Schur Algorithms for Joint-Detection in TD-CDMA based Mobile Radio Systems

Marius Vollmer, Martin Haardt, Jürgen Götze

Marius Vollmer works on his PhD thesis both at the University of Dortmund and at Siemens AG.
Jürgen Götze is head of the Information Processing Lab at the University of Dortmund.
Martin Haardt works at Siemens AG, ICN CA CTO 7.

Abstract

Third generation mobile radio systems will employ TD-CDMA in their TDD mode. In a TD-CDMA mobile radio system, joint-detection is equivalent to solving a least squares problem with a system matrix that exhibits some form of block-Toeplitz structure. This structure can be successfully exploited by using variations of the Schur algorithm for computing the QR decomposition of this system matrix. Together with a displacement representation, the Schur algorithm can be straightforwardly adapted to a wide variety of matrix structures. In this paper we show this approach for two concrete manifestations of the TD-CDMA system matrix: first for a very compact, block-Toeplitz structure; and second for the less favorable Toeplitz-block structure that arises when decision feedback is added to the data detection process.

Keywords

TD-CDMA, joint detection, least-squares, block-Toeplitz, Toeplitz-block, displacement representation, Schur algorithm.

I. INTRODUCTION

In January 1998, the European standardization body for third generation mobile radio systems, the ETSI Special Mobile Group (SMG), has agreed on a radio access scheme for third generation mobile radio systems, called Universal Mobile Telecommunications System (UMTS). This agreement recommends the use of WCDMA (wideband CDMA) in the paired portion of the radio spectrum (FDD mode) and TD-CDMA (Time Division CDMA) in the unpaired portion (TDD mode). TD-CDMA is a TDMA based system where multiple users are separated by spreading codes within one time slot.

To overcome the near/far problem of traditional CDMA systems, receiver structures have been proposed for TD-CDMA that perform joint (or multi-user) detection [1]. A joint detector combines the knowledge about all users that share one time slot into one large system of equations [2], [1], [3], [4]. This knowledge consists of the channel impulse responses that have been estimated from training sequences, the spreading codes, and the received antenna samples. The resulting system of equations can be very large and thus algorithms must be developed to exploit its special structural characteristics, namely its band and block-Toeplitz structure. In [5] an approach based on the Cholesky algorithm was presented. The band structure of the system matrix leads to an approximate block-Toeplitz structure in the desired Cholesky factor. This has been exploited by computing the Cholesky factor of a smaller subproblem and using it to approximate the complete Cholesky factor from copies of that smaller factor [6].

In this paper, we show how the Toeplitz structure of the system matrix can be directly exploited by the Schur algorithm. While the resulting computational complexity is not significantly lower than for the approximating Cholesky algorithm, the Schur algorithm is more amendable to a fine-grained parallel implementation with systolic processor arrays. In addition, it provides a flexible framework for engineering joint-detection algorithms in situations where the approximated Cholesky algorithm is not efficient. We show how this flexibility can be put to good use when decision feedback is introduced into the joint detection process.

After detailing the system model of the TD-CDMA system and explaining the basic joint detection process in sections II and III, this paper shows how to derive a generalized Schur algorithm that can cope with the specific kind of Toeplitz-derived structure of the system matrix (section IV) and how to apply approximation techniques to it (section V). In section VI, we present comparisons of the computational complexity for two algorithms, and then conclude.

II. System Model

Performing joint detection can be reduced to solving a system of linear equations. In the sequel, we will explain the construction of this system of equations and the underlying data model.

A. TD-CDMA

In the TD-CDMA system, K CDMA codes are simultaneously active on the same frequency and in the same time slot. The different spreading codes allow the signal separation at the receiver. According to the required data rate, a given user might use several CDMA codes and/or time slots. The frame structure for this time-slotted CDMA concept is illustrated in Figure 1, where B, T_{fr}, N_{fr}, T_{bu}, and K denote the bandwidth of a frequency slot, the duration of a TDMA frame, the number of bursts per TDMA frame, the burst duration, and the number of CDMA codes per frequency and time slot, respectively. A burst consists of a guard interval and two data blocks (of N symbols each), separated by a user specific midamble which contains L_m chips and is used for channel estimation [3], see Figure 1. In Figure 2, the structure of one time slot is illustrated for the kth midamble and the kth spreading code. Here, Q denotes the spreading factor of the data symbols.

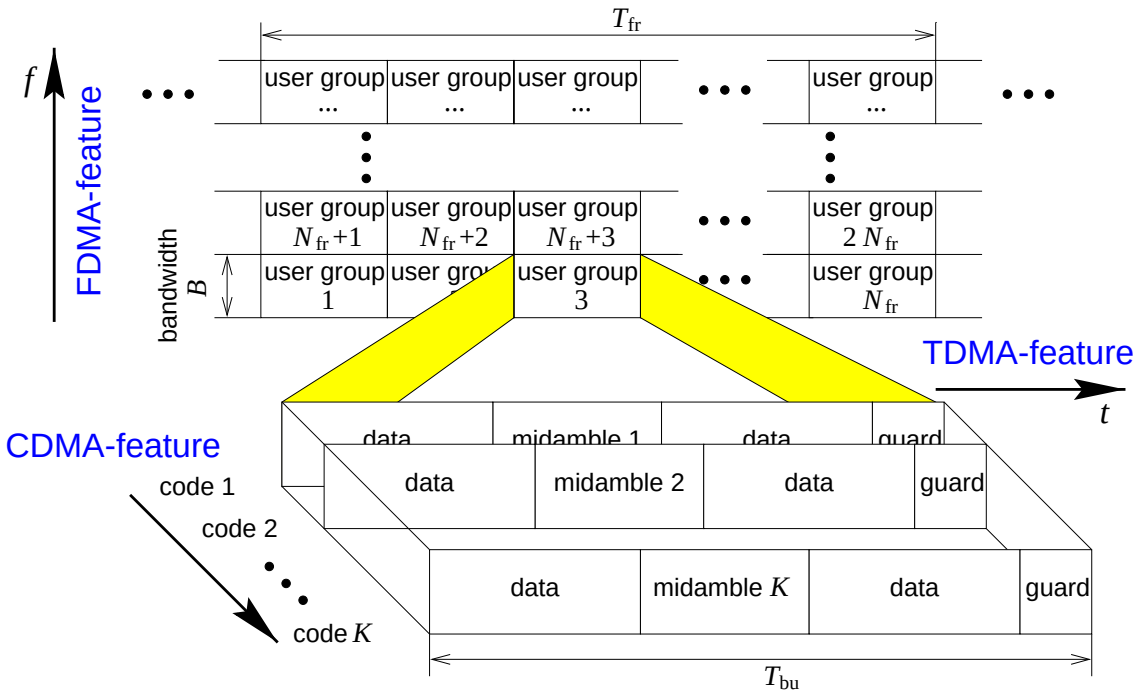


Fig. 1. Frame structure of the TD-CDMA system. Here, B, $T_{\rm fr}$, $N_{\rm fr}$, $T_{\rm bu}$, and K denote the bandwidth of a frequency slot, the duration of a TDMA frame, the number of bursts per TDMA frame, the burst duration, and the number of CDMA codes per frequency and time slot, respectively.

B. Discrete Time Data Model

The received measurements at the mth antenna during the duration of one time slot are pictured in Figure 3. The parts labeled *a* of the received measurement vector are only influenced by the corresponding data blocks, the *c* part is exclusively determined by the transmitted midambles, and the *b* parts of the measurement vector are influenced by the transmitted midambles and the corresponding data blocks. The channel impulse response (CIR) vectors between the kth mobile and the mth antenna $\boldsymbol{h}^{(k,m)} \in \mathbb{C}^W$ are estimated from the *c* part of this received measurement vector as, for instance, described in [7].

Let us combine the N data symbols $d_n^{(k)}$, $1 \le n \le N$, that are transmitted on the kth spreading code during one data block (half burst) to the vector

$$\boldsymbol{d}^{(k)} = \begin{bmatrix} d_1^{(k)} \\ d_2^{(k)} \\ \vdots \\ d_N^{(k)} \end{bmatrix} \in \mathbb{C}^N, \quad 1 \le k \le K. \tag{1}$$

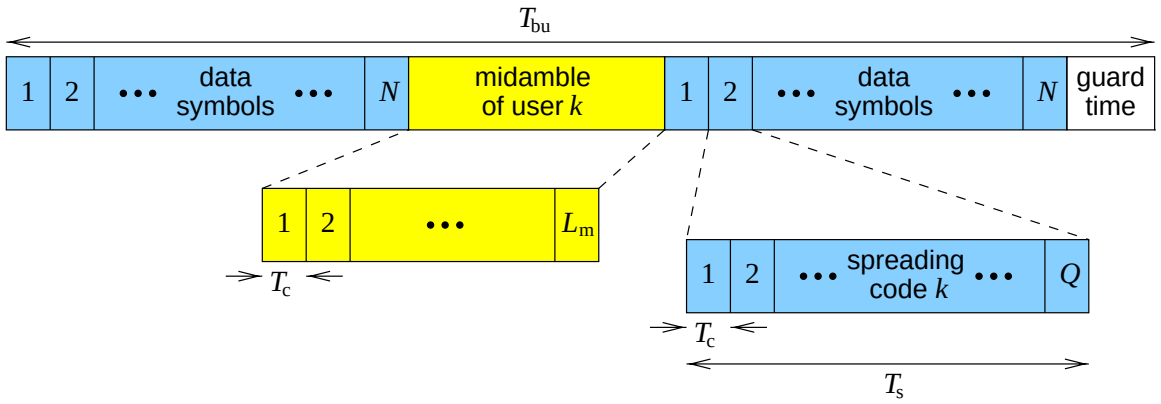


Fig. 2. Time slot structure of the TD-CDMA system. Here, T_{bu}, T_{s} T_{c}, and Q denote the burst duration, the symbol duration, the chip duration, and the spreading factor of the data symbols, respectively.

