COMPUTATIONALLY EFFICIENT BLIND MMSE RECEIVERS FOR LONG CODE WCMDA USING TIME-VARYING SYSTEMS THEORY

Alle-Jan van der Veen [*,1] **Lang Tong** [**]

[*] *Delft Univ. of Technology, Dept. Electrical Eng./DIMES, Mekelweg 4, 2628 CD Delft, The Netherlands*
[**] *Cornell University, Dept. Electrical Eng., 326 Frank H.T. Rhodes Hall, Ithaca, NY 14853, USA*

Abstract: UMTS systems will employ long-code wideband CDMA modulation schemes. Receivers for this system are for computational reasons usually based on simple matched-filter techniques, and hence suffer from multiaccess interference. Decorrelating RAKE and MMSE receivers do not have this problem but, until now, were considered as too complex, due to the inversion of a large code matrix. As is shown in this paper, the code matrix can be interpreted as a time-varying system. Efficient implementations are then possible by carrying out the inversion using time-varying state space theory, yielding a complexity comparable to that of the conventional RAKE receiver.

Keywords: Long-code W-CDMA, decorrelating RAKE receiver, MMSE receiver, time-varying system theory, computational structures

1. INTRODUCTION

Current receivers for long-code (or aperiodic spreading code) wideband CDMA are typically based on RAKE receivers, i.e. banks of matched filters which correlate the received data with the desired user's code, followed by a combining of the outputs (RAKE fingers). Since multiuser interference is not completely cancelled, the performance degrades, especially when the network is heavily loaded and power control imperfect.

In this paper, we consider the uplink (mobiles to base station) and assume that the base station knows all codes. We model multiuser interference explicitly and propose a blind decorrelating RAKE and MMSE receiver to estimate the channel and user symbols, based on all samples in a frame. The decorrelating RAKE was presented earlier by us in (Tong *et al.*, 2002*b*; Tong *et al.*, 2002*a*) with an emphasis on identifiability and performance; the MMSE receiver is similar. Here, we also take the noise covariance into account and focus in particular on the efficient implementation of these receivers.

The decorrelating matched filter asks for the inversion of a code matrix whose long dimension is equal to the number of chips over the complete frame. This is a formidable task, but fortunately, the sparse structure of this matrix admits computationally efficient techniques. As an application of the work in (Dewilde and van der Veen, 1998) on the inversion of infinite-size matrices, we derive efficient time-varying state-space implementations of the various steps in the algorithm.

Blind channel estimation and multiuser detection for long code CDMA has been considered by a number of other authors. In particular, second order moment techniques (Zoltowski *et al.*, 1996; Liu and Zoltowski, 1997; Sidiropoulos and Bro, 1999; Xu and Tsatanis, 2000; Escudero *et al.*, 2001) rely on the convergence of time averages, which often requires hundreds to thousands of symbols. Although related, Weiss and Friedlander (Weiss and Friedlander, 1999) focus on the down link where users can be considered synchronous.

[1] The research of A.J. van der Veen was supported in part by the Dutch Min. Econ. Affairs under TSIT 1025 "Beyond-3G".

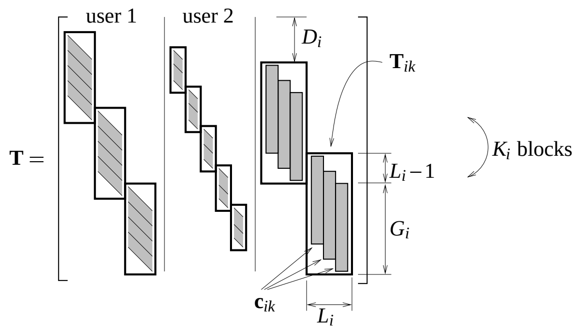

Fig. 1. Structure of the code matrix $\mathbf{T}$.

2. DATA MODEL

We consider the uplink of a slotted system with I asynchronous users. In a frame, the i-th user transmits a vector $\mathbf{s}_i$ consisting of K_i symbols s_{ik}. Each symbol is spread by an aperiodic code $\mathbf{c}_{ik}$ of length G_i. After multipath propagation over a channel with length L_i chips and relative delay D_i, pulse shape matched filtering and sampling at the chiprate, the receiver stacks the received samples in a frame in a vector $\mathbf{y}$. (Oversampling is equivalent to a system with multiple receive antennas.) The contribution of s_{ik} is a linear combination of the transmitted signal $\mathbf{c}_{ik}s_{ik}$, plus delays of it, properly scaled by the L_i channel coefficients collected in a vector $\mathbf{h}_i$, or

$$\mathbf{y}_{ik} = \mathbf{T}_{ik}\mathbf{h}_i\mathbf{s}_{ik}\,, \quad k = 1, \cdots, K_i\,.$$

$\mathbf{T}_{ik}$ is a Toeplitz matrix whose L_i columns consist of shifts of the code $\mathbf{c}_{ik}$. Including all users and the noise, we have

$$\begin{aligned} \mathbf{y} &= \mathbf{THs} + \mathbf{w} \qquad (1)\\ \mathbf{T} &:= [\mathbf{T}_{11}\cdots\mathbf{T}_{1,G_i}, \cdots, \mathbf{T}_{I1}\cdots\mathbf{T}_{I,G_I}]\\ \mathbf{H} &:= \mathrm{diag}(\mathbf{I}_{K_1}\otimes\mathbf{h}_1, \cdots, \mathbf{I}_{K_I}\otimes\mathbf{h}_I)\,. \end{aligned}$$

where matrix $\mathbf{H}$ is block diagonal with $\mathbf{I}\otimes\mathbf{h}_i$ as the ith block, vector $\mathbf{s}$ is a stacking of all symbol vectors, and $\mathbf{w}$ is a vector representing the additive Gaussian noise. The structure of the code matrix $\mathbf{T}$ is illustrated in figure 1. Note that different spreading gains G_i are part of the model. Multiple antennas are a simple extension.

