Determining the Initial States in Forward-Backward Filtering

Fredrik Gustafsson

Abstract— **Forward-backward filtering is a common tool in off-line filtering for implementing non-causal filters. Filtering first forward and then backward or the other way around does not give the same result generally. Here we propose a method to choose the initial state to obtain uniqueness and to remove transients at both ends.**

I. Introduction

In batch wise signal processing, a non-causal IIR filter can be implemented by applying one causal filter to the signal forward in time as usual, and a second anti-causal filter backward on the filtered signal. We will refer to this procedure as *forward-backward filtering*. For example, forward-backward filtering is needed in the following applications:

- Zero-phase filtering using IIR filters.
- Implementation of non-causal Wiener filters or other filters with poles both inside and outside the unit circle.

Forward-backward filtering for implementing zero-phase filters is used in various software packages, see for instance the function `filtfilt` in Matlab's Signal Processing Toolbox [2]. However, this issue is not much discussed in standard text books. The design of non-causal Wiener filters leads to a filter with one stable and one unstable factor as discussed in for instance [1]. The unstable factor has to be implemented using backward filtering. In both cases, there might be a problem with transients at the beginning and the end of the signal.

A filter $G(q)$ becomes all-pass, or zero phase, when applied first forward, $y_f(t) = G(q)u(t)$, and then backward on the reversed filtered signal and finally the output is reversed again, $y_{fb}^R(t) = G(q)y_f^R(t)$. The superindex R stands for reversed sequences, the indices f for forward and b for backward and q denotes the shift operator $qu(t) = u(t+1)$. We can also first filter backward and then forward and obtain $y_{bf}(t)$. In both cases, the total effect is a zero phase filter with transfer function $|G(e^{i\omega})|^2$. However, we have $y_{fb}(t) \neq y_{bf}(t)$ generally due to unknown initial conditions, and this is not satisfactory from the logical point of view. Besides, the FB (forward-backward) filtered sequence has visible transients at the end and the BF filtered sequence at the beginning.

One way to obtain symmetry is to implement a filter with poles both inside and outside the unit circle by using the partial fractions of the stable poles and unstable poles respectively. This is the standard approach in non-causal Wiener filtering. The stable filter is applied forward on the input and the unstable filter backward and the filter output is the sum of these terms. Since both filters are applied to the input, the result is of course independent of the order of computations. However, there is still a problem with transients, this time at both ends, due to unknown initial conditions.

We will here determine the initial conditions that make $y_{bf}(t) = y_{fb}(t)$ and at the same time removes the transients. That is, the initial condition of one filter is chosen to match the end of the other filter. Instead of solving the system of equations arising from the condition $y_{bf}(t) = y_{fb}(t)$, we will work in a least squares framework and minimize $\|y_{bf}(t) - y_{fb}(t)\|_2^2$. The advantage is that we immediately get a feasible implementation. The method is illustrated on filtered white noise and an EKG signal. The improvement for short data records or long filter impulse responses is obvious.

Manuscript received May 31, 1994; revised September 8, 1995. The associate editor coordinating the review of this paper and approving it for publication was Prof. Demetris Lainiotis. The author is with the Department of Electrical Engineering, Linköping University, S-58183 Linköping, Sweden. This is an electronic version maintained by the author.

II. Determining the initial state

A linear filter can always be expressed as

$$Y = \mathcal{H}U + \mathcal{O}x_0,$$

where $Y = (y_0, y_1, .., y_{N-1})^T$ is the vector of outputs, $U = (u_0, u_1, .., u_{N-1})^T$ is the vector of inputs to the filter and x_0 its initial state. $\mathcal{H}$ is a Toeplitz matrix of impulse response coefficients

$$\mathcal{H} = \begin{pmatrix} h_0 & 0 & \cdots & 0 \\ h_1 & h_0 & 0 & \vdots \\ \vdots & & \ddots & \\ h_{N-1} & \cdots & h_1 & h_0 \end{pmatrix} \tag{1}$$

where N is the number of inputs, and $\mathcal{O}$ is an "observability" matrix. If the filter is of order n, so x_0 has dimension $n \times 1$, the filter can be written in state space form

$$\begin{aligned} x(t+1) &= Fx(t) + Gu(t) \\ y(t) &= Hx(t) + Du(t) \end{aligned}$$

where the state x is an $n \times 1$ vector and F, G, H, D are matrices of appropriate dimensions. Then we can write

$$\mathcal{H} = \begin{pmatrix} D & 0 & \cdots & 0 \\ HG & D & 0 & \vdots \\ \vdots & & \ddots & \\ HF^{N-2}G & \cdots & HG & D \end{pmatrix} \tag{2}$$

$$\mathcal{O} = \begin{pmatrix} H \\ HF \\ \vdots \\ HF^{N-1} \end{pmatrix} \tag{3}$$