The kth spreading code consists of Q complex chips $c_q^{(k)}$, $1 \leq q \leq Q$, and is denoted as

$$\boldsymbol{c}^{(k)} = \begin{bmatrix} c_1^{(k)} \\ c_2^{(k)} \\ \vdots \\ c_Q^{(k)} \end{bmatrix} \in \mathbb{C}^Q, \quad 1 \leq k \leq K.$$

With this definition, the block-diagonal spreading matrix $\boldsymbol{C}^{(k)}$ that corresponds to the kth code can be written as

$$\boldsymbol{C}^{(k)} = \boldsymbol{I}_N \otimes \boldsymbol{c}^{(k)} = \begin{bmatrix} \boldsymbol{c}^{(k)} & & & \\ & \boldsymbol{c}^{(k)} & & \\ & & \ddots & \\ & & & \boldsymbol{c}^{(k)} \end{bmatrix} \in \mathbb{C}^{NQ \times N}. \tag{2}$$

Assume that K spreading codes are transmitted at the same time. After eliminating the influence of the midamble, cf. Figure 3, the received measurements at the mth antenna obey the following linear model:

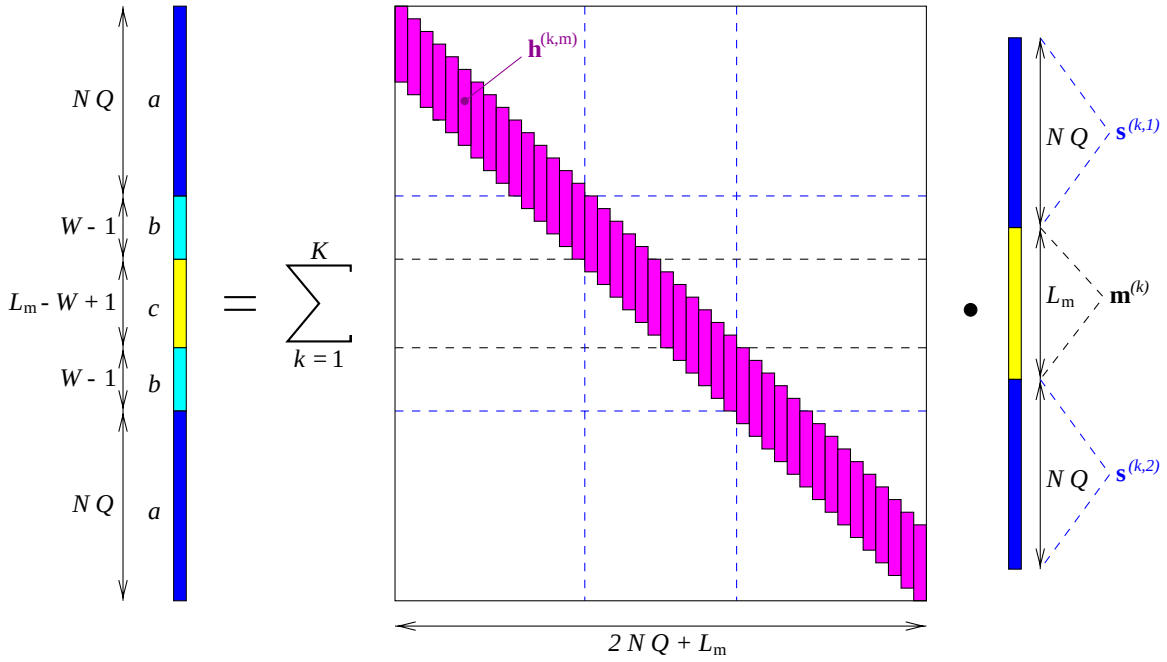


Fig. 3. Received measurements at the mth antenna during the duration of one time slot. In this illustration, the additive noise and inter-cell-interference are not considered. The parts labeled *a* of the received measurement vector are only influenced by the corresponding data block, the *c* part only by the transmitted midamble, and the *b* parts by the midamble and the corresponding data blocks.

$$\boldsymbol{x}^{(m)} = \sum_{k=1}^{K} \underbrace{\begin{bmatrix} \boldsymbol{h}^{(k,m)} & & \\ & \ddots & \\ & & \boldsymbol{h}^{(k,m)} \end{bmatrix}}_{\boldsymbol{H}^{(k,m)}} \cdot \boldsymbol{C}^{(k)} \cdot \boldsymbol{d}^{(k)} + \begin{bmatrix} n_1^{(m)} \\ n_2^{(m)} \\ \vdots \\ n_{NQ+W-1}^{(m)} \end{bmatrix} =$$

$$= \sum_{k=1}^{K} \underbrace{\boldsymbol{H}^{(k,m)} \cdot \boldsymbol{C}^{(k)}}_{\boldsymbol{B}^{(k,m)}} \cdot \boldsymbol{d}^{(k)} + \boldsymbol{n}^{(m)}, \qquad 1 \le m \le M \tag{3}$$

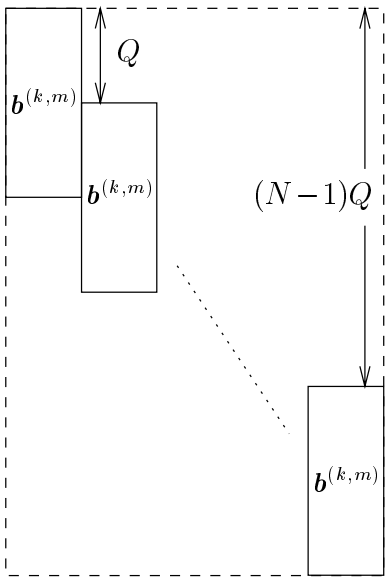
with

$$\boldsymbol{x}^{(m)} = \begin{bmatrix} x_1^{(m)} \\ x_2^{(m)} \\ \vdots \\ x_{NQ+W-1}^{(m)} \end{bmatrix}. \tag{4}$$

Notice that $\boldsymbol{H}^{(k,m)} \in \mathbb{C}^{(NQ+W-1)\times(NQ)}$ is a Toeplitz matrix that contains estimates of the channel impulse response (CIR) vector $\boldsymbol{h}^{(k,m)} \in \mathbb{C}^W$ of the kth user (corresponding to the kth spreading code) at the mth antenna. By comparing (3) with Figure 3, we can identify $\boldsymbol{s}^{(k,\ell)} = \boldsymbol{C}^{(k)} \cdot \boldsymbol{d}^{(k,\ell)}, \ell = 1, 2$, where the parameter ℓ determines whether the first or the second data block of a time slot is treated. In the sequel, we will omit the parameter ℓ for notational convenience.

Using the definition of $\boldsymbol{B}^{(k,m)}$ in (3), $\boldsymbol{x}^{(m)}$ may be expressed as

$$\boldsymbol{x}^{(m)} = \sum_{k=1}^{K} \boldsymbol{B}^{(k,m)} \boldsymbol{d}^{(k)} + \boldsymbol{n}^{(m)} = \tag{5}$$



$$= \sum_{k=1}^{K} \cdot \boldsymbol{d}^{(k)} + \boldsymbol{n}^{(m)}, \quad 1 \le m \le M,$$

where $\boldsymbol{B}^{(k,m)} \in \mathbb{C}^{(NQ+W-1)\times N}$ is a block-Toeplitz matrix consisting of combined CIR vectors $\boldsymbol{b}^{(k,m)}$ that can be expressed as the convolution of the channel impulse response (CIR) vector

$\boldsymbol{h}^{(k,m)}$ with the corresponding spreading code $\boldsymbol{c}^{(k)}$, i.e.,

$$\boldsymbol{b}^{(k,m)} = \begin{bmatrix} b_1^{(k,m)} \\ b_2^{(k,m)} \\ \vdots \\ b_{Q+W-1}^{(k,m)} \end{bmatrix} = \boldsymbol{h}^{(k,m)} * \boldsymbol{c}^{(k)} \in \mathbb{C}^{Q+W-1}, \quad 1 \leq k \leq K, \quad 1 \leq m \leq M. \tag{6}$$

The combined CIR vectors $\boldsymbol{b}^{(k,m)}$ of the kth user at all M antennas can be simplified to a single combined CIR vector in the space-time domain

$$\boldsymbol{b}^{(k)} = \text{vec}\left\{ \begin{bmatrix} \boldsymbol{b}^{(k,1)T} \\ \boldsymbol{b}^{(k,2)T} \\ \vdots \\ \boldsymbol{b}^{(k,M)T} \end{bmatrix} \right\} = \text{vec}\left\{ \begin{bmatrix} b_1^{(k,1)} & b_2^{(k,1)} & \cdots & b_{Q+W-1}^{(k,1)} \\ b_1^{(k,2)} & b_2^{(k,2)} & \cdots & b_{Q+W-1}^{(k,2)} \\ \vdots & \vdots & \cdot & \vdots \\ b_1^{(k,M)} & b_2^{(k,M)} & \cdots & b_{Q+W-1}^{(k,M)} \end{bmatrix} \right\} \in \mathbb{C}^{M(Q+W-1)}, \tag{7}$$

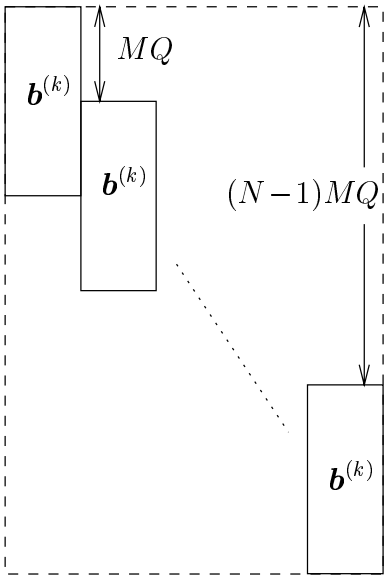
$1 \leq k \leq K$. Here, vec$\{\boldsymbol{A}\}$ denotes the reshaping of a matrix into a vector, such that the first elements of the vector are formed by the first column of the matrix, the next elements by the second column, and so on. Moreover, let us define the space-time array measurement vector (during one half burst) as

$$\boldsymbol{x} = \text{vec}\left\{ \begin{bmatrix} \boldsymbol{x}^{(1)T} \\ \boldsymbol{x}^{(2)T} \\ \vdots \\ \boldsymbol{x}^{(M)T} \end{bmatrix} \right\} = \text{vec}\left\{ \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_{NQ+W-1}^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_{NQ+W-1}^{(2)} \\ \vdots & \vdots & \cdot & \vdots \\ x_1^{(M)} & x_2^{(M)} & \cdots & x_{NQ+W-1}^{(M)} \end{bmatrix} \right\} \in \mathbb{C}^{M(NQ+W-1)}. \tag{8}$$

Using equation (5), the vector $\boldsymbol{x}$ may then be expressed as

$$\boldsymbol{x} = \sum_{k=1}^{K} \boldsymbol{B}^{(k)} \boldsymbol{d}^{(k)} + \boldsymbol{n} \tag{9}$$

$$= \sum_{k=1}^{K} \cdot \boldsymbol{d}^{(k)} + \boldsymbol{n}, \tag{10}$$



where the matrix $\boldsymbol{B}^{(k)}$ is constructed from $\boldsymbol{b}^{(k)}$ in the same way as $\boldsymbol{B}^{(k,m)}$ is constructed from $\boldsymbol{b}^{(k,m)}$. In equation (9), the space-time vector

$$\boldsymbol{n} = \text{vec}\left\{\begin{bmatrix} \boldsymbol{n}^{(1)T} \\ \boldsymbol{n}^{(2)T} \\ \vdots \\ \boldsymbol{n}^{(M)T} \end{bmatrix}\right\} = \text{vec}\left\{\begin{bmatrix} n_1^{(1)} & n_2^{(1)} & \cdots & n_{NQ+W-1}^{(1)} \\ n_1^{(2)} & n_2^{(2)} & \cdots & n_{NQ+W-1}^{(2)} \\ \vdots & \vdots & \cdot & \vdots \\ n_1^{(M)} & n_2^{(M)} & \cdots & n_{NQ+W-1}^{(M)} \end{bmatrix}\right\} \in \mathbb{C}^{M(NQ+W-1)}. \tag{11}$$

models inter-cell interference, i.e., dominant interferers from adjacent cells, and additive (thermal) noise.

