MULTIPLE PARTICLE FILTERING

Petar M. Djurić, Ting Lu, and Mónica F. Bugallo

Department of Electrical and Computer Engineering
Stony Brook University, Stony Brook, NY 11794, USA
e-mail: {djuric,tinglu,monica}@ece.sunysb.edu

ABSTRACT

Particle filtering is a sequential signal processing methodology that uses discrete random measures composed of particles and weights to approximate probability distributions of interest. The quality of approximation depends on many factors including the number of particles used for filtering and the way new particles are generated by the filter. The problem of good approximation becomes increasingly challenging as the dimension of the state space increases. In this paper, we address a possible solution for improved particle filtering in high dimensional cases by using a set of particle filters operating on partitioned subspaces of the complete state space. We provide simulation results that show the feasibility of the proposed approach.

Index Terms— recursive estimation, filtering, dynamic systems

1. INTRODUCTION

Particle filtering has become an important methodology for sequential signal processing. This can be attested by the amount of attention it has received in recent years not only by theoreticians, but by practitioners too. In the past few years there have been several special journal issues and books where particle filtering has been the main or one of the main topics of interest [1], [2], [3], [4], [5].

Particle filtering provides approximate solutions to finding filtering, predictive, and smoothing densities of interest. The approximation is based on discrete representation of these densities by samples from the space of unknowns and weights associated to the samples. The method is based on the Bayes' theory and is composed of three steps: (1) generation of particles (samples from the space of unknowns), (2) computation of the particle weights, and (3) resampling. The last step does not have to be implemented at every time instant but it is essential for accurate performance of the filter.

In many problems of sequential signal processing the dimension of the state space may be very large. That usually requires a very large number of particles for satisfactory performance of the particle filter. If we keep in mind that particle filtering is a computationally expensive methodology and that its computational complexity grows with the number of particles, addressing high dimensional problems with particle filtering becomes of great importance.

Here we propose a particle filtering approach for dealing with high dimensional state spaces. We assume that our interest is to find the marginal posterior densities of the subvectors of the state vector $\mathbf{x}_t$, and we denote them by $\mathbf{x}_{k,t}$, $k = 1, 2, \cdots, K$. This set of subvectors form a partition of $\mathbf{x}_t$. In other words, we assume that there is no need to obtain the joint posterior of the complete state. Therefore, we decompose the state space into separate subspaces and we run different particle filters in each of the subspaces. We will refer to this method as multiple particle filtering.

For example, if we have a problem where the state space is 15-dimensional, we could run a particle filter where the state vector has a dimension of 15. With the proposed method, we first partition the state space, say, into three state subspaces of dimension five each, and run three separate particle filters, each of them tracking the state in one of the state subspaces. The choice of the subspaces is obviously not unique and has to be made judiciously. Our method will have some important advantages over the standard particle filtering approach. The back side of the coin is that with multiple particle filtering, we give up some of the information that the 15-dimensional particle filter could provide.

The paper is organized as follows. First we formulate the problem in Section 2. In Section 3, we explain the proposed method and discuss some of its advantages and disadvantages. The simulation results that demonstrate the method's performance are presented in Section 4. The paper is concluded with final thoughts given in Section 5.

2. THE PROBLEM

Suppose that we have a model of a system described by the set of equations given by

$$\mathbf{x}_t = g_x(\mathbf{x}_{t-1}, \mathbf{u}_t) \quad (1)$$

$$\mathbf{y}_t = g_y(\mathbf{x}_t, \mathbf{w}_t) \quad (2)$$

where the time index t is discrete and $t = 1, 2, ...$, and the symbols have the following meaning:

- $\mathbf{x}_t \in \mathbb{R}^{d_x}$ is a system state at time t of dimension d_x,
- $\mathbf{u}_t \in \mathbb{R}^{d_u}$ is a state noise vector at time t of dimension d_u,
- $g_x : \mathbb{R}^{d_x} \times \mathbb{R}^{d_u} \to \mathbb{R}^{d_x}$ is a state transition function that may be nonlinear,
- $\mathbf{y}_t \in \mathbb{R}^{d_y}$ is a vector of observations collected at time t, where the observations are a function of the system state,
- $g_y : \mathbb{R}^{d_x} \times \mathbb{R}^{d_v} \to \mathbb{R}^{d_y}$ is a measurement function of the state, and it, too, may be nonlinear, and
- $\mathbf{w}_t \in \mathbb{R}^{d_w}$ is the observation noise vector at time t of dimension d_w.