We will assume that the code matrix $\mathbf{T}$ is known, "tall" and has full column rank. This implies that the receiver knows the codes, the delay offsets D_i, and the number of paths L_i of all users.

3. BLIND RECEIVER ALGORITHMS

3.1 *Conventional RAKE*

The conventional RAKE receiver consists of a bank of matched filters and projects the received signal into the code domains of the individual users, by correlating with several shifts of the code vectors, or $\mathbf{r} = \mathbf{T}^H\mathbf{y}$. Since the codes are not exactly orthogonal (let alone shift-orthogonal), $\mathbf{T}^H\mathbf{T} \neq \mathbf{I}$, and contributions of each user enter into the projections of any other user. This makes the performance interference-limited.

3.2 *Decorrelating RAKE*

The proposed decorrelating RAKE uses a decorrelating matched filter, or $\mathbf{T}^\dagger = (\mathbf{T}^H\mathbf{T})^{-1}\mathbf{T}^H$. This removes all multi-user interference. The output of the decorrelating matched filter is given by

$$\mathbf{u} = \mathbf{T}^\dagger\mathbf{y} = \mathrm{diag}(\mathbf{I}\otimes\mathbf{h}_1, \cdots, \mathbf{I}\otimes\mathbf{h}_I)\,\mathbf{s} + \mathbf{n}\,, \qquad (2)$$

where $\mathbf{n} = \mathbf{T}^\dagger\mathbf{w}$ is now a colored noise vector. After computing $\mathbf{u}$, we estimate the channel and the data symbols, blindly and independently for each user. Partition $\mathbf{u}$ into segments $\mathbf{u}_{ik}$ of length L_i. The structure of $\mathbf{u}$ implies that $\mathbf{u}_{ik}$ corresponds to symbol k of user i,

$$\mathbf{u}_{ik} = \mathbf{h}_i s_{ik} + \mathbf{n}_{ik}\,, \quad k = 1, \cdots, K_i\,, \qquad (3)$$

and is free from multiuser interference. Collecting all data for user i gives $\mathbf{U}_i = [\mathbf{u}_{i1}, \cdots, \mathbf{u}_{iK_i}] = \mathbf{h}_i\mathbf{s}_i^T + \mathbf{N}_i$. This is a rank-1 data model, and estimates of $\mathbf{h}_i$ and $\mathbf{s}_i$ (with an unknown scaling factor) are found from a rank-one factorization of $\mathbf{U}_i$. In other words, denoting

$$\mathbf{\Psi}_i := \frac{1}{K_i}\sum_{k=1}^{K_i}\mathbf{u}_{ik}\mathbf{u}_{ik}^H\,, \qquad (4)$$

we obtain the least squares estimates

$$\hat{\mathbf{h}}_i = \arg\max_{\|\mathbf{g}\|=1}\mathbf{g}^H\mathbf{\Psi}_i\mathbf{g}\,, \quad \hat{s}_{ik} = \hat{\mathbf{h}}_i^H\mathbf{u}_{ik}\,. \qquad (5)$$

The solution $\hat{\mathbf{h}}_i$ is given as the dominant eigenvector of $\mathbf{\Psi}_i$. The scaling ambiguity is resolved by a single pilot symbol. See (Tong *et al.*, 2002*b*; Tong *et al.*, 2002*a*) for further results and performance simulations.

3.3 *Whitened Estimator*

The channel and symbol estimator given in (5) did not take into account that the noise process $\mathbf{n}_{ik}$ is colored, both in k and in its components. If we ignore the coloring in k, then a simple whitening approach can be applied. Specifically, since $\mathbf{n} = \mathbf{T}^\dagger\mathbf{w}$, we have that $\mathbf{n}_{ik} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{\Sigma}_{ik})$ where $\mathbf{\Sigma}_{ik}$ is an $L_i \times L_i$ submatrix on the diagonal of $\mathbf{T}^\dagger(\mathbf{T}^\dagger)^H$. We have

$$\begin{aligned} \mathrm{E}(\mathbf{\Psi}_i) &= \frac{\|\mathbf{s}_i\|^2}{K_i}\mathbf{h}_i\mathbf{h}_i^H + \sigma^2\mathbf{\Delta}_i\,,\\ \mathbf{\Delta}_i &:= \frac{1}{K_i}\sum_{k=1}^{K_i}\mathbf{\Sigma}_{ik} \end{aligned}$$

where $\mathbf{\Delta}_i$ is a known matrix. The channel can then be estimated from the following modification which whitens the noise on $\mathbf{\Psi}_i$:

$$\mathbf{g}_* = \arg\max_{\|\mathbf{g}\|=1} \mathbf{g}^H(\boldsymbol{\Delta}_i^{-1/2}\boldsymbol{\Psi}_i\boldsymbol{\Delta}_i^{-H/2})\mathbf{g}$$
$$\hat{\mathbf{h}}_i = \boldsymbol{\Delta}^{1/2}\mathbf{g}_* .$$

The symbol estimator given in (5) is replaced by $\hat{s}_{ik} = \hat{\mathbf{h}}_i^H \boldsymbol{\Sigma}_{ik}^{-1}\mathbf{u}_{ik}$.

3.4 *MMSE Receiver*

Based on the data model (1), the estimated data sequence by a linear minimum mean square error (MMSE) receiver is known to be

$$\hat{\mathbf{s}} = (\mathbf{H}^H\mathbf{T}^H\mathbf{T}\mathbf{H} + \sigma^2\mathbf{I})^{-1}\mathbf{H}^H\mathbf{T}^H\mathbf{y} . \qquad (6)$$

This receiver can be implemented using the previously estimated channel matrix $\mathbf{H}$, and assuming that the noise power σ^2 is known. Compared to the decorrelating RAKE, the MMSE can have a significantly improved performance. It is also one of two similar steps in an iterative LS estimator (Hieu and van der Veen, 2003).