It is desirable to simplify the explicit summation over all users in equation (9) to a single matrix vector product such that we get the final form of the system equation:

$$\boldsymbol{x} = \boldsymbol{T}\boldsymbol{d} + \boldsymbol{n}. \tag{12}$$

However, there are multiple ways to combine the matrices $\boldsymbol{B}^{(k)}$ for all K users into the single system matrix $\boldsymbol{T}$. This arrangement dictates the sequence in which the estimated data symbols become available during the joint detection process, as we will see below. The arrangement that leads to a low computational complexity and to the approximation opportunities that have been

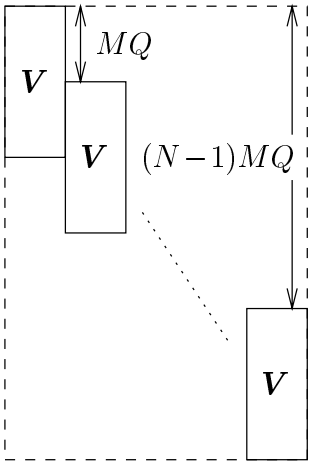
exploited in [6] does not offer the right sequence to properly introduce decision feedback into the joint detector. We will first construct $\boldsymbol{T}$ such that the resulting cmputational complexity is minimized. Later, after introducing decision feedback itself, we will show the construction of a slightly different system matrix $\tilde{\boldsymbol{T}}$ that allows decision feedback to be more effective.

Using the definition of $\boldsymbol{d}^{(k)} \in \mathbb{C}^N$ in (1), the transmitted data symbols of all K users are combined in the following fashion,

$$\boldsymbol{d} = \text{vec}\left\{\begin{bmatrix} \boldsymbol{d}^{(1)T} \\ \boldsymbol{d}^{(2)T} \\ \vdots \\ \boldsymbol{d}^{(K)T} \end{bmatrix}\right\} = \text{vec}\left\{\begin{bmatrix} d_1^{(1)} & d_2^{(1)} & \cdots & d_N^{(1)} \\ d_1^{(2)} & d_2^{(2)} & \cdots & d_N^{(2)} \\ \vdots & \vdots & \cdot & \vdots \\ d_1^{(K)} & d_2^{(K)} & \cdots & d_N^{(K)} \end{bmatrix}\right\} = \begin{bmatrix} \boldsymbol{d}_1 \\ \boldsymbol{d}_2 \\ \vdots \\ \boldsymbol{d}_N \end{bmatrix} \in \mathbb{C}^{NK}. \quad (13)$$

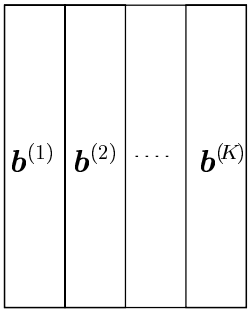
This leads to the follwing construction of $\boldsymbol{T}$

$$\boldsymbol{T} = \in \mathbb{C}^{M(NQ+W-1)\times NK}. \quad (14)$$



where the matrix

$$\boldsymbol{V} = \in \mathbb{C}^{M(Q+W-1)\times K} \quad (15)$$



contains the combined CIR vectors of all K users in the space-time domain.

III. Joint Data Detection via Block Linear Equalization

Given the linear space-time data model in equation (12), we want to find a linear estimate of the N data symbols transmitted by each of the K users during the duration of one block (half burst), i.e., a *block linear equalizer*, such that

$$\hat{\boldsymbol{d}} = \boldsymbol{W}^H \boldsymbol{x} = \begin{bmatrix} \boldsymbol{w}_1^H \\ \boldsymbol{w}_2^H \\ \vdots \\ \boldsymbol{w}_{NK}^H \end{bmatrix} \boldsymbol{x} \in \mathbb{C}^{NK}. \tag{16}$$

In the sequel, two alternative solutions are presented.

A. Least-Squares Solution

In the first case, we choose the space-time weighting matrix $\boldsymbol{W}^H \in \mathbb{C}^{(NK)\times M(NQ+W-1)}$ in (16) such that the Euclidean norm of the error

$$\|\boldsymbol{x} - \boldsymbol{T} \cdot \boldsymbol{d}\|_2^2$$

is minimized. It is given by the the standard least squares (LS) solution, where $\boldsymbol{W}^H$ is equal to the Moore-Penrose pseudo inverse (generalized inverse) of $\boldsymbol{T}$, i.e.,

$$\hat{\boldsymbol{d}} = \boldsymbol{W}^H \boldsymbol{x} = \boldsymbol{T}^+ \boldsymbol{x} = \left(\boldsymbol{T}^H \boldsymbol{T}\right)^{-1} \boldsymbol{T}^H \cdot \boldsymbol{x}. \tag{17}$$

Notice that the LS solution does not take inter-cell interference into account, i.e., strong interferers from adjacent cells are not canceled.

B. Weighted Least-Squares Solution

Alternatively, we want to find NK space-time weight vectors $\boldsymbol{w}_i^H$ that minimize the variance of the estimated data symbols $\boldsymbol{w}_i = \text{argmin}_{\boldsymbol{w}_i} \, \text{E}\{|\hat{d}_n^{(k)}|^2\}$ with

$$\hat{d}_n^{(k)} = \boldsymbol{w}_i^H \boldsymbol{x}, \quad 1 \le n \le N, \quad 1 \le k \le K, \quad i = k + (n-1)K,$$

such that

$$\boldsymbol{W}^H \boldsymbol{T} = \begin{bmatrix} \boldsymbol{w}_1^H \\ \boldsymbol{w}_2^H \\ \vdots \\ \boldsymbol{w}_{NK}^H \end{bmatrix} \boldsymbol{T} = \boldsymbol{I}_{NK}. \tag{18}$$

In the literature, this approach is called minimum variance distortionless response (MVDR) solution, zero-forcing block linear equalizer, linear minimum variance unbiased estimate, weighted least squares estimate, or Gauss-Markov estimate [8]. The solution of this constrained optimization problem is given by

$$\hat{\boldsymbol{d}} = \boldsymbol{W}^H \boldsymbol{x} = \left(\boldsymbol{T}^H \boldsymbol{R}_{nn}^{-1} \boldsymbol{T} \right)^{-1} \boldsymbol{T}^H \boldsymbol{R}_{nn}^{-1} \cdot \boldsymbol{x}, \tag{19}$$

where we assume that the space-time covariance matrix of the inter-cell-interference-plus-noise $\boldsymbol{R}_{nn} = \mathrm{E}\{\boldsymbol{n}\boldsymbol{n}^H\}$ is non-singular.[1] It contains contributions from the dominant interferers form adjacent cells and from the additive noise. Notice that the weighted least-squares solution (19) simplifies to the standard least-squares solution (17) if $\boldsymbol{R}_{nn} = \sigma_N^2 \boldsymbol{I}_{M(NQ+W-1)}$. Furthermore, the solution given in (19) minimizes the following weighted norm of the error

$$\| \boldsymbol{L}_{nn} (\boldsymbol{x} - \boldsymbol{T} \cdot \boldsymbol{d}) \|_2^2$$

where $\boldsymbol{L}_{nn}$ is a square root of $\boldsymbol{R}_{nn}^{-1}$ such that

$$\boldsymbol{L}_{nn}^H \boldsymbol{L}_{nn} = \boldsymbol{R}_{nn}^{-1}.$$

This relation can be used to easily extend an algorithm that computes the least-squares solution to one that computes the weighted least-squares solution. Alternatively, one could use the algorithm due to Paige to solve the *generalized least squares* problem directly [9].

C. Decision Feedback

As we will see in the sequel, the last step of the proposed estimation procedure is the solution of a triangular system of linear equations, i.e.

$$\boldsymbol{R}\boldsymbol{d} = \boldsymbol{z}, \tag{20}$$

[1] It can be shown that the MVDR solution also minimizes the variance of the estimation error $\mathrm{E}\{|e_n^{(k)}|^2\}$ subject to (18), where $e_n^{(k)}$ is defined as $e_n^{(k)} = d_n^{(k)} - \hat{d}_n^{(k)} = d_n^{(k)} - \boldsymbol{w}_i^H \boldsymbol{x}, 1 \le k \le K, 1 \le n \le N, i = k + (n-1)K$.