We introduce the row and column reversing operators R and C with the following easily checked properties:

$$\begin{aligned} A &= BC \Rightarrow \\ A^R &= B^R C \\ A^C &= BC^C \\ BC^R &= B^C C \\ (BC^{RC})^{RC} &= B^{RC} C \\ A^{RC} &= A^T \text{ if } A \text{ Toeplitz} \end{aligned}$$

A simple proof is obtained by multiplying $A = BC$ by the permutation matrix I^R, where $I^R I^R = I$, on different places.

Generally, the forward and backward filters are different. The forward and backward filters are characterized by the pairs $(\mathcal{H}_f, \mathcal{O}_f)$ and $(\mathcal{H}_b, \mathcal{O}_b)$, respectively. Using the notation x_0 and x_{N-1} for the initial conditions for the forward and backward filters, the result of forward-backward and backward-forward filtering can be written as

$$\begin{aligned} Y_{fb} &= (\mathcal{H}_b Y_f^R + \mathcal{O}_b x_{N-1})^R \\ &= \mathcal{H}_b^R (\mathcal{H}_f U + \mathcal{O}_f x_0)^R + \mathcal{O}_b^R x_{N-1} \\ &= \mathcal{H}_b^R \mathcal{H}_f^R U + \mathcal{H}_b^R \mathcal{O}_f^R x_0 + \mathcal{O}_b^R x_{N-1} \\ &= \mathcal{H}_b^T \mathcal{H}_f U + \mathcal{H}_b^T \mathcal{O}_f x_0 + \mathcal{O}_b^R x_{N-1} \end{aligned}$$

and

$$\begin{aligned} Y_{bf} &= \mathcal{H}_f Y_b^R + \mathcal{O}_f x_0 \\ &= \mathcal{H}_f (\mathcal{H}_b U^R + \mathcal{O}_b x_{N-1})^R + \mathcal{O}_f x_0 \\ &= \mathcal{H}_f^C \mathcal{H}_b^C U + \mathcal{H}_f \mathcal{O}_b^R x_{N-1} + \mathcal{O}_f x_0 \\ &= \mathcal{H}_f \mathcal{H}_b^T U + \mathcal{H}_f \mathcal{O}_b^R x_{N-1} + \mathcal{O}_f x_0 \end{aligned}$$

respectively. Note that $\mathcal{H}_b^T \mathcal{H}_f \neq \mathcal{H}_f \mathcal{H}_b^T$. That is, the use of zero initial conditions does not solve our problem.

It turns out that there is always a solution to $Y_{fb} - Y_{bf} = 0$. We will not prove this statement. Instead we formulate the problem as minimizing $\|Y_{fb} - Y_{bf}\|_2^2$, which turns out to be easier to solve. We have

$$\begin{aligned} Y_{fb} - Y_{bf} &= (\mathcal{H}_b^T \mathcal{H}_f - \mathcal{H}_f \mathcal{H}_b^T) U + (\mathcal{H}_b^T - I) \mathcal{O}_f x_0 \\ &+ (I - \mathcal{H}_f) \mathcal{O}_b^R x_{N-1} \end{aligned}$$

Since this is a linear expression in x_0 and x_{N-1}, the minimizing arguments are given by the least squares estimate

$$\begin{pmatrix} x_0^{\text{opt}} \\ x_{N-1}^{\text{opt}} \end{pmatrix} = \left[(\mathcal{H}_b^T - I)\mathcal{O}_f \ , \ (I - \mathcal{H}_f)\mathcal{O}_b^R \right]^\dagger \left(\mathcal{H}_b^T \mathcal{H}_f - \mathcal{H}_f \mathcal{H}_b^T \right) U \quad (4)$$