4. EFFICIENT IMPLEMENTATIONS

The code matrix $\mathbf{T}$ can be very large. Without an efficient technique to compute and apply the left inverse $\mathbf{T}^\dagger = (\mathbf{T}^H\mathbf{T})^{-1}\mathbf{T}^H$, the proposed receiver structures would not be feasible. Fortunately, $\mathbf{T}$ is sparse. Using the Matlab sparse toolbox, $\mathbf{u} = \mathbf{T}^\dagger\mathbf{y}$ can be computed efficiently via a sparse QR factorization $\mathbf{T} = \mathbf{Q}\mathbf{R}$, and $\mathbf{u} = \mathbf{R}^{-1}\mathbf{Q}^H\mathbf{y}$, or, avoiding the storage of $\mathbf{Q}$, as

$$\begin{bmatrix} \mathbf{R} & \mathbf{v} \\ \mathbf{0} & \varepsilon \end{bmatrix} := \mathrm{qr}([\mathrm{sparse}(\mathbf{T}) \;\; \mathbf{y}])$$
$$\mathbf{u} := \mathbf{R}\backslash\mathbf{v}$$

$\mathbf{v} = \mathbf{Q}^H\mathbf{y}$, and $\mathbf{R}\backslash\mathbf{v}$ denotes $\mathbf{R}^{-1}\mathbf{v}$, implemented efficiently via backsubstitution. This does not reveal how the sparse computations can actually be implemented in a practical system. It is also unclear how the noise whitening (computation of $\boldsymbol{\Sigma}_{ik}$) can be implemented efficiently. Explicit computation of $\boldsymbol{\Sigma} = \mathbf{R}^{-1}\mathbf{R}^{-H}$ is to be avoided because $\mathbf{R}^{-1}$ is not sparse even if $\mathbf{R}$ is. In this section, we show how time-varying state space representations can be used for this purpose. The theory behind it is available in (Dewilde and van der Veen, 1998).

4.1 *State Space Representation of a Matrix*

Consider an input signal $\mathbf{u}$ and output signal $\mathbf{y}$, with arbitrary block-partitioning $\mathbf{u} = [\mathbf{u}_1^T, \cdots, \mathbf{u}_N^T]^T$, $\mathbf{y} = [\mathbf{y}_1^T, \cdots, \mathbf{y}_N^T]^T$. The partitioning introduces the notion of "time", or a stage in a computational procedure. The blocks do not need to be of equal size, and some dimensions can even be zero (such a block is denoted by "•").

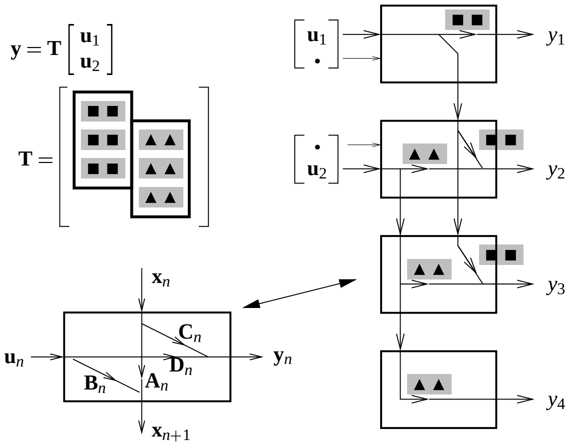

Fig. 2. Computational network for $\mathbf{T} = [\mathbf{T}^{(1)} \;\; \mathbf{T}^{(2)}]$.

A time-varying state space realization has the form

$$\begin{cases} \mathbf{x}_{n+1} = \mathbf{A}_n\mathbf{x}_n + \mathbf{B}_n\mathbf{u}_n \\ \mathbf{y}_n = \mathbf{C}_n\mathbf{x}_n + \mathbf{D}_n\mathbf{u}_n \end{cases}$$
$$\Leftrightarrow$$
$$\begin{bmatrix} \mathbf{x}_{n+1} \\ \mathbf{y}_n \end{bmatrix} = \mathbf{T}_n \begin{bmatrix} \mathbf{x}_n \\ \mathbf{u}_n \end{bmatrix}, \quad \mathbf{T}_n = \begin{bmatrix} \mathbf{A}_n & \mathbf{B}_n \\ \mathbf{C}_n & \mathbf{D}_n \end{bmatrix}$$

The realization starts at time 1 with $\mathbf{x}_1 = \bullet$ (or: no state), and ends with $\mathbf{x}_{N+1} = \bullet$. Hence $\mathbf{A}_1 = \bullet$, $\mathbf{A}_N = \bullet$, $\mathbf{C}_1 = \bullet$, $\mathbf{B}_N = \bullet$.

A time-varying state-space realization specifies a linear mapping of $\mathbf{u}$ to $\mathbf{y}$, hence a matrix $\mathbf{T}$ such that $\mathbf{y} = \mathbf{T}\mathbf{u}$. In particular, it defines a factorization of $\mathbf{T}$ into factors $\mathbf{T}_n$.

Lemma 1. Let be given a time-varying realization $\mathbf{T}_n = \{\mathbf{A}_n, \mathbf{B}_n, \mathbf{C}_n, \mathbf{D}_n\}$ of $\mathbf{T}$. Then $\mathbf{T} = \tilde{\mathbf{T}}_N \cdots \tilde{\mathbf{T}}_2\tilde{\mathbf{T}}_1$ where $\tilde{\mathbf{T}}_n$ is an embedding of $\mathbf{T}_n$,

$$\tilde{\mathbf{T}}_n := \left[\begin{array}{c|ccccccc} \mathbf{A}_n & & & \mathbf{B}_n & & & \\ \hline & \mathbf{I} & & & & & \\ & & \ddots & & & & \\ & & & \mathbf{I} & & & \\ \mathbf{C}_n & & & & \mathbf{D}_n & & \\ & & & & & \mathbf{I} & \\ & & & & & & \ddots \\ & & & & & & & \mathbf{I} \end{array}\right]$$

(there are $n-1$ and $N-n$ identity matrices in the diagonal sequences, respectively.) Moreover, matrix $\mathbf{T}$ is block-lower triangular and has the form

$$\mathbf{T} = \begin{bmatrix} \mathbf{D}_1 & & & \\ \mathbf{C}_2\mathbf{B}_1 & \mathbf{D}_2 & & \\ \vdots & \ddots & \ddots & \\ \mathbf{C}_N\mathbf{A}_{N-1}\cdots\mathbf{A}_2\mathbf{B}_1 & \cdots & \mathbf{C}_N\mathbf{B}_{N-1} & \mathbf{D}_N \end{bmatrix} .$$

Conversely, if a matrix $\mathbf{T}$ has this form, then it has a state space realization $\mathbf{T}_n = \{\mathbf{A}_n, \mathbf{B}_n, \mathbf{C}_n, \mathbf{D}_n\}$.