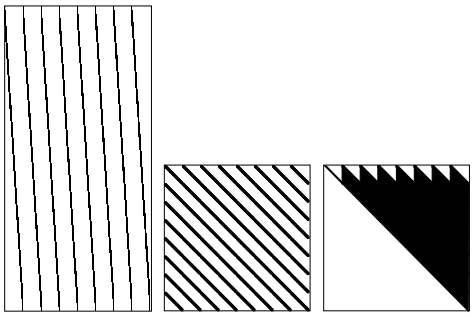
where $\boldsymbol{R} \in \mathbb{C}^{NK \times NK}$ is upper triangular. Equation (20) can easily be solved via back-substitution. It has been proposed to include decision feedback into this process to improve the bit error rate performance of the transmission system [1], [2]. Figure 4 shows the procedure used for solving (20) with decision feedback. The function $\mathtt{map}(x)$ maps x to the nearest member of the symbol alphabet, i.e., it implements the decision. As can be seen, the first decision influences all symbols contained in $\boldsymbol{d}$. Therefore, it would be advantageous to arrange $\boldsymbol{d}$ in such a way that the symbols of the strongest user are found in the last elements of $\boldsymbol{d}$. Let us denote this new arrangement of the data symbols of all users by $\tilde{\boldsymbol{d}}$ and the corresponding system matrix by $\tilde{\boldsymbol{T}}$. Then we have

$$\tilde{\boldsymbol{d}} = \begin{bmatrix} \boldsymbol{d}^{(1)} \\ \boldsymbol{d}^{(2)} \\ \vdots \\ \boldsymbol{d}^{(K)} \end{bmatrix} \quad \text{and} \quad \tilde{\boldsymbol{T}} = \begin{bmatrix} \boldsymbol{B}^{(1)} & \boldsymbol{B}^{(2)} & \cdots & \boldsymbol{B}^{(K)} \end{bmatrix}. \tag{21}$$

The structure of $\tilde{\boldsymbol{T}}$ and related matrices is depicted in Figure 5. For comparison, Figure 6 shows the structures of the corresponding matrices for $\boldsymbol{T}$ as defined in (14). As can be seen, the matrices $\boldsymbol{T}$ and $\boldsymbol{T}^H\boldsymbol{T}$ have a *block-Sylvester* structure and the strong band structure is inherited by the Cholesky factor $\boldsymbol{R}$. On the other hand, $\tilde{\boldsymbol{T}}$ and $\tilde{\boldsymbol{T}}^H\tilde{\boldsymbol{T}}$ are *Sylvester-block* structured and the Cholesky factor $\tilde{\boldsymbol{R}}$ does not inherit their sparseness. The latter fact is the main reason why the previously used Cholesky algorithm can not be sucessfully applied to the decision feedback problem.

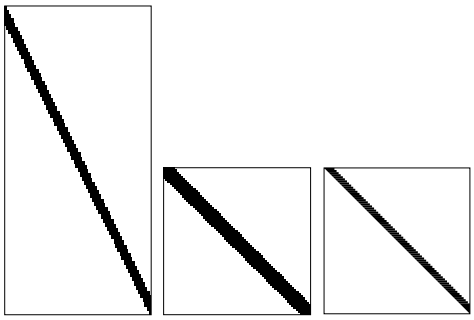
for i **from** n **down to** 1
  $\boldsymbol{d}(i) \leftarrow \big(\boldsymbol{z}(i) - \sum_{j=i+1}^{n} \boldsymbol{R}(i,j)\boldsymbol{t}(j)\big) \big/ \boldsymbol{R}(i,i)$
  $\boldsymbol{t}(i) \leftarrow \mathtt{map}(\boldsymbol{d}(i))$
end

Fig. 4. Performing decision feedback while back substituting. The function $\mathtt{map}(x)$ maps x to the nearest member of the symbol alphabet, i.e., it implements the decision.



(a) $\tilde{\boldsymbol{T}}$ (b) $\tilde{\boldsymbol{T}}^H\tilde{\boldsymbol{T}}$ (c) $\tilde{\boldsymbol{R}}$

Fig. 5. The sparseness of the matrices for decision feedback. The black regions represent non-zero matrix elements. In the depicted example, the following parameters were chosen: $N = 69$, $K = 8$, $Q = 16$, $W = 60$



(a) $\boldsymbol{T}$ (b) $\boldsymbol{T}^H\boldsymbol{T}$ (c) $\boldsymbol{R}$

Fig. 6. The sparseness of the compact matrices. The black regions represent non-zero matrix elements. In the depicted example, the following parameters were chosen: $N = 69$, $K = 8$, $Q = 16$, $W = 60$

IV. The Schur Algorithm

The Schur algorithm is a way to efficiently compute the QR decomposition of a Toeplitz-structured matrix. Such a QR decomposition can be used to find the solution (17) of the least squares problem [9]. In the following, we will outline the general Schur algorithm for matrices that have some kind of Toeplitz derived structure. As we go along, we will use the insights to solve (17) for both the block-Toeplitz matrix $\boldsymbol{T}$ in (14), as well as the Toeplitz-block matrix $\tilde{\boldsymbol{T}}$. In the following we will therefore not distinguish between $\boldsymbol{T}$ and $\tilde{\boldsymbol{T}}$. Instead, we will use the general matrix $\boldsymbol{U}$ representing both.

The QR decomposition of $\boldsymbol{U} \in \mathbb{C}^{n\times m}$ consists of finding two matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ such that

$$\boldsymbol{U} = \boldsymbol{Q}\boldsymbol{R} \tag{22}$$

where $\boldsymbol{Q} \in \mathbb{C}^{n\times m}$ is a unitary matrix, and $\boldsymbol{Q}^H\boldsymbol{Q} = \boldsymbol{I}$. The matrix $\boldsymbol{R} \in \mathbb{C}^{m\times m}$ is upper triangular and is also known as the Cholesky factor of $\boldsymbol{U}^H\boldsymbol{U}$. It can be used to find the vector $\boldsymbol{a}$ that minimizes

$$\|\boldsymbol{U}\boldsymbol{a} - \boldsymbol{b}\|_2 \tag{23}$$

by solving the triangular system of equations

$$\boldsymbol{R}\boldsymbol{a} = \boldsymbol{Q}^H\boldsymbol{b}. \tag{24}$$

It should be noted that although $\boldsymbol{U}$ is Toeplitz structured, the resulting matrix $\boldsymbol{R}$ does not inherit this structure. Also, it is possible to compute the right hand side $\boldsymbol{Q}^H\boldsymbol{b}$ in parallel to computing $\boldsymbol{R}$. Therefore, we will concentrate on a procedure to compute $\boldsymbol{R}$ and will later show how to extend it to compute $\boldsymbol{Q}^H\boldsymbol{b}$ as well.

The most general form of the Schur algorithm [10] starts from a *displacement representation* of the Gramian matrix $\boldsymbol{U}^H\boldsymbol{U}$ that is easy to obtain and then proceeds to transform this decomposition into $\boldsymbol{R}$ via *orthogonal* and *hyperbolic rotations* [11], [12]. The displacement representation consists of finding a way to exploit the whole structure of a matrix. For the positive definite, Hermitian matrix $\boldsymbol{S} = \boldsymbol{U}^H\boldsymbol{U}$, the displacement representation can be expressed as

$$\boldsymbol{S} - \boldsymbol{Z}\boldsymbol{S}\boldsymbol{Z}^T = \sum_{i=1}^{L} \boldsymbol{\alpha}_i^H\boldsymbol{\alpha}_i - \sum_{i=1}^{L} \boldsymbol{\beta}_i^H\boldsymbol{\beta}_i. \tag{25}$$

The matrix $\boldsymbol{Z}$ is a *shift matrix* that has been constructed to reflect the Toeplitz structure of $\boldsymbol{S}$. It is different depending on whether we want to invert $\boldsymbol{T}$ or $\tilde{\boldsymbol{T}}$.

The vectors $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ are $1 \times m$ row vectors, i.e., the outer product $\boldsymbol{\alpha}_i^H\boldsymbol{\alpha}_i$ is a $m \times m$ matrix. These vectors are called the *generators* of $\boldsymbol{S}$.

The advantage of the Schur algorithm for structured matrices is that it is sufficient to work with the generators $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ (instead of the full matrix $\boldsymbol{S}$) to compute $\boldsymbol{R}$. The parameter L determines how many vector pairs are needed to express $\boldsymbol{S}$. It, too, depends on the precise Toeplitz structure of $\boldsymbol{S}$. The number $2L$ is called the *displacement rank* of $\boldsymbol{S}$.

A. Finding the generators

The first step in finding the generators $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ is to choose an appropriate shift matrix $\boldsymbol{Z}$. Such a matrix should make $\boldsymbol{D} = \boldsymbol{S} - \boldsymbol{Z}\boldsymbol{S}\boldsymbol{Z}^T$ as sparse as possible. In addition, the transformation $\boldsymbol{S} \rightarrow \boldsymbol{D}$ should only introduce zeros but should not change the remaining elements of $\boldsymbol{S}$. The last condition is required to ensure that the benefits gained from the fact that $\boldsymbol{S}$ is positive semi-definite are not lost. That requirement also means that $\boldsymbol{D}$ does not need to be computed explicitely since the introduced zeros are predictable and the rest can be read directly from $\boldsymbol{S}$.

Once a suitable $\boldsymbol{Z}$ has been chosen, we need to compute one generator pair $(\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i)$ for each row that does not contain only zeros. For row j, the generators are computed according to

$$\boldsymbol{\alpha}_i = \boldsymbol{D}(j,:)/\sqrt{\boldsymbol{D}(j,j)}, \quad \boldsymbol{\beta}_i = \boldsymbol{\alpha}_i \text{ except } \boldsymbol{\beta}_i(j) = 0 \tag{26}$$

After the generator pair has been computed for such a row, its contribution is removed from $\boldsymbol{D}$ before computing the next generator pair.

For the two specific matrices $\boldsymbol{T}$ and $\tilde{\boldsymbol{T}}$, Figure 7 and 8 show which shift matrix to use and how to compute the generators, respectively. The definition of such functions uses a Matlab inspired notation. The arrow $\leftarrow$ denotes assignment: the value on the right side of it is computed and then assigned to the entity denoted on the left hand side. Matrix and vector subscription is denoted as $\boldsymbol{X}(r, c)$ where r selects rows of $\boldsymbol{X}$ and c selects columns. Both r and c are ranges. A range can have the following forms with the indicated meaning:

a	The single index a.
a:e	All indices from a to e inclusive.
a:s:e	All indices from a to e inclusive with a step size of s.
:	All indices that are valid as determined by the context.