Note that the functions $g_x(\cdot)$ and $g_y(\cdot)$ may vary with time. This does not change anything in our work, and therefore we assume that the forms of the functions with time remain unchanged and that these forms are *known*.

This work has been supported by the National Science Foundation under CCF-0515246 and the Office of Naval Research under Award N00014-06-1-0012.

We assume that the dimension of the state vector d_x is greater than one, i.e., $d_x > 1$. We represent the state vector by subvectors (with possibly different sizes), i.e., $\mathbf{x}^\top = [\mathbf{x}_1^\top \mathbf{x}_2^\top \cdots \mathbf{x}_K^\top]$. Therefore the state equation (1) can be written as

$$\begin{bmatrix} \mathbf{x}_{1,t} \\ \mathbf{x}_{2,t} \\ \vdots \\ \mathbf{x}_{K,t} \end{bmatrix} = \begin{bmatrix} g_1(\mathbf{x}_{1,t-1}, \mathbf{x}_{2,t-1}, \cdots, \mathbf{x}_{K,t-1}, \mathbf{u}_t) \\ g_2(\mathbf{x}_{1,t-1}, \mathbf{x}_{2,t-1}, \cdots, \mathbf{x}_{K,t-1}, \mathbf{u}_t) \\ \vdots \\ g_K(\mathbf{x}_{1,t-1}, \mathbf{x}_{2,t-1}, \cdots, \mathbf{x}_{K,t-1}, \mathbf{u}_t) \end{bmatrix} \tag{3}$$

where the functions $g_k(\cdot)$ are not necessarily all identical.

Our interest is to find the marginal posterior densities of the state vectors $\mathbf{x}_{k,t}$, $k = 1, 2, \cdots, K$. This can be done, for example, by tracking the complete state vector and then marginalizing the complete joint posterior accordingly.

When we apply particle filtering for the stated task, we basically form a random measure, which at time instant $t-1$ is given by

$$\chi_{t-1} = \left\{ \mathbf{x}_{t-1}^{(m)}, w_{t-1}^{(m)} \right\}_{m=1}^{M} \tag{4}$$

with particles $\mathbf{x}_{t-1}^{(m)}$ and their associated weights $w_{t-1}^{(m)}$, and update it when the new measurement $\mathbf{y}_t$ becomes available. This is done by first proposing particles for $\mathbf{x}_t$ by drawing them from a proposal function $\pi(\mathbf{x}_t)$ and then computing their weights $w_t^{(m)}$ by

$$w_t^{(m)} \propto w_{t-1}^{(m)} \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(m)}) p(\mathbf{x}_t^{(m)})}{\pi(\mathbf{x}_t^{(m)})}. \tag{5}$$

If the dimension of the state-space, d_x, is large, for accurate sequential estimation of the evolving state this would most likely require a very large number of particles. *The problem is to find alternative schemes that would alleviate the explosion of necessary number of particles with the increase of state dimension.*

3. MULTIPLE PARTICLE FILTERS

One approach to avoid the need for too many particles for accurate state estimation is to partition the state-space into subspaces and in each subspace run a separate particle filter. Here we describe in detail one way of accomplishing this and discuss some alternative.

3.1. Proposed method

Following the idea of using multiple particle filters, we assign to each state vector $\mathbf{x}_{k,t}$ a particle filter. Each subspace of the state-space has a random measure $\chi_{k,t} = \left\{ \mathbf{x}_{k,t}^{(m)}, w_{k,t}^{(m)} \right\}_{m=1}^{M_k}$, where M_k is the number of particles used by the $k-$th particle filter. Clearly, the particle propagation and resampling step of the filter can be implemented in the usual way and the main question for the implementation is the update of the particle weights of the $k-$th measure from $w_{k,t-1}^{(m)}$ to $w_{k,t}^{(m)}$. The reason why this is a nontrivial question can be seen from (5), where an important factor in the update is the likelihood function $p(\mathbf{y}_t|\mathbf{x}_t^{(m)})$. The update of $w_{k,t-1}^{(m)}$ should be theoretically carried out either by