The inherent causality translates to $\mathbf{T}$ being block-lower triangular. However, by playing with dimensions, *any* matrix can fit this model, as the next examples illustrate. Consider first an an arbitrary $N \times L$ matrix $\mathbf{T}$, with rows $\mathbf{t}_n^H$. A (trivial) realization that models $\mathbf{y} = \mathbf{Tu}$ is obtained by setting $\mathbf{u}_1 = \mathbf{u}$, $\mathbf{u}_2 = \cdots = \mathbf{u}_N = \bullet$ (i.e., the complete input vector is entered at time 1), and

$$\begin{aligned}
\begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_1 \\ \mathbf{C}_1 & \mathbf{D}_1 \end{bmatrix} &= \begin{bmatrix} \bullet & \mathbf{I} \\ \bullet & \mathbf{t}_1^H \end{bmatrix} \\
\begin{bmatrix} \mathbf{A}_n & \mathbf{B}_n \\ \mathbf{C}_n & \mathbf{D}_n \end{bmatrix} &= \begin{bmatrix} \mathbf{I} & \bullet \\ \mathbf{t}_n^H & \bullet \end{bmatrix}, \quad n = 2, \cdots, N-1 \\
\begin{bmatrix} \mathbf{A}_N & \mathbf{B}_N \\ \mathbf{C}_N & \mathbf{D}_N \end{bmatrix} &= \begin{bmatrix} \bullet & \bullet \\ \mathbf{t}_N^H & \bullet \end{bmatrix}
\end{aligned}$$

As a second example, let $\mathbf{T} = [\mathbf{T}^{(1)} \ \ \mathbf{T}^{(2)}]$ be an arbitrary block-partitioned matrix, where $\mathbf{T}^{(1)}$ has realization $\{\mathbf{A}_n^{(1)}, \mathbf{B}_n^{(1)}, \mathbf{C}_n^{(1)}, \mathbf{D}_n^{(1)}\}$ and $\mathbf{T}^{(2)}$ has realization $\{\mathbf{A}_n^{(2)}, \mathbf{B}_n^{(2)}, \mathbf{C}_n^{(2)}, \mathbf{D}_n^{(2)}\}$. Then

$$\mathbf{T}_n = \left[\begin{array}{cc|cc} \mathbf{A}_n^{(1)} & 0 & \mathbf{B}_n^{(1)} & 0 \\ 0 & \mathbf{A}_n^{(2)} & 0 & \mathbf{B}_n^{(2)} \\ \hline \mathbf{C}_n^{(1)} & \mathbf{C}_n^{(2)} & \mathbf{D}_n^{(1)} & \mathbf{D}_n^{(2)} \end{array}\right]$$

is a realization of $\mathbf{T}$. Its structure is shown in Fig. 2.

The code matrix $\mathbf{T}$ in our case has a block structure as shown in Fig. 1. By combining the two examples, we can represent any code matrix $\mathbf{T}$. The number of state space time points equals the number of rows of $\mathbf{T}$. The input vector is partitioned in blocks of L_i entries which enter the system at appropriate time points, determined by the starting points of the individual code blocks. The state dimension at each time point is the number of nonzero entries in the corresponding row of $\mathbf{T}$.

4.2 *QR Factorization and Inversion in State Space*

To compute the left inverse $\mathbf{T}^\dagger$, our aim is to first compute a QR factorization $\mathbf{T} = \mathbf{QR}$ where $\mathbf{Q}^H\mathbf{Q} = \mathbf{I}$ and $\mathbf{R}$ is square and lower triangular, and then to invert each of the factors: $\mathbf{T}^\dagger = \mathbf{R}^{-1}\mathbf{Q}^H$. The computation of the QR factorization can be done in state space, as is demonstrated by the following theorem (cf. (Dewilde and van der Veen, 1998), p.156]).

For $\mathbf{T}$ with realization $\{\mathbf{A}_n, \mathbf{B}_n, \mathbf{C}_n, \mathbf{D}_n\}_{n=1,\cdots;N}$, consider the recursion (economy-size QR factorizations)

$$\begin{aligned}
&\mathbf{Y}_{N+1} = \bullet \\
&\begin{bmatrix} \mathbf{Y}_{n+1}\mathbf{A}_n & \mathbf{Y}_{n+1}\mathbf{B}_n \\ \mathbf{C}_n & \mathbf{D}_n \end{bmatrix} =: \underbrace{\begin{bmatrix} \mathbf{A}_n^Q & \mathbf{B}_n^Q \\ \mathbf{C}_n^Q & \mathbf{D}_n^Q \end{bmatrix}}_{\mathbf{Q}_n} \begin{bmatrix} \mathbf{Y}_n & 0 \\ \mathbf{C}_n^R & \mathbf{D}_n^R \end{bmatrix}, \qquad (7) \\
&n = N, N-1, \cdots, 1
\end{aligned}$$

where $\mathbf{Q}_n$ is isometric ($\mathbf{Q}_n^H\mathbf{Q}_n = \mathbf{I}$), and the right factor is lower triangular (possibly staircase) and partitioned such that $\mathbf{Y}_n$ has the same number of columns as $\mathbf{A}_n$, $\mathbf{D}_n^R$ has the same number of columns as $\mathbf{D}_n$, and both $\mathbf{Y}_n$ and $\mathbf{D}_n^R$ are full row rank.