Here $\dagger$ denotes pseudo-inverse. By substituting these estimated initial states into the FB and BF filters we get

$$\begin{aligned} Y_{fb}^{\text{opt}} &= \mathcal{H}_b^T \mathcal{H}_f U + \left[\mathcal{H}_b^R \mathcal{O}_f^R \ , \ \mathcal{O}_b^R \right] \\ &\quad \left[(\mathcal{H}_b^T - I)\mathcal{O}_f \ , \ (I - \mathcal{H}_f)\mathcal{O}_b^R \right]^\dagger \left(\mathcal{H}_b^T \mathcal{H}_f - \mathcal{H}_f \mathcal{H}_b^T \right) U \\ Y_{bf}^{\text{opt}} &= \mathcal{H}_f \mathcal{H}_b^T U + \left[\mathcal{O}_f \ , \ \mathcal{H}_f \mathcal{O}_b^R \right] \\ &\quad \left[(\mathcal{H}_b^T - I)\mathcal{O}_f \ , \ (I - \mathcal{H}_f)\mathcal{O}_b^R \right]^\dagger \left(\mathcal{H}_b^T \mathcal{H}_f - \mathcal{H}_f \mathcal{H}_b^T \right) U \end{aligned}$$

The expressions given above are not useful for implementation, since the $N \times N$ matrices $\mathcal{H}_f$ and $\mathcal{H}_b$ have to be formed. In the next section, we describe how (4) can be solved without using $\mathcal{H}$ explicitly.

III. Fast implementation

In this section we describe how Y_{fb} can be computed efficiently, which, without proof, is claimed to be the same as Y_{bf}. First an exact method is detailed, where the initial states are computed with complexity linear in data length. In the next section, a constant complexity approximation is proposed.

We first compute the nominal values of Y_{fb} and Y_{bf} for zero initial conditions using standard filter routines. That is, we have

$$\begin{aligned} Y_{fb}^0 &= \mathcal{H}_b^T \mathcal{H}_f U \\ Y_{bf}^0 &= \mathcal{H}_f \mathcal{H}_b^T U \end{aligned}$$

Note that the filters are not implemented using Hankel matrices $\mathcal{H}$. We also need arbitrary state space realizations of the filters, (F_f, G_f, H_f, D_f) and (F_b, G_b, H_b, D_b). The "observability" matrices $\mathcal{O}_f$ and $\mathcal{O}_b$ are computed according to (3). Now, we just have to compute the quantities

$$\mathcal{S}_{bf} \triangleq \mathcal{H}_b \mathcal{O}_f^R \quad (5)$$

$$\mathcal{S}_{fb} \triangleq \mathcal{H}_f \mathcal{O}_b^R \quad (6)$$

and we can express the least squares estimate as

$$\begin{pmatrix} x_0^{\text{opt}} \\ x_{N-1}^{\text{opt}} \end{pmatrix} = \left[(\mathcal{S}_{fb}^R - \mathcal{O}_f) \ \ \mathcal{O}_b^R - \mathcal{S}_{bf} \right]^\dagger (Y_{bf}^0 - Y_{fb}^0).$$

Thus, we have

$$Y_{fb}^{\text{opt}} = Y_{fb}^0 + \left[\mathcal{S}_{bf}^R \ , \ \mathcal{O}_b^R \right] \left[(\mathcal{S}_{fb}^R - \mathcal{O}_f) \ \ \mathcal{O}_b^R - \mathcal{S}_{bf} \right]^\dagger (Y_{bf}^0 - Y_{fb}^0). \quad (7)$$

Next is shown that $\mathcal{S}_{fb}$ can be computed without forming the $N \times N$ matrix $\mathcal{H}_f$. The vector of impulse response coefficients can be expressed in h_f as

$$h_f = \begin{bmatrix} D_f \\ \mathcal{O}_f(1:N-1,:) G_f \end{bmatrix}.$$

Here $\mathcal{O}_f(1:N-1,:)$ means rows 1 to $N-1$ of $\mathcal{O}_f$. Note that h_f is the first column of $\mathcal{H}_f$. Let $\mathcal{S}_{fb}(t,:)$ denote the t'th row of $\mathcal{S}_{fb}$ and $h_f(t)$ the t'th component of h_f. From (6), (2) and (3) we have

$$\begin{aligned} \mathcal{S}_{fb}(N,:) &= (h_f^R)^T \mathcal{O}_b^R = (h_f^T)^C \mathcal{O}_b^R = h_f^T \mathcal{O}_b \\ \mathcal{S}_{fb}(N-1,:) &= \mathcal{S}_{fb}(N,:) F_b - h_f(N) H_b F_b^{N-1} \\ \mathcal{S}_{fb}(t-1,:) &= \mathcal{S}_{fb}(t,:) F_b - h_f(t) H_b F_b^{N-1} \end{aligned}$$