$$w_{k,t}^{(m)} \propto w_{k,t-1}^{(m)} \frac{p(\mathbf{y}_t|\mathbf{x}_{k,t}^{(m)}, \mathbf{x}_{-k,t}) p(\mathbf{x}_{k,t}^{(m)}|\mathbf{x}_{k,t-1}^{(m)}, \mathbf{x}_{-k,t-1})}{\pi_k(\mathbf{x}_{k,t}^{(m)}|\mathbf{x}_{k,t-1}^{(m)}, \mathbf{x}_{-k,t-1}, \mathbf{y}_t)} \tag{6}$$

where $\mathbf{x}_{-k,t}$ is the state vector that does not contain the elements of $\mathbf{x}_{k,t}$ and $\pi_k(\cdot)$ is the proposal function of the $k-$th filter, or we could use

$$w_{k,t}^{(m)} \propto w_{k,t-1}^{(m)} \frac{p(\mathbf{y}_t|\mathbf{x}_{k,t}^{(m)}) p(\mathbf{x}_{k,t}^{(m)}|\mathbf{x}_{k,t-1}^{(m)})}{\pi_k(\mathbf{x}_{k,t}^{(m)}|\mathbf{x}_{k,t-1}^{(m)}, \mathbf{y}_t)}. \tag{7}$$

The former update requires knowledge of $\mathbf{x}_{-k,t}$ which is not available, and the latter, the ability to marginalize, which is technically intractable. Here we propose that the update is implemented by

$$w_{k,t}^{(m)} \propto w_{k,t-1}^{(m)} \frac{p(\mathbf{y}_t|\mathbf{x}_{k,t}^{(m)}, \tilde{\mathbf{x}}_{-k,t}) p(\mathbf{x}_{k,t}^{(m)}|\mathbf{x}_{k,t-1}^{(m)}, \hat{\mathbf{x}}_{-k,t-1})}{\pi_k(\mathbf{x}_{k,t}^{(m)}|\mathbf{x}_{k,t-1}^{(m)}, \hat{\mathbf{x}}_{-k,t-1}, \mathbf{y}_t)} \tag{8}$$

where $\hat{\mathbf{x}}_{-k,t-1}$ is the estimated value of all the states at time t except of $\mathbf{x}_{k,t}$, i.e.,

$$\hat{\mathbf{x}}_{-k,t}^\top = [\hat{\mathbf{x}}_{1,t}^\top \hat{\mathbf{x}}_{2,t}^\top \cdots \hat{\mathbf{x}}_{k-1,t}^\top \hat{\mathbf{x}}_{k+1,t}^\top \cdots \hat{\mathbf{x}}_{K,t}^\top]$$

and similarly, $\tilde{\mathbf{x}}_{-k,t}$ is the predicted value of all the states except of $\mathbf{x}_{k,t}$. The predictions can be obtained in one of many ways. For example, we can use the prediction defined by

$$\tilde{\mathbf{x}}_{j,t} = \sum_{m=1}^{M_j} w_{j,t-1}^{(m)} \mathbf{x}_{j,t}^{(m)} \tag{9}$$

for $j = 1, 2, \cdots, K$. This implies that at every time instant all the particle filters obtain the predictions of their states and provide this information to the remaining particle filters. The particle filters use the exchanged information for computing the weights of their particles and eventually for generation of new particles.

3.2. Alternatives

Alternative methods would be based similarly on exchanging predictions of the states obtained with different methods or by using Rao-Blackwellization (RB). Recall that RB is a variance reduction technique [6], where states that are considered nuisance states are integrated out analytically. As a result, the dimensionality of the space of interest is decreased and the applied estimation methods become more accurate.

The idea of RB has already been explored in the context of particle filtering, for example in [7] and [8]. Suppose that the state-space is partitioned into two subspaces where the state vector from one of the subspaces follows a conditionally linear Gaussian model given the other state vector. Then, one can use Kalman filtering for tracking the conditionally linear state vector and particle filtering for tracking the conditioning state vector.