The operator $\otimes$ denotes the Kronecker product defined as

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11} & a_{12} & \cdots \\ a_{21} & a_{22} & \\ \vdots & & \ddots \end{bmatrix} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & a_{12}\boldsymbol{B} & \cdots \\ a_{21}\boldsymbol{B} & a_{22}\boldsymbol{B} & \\ \vdots & & \ddots \end{bmatrix}.$$

The matrix $\boldsymbol{I}_K$ is the $K \times K$ identity matrix and $\boldsymbol{Z}_N^{(1)}$ is the $N \times N$ *unit shift matrix*

$$\boldsymbol{Z}_N^{(1)} = \begin{bmatrix} 0 & & & & \\ 1 & 0 & & & \\ 0 & 1 & 0 & & \\ \vdots & & \ddots & \ddots & \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}. \tag{27}$$

$\boldsymbol{Z} \leftarrow \boldsymbol{Z}_N^{(1)} \otimes \boldsymbol{I}_K$
$\boldsymbol{S} \leftarrow \boldsymbol{T}^H \boldsymbol{T}$
$\boldsymbol{D} \leftarrow \boldsymbol{S} - \boldsymbol{Z}\boldsymbol{S}\boldsymbol{Z}^T$
for $i = 1 : K$
  $\boldsymbol{\alpha}_i \leftarrow \boldsymbol{D}(i,:) / \sqrt{\boldsymbol{D}(i,i)}$
  $\boldsymbol{\beta}_i \leftarrow \boldsymbol{\alpha}_i$
  $\boldsymbol{\beta}_i(i) \leftarrow 0$
  $\boldsymbol{D}(:,i) \leftarrow \boldsymbol{0}$
end

Fig. 7. Computing the generators for $\boldsymbol{T}$

$\boldsymbol{Z} \leftarrow \boldsymbol{I}_K \otimes \boldsymbol{Z}_N^{(1)}$
$\boldsymbol{S} \leftarrow \tilde{\boldsymbol{T}}^H \tilde{\boldsymbol{T}}$
$\boldsymbol{D} \leftarrow \boldsymbol{S} - \boldsymbol{Z}\boldsymbol{S}\boldsymbol{Z}^T$
for $i = 1 : K$
  $j \leftarrow (i-1)N + 1$
  $\boldsymbol{\alpha}_i \leftarrow \boldsymbol{D}(j,:) / \sqrt{\boldsymbol{D}(j,j)}$
  $\boldsymbol{\beta}_i \leftarrow \alpha_i$
  $\boldsymbol{\beta}_i(j) \leftarrow 0$
  $\boldsymbol{D}(:,j) \leftarrow \boldsymbol{0}$
end

Fig. 8. Computing the generators for $\tilde{\boldsymbol{T}}$

B. Finding $\boldsymbol{R}$ from the generators

The displacement representation (25) can be rewritten to directly express $\boldsymbol{S}$ by introducing *Krylov* matrices. The Krylov matrix $\mathcal{U}(\boldsymbol{\xi}, \boldsymbol{Z})$ of a row vector $\boldsymbol{\xi}$ with respect to $\boldsymbol{Z}$ is (for our purposes) a $m \times m$ matrix defined as

$$\mathcal{U}(\boldsymbol{\xi}, \boldsymbol{Z}) = \begin{bmatrix} \boldsymbol{\xi} \\ \boldsymbol{\xi Z}^T \\ \vdots \\ \boldsymbol{\xi Z}^{(m-1)T} \end{bmatrix}. \tag{28}$$

Note that $\boldsymbol{\mathcal{U}}(\boldsymbol{\xi}, \boldsymbol{Z})$ is an upper triangular matrix. In the sequel we omit the $\boldsymbol{Z}$ argument to the Krylov constructor for simplicity. With this definition, (25) can be rewritten as

$$\boldsymbol{S} = \sum_{i=1}^{L} \mathcal{U}(\boldsymbol{\alpha}_i)^H \mathcal{U}(\boldsymbol{\alpha}_i) - \sum_{i=1}^{L} \mathcal{U}(\boldsymbol{\beta}_i)^H \mathcal{U}(\boldsymbol{\beta}_i). \tag{29}$$

The goal of the Schur algorithm is now to gradually eliminate the contributions of $\mathcal{U}(\boldsymbol{\alpha}_2)$, $\mathcal{U}(\boldsymbol{\alpha}_3)$, ..., $\mathcal{U}(\boldsymbol{\alpha}_L)$ and $\mathcal{U}(\boldsymbol{\beta}_1)$, $\mathcal{U}(\boldsymbol{\beta}_2)$, ..., $\mathcal{U}(\boldsymbol{\beta}_L)$ so that only the first term of the first sum remains. The Schur algorithm preserves the upper triangular structure of the matrices. This transformation can be summarized as follows

$$\begin{aligned} &\mathcal{U}(\boldsymbol{\alpha}_1) \rightarrow \boldsymbol{R}, && \boldsymbol{R} \text{ upper triangular} \\ &\mathcal{U}(\boldsymbol{\alpha}_i) \rightarrow \boldsymbol{0}, && 1 < i \leq L \\ &\mathcal{U}(\boldsymbol{\beta}_i) \rightarrow \boldsymbol{0}, && 1 \leq i \leq L \end{aligned} \tag{30}$$

It can be seen that with this transformation, the remaining matrix $\boldsymbol{R}$ must indeed be the $\boldsymbol{R}$ factor of the QR decomposition of $\boldsymbol{U}$:

$$\boldsymbol{S} = \boldsymbol{U}^H \boldsymbol{U} = \boldsymbol{R}^H \boldsymbol{Q}^H \boldsymbol{Q} \boldsymbol{R} = \boldsymbol{R}^H \boldsymbol{R}. \tag{31}$$

To derive the actual Schur algorithm, we rewrite (29) into the following form:

$$\boldsymbol{S} = \boldsymbol{X}^H \boldsymbol{J} \boldsymbol{X} \tag{32}$$

where

$$\boldsymbol{X} = \begin{bmatrix} \mathcal{U}(\boldsymbol{\alpha}_1) \\ \mathcal{U}(\boldsymbol{\alpha}_2) \\ \vdots \\ \mathcal{U}(\boldsymbol{\alpha}_L) \\ \mathcal{U}(\boldsymbol{\beta}_1) \\ \mathcal{U}(\boldsymbol{\beta}_2) \\ \vdots \\ \mathcal{U}(\boldsymbol{\beta}_L) \end{bmatrix} \in \mathbb{C}^{2mL \times m} \text{ and } \boldsymbol{J} = \begin{bmatrix} \boldsymbol{I} & & & & & & \\ & \boldsymbol{I} & & & & & \\ & & \ddots & & & & \\ & & & \boldsymbol{I} & & & \\ & & & & -\boldsymbol{I} & & \\ & & & & & -\boldsymbol{I} & \\ & & & & & & \ddots \\ & & & & & & -\boldsymbol{I} \end{bmatrix} \in \mathbb{C}^{2mL \times 2mL}. \quad (33)$$

Performing the transformation (30) can now be restated as finding a matrix $\boldsymbol{\Theta} \in \mathbb{C}^{2mL \times 2mL}$ such that

$$\boldsymbol{\Theta}\boldsymbol{X} = \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{bmatrix} \text{ and } \boldsymbol{\Theta}^H \boldsymbol{J} \boldsymbol{\Theta} = \boldsymbol{J}. \quad (34)$$

The complete transformation $\boldsymbol{\Theta}$ is constructed from *elementary rotations* $\boldsymbol{\Theta}_i$ each introducing one zero into $\boldsymbol{X}$. All transformations are applied from the left to $\boldsymbol{X}$ such that

$$\boldsymbol{\Theta} = \cdots \boldsymbol{\Theta}_3 \boldsymbol{\Theta}_2 \boldsymbol{\Theta}_1. \quad (35)$$

The condition $\boldsymbol{\Theta}^H \boldsymbol{J} \boldsymbol{\Theta} = \boldsymbol{J}$ is needed to ensure that the transformation preserves the validity of (32). When this condition holds, $\boldsymbol{\Theta}$ is said to be $\boldsymbol{J}$*-orthogonal*. The easiest way to enforce $\boldsymbol{\Theta}$ to be $\boldsymbol{J}$-orthogonal is to construct each elementary transformation so that it is $\boldsymbol{J}$-orthogonal as well.

The general structure of a elementary transformation that introduces one zero into $\boldsymbol{X}$ is

$$\boldsymbol{\Theta}_i = \mathcal{E}(\boldsymbol{K}, k, l) = \begin{bmatrix} & & k & & l & & \\ 1 & & 0 & & 0 & & 0 \\ & \ddots & \vdots & & \vdots & & \\ 0 & \cdots & k_{11} & \cdots & k_{12} & \cdots & 0 \\ & & \vdots & \ddots & \vdots & & \\ 0 & \cdots & k_{21} & \cdots & k_{22} & \cdots & 0 \\ & & \vdots & & \vdots & \ddots & \\ 0 & & 0 & & 0 & & 1 \end{bmatrix} \begin{matrix} \\ \\ k \\ \\ l \\ \\ \\ \end{matrix} \quad \text{where } \boldsymbol{K} = \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}. \tag{36}$$

If such a transformation $\boldsymbol{\Theta}_i$ is multiplied onto $\boldsymbol{X}$ from the left, it will only affect rows k and l of $\boldsymbol{X}$. The *transformation kernel* $\boldsymbol{K}$ must be chosen so that the desired zero turns up in row l after the transformation has been applied, and so that the condition $\boldsymbol{\Theta}_i \boldsymbol{J} \boldsymbol{\Theta}_i = \boldsymbol{J}$ is satisfied. The first requirement can be expressed as

$$\boldsymbol{K} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix} \tag{37}$$

where a and b are elements from row k and row l of $\boldsymbol{X}$, respectively. After $\boldsymbol{\Theta}_i$ has been applied to $\boldsymbol{X}$, the place that contained b will be zero.

Due to the structure of $\boldsymbol{J}$ in (33) and since being $\boldsymbol{J}$-orthogonal implies being $(-\boldsymbol{J})$-orthogonal, the kernel $\boldsymbol{K}$ must satisfy one of two orthogonality constraints, depending on the choice of k and l. The transformations that correspond to these two constraints are called *unitary* and *hyperbolic*, respectively.