Thus, $\mathcal{S}_{fb}$ can be computed by the recursion

$$\begin{aligned} \mathcal{S}_{fb}(N,:) &= h_f^T \mathcal{O}_b \\ \mathcal{S}_{fb}(t-1,:) &= \mathcal{S}_{fb}(t,:) F_b - h_f(t) H_b F_b^{N-1}, \ t = N, N-1, ..., 2. \end{aligned}$$

and $\mathcal{S}_{bf}$ is computed similarly. In this way, we do not have to form the $N \times N$ matrix $\mathcal{H}$ explicitly, but only the $N \times n$ matrices $\mathcal{O}$ and $\mathcal{S}$. The major computational step is a pseudo-inversion of an $N \times 2n$ matrix.

IV. Fixed complexity approximation

For long signals, it seems to be overkill to work with the complete impulse response vector h and observability matrix $\mathcal{O}$. If we assume that the impulse response is approximately zero for $t > M$ a logical approximation in (7) is to replace F^t by 0 if $t > M$. This leads to an $\hat{\mathcal{O}}$ having non-zero elements in its first M rows only, and $\hat{\mathcal{S}}$ having non-zero elements in its last M rows. Thus, we are faced with a least squares problem with $2M$ equations and $2n$ unknowns, so all that is needed is to compute the pseudo-inverse of a $2M \times 2n$ matrix in (7). That is, the computational burden depends on the rate of decay of the impulse response, which in turn depends on the proximity of the filters' poles to the unit circle, and not on the data length.

V. Simulations

In this section, a band-pass Butterworth filter applied to a white noise sequence[1] will be examined. The filter is of tenth order with (normalized) cut-off frequencies 0.2 and 0.25 and the noise sequence is of length 200. Figure 1 shows first Y_{fb}^{filtfilt} and Y_{bf}^{filtfilt} using the initial states in Matlab's `filtfilt`[2] function in the signal processing toolbox and second the result using zero initial states x_0 and x_{N-1}, *i.e.* $Y_{fb}^0 = \mathcal{H}^T \mathcal{H} U$ and $Y_{bf}^0 = \mathcal{H} \mathcal{H}^T U$. Then follows the signals Y_{fb}^{opt} and Y_{bf}^{opt} from (7) using optimized initial states. Finally, the result Y_{fb}^{pf} and Y_{bf}^{pf} using partial fractions,

$$|H(q)|^2 = \frac{B(q)}{A(q)} \frac{B(q^{-1})}{A(q^{-1})} = \frac{C(q)}{A(q)} + \frac{D(q)}{A(q^{-1})},$$

[1] The result can be reproduced using Matlab's Gaussian distribution with seed 0

[2] Version 4.1 and earlier of Matlab has a bug. With the following definition of the initial state `zi`, the function works as intended: `zi = (eye(nfilt-1) + [aa(2:nfilt)' [-eye(nfilt-2); zeros(1,nfilt-2)]])\(bb(2:nfilt)-bb(1)*aa(2:nfilt))';`