In general, however, our system may not contain conditionally linear states. What can one do in that case? One option is to linearize the system over the space that we want to marginalize. This would then lead to a combination of extended Kalman filtering (EKF) and RB. The method could be applied as follows: for the $k-$th subspace we linearize the system so that $\mathbf{x}_{1,t}$, $\mathbf{x}_{2,t}$, $\cdots$, $\mathbf{x}_{k-1,t}$, $\mathbf{x}_{k+1,t}$, $\cdots$, $\mathbf{x}_{K,t}$ are conditionally linear given $\mathbf{x}_{k,t}$, and they are estimated within that subspace using EKF. Each particle filter will then have its own set of linearized states which will be marginalized by using Kalman filtering.

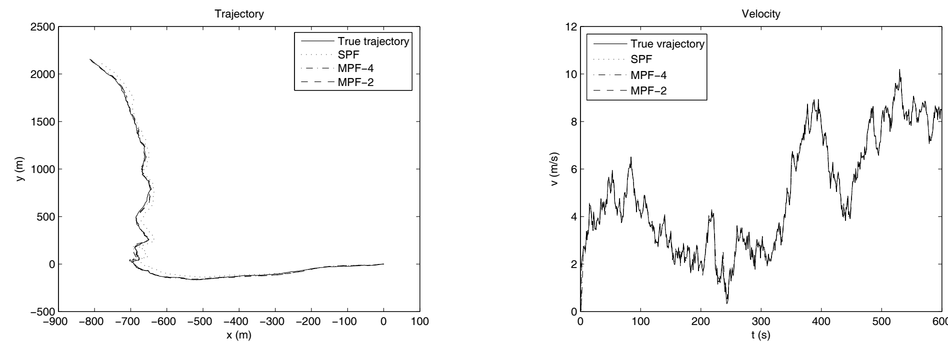

Fig. 1. A target trajectory and its estimates by various PFs (SPF with $M = 800$, MPF-2 with $M = 400$, and MPF-4 with $M = 200$ particles).

3.3. Discussion

In terms of implementation, the proposed method is expected to require less particles to achieve the same accuracy like particle filters operating on the complete state space. When comparisons are made among the particle filters one can argue that it is important that the computation time of the filters is the same. If one uses one CPU for each particle filter, the number of particles of the multiple particle filter and standard particle filter should be the same per particle filter. Clearly, in that case the total number of particles of the multiple particle filter will be K times bigger than that of the standard particle filter, but the multiple particle filter will require better computational resources. On the other hand, if the particle filtering is executed on one sequential machine, the number of particles of each of the particle filters of the multiple particle filtering scheme will be K times less than that of the standard particle filter.

An important issue in the implementation of the multiple particle filter is the partitioning of the state space. The performance of the filter may strongly depend on that.

One advantage of the multiple particle filter is that it may allow for different number of particles for the various subspaces. This feature should be used in particular when the dimensions of the subspaces are different.

The multiple particle filter will most likely have degraded performance in cases when the posteriors of the marginalized states are multimodal. When this arises, the multiple particle filter should not be used.

4. SIMULATION RESULTS

A standard application of particle filtering is target tracking. In our simulations, we considered the problem of tracking a vehicle moving along a two-dimensional space. The tracking was modeled by the following state-space system:

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_t \qquad \text{state equation} \qquad (10)$$

$$\mathbf{y}_t = g_y(\mathbf{x}_t) + \mathbf{w}_t \qquad \text{observation equation} \qquad (11)$$

where $\mathbf{x}_t$ is the state vector in the two-dimensional plane, $\mathbf{A}$ and $\mathbf{B}$ are state-transition and noise-transition matrices, respectively, $\mathbf{y}_t$ is a vector of observations, and $\mathbf{u}_t$ and $\mathbf{w}_t$ are the state and observation noise processes. The state $\mathbf{x}_t$ consisted of the target location at time instant t, $\mathbf{l}_t = [l_{x,t} \; l_{y,t}]^\top$ and its velocity $\mathbf{v}_t = [v_{x,t} \; v_{y,t}]^\top$. The units of position and velocity were meter and meter/second, respectively.