Theorem 2. If $\mathbf{T}$ is full column rank, then all $\mathbf{D}_n^R$ are square, lower triangular and invertible. Define the realizations

$$\mathbf{Q}_n = \begin{bmatrix} \mathbf{A}_n^Q & \mathbf{B}_n^Q \\ \mathbf{C}_n^Q & \mathbf{D}_n^Q \end{bmatrix}, \qquad \mathbf{R}_n = \begin{bmatrix} \mathbf{A}_n & \mathbf{B}_n \\ \mathbf{C}_n^R & \mathbf{D}_n^R \end{bmatrix}.$$

Then $\mathbf{T} = \mathbf{QR}$, where $\mathbf{Q}$ is specified by $\mathbf{Q}_n$ and is isometric ($\mathbf{Q}^H\mathbf{Q} = \mathbf{I}$), and $\mathbf{R}$ is specified by $\mathbf{R}_n$ and is lower triangular and invertible.

PROOF Recall the factorization $\mathbf{T} = \tilde{\mathbf{T}}_N\tilde{\mathbf{T}}_{N-1}\cdots\tilde{\mathbf{T}}_1$ and consider the first factor, $\mathbf{T}_N$. Since $\mathbf{A}_N = \bullet$, $\mathbf{B}_N = \bullet$, and $\mathbf{Y}_{N+1} = \bullet$,

$$\mathbf{T}_N = \begin{bmatrix} \mathbf{A}_N & \mathbf{B}_N \\ \mathbf{C}_N & \mathbf{D}_N \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_{N+1}\mathbf{A}_N & \mathbf{Y}_{N+1}\mathbf{B}_N \\ \mathbf{C}_N & \mathbf{D}_N \end{bmatrix}.$$

The first step in the recursion is the QR factorization

$$\mathbf{Q}_N^H\mathbf{T}_N = \begin{bmatrix} \mathbf{A}_N^Q & \mathbf{B}_N^Q \\ \mathbf{C}_N^Q & \mathbf{D}_N^Q \end{bmatrix}^H \begin{bmatrix} \mathbf{Y}_{N+1}\mathbf{A}_N & \mathbf{Y}_{N+1}\mathbf{B}_N \\ \mathbf{C}_N & \mathbf{D}_N \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_N & 0 \\ \mathbf{C}_N^R & \mathbf{D}_N^R \end{bmatrix}$$

Premultiplying $\mathbf{T}$ by $\tilde{\mathbf{Q}}_N^H$ gives

$$\begin{aligned}
&\tilde{\mathbf{Q}}_N^H\mathbf{T} = \\
&= \left[\begin{array}{c|cc} \mathbf{A}_N^Q & & \mathbf{B}_N^Q \\ \hline & \mathbf{I} & \\ \mathbf{C}_N^Q & & \mathbf{D}_N^Q \end{array}\right]^H \left[\begin{array}{c|cc} \mathbf{Y}_{N+1}\mathbf{A}_N & & \mathbf{Y}_{N+1}\mathbf{B}_N \\ \hline & \mathbf{I} & \\ \mathbf{C}_N & & \mathbf{D}_N \end{array}\right] \tilde{\mathbf{T}}_{N-1}\cdots\tilde{\mathbf{T}}_1 \\
&= \left[\begin{array}{c|cccc} \mathbf{Y}_N & & & & 0 \\ \hline & \mathbf{I} & & & \\ & & \ddots & & \\ & & & \mathbf{I} & \\ \mathbf{C}_N^R & & & & \mathbf{D}_N^R \end{array}\right] \left[\begin{array}{c|cccc} \mathbf{A}_{N-1} & & \mathbf{B}_{N-1} & & \\ \hline & \mathbf{I} & & & \\ & & \ddots & & \\ \mathbf{C}_{N-1} & & \mathbf{D}_{N-1} & & \\ & & & & \mathbf{I} \end{array}\right] \tilde{\mathbf{T}}_{N-2}\cdots\tilde{\mathbf{T}}_1 \\
&= \left[\begin{array}{c|ccc} \mathbf{Y}_N\mathbf{A}_{N-1} & & \mathbf{Y}_N\mathbf{B}_{N-1} & \\ \hline & \mathbf{I} & & \\ & & \ddots & \\ \mathbf{C}_{N-1} & & \mathbf{D}_{N-1} & \\ \mathbf{C}_N^R\mathbf{A}_{N-1} & & \mathbf{C}_N^R\mathbf{B}_{N-1} & \mathbf{D}_N^R \end{array}\right] \tilde{\mathbf{T}}_{N-2}\cdots\tilde{\mathbf{T}}_1
\end{aligned}$$

We subsequently obtain

$$\begin{aligned}
&\tilde{\mathbf{Q}}_{N-1}^H\tilde{\mathbf{Q}}_N^H\mathbf{T} = \\
&\left[\begin{array}{c|ccc} \mathbf{Y}_{N-1} & & 0 & \\ \hline & \mathbf{I} & & \\ & & \ddots & \\ \mathbf{C}_{N-1}^R & & \mathbf{D}_{N-1}^R & \\ \mathbf{C}_N^R\mathbf{A}_{N-1} & & \mathbf{C}_N^R\mathbf{B}_{N-1} & \mathbf{D}_N^R \end{array}\right] \left[\begin{array}{c|ccc} \mathbf{A}_{N-2} & \mathbf{B}_{N-2} & & \\ \hline & \mathbf{I} & & \\ \mathbf{C}_{N-2} & \mathbf{D}_{N-2} & & \\ & & \mathbf{I} & \\ & & & \mathbf{I} \end{array}\right] \\
&\cdot\tilde{\mathbf{T}}_{N-3}\cdots\tilde{\mathbf{T}}_1 = \\
&\left[\begin{array}{c|ccc} \mathbf{Y}_{N-1}\mathbf{A}_{N-2} & \mathbf{Y}_{N-1}\mathbf{B}_{N-2} & & \\ \hline & \mathbf{I} & & \\ \mathbf{C}_{N-2} & \mathbf{D}_{N-2} & & \\ \mathbf{C}_{N-1}^R\mathbf{A}_{N-2} & \mathbf{C}_{N-1}^R\mathbf{B}_{N-2} & \mathbf{D}_{N-1}^R & \\ \mathbf{C}_N^R\mathbf{A}_{N-1}\mathbf{A}_{N-2} & \mathbf{C}_N^R\mathbf{A}_{N-1}\mathbf{B}_{N-2} & \mathbf{C}_N^R\mathbf{B}_{N-1} & \mathbf{D}_N^R \end{array}\right] \\
&\cdot\tilde{\mathbf{T}}_{N-3}\cdots\tilde{\mathbf{T}}_1
\end{aligned}$$