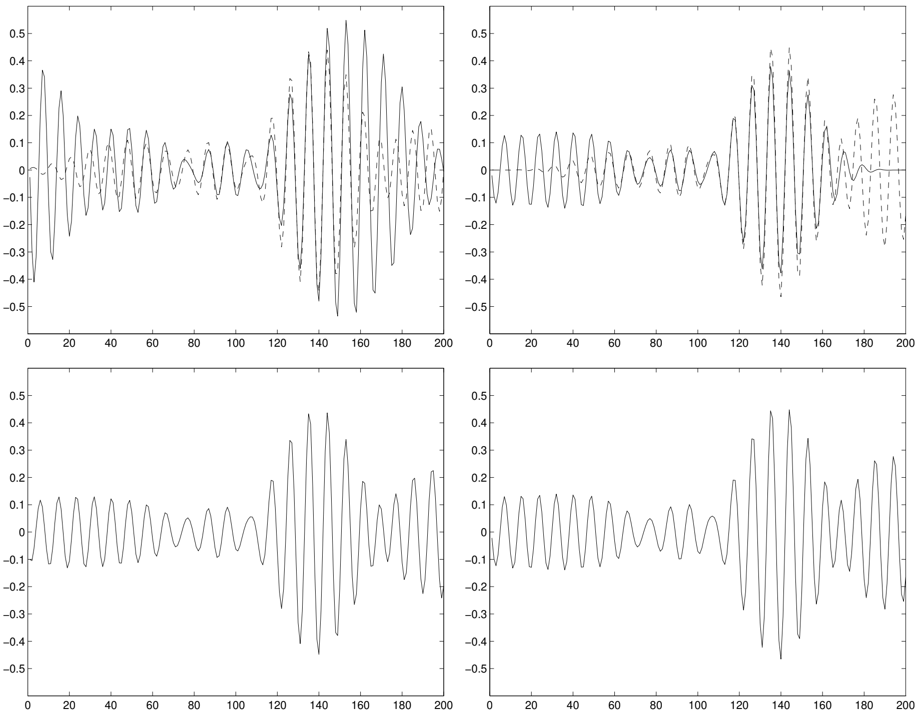

Fig. 1. The forward-backward (solid) and backward-forward (dashed) filtered signal using first Matlab's `filtfilt` function (upper left), then zero initial condition (upper right), optimized initial conditions (lower left) and finally using partial fractions (lower right).

is shown. As seen, the last two curves are identical showing the symmetry of the proposed initialization method. The 2-norms are as follows:

$$\begin{aligned} \|Y_{fb}^{\text{filtfilt}} - Y_{bf}^{\text{filtfilt}}\|_2 &= 2.2 \\ \|Y_{fb}^{0} - Y_{bf}^{0}\|_2 &= 1.2 \\ \|Y_{fb}^{\text{opt}} - Y_{bf}^{\text{opt}}\|_2 &= 6 \cdot 10^{-10} \end{aligned}$$

Thus, there is a significant difference depending on the order of forward and backward filtering for the first two choices of initial conditions. The transient from the filter is clearly visible either at the beginning or the end. The proposed method and the partial fraction method are perfectly symmetric. Note, however, that the results are not identical for these two methods.

To highlight the transients due to unknown initial conditions in the partial fraction method, a very simple zero-phase high-pass filter is used,

$$y(t) = \left|\frac{1}{q+0.8}\right|^2 u(t) = \frac{125/36}{q+1.25} - \frac{20/9}{q+0.8}$$

and a linear trend $15 + 15t/200$ is added to the signal $u(t)$ used in the previous example. The result is shown in Figure 2. The transients for the partial fraction method are obvious. Of course, some *ad-hoc* rules can be used to determine initial conditions to compensate for an offset or a trend as in this example, one example being Matlab's `filtfilt`. The point is, that the proposed method computes the initial conditions for each filter to match the end of the other filter response when the transient hopefully has disappeared.

VI. Filtering an EKG signal

The first plot in Figure 3 shows an EKG signal with a 50 Hz disturbance. The objective with forward-backward filtering here is to get rid of the disturbance while not perturbing the shape of the true signal. A fourth order Butterworth stop-band filter with cutoff frequencies 45 and 55 Hz will be used. Matlab's `filtfilt` function is used to demonstrate an *ad-hoc* choice of initial conditions, and the result is shown in the second plot. There is a significant transient at the beginning. Interestingly, it turns out that the result improves somewhat with zero initial conditions. The third plot is obtained with the partial fraction method. Here the transient is less, but on the other hand it appears at both ends. The new method gives a superior result. The approximation described in Section IV gives an almost identical result when using only 10 samples at each end. Even using only three samples gives a fair result. For this particular signal with a known 50 Hz disturbance, we can compute the true signal with model based methods. The results are summarized below:

Method	$\|Y_{bf} - Y_{fb}\|$	$\|Y_{bf} - Y_{true}\|$
`filtfilt`	2.34	2.66
Zero condition	1.75	2.68
Partial fractions	0	3.03
Optimal	$6 \cdot 10^{-13}$	2.34
Suboptimal, $M = 20$	0.30	2.35
Suboptimal, $M = 10$	0.41	2.36
Suboptimal, $M = 3$	1.18	2.56