In the simulations, we used four static sensors, all located at the same spot, $\mathbf{r} = [-10,000 \;\; 20,000]^\top$. The first sensor was measuring the signal strength, the second, the angle of signal arrival, the third, absolute velocity of the target, and the fourth, the direction of motion. The functions describing the sensor measurements were as follows:

$$g_{y,1}(\mathbf{x}_t) = 10\log_{10}\left(\frac{P_0}{\|\mathbf{r} - \mathbf{l}_t\|^\alpha}\right) \qquad (12)$$

$$g_{y,2}(\mathbf{x}_t) = \angle(\mathbf{l}_t, \mathbf{r}) \qquad (13)$$

$$g_{y,3}(\mathbf{x}_t) = \|\mathbf{v}_t\| \qquad (14)$$

$$g_{y,4}(\mathbf{x}_t) = \angle(\mathbf{v}_t) \qquad (15)$$

where $\|\cdot\|$ denotes norm of vector, $\angle(\cdot)$ is the relative angle of the vector, P_0 is the power of the signal measured at unit distance, and α is the attenuation coefficient of the fading channel.

We tested the tracking of a target with constant velocity. The transition matrix was given by

$$\mathbf{A}_{4\times4} = \begin{bmatrix} \mathbf{I}_2 & T_s\mathbf{I}_2 \\ \mathbf{0}_2 & \mathbf{I}_2 \end{bmatrix} \qquad (16)$$

where T_s was sampling period, $\mathbf{I}_2$ and $\mathbf{0}_2$ were the identity and zero matrices, respectively, and $\mathbf{B} = \mathbf{I}_4$. The state noise process $\mathbf{u}_t$ was modeled as independent Gaussian, with different variances for position and velocity, i.e.,

$$u_{1,t}, u_{2,t} \sim \mathcal{N}(0,1), \quad u_{3,t}, u_{4,t} \sim \mathcal{N}(0,0.1).$$

The observation noise process $\mathbf{w}_t$ was also Gaussian and had different variances for each component,

$$w_{1,t} \sim \mathcal{N}(0,10^{-4}), \quad w_{2,t} \sim \mathcal{N}(0,10^{-4})$$

$$w_{3,t} \sim \mathcal{N}(0,10^{-5}), \quad w_{4,t} \sim \mathcal{N}(0,10^{-4}).$$

We tracked the target for 10 minutes where the measurements were sampled with $T_s = 1$ second. The filters were applied with different number of particles as will be explained further below. In implementing the multiple particle filter (MPF) we partitioned the state space according to $\mathbf{x}_t = [x_{1,t} \;\; x_{2,t}]^\top$ where $\mathbf{x}_{1,t} = [l_{x,t} \; l_{y,t}]^\top$ and $\mathbf{x}_{2,t} = [\; v_{x,t} \;\; v_{y,t}]^\top$ and we refer to this particle filter as MPF-2. So MPF-2 was composed of two particle filters running in parallel. Then we also partitioned the state space as $\mathbf{x}_t = [x_{1,t} \; x_{2,t} \; x_{3,t} \; x_{4,t}]^\top = [l_{x,t} \; l_{y,t} \; v_{x,t} \; v_{y,t}]^\top$, and we refer to this particle filter as MPF-4. Thus, in this case we ran simultaneously four different particle filters.

We computed the mean-square error (MSE) of the target's location and velocity during the time of tracking. The MSE was

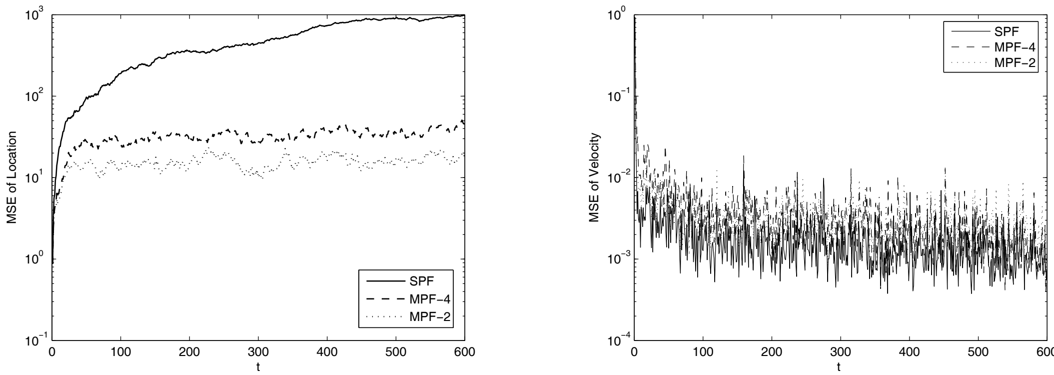

Fig. 2. MSE's of location and velocity by various PFs (SPF with $M = 800$, MPF-2 with $M = 400$, and MPF-4 with $M = 200$ particles).