Following the recursion this way, we finally obtain

$$\tilde{\mathbf{Q}}_1^H \cdots \tilde{\mathbf{Q}}_N^H \mathbf{T} = \left[\begin{array}{c|cccc} \mathbf{Y}_1 & & & & \\ \hline \mathbf{C}_1^R & \mathbf{D}_1^R & & & \\ \mathbf{C}_2^R \mathbf{A}_1 & \mathbf{C}_2^R \mathbf{B}_1 & \mathbf{D}_2^R & & \\ \vdots & \vdots & \ddots & \ddots & \\ \mathbf{C}_N^R \mathbf{A}_{N-1} \cdots \mathbf{A}_1 & \mathbf{C}_N^R \mathbf{A}_{N-1} \cdots \mathbf{A}_2 \mathbf{B}_1 & \cdots & \cdots & \mathbf{D}_N^R \end{array}\right].$$

Note that $\mathbf{A}_1 = \cdot$ so that the first column has zero width. Hence $\mathbf{Y}_1 = \cdot$ (since the $\mathbf{Y}_k$ are wide) and also the first row has empty dimensions. It follows that

$$\tilde{\mathbf{Q}}_1^H \cdots \tilde{\mathbf{Q}}_N^H \mathbf{T} = \begin{bmatrix} \mathbf{D}_1^R & & & \\ \mathbf{C}_2^R \mathbf{B}_1 & \mathbf{D}_2^R & & \\ \vdots & \ddots & \ddots & \\ \mathbf{C}_N^R \mathbf{A}_{N-1} \cdots \mathbf{A}_2 \mathbf{B}_1 & \cdots & \cdots & \mathbf{D}_N^R \end{bmatrix} = \mathbf{R}$$

This is equal to $\mathbf{Q}^H \mathbf{T} = \mathbf{R}$, where $\mathbf{R}$ is lower triangular. Lemma 1 shows that $\mathbf{R} = \tilde{\mathbf{R}}_N \cdots \tilde{\mathbf{R}}_1$, so that $\mathbf{R}$ has the advertised state space realization. Since $\mathbf{T}$ is full column rank, all $\mathbf{D}_N^R$ are square and invertible, so that $\mathbf{R}$ is square and invertible. $\mathbf{Q}$ is isometric since each of its factors $\mathbf{Q}_n$ is isometric. □

The structure of the factorization is shown in Fig. 3(a). Note that in our application, $\mathbf{A}_n$ and $\mathbf{B}_n$ are trivial: embeddings of identity matrices of appropriate sizes. Hence the multiplication by $\mathbf{Y}_{n+1}$ is trivial and the only actual work in (7) is the QR factorization.

Theorem 3. Suppose that $\mathbf{R}$ is a square invertible lower triangular matrix. Then its inverse is lower triangular too. If $\mathbf{R}$ has state space realization

$$\mathbf{R}_n = \begin{bmatrix} \mathbf{A}_n^R & \mathbf{B}_n^R \\ \mathbf{C}_n^R & \mathbf{D}_n^R \end{bmatrix}, \qquad n = 1, \cdots, N$$

then $\mathbf{S} := \mathbf{R}^{-1}$ has state space realization

$$\mathbf{S}_n = \begin{bmatrix} \mathbf{A}_n^R - \mathbf{B}_n^R \mathbf{D}_n^{R^{-1}} \mathbf{C}_n^R & \mathbf{B}_n^R \mathbf{D}_n^{R^{-1}} \\ -\mathbf{D}_n^{R^{-1}} \mathbf{C}_n^R & \mathbf{D}_n^{R^{-1}} \end{bmatrix}, \; n = 1, \cdots, N$$

PROOF Note that $\mathbf{R}\mathbf{u} = \mathbf{y} \Leftrightarrow \mathbf{S}\mathbf{y} = \mathbf{u}$, hence $\mathbf{S}$ maps $\mathbf{y}$ to $\mathbf{u}$. Since $\mathbf{S}$ is lower triangular (causal),

$$\begin{aligned} \mathbf{y}_n &= \mathbf{C}_n^R \mathbf{x}_n + \mathbf{D}_n^R \mathbf{u}_n \\ \Leftrightarrow \mathbf{u}_n &= -\mathbf{D}_n^{R^{-1}} \mathbf{C}_n^R \mathbf{x}_n + \mathbf{D}_n^{R^{-1}} \mathbf{y}_n \end{aligned}$$

Backsubstitution in $\mathbf{x}_{n+1} = \mathbf{A}_n^R \mathbf{x}_n + \mathbf{B}_n^R \mathbf{u}_n$ gives the result. □

The left-inverse of the isometric factor $\mathbf{Q}$ is $\mathbf{Q}^H$, with *anticausal* state space realization (backward recursion)

$$\begin{cases} \mathbf{x}_n = \mathbf{A}_n^{Q^H} \mathbf{x}_{n+1} + \mathbf{C}_n^{Q^H} \mathbf{u}_n \\ \mathbf{y}_n = \mathbf{B}_n^{Q^H} \mathbf{x}_{n+1} + \mathbf{D}_n^{Q^H} \mathbf{u}_n \end{cases} \quad n = N, N-1, \cdots, 1.$$

The preceding theorems can be used to invert more general matrices, in particular the code matrix $\mathbf{T}$. We obtain an implementation of $\mathbf{T}^\dagger = \mathbf{S}\mathbf{Q}^H$ in factored form, where $\mathbf{T}^\dagger$, $\mathbf{R}$ and $\mathbf{Q}$ are never explicitly evaluated. The structure of the computational network is shown in Fig. 3(b). As is seen from this structure, the "complexity" of $\mathbf{T}$ and $\mathbf{T}^\dagger$ is the same, even if $\mathbf{T}^\dagger$ is a full matrix without visible sparse structure.