Unitary kernels arise if both row k and row l of $\boldsymbol{X}$ belong to the region formed by the Krylov matrices $\mathcal{U}(\boldsymbol{\alpha}_i)$, or both rows belong to the region of the $\mathcal{U}(\boldsymbol{\beta}_i)$:

$$\boldsymbol{K}^H \begin{bmatrix} 1 & \\ & 1 \end{bmatrix} \boldsymbol{K} = \begin{bmatrix} 1 & \\ & 1 \end{bmatrix}. \tag{38}$$

The unitary kernel is defined as

$$\boldsymbol{K} = \mathcal{G}(a, b) = \begin{cases} \frac{1}{\sqrt{1+|\rho|^2}} \begin{bmatrix} 1 & \rho^* \\ -\rho & 1 \end{bmatrix}, & \rho = \frac{b}{a}, \quad \text{for } a \neq 0, \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, & \text{for } a = 0 \end{cases} \tag{39}$$

The hyperbolic case arises when row k of $\boldsymbol{X}$ is in the $\mathcal{U}(\boldsymbol{\alpha}_i)$ region and row l is in the $\mathcal{U}(\boldsymbol{\beta}_i)$ region:

$$\boldsymbol{K}^H \begin{bmatrix} 1 & \\ & -1 \end{bmatrix} \boldsymbol{K} = \begin{bmatrix} 1 & \\ & -1 \end{bmatrix}. \tag{40}$$

The corresponding hyperbolic kernel is defined as

$$\boldsymbol{K} = \mathcal{H}(a,b) = \frac{1}{\sqrt{1-|\rho|^2}} \begin{bmatrix} 1 & -\rho^* \\ -\rho & 1 \end{bmatrix}, \qquad \rho = \frac{b}{a}, \quad |a| > |b|. \tag{41}$$

The next step is to find a specific sequence of elementary transformations $\boldsymbol{\Theta}_i$ that actually implements the complete transformation (34). This sequence cannot be chosen arbitrarily because care must be taken not to destroy desired zeros during one of the later elementary transformations. Figure 9 shows the first two steps for a system with $L = 2$. As one can see, the non-zero triangles in the Krylov matrices are made smaller by removing one diagonal after the other. One step consists of first using unitary rotations to cancel the diagonals within the group formed by the $\boldsymbol{\alpha}_i$, and within the groups formed by the $\boldsymbol{\beta}_i$; then, hyperbolic rotations are used to cancel the diagonal formed by $\boldsymbol{\beta}_1$ with $\boldsymbol{\alpha}_1$. After each step, the next row of the result is available.

One can deduce from Figure 9 which transformations are redundant due to the structure of the matrices and how this structure changes from step to step. All matrices except the first one — the one that originally was $\mathcal{U}(\boldsymbol{\alpha}_1)$ and that is going to be transformed to $\boldsymbol{R}$ — can be expressed by the Krylov construction (28) throughout the whole transformation. Therefore, a lot of transformations are redundant because their result is already known from applying the transformation to the first row of the Krylov matrices. The first matrix is gradually transformed to $\boldsymbol{R}$ and thus it looses its Krylov nature. However, the transformation proceeds row-wise: each step produces one more row of $\boldsymbol{R}$ because the following steps will only affect the remaining rows. The remaining rows *are* expressible by the Krylov construction. To go from one step to the next, we need to go from one row of the matrix to the next; according to (28), this can be done by shifting $\boldsymbol{\alpha}_1$ with $\boldsymbol{Z}$. It is therefore sufficient to work only with the row vectors $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ and a place to store the result $\boldsymbol{R}$ to carry out the Schur algorithm. Figure 10 repeats Figure 9 but shows only the relevant computations.

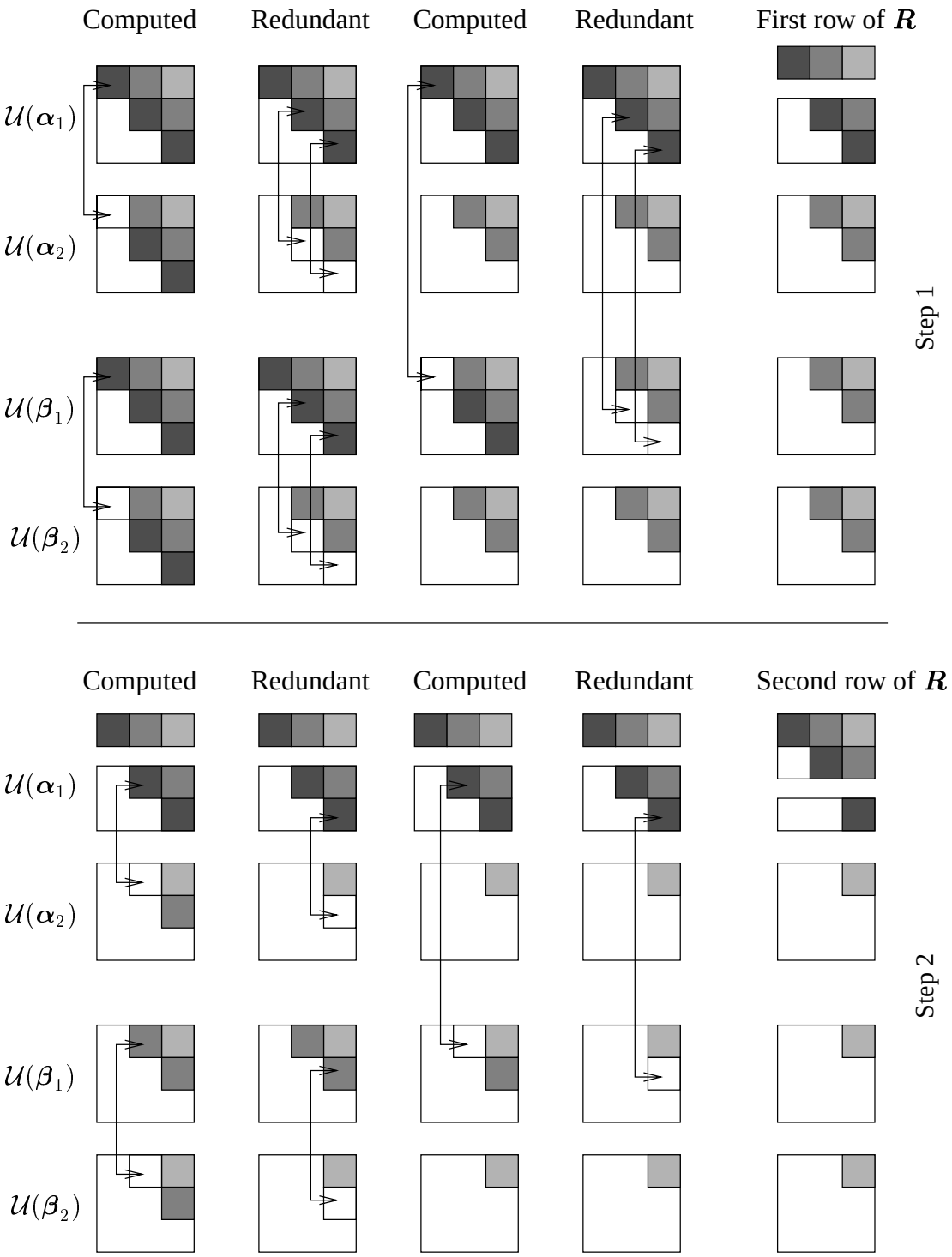


Fig. 9. The first two steps in the Schur algorithm. In this example, $L = 2$.

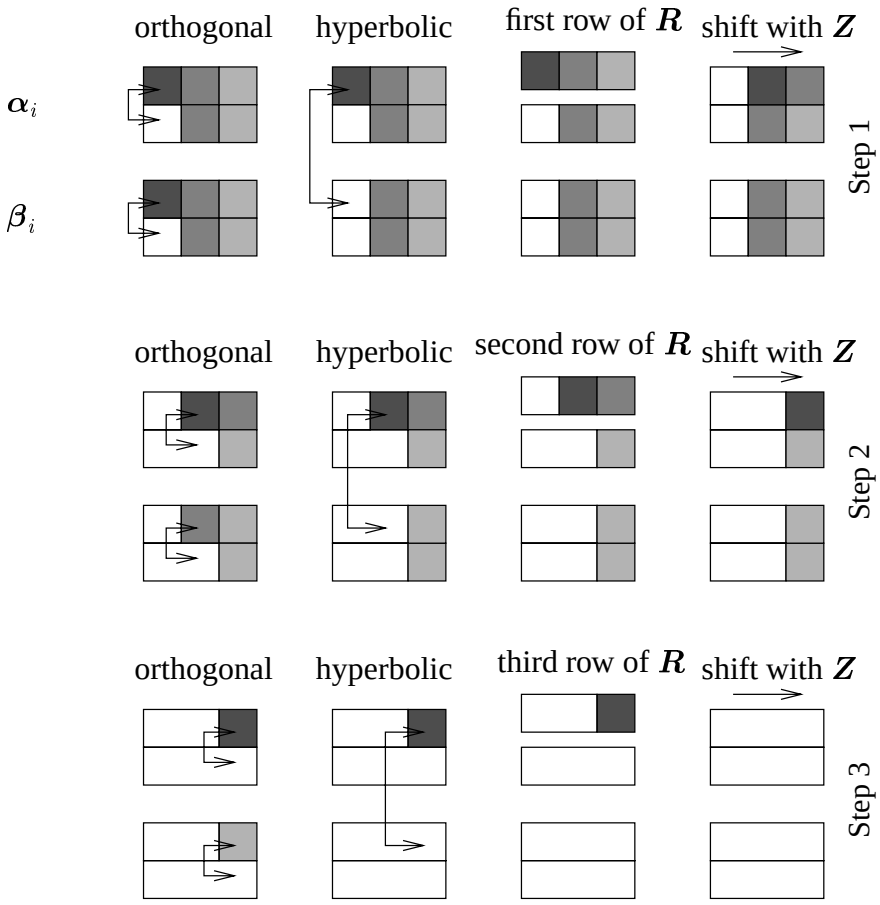


Fig. 10. The Schur algorithm where only the relevant computations are shown.

C. The right hand side

As mentioned in the beginning, the right hand side $\boldsymbol{Q}^H\boldsymbol{b}$ can be computed in parallel to computing $\boldsymbol{R}$. This is done by finding a vector $\boldsymbol{y}$ such that

$$\boldsymbol{\Theta}\boldsymbol{y} = \begin{bmatrix} \boldsymbol{q} \\ * \\ \vdots \end{bmatrix} \quad \text{and} \quad \boldsymbol{q} = \boldsymbol{Q}^H\boldsymbol{b}, \tag{42}$$

where $*$ denotes vector elements of no further interest. The vector $\boldsymbol{y}$ can be deduced from the relation

$$\boldsymbol{b}^H\boldsymbol{U} = \boldsymbol{y}^H\boldsymbol{J}\boldsymbol{X}, \tag{43}$$

since after the transformation Θ has been applied to both $\boldsymbol{X}$ and $\boldsymbol{y}$, we get

$$\boldsymbol{b}^H \boldsymbol{U} = \begin{bmatrix} \boldsymbol{q}^H & * & \cdots \end{bmatrix} \boldsymbol{J} \begin{bmatrix} \boldsymbol{R} \\ 0 \\ \vdots \end{bmatrix} = \boldsymbol{q}^H \boldsymbol{R}. \tag{44}$$

Substituting $\boldsymbol{U} = \boldsymbol{Q}\boldsymbol{R}$ into (44) shows that indeed $\boldsymbol{q} = \boldsymbol{Q}^H \boldsymbol{b}$ if $\boldsymbol{y}$ has been chosen according to (43).