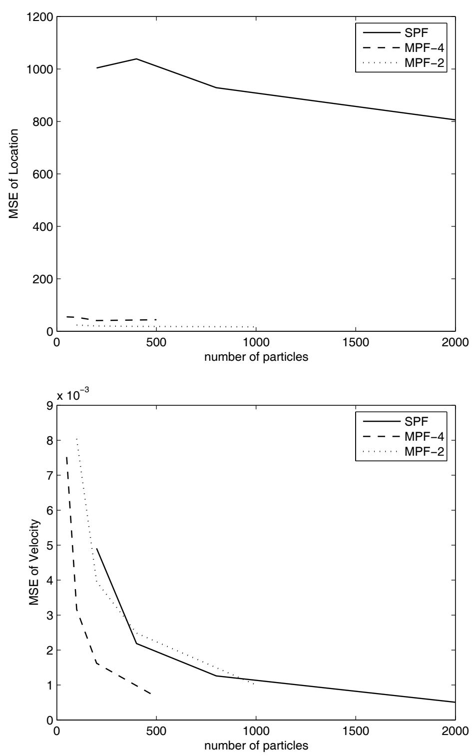

Fig. 3. MSE's of location and velocity as functions of number of used particles for the SPF, MPF-2 and MPF-4.

computed from 50 independent realizations. In all the realizations, the target started moving in the proximity of the origin of the coordinate system. For comparison purposes, we also implemented the standard particle filter (SPF).

In Fig. 1 on the left, we can see one realization of a trajectory and its estimates obtained by SPF with $M = 800$, MPF-2 with $M = 400$, and MPF-4 with $M = 200$ particles and in Fig. 1 on the right, the corresponding track of the velocity of the target and its estimates. Clearly, all the methods were able to estimate the trajectory and velocity of the target with good accuracy.

In Figure 2, we can see the MSEs of the location (left) and velocity (right) of the target obtained by the different methods. We observe that the trajectory was best estimated by MPF-2 and the worst by SPF. The ranking in performance according to the MSEs of velocity is less clear. However, it is obvious that the performance of all the methods was approximately the same.

Finally in Figure 3 we present the average MSEs per sample for estimated location (top) and velocity (bottom), respectively, as functions of used number of particles. The MSEs were computed for the last 60 seconds of tracking. We see from the figures that MPF outperformed the SPF.

5. CONCLUSIONS

An important challenge in particle filtering is the need for very large number of particles when the dimensions of the states are even moderately large. One approach to combat this problem is to partition the state space into subspaces and run separate particle filters for these subspaces. In the paper we presented one possible implementation of this idea. The computer simulations showed that the proposed method outperformed the standard particle filter.

6. REFERENCES

[1] A. Doucet, N. de Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Springer, New York, 2001.

[2] P. M. Djurić and S. J. Godsill, Eds., *Special Issue on Monte Carlo Methods for Statistical Signal Processing*, vol. 2004, IEEE Transactions on Signal Processing, 2002.

[3] P. M. Djurić, S. Godsill, and A. Doucet, Eds., *Special Issue on Particle Filtering in Signal Processing*, EURASIP Journal of Applied Signal Processing, 2004.

[4] J. S. Liu, *Monte Carlo Strategies in Scientific Computing*, Springer, New York, 2001.

[5] B. Ristić, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter*, Artech House, Boston, MA, 2004.

[6] G. Casella and C. P. Robert, "Rao-Blackwellization of sampling schemes," *Biometrika*, vol. 84, pp. 81–94, 1996.

[7] R. Chen and J. S. Liu, "Mixture Kalman filters," *Journal of the Royal Statistical Society*, vol. 62, no. Part 3, pp. 493–508, 2000.

[8] T. Schon, F. Gustafsson, and P. Nordlund, "Marginalized particle filters for mixed linear/nonlinear state-space models," *IEEE Transactions on Signal Processing*, vol. 50, no. 7, pp. 2279–2289, 2005.