4.3 *Computation of* $\mathbf{\Sigma}_{ik}$

In the computation of the noise covariance, expressions for $\mathbf{\Sigma}_{ik}$ are needed. We can apply the following theorem:

Theorem 4. Let $\mathbf{T}$ have state space realization $\{\mathbf{A}_n, \mathbf{B}_n, \mathbf{C}_n, \mathbf{D}_n\}$. A realization for the lower triangular part of $\mathbf{N} := \mathbf{T}\mathbf{T}^H$ is given by

$$\mathbf{N}_n = \begin{bmatrix} \mathbf{A}_n & \mathbf{A}_n \mathbf{\Lambda}_n \mathbf{C}_n^H + \mathbf{B}_n \mathbf{D}_n^H \\ \mathbf{C}_n & \mathbf{C}_n \mathbf{\Lambda}_n \mathbf{C}_n^H + \mathbf{D}_n \mathbf{D}_n^H \end{bmatrix}$$

where $\mathbf{\Lambda}_n$ is specified by the forward recursion

$$\mathbf{\Lambda}_1 = \cdot; \quad \mathbf{\Lambda}_{n+1} = \mathbf{A}_n \mathbf{\Lambda}_n \mathbf{A}_n^H + \mathbf{B}_n \mathbf{B}_n^H \qquad n = 1, 2, \cdots, N,$$

PROOF By inspection of Fig. 3(c) and following the mapping of $\mathbf{x}_n, \mathbf{u}_n$ to $\mathbf{x}_{n+1}, \mathbf{y}_n$. The causal part of the state is $\mathbf{x}_n$, the non-causal part is $\mathbf{x}'_n$, and $\mathbf{\Lambda}_n$ represents the transfer of $\mathbf{x}'_n$ to $\mathbf{x}_n$. (A formal proof appears in (Dewilde and van der Veen, 1998, p.366).) □

The preceding recursions are useful in the computation of the noise covariance after the decorrelating matched filter. If $\mathbf{w}$ is a white noise vector with power normalized to $\sigma^2 = 1$, and $\mathbf{n} = \mathbf{T}^\dagger \mathbf{w} = (\mathbf{T}^H \mathbf{T})^{-1} \mathbf{T}^H \mathbf{w}$, then the covariance of $\mathbf{n}$ is given by

$$\mathbf{\Sigma} := \mathrm{E}(\mathbf{n}\mathbf{n}^H) = (\mathbf{T}^H \mathbf{T})^{-1} = \mathbf{S}\mathbf{S}^H$$

where $\mathbf{T} = \mathbf{Q}\mathbf{R}$ and $\mathbf{S} = \mathbf{R}^{-1}$. A state space realization of $\mathbf{S}$ was derived before. Thus, theorem 4 (applied to $\mathbf{S}$) gives a recursion to compute a realization for the lower part of $\mathbf{S}\mathbf{S}^H$. The upper part is simply the transpose.

In the identification algorithm in section 3.3, we are only interested in the main (block)-diagonal of $\mathrm{E}(\mathbf{n}\mathbf{n}^H)$ (the auto-covariances of size $L_i \times L_i$). In this case, it suffices to compute

$$\begin{aligned} \mathrm{E}(\mathbf{n}_n \mathbf{n}_n^H) &= \mathbf{C}_n^S \mathbf{\Lambda}_n \mathbf{C}_n^{S^H} + \mathbf{D}_n^S \mathbf{D}_n^{S^H} \\ \mathbf{\Lambda}_{n+1} &= \mathbf{A}_n^S \mathbf{\Lambda}_n \mathbf{A}_n^{S^H} + \mathbf{B}_n^S \mathbf{B}_n^{S^H} \end{aligned}$$

4.4 *Computation of the MMSE Receiver in State Space*

Recall the MMSE receiver (6). It is known that equations of this form can be efficiently computed via a QR factorization. Indeed, note that

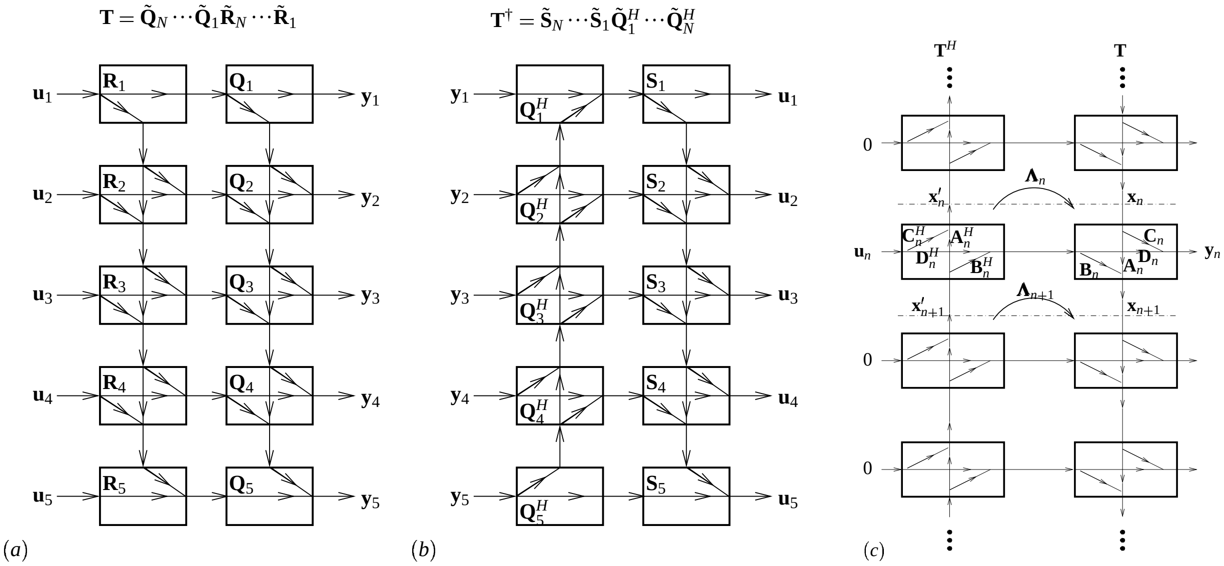

Fig. 3. (a) Structure of the QR factorization, (b) structure of the inverse, (c) structure of $\mathbf{TT}^H$.