There is a lot of freedom in choosing $\boldsymbol{y}$. The following restriction on the structure of $\boldsymbol{y}$ simplifies the process:

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_L \\ \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_L \end{bmatrix}. \tag{45}$$

With this construction, (43) can be expanded into

$$\boldsymbol{b}^H \boldsymbol{U} = \sum_{i=1}^{L} \boldsymbol{y}_i^H (\mathcal{U}(\boldsymbol{\alpha}_i) - \mathcal{U}(\boldsymbol{\beta}_i)). \tag{46}$$

The nice property of this formulation is that $\mathcal{U}(\boldsymbol{\alpha}_i) - \mathcal{U}(\boldsymbol{\beta}_i)$ has only columns with at most one non-zero entry, all these non-zero entries have the same value, and no $\mathcal{U}(\boldsymbol{\alpha}_j) - \mathcal{U}(\boldsymbol{\beta}_j)$ with $i \neq j$ has non-zero entries in the same columns, when $\boldsymbol{Z}$ has been chosen according to the requirements above.

Stated in another way, equation (46) can be partitioned column-wise. In addition, due to the structure of $\boldsymbol{Z}$ (both variants), only the first N elements of each $\boldsymbol{y}_i$ do really matter in the calculations. In the sequel, we will use the abbreviation $\hat{\boldsymbol{y}}_i = \boldsymbol{y}_i(1:N)$ to simplify the notation accordingly.

The column-wise partitioning depends on the precise structure of $\boldsymbol{Z}$. Therefore, we have two methods to compute the $\hat{\boldsymbol{y}}_i$, see Figure 11 and 12, respectively.

Once we have computed the relevant parts of $\boldsymbol{y}$, we need to apply Θ to them, including the parts of Θ that are redundant. We can do this by transposing each $\hat{\boldsymbol{y}}_i$ into a row vector, pasting it

for $i = 1 : K$

$\quad \hat{\boldsymbol{y}}_i = \boldsymbol{T}^H(:, i : K : (N-1)K + i)\boldsymbol{x}/\boldsymbol{\alpha}_i(i)$

end

Fig. 11. Computing the right hand-side generators for $\boldsymbol{T}$

for $i = 1 : K$

$\quad j = (i-1)N + 1$

$\quad \hat{\boldsymbol{y}}_i = \tilde{\boldsymbol{T}}^H(:, j : j + N - 1)\boldsymbol{x}/\boldsymbol{\alpha}_i(j)$

end

Fig. 12. Computing the right hand-side generators for $\tilde{\boldsymbol{T}}$

onto the right of the corresponding $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$, and then applying the elementary transformations to this longer vector. Care must be taken to properly shift $\hat{\boldsymbol{y}}_1$ when $\boldsymbol{\alpha}_1$ is shifted. Taken together, this leads to the Schur algorithm, which is depicted in Figure 13.

V. Approximations

In general, the matrix $\boldsymbol{R}$ does not inherit the structure of $\boldsymbol{S}$. However, the sparseness of $\boldsymbol{S}$ leads to an approximate Toeplitz-derived structure in $\boldsymbol{R}$ [6]. It is possible to exploit this fact by computing only the relevant parts of $\boldsymbol{R}$ and fill the rest with copies from that part. Figure 14 shows how this can be done on a row-by-row basis which is well suited for the Schur algorithm. For the decision feedback system matrix $\tilde{\boldsymbol{T}}$, the approximations are interleaved with the algorithm, i.e., the receiver actually computes the first d rows of $\boldsymbol{R}$, then copies the last of them until N rows are filled and then resumes to compute the next few rows.

Figure 15 shows the bit error rate performance of a TD-CDMA system that uses the Schur algorithm for decision feedback in the receiver applying the approximations as shown in Figure 14.

These approximations can be used to save a large amount of computations without degrading the bit error rate performance of the system.

for i **from** 1 **to** K

$\boldsymbol{y}_i^{(a)} \leftarrow \boldsymbol{y}_i^{(b)} \leftarrow \hat{\boldsymbol{y}}_i$

end

for j **from** 1 **to** NK

for i **from** 2 **to** K

$$\begin{bmatrix} \boldsymbol{\alpha}_1 \ \boldsymbol{y}_1^{(a)T} \\ \boldsymbol{\alpha}_i \ \boldsymbol{y}_i^{(a)T} \end{bmatrix} \leftarrow \mathcal{G}(\boldsymbol{\alpha}_1(j), \boldsymbol{\alpha}_i(j)) \begin{bmatrix} \boldsymbol{\alpha}_1 \ \boldsymbol{y}_1^{(a)T} \\ \boldsymbol{\alpha}_i \ \boldsymbol{y}_i^{(a)T} \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{\beta}_1 \ \boldsymbol{y}_1^{(b)T} \\ \boldsymbol{\beta}_i \ \boldsymbol{y}_i^{(b)T} \end{bmatrix} \leftarrow \mathcal{G}(\boldsymbol{\beta}_1(j), \boldsymbol{\beta}_i(j)) \begin{bmatrix} \boldsymbol{\beta}_1 \ \boldsymbol{y}_1^{(b)T} \\ \boldsymbol{\beta}_i \ \boldsymbol{y}_i^{(b)T} \end{bmatrix}$$

end

$$\begin{bmatrix} \boldsymbol{\alpha}_1 \ \boldsymbol{y}_1^{(a)T} \\ \boldsymbol{\beta}_1 \ \boldsymbol{y}_1^{(b)T} \end{bmatrix} \leftarrow \mathcal{H}(\boldsymbol{\alpha}_1(j), \boldsymbol{\beta}_1(j)) \begin{bmatrix} \boldsymbol{\alpha}_1 \ \boldsymbol{y}_1^{(a)T} \\ \boldsymbol{\beta}_1 \ \boldsymbol{y}_1^{(b)T} \end{bmatrix}$$

$\boldsymbol{R}(j,:) \leftarrow \boldsymbol{\alpha}_1$

$\boldsymbol{q}(j) \leftarrow \boldsymbol{y}_1^{(a)}(1)$

$\boldsymbol{\alpha}_1 \leftarrow \boldsymbol{\alpha}_1 \boldsymbol{Z}^T$

$\boldsymbol{y}_1^{(a)} \leftarrow \begin{bmatrix} \boldsymbol{y}_1^{(a)}(2:N) & 0 \end{bmatrix}$

end

Fig. 13. The Schur algorithm. The matrices $\boldsymbol{y}_i^{(a)}$ and $\boldsymbol{y}_i^{(b)}$ carry the right hand side through the algorithm.

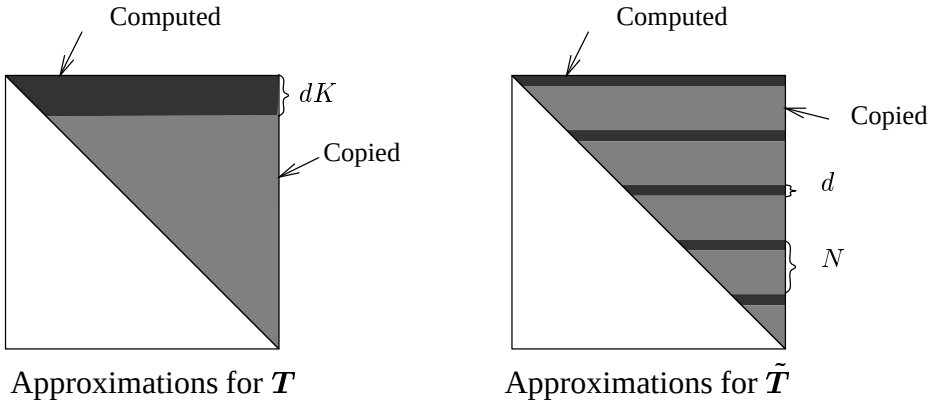


Fig. 14. Row-by-row approximation

VI. Computational Complexity

Algorithms that cannot exploit the Toeplitz structure of $\boldsymbol{S}$ have a computational complexity of $O(N^3K^3)$.[2]

The generalized Schur algorithm on the other hand can exploit the Toeplitz structure and achieves a complexity of $O(N^2K^3)$. The number of symbols per block N is generally much larger than the number of users K. By using approximations one can reduce the complexity of the decomposition to $O(NK^3)$ because the number of rows that need to be computed out of every N depends only on the degree of inter symbol interference. However, the back-substitution remains of order $O(N^2K^2)$ because there is no band-structure it could exploit in the decision feedback case.

Figures 16, 17, and 18 show the number of multiplications necessary for performing joint-detection with decision feedback for two algorithms. Note that the Figures use a logarithmic scale. The first algorithm, denoted as "Cholesky" in the legend, consists of a Cholesky decomposition of $\boldsymbol{S}$ while applying the approximations outlined in section V. The algorithm denoted by "Schur" is the one presented in this paper, featuring the same approximations. Each Figure shows the number of multiplications required to perform joint detection of one burst. Such a burst consists of two data blocks with N symbols each.

VII. Conclusions

By using a generalized Schur-type algorithm it is possible to exploit any variant of Toeplitz structure. For a system matrix $\boldsymbol{T}$ that has been arranged as in Figure 6, there is no significant difference in the computational complexity of the standard Cholesky algorithm and a Schur-type algorithm. This is due to the dominance of the band structure. However, when the decision feedback formulation is used (Figure 5), such that $\tilde{\boldsymbol{T}}$ looses its band structure, the Schur-type algorithms can exploit the remaining Toeplitz-block structure and, therefore, allow the computational complexity to be reduced by a significant amount.

Approximations allow another large reduction of the computational effort needed to solve the joint detection problem. This has been achieved by generalizing the known approximation schemes.

[2]The notation $O(f(n))$ means that the real complexity is some function $g(n)$ with $|g(n)| < C|f(n)|$ for some constant C and large enough n.

