated.

This is perhaps even more apparent in our second algorithm which can serve either as the interprocessor interface step or as a stand-alone parallel smoothing algorithm. In this algorithm, the individual estimates from the first local processing stage (or the new data in the stand-alone mode) serve as *initial condition* for our dynamic model which evolves finer to coarser subinterval partitions by merging subintervals and keeping track only of the resulting exterior boundaries - i.e., the dynamics in this case are essentially nothing more than decimation! Moreover, the dynamic relationships between subinterval boundaries in this case essentially play the role of measurements! This leads us naturally to the consideration of dynamic models on dyadic trees, a topic that has also arisen quite independently, from the development of statistical filtering methods related to the wavelet transform. We refer the reader to [14] for complete expositions of this topic.

Finally, as we indicated in the introduction, the structure of Algorithm # 2 can be easily extended to multiple dimensions. For example consider a Markov random field [15] on a 2-D rectangular grid, and suppose we partition the data array into many smaller rectangles. A natural parallel processing structure in this case is a first, parallel, outward processing step within each sub-rectangle, followed by an exchange of boundary information and a subsequent parallel inward processing step. Looking more carefully at the boundary exchange step, we can imagine performing it in exactly the same way as in Algorithm # 2: merging smaller rectangles into larger ones (in parallel) and propagating information about the resulting outward boundaries. Note that t his also can be organized to have a dyadic tree structure and thus is naturally matched to the hypercube architecture. Obviously, while what we have just described is superficially identical to what we have discussed in this paper, there are substantial differences since outward and inward processing on rectangles is quite different from that on intervals and the same is obviously true about the relationship between rectangle and interval boundaries! Thus, the development of methods for 2-D smoothing that realize the structure we have described is far from trivial. An investigation of this problem is currently underway and will be reported on in [13].

References

[1] Bello, Martin G., Alan S. Willsky, Bernard C. Levy, and David A. Castanon, *Smoothing Error Dynamics and Their Use in the Solution of Smoothing and Mapping Problems*, IEEE Transactions on Information Theory, Vol IT-32, No.4, July 1986

[2] Levy, Bernard C., David A. Castanon, George C. Verghese, and Alan S. Willsky, *A Scattering Framework for Decentralized Estimation Problems*, Automatica Vol 19, No 4, pp373-384, 1983

[3] Speyer J.L., *Computation and Transmission Requirements for a Decentralized Linear-Quadratic-Gaussian Control Problem*, IEEE Transactions on Automatic Control, Vol AC-24, pp266, 1979

[4] Willsky, Alan S., Martin G. Bello, David A. Castanon, Bernard C Levy and George C Verghese, *Combining and Updating of Local Estimates and Regional Maps along*

one Dimensional Tracks, IEEE Transactions on Automatic Control, Vol AC-27, No.4, pp799-813, August 1982

[5] Hashemipour, Hamid R., Sumit Roy, and Alan J. Laub, *Decentralized Structures for Parallel Kalman Filtering*, IEEE Transactions on Automatic Control, Vol AC-33, No.1, January 1988

[6] Morf, M., J.R. Dobbins, B. Freidlander, and T. Kailath, *Square Root Algorithms for Parallel Processing in Optimal Estimation*, Automatica, Vol-15, pp. 299-306, 1979

[7] Tewfik, A.H., B.C. Levy, and A.S. Willsky, *A New Distributed Smoothing Algorithm*, MIT Laboratory for Information and Decision Systems, (LIDS-P- 1501), Aug 1988

[8] Catlin, Donald E.,*Estimation of random States in General Linear Models*, IEEE Transactions on automatic control, Vol. 36, No.2, February 1991

[9] Campbell, S.L., and C.D. Meyer, Generalized Inverses of Linear Transformations. London: Pitman, 1979

[10] Ljung Lennart, and Thomas Kailath , *A Unified Approach to Smoothing Formulas*, Automatica, Vol. 12, pp147-157, 1976

[11] Nikoukah, Ramine *A Deterministic and Stochastic Theory for Two-point Boundary Value Descriptor Systems*, MIT-Laboratory for information and Decision Systems, LIDS-TH-1820

[12] Nikoukhah, R., A.S. Willsky, and B.C. Levy, *Kalman Filtering and Riccati Equations for Descriptor Systems* Proceedings of the 29^{th} IEEE Conference on Decision and Control, Dec 1990

[13] Taylor, Darrin *Parallel Estimation on Two Dimensional Systems* Ph.D. Thesis, MIT, Aug 1991

[14] Chou, K.C., *A Stochastic Modeling Approach MultiScale Signal Processing* Ph.D Thesis, MIT, Jun 1991

[15] Levy, B.C., M.B. Adams, A.S. Willsky, *Solution and Linear Estimation of 2-D Nearest neighbor Models* Proceedings of the IEEE Vol.78, No. 4, April 1990