$$\begin{aligned}\hat{\mathbf{s}} &= (\mathbf{H}^H\mathbf{T}^H\mathbf{T}\mathbf{H}+\sigma^2\mathbf{I})^{-1}\mathbf{H}^H\mathbf{T}^H\mathbf{y} \qquad (8)\\ &= (\mathbf{H}^H\mathbf{T}^H\mathbf{T}\mathbf{H}+\sigma^2\mathbf{I})^{-1}\begin{bmatrix}\mathbf{H}^H\mathbf{T}^H & \sigma\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{y}\\ \mathbf{0}\end{bmatrix}\\ &= \underbrace{\begin{bmatrix}\mathbf{TH}\\ \sigma\mathbf{I}\end{bmatrix}}_{\mathbf{M}}^{\dagger}\begin{bmatrix}\mathbf{y}\\ \mathbf{0}\end{bmatrix}\end{aligned}$$

Thus, if $\mathbf{M} =: \mathbf{Q}^M\mathbf{R}^M$ is an economy-size QR factorization for $\mathbf{M}$ (where $\mathbf{R}^M$ is square triangular, and $\mathbf{Q}^M$ is tall and isometric), then

$$\hat{\mathbf{s}} = (\mathbf{R}^M)^{-1}(\mathbf{Q}^M)^H\begin{bmatrix}\mathbf{y}\\ \mathbf{0}\end{bmatrix}.$$

The QR factorization and factor inversion can be done in state space as before. Thus, $\hat{\mathbf{s}}$ is the output of a computational structure similar to the one in Fig. 3(b). The only new aspect is the derivation of a realization for $\mathbf{M}$.

A realization $\{\mathbf{A}_n,\mathbf{B}_n,\mathbf{C}_n,\mathbf{D}_n\}$ for $\mathbf{T}$ is already known. $\mathbf{H}$ is block-diagonal, with blocks $\mathbf{h}_i$ matching the inputs of $\mathbf{T}$. Define

$$\begin{aligned}\mathbf{H}_n &:= \begin{bmatrix}\boldsymbol{\beta}_{1,n} & & \\ & \ddots & \\ & & \boldsymbol{\beta}_{I,n}\end{bmatrix}\\ \boldsymbol{\beta}_{i,n} &:= \begin{cases}\mathbf{h}_i, & \mathbf{T}\text{ has an input for user } i \text{ at } n\\ \cdot, & \text{otherwise.}\end{cases}\end{aligned}$$

A realization for $\mathbf{TH}$ is then given by

$$(\mathbf{TH})_n = \begin{bmatrix}\mathbf{A}_n & \mathbf{B}_n\mathbf{H}_n\\ \mathbf{C}_n & \mathbf{D}_n\mathbf{H}_n\end{bmatrix}, \quad n = 1,\cdots,N.$$

Finally, a realization for $\mathbf{M}$ is simply obtained by extending the D-matrix by $\sigma\mathbf{I}$:

$$\mathbf{M}_n = \left[\begin{array}{c|c}\mathbf{A}_n & \mathbf{B}_n\mathbf{H}_n\\ \hline \mathbf{C}_n & \mathbf{D}_n\mathbf{H}_n\\ \mathbf{0} & \sigma\mathbf{I}\end{array}\right], \quad n = 1,\cdots,N.$$

5. REFERENCES

Dewilde, P. and A.J. van der Veen (1998). *Time-Varying Systems and Computations*. Kluwer Academic Publishers. Dordrecht, The Netherlands.

Escudero, C., U. Mitra and D. Slock (2001). A Toeplitz displacement method for blind multipath estimation for Long Code DS/CDMA signals. *IEEE Trans. Signal Processing* **SP-48**(3), 654–665.

Hieu, D.Q. and A.J. van der Veen (2003). Single- and multi-user blind receivers for long code WCDMA. In: *IEEE workshop on Signal Proc. Adv. in Wireless Comm. (SPAWC)*. Rome (Italy).

Liu, H. and M. Zoltowski (1997). Blind equalization in antenna array CDMA systems. *IEEE Trans. Signal Processing* **45**(1), 161172.

Sidiropoulos, N. and R. Bro (1999). User separation in DS-CDMA Systems with unknown Long PN Spreading Codes. In: *Proc. SPAWC*. Annapolis, MD.. pp. 194–197.

Tong, Lang, A.J. van der Veen and P. Dewilde (2002*a*). Channel estimation for long-code WCDMA. In: *Proc. IEEE ISCAS*. IEEE. Scotsdale (AZ).

Tong, Lang, A.J. van der Veen and P. Dewilde (2002*b*). A new decorrelating RAKE receiver for long-code WCDMA. In: *36th Ann. Conf. Information Sciences and Systems (CISS)*. Princeton (NJ).

Weiss, A. and B. Friedlander (1999). Channel estimation for DS-CDMA downlink with aperiodic spreading codes. *IEEE Tr. Comm.* **47**(10), 1561–1569.

Xu, Z. and M. Tsatanis (2000). Blind channel estimation for long code multiuser CDMA systems. *IEEE Trans.Signal Processing* **SP-48**(4), 988–1001.

Zoltowski, M., Y. Chen and J. Ramos (1996). Blind 2D RAKE receivers based on space-time adaptive MVDR processing for IS-95 CDMA system. In: *Proceedings of the 15th IEEE MILCOM*. Atlanta, GA. pp. 618–622.