VII. Conclusion

We have here detailed a method to choose the initial conditions for filters to be applied forward and backward to a signal. The objectives were to get the same result independently of the order of filtering and to minimize transients due to offsets and trends in the data. Zero initial conditions or other *ad-hoc* initializations can by examples easily be shown to give undesired transients at the beginning or end of the data sequence depending on the order of application of the forward and backward filters. Transients are also a problem when using partial fractions of the stable and unstable parts. The proposed method

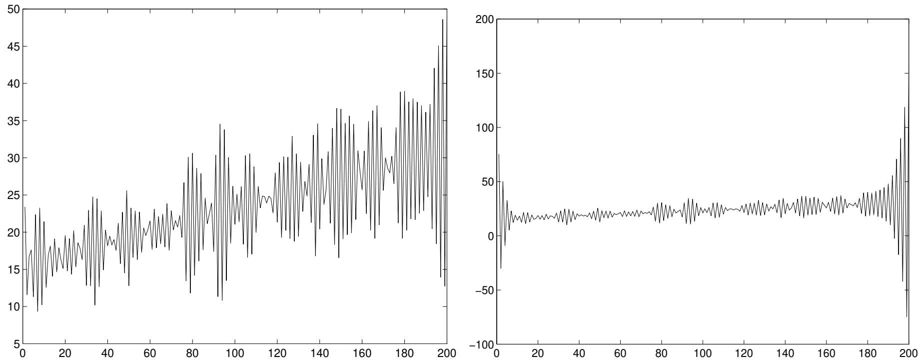

Fig. 2. The filtered signal using optimized initial conditions and partial fractions.

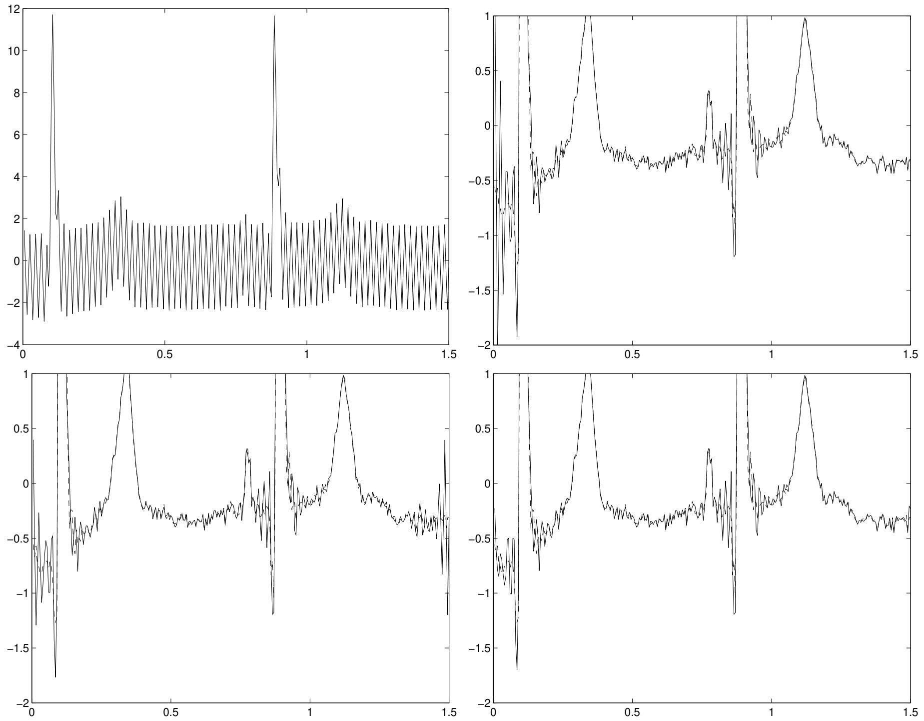

Fig. 3. An EKG signal with a 50 Hz disturbance (upper left). The signal is filtered through a zero-phase Butterworth filter using Matlab's `filtfilt` (upper right), partial fractions (lower left) and finally using the new method (lower right). Dashed curve shows the true signal in the last three plots.

gives no transients and is symmetric, which was illustrated by simulations and on a real EKG signal.

References

[1] T. Kailath. *Lectures on Wiener and Kalman filtering.* Springer-Verlag, Wien, 1981.

[2] MATLAB. *Signal Processing Toolbox User's Guide.* The MathWorks, Inc, Natick, MA, 1993.