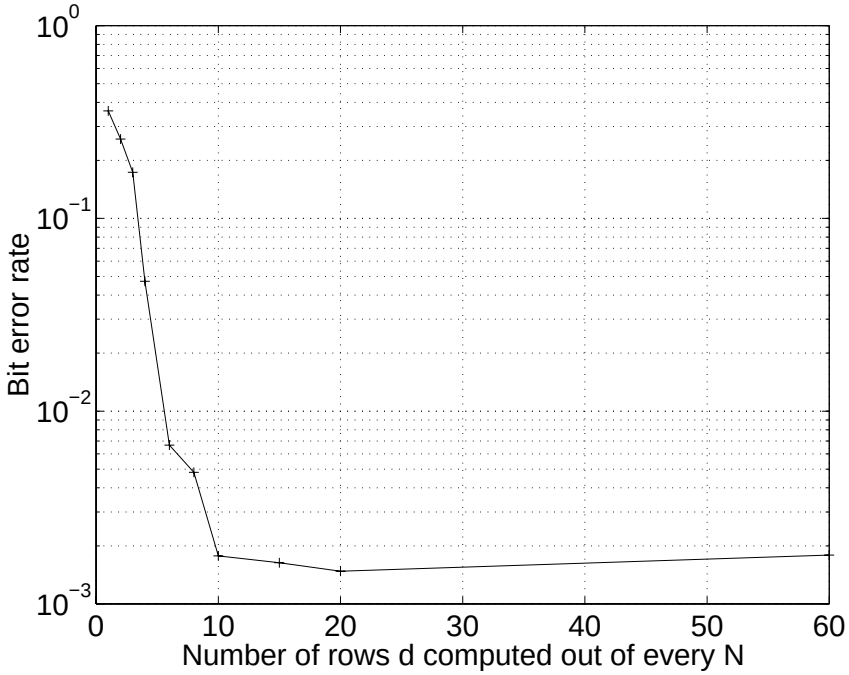


Fig. 15. Performance degradation of approximations for a varying number of computed rows, d. $K = 4$, $Q = 16$, $W = 60$, $N = 69$, $M = 1$.

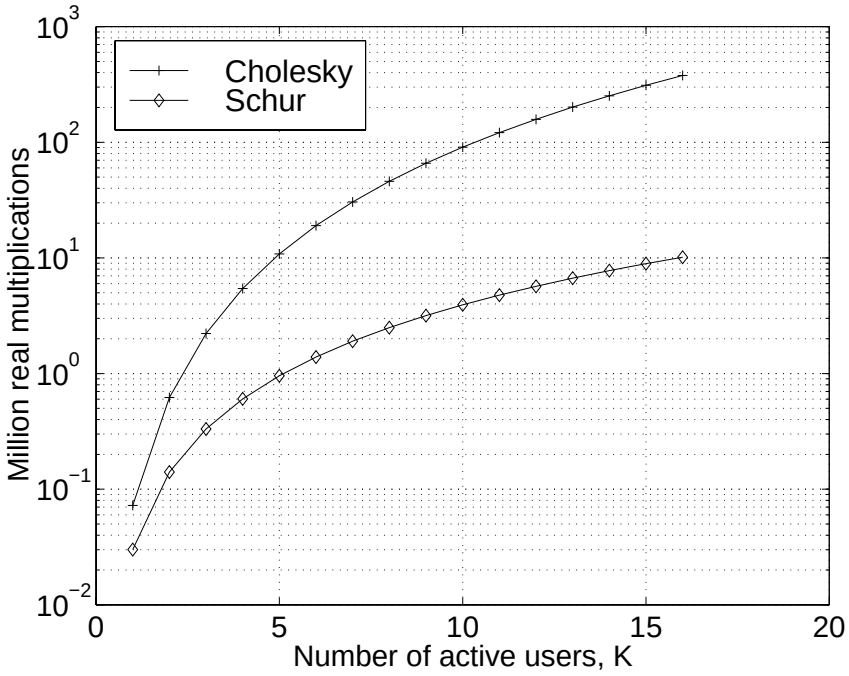


Fig. 16. Computational complexity for a varying number of active codes, K. $Q = 16$, $W = 57$, $N = 69$, $d = 10$, $M = 1$.

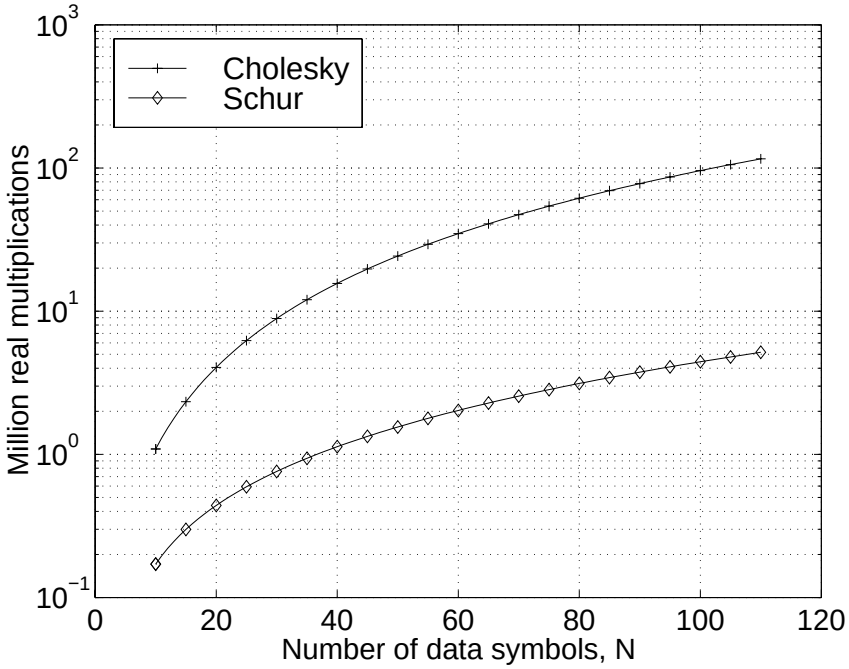


Fig. 17. Computational complexity for a varying number of symbols per data block, N. $Q = 16$, $W = 57$, $K = 8$, $d = 10$, $M = 1$.

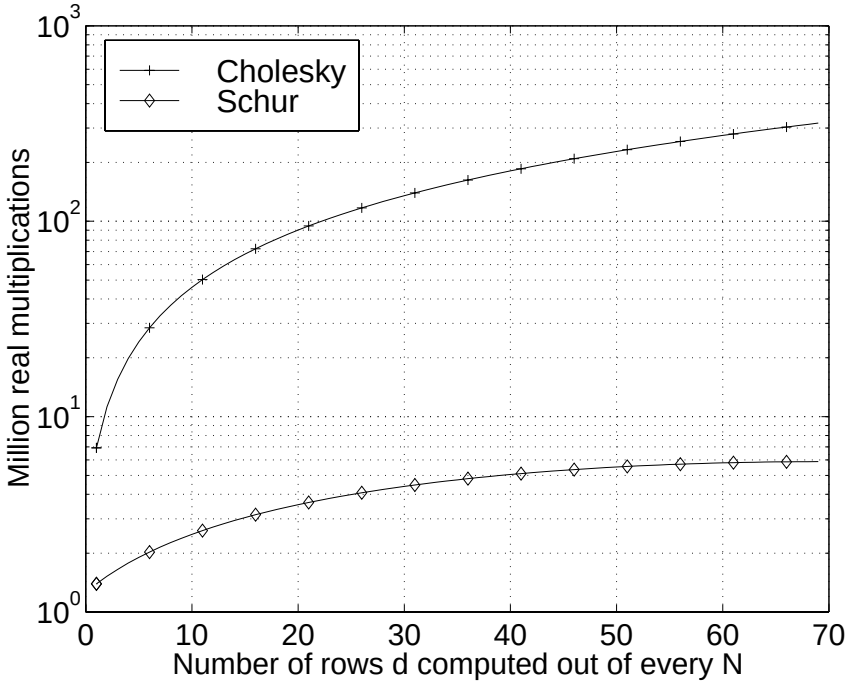


Fig. 18. Computational complexity for a varying number of computed rows d out of every N. $Q = 16$, $W = 57$, $K = 8$, $N = 69$, $M = 1$.

REFERENCES

[1] P. Jung and J. J. Blanz, "Joint detection with coherent receiver antenna diversity in CDMA mobile radio systems," *IEEE Trans. on Vehicular Technology*, vol. 44, pp. 76–88, 1995.

[2] P. Jung, J. J. Blanz, and P. W. Baier, "Coherent receiver antenna diversity for CDMA mobile radio systems using joint detection," in *Proc. 4th IEEE Int. Symp. on Personal, Indoor and Mobile Radio Commun. (PIMRC)*, Yokohama, Japan, Sept. 1993, pp. 488–492.

[3] J. J. Blanz, A. Klein, M. Naßhan, and A. Steil, "Performance of a cellular hybrid C/TDMA mobile radio system applying joint detection and coherent receiver antenna diversity," *IEEE J. Select. Areas Commun.*, vol. 12, pp. 568–579, May 1994.

[4] T. Ottosson, *Coding, Modulation and Multiuser Decoding for DS-CDMA Systems*, Ph.D. thesis, Chalmers University of Technology, Göteborg, Sweden, 1997.

[5] J. Blanz, A. Klein, M. Naßhan, and Andreas Steil, "Performance of a Cellular Hybrid C/TDMA Mobile Radio System Applying Joint Detection and Coherent Receiver Antenna Diversity," *IEEE Journal on Selected Areas in Communications*, vol. 12, no. 4, May 1994.

[6] J. Mayer, J. Schlee, and T. Weber, "Realtime feasibility of joint detection CDMA," in *Proc. 2nd European Personal Mobile Communications Conference*, Bonn, Germany, Sept. 1997, pp. 245–252.

[7] B. Steiner and P. Jung, "Optimum and suboptimum channel estimation for the uplink of CDMA mobile radio systems with joint detection," *European Trans. on Telecommunications and Related Techniques*, vol. 5, pp. 39–50, 1994.

[8] D. G. Luenberger, *Optimization by Vector Space Methods*, John Wiley and Sons, New York, NY, 1969.

[9] G. H. Golub and C. F. van Loan, *Matrix Computations*, The John Hopkins University Press, third edition, 1996.

[10] T. Kailath and J. Chun, "Generalized Displacement Structure for Block-Toeplitz, Toeplitz-Block, and Toeplitz-Derived Matrices," *SIAM J. Matrix Anal. Appl*, vol. 15, no. 1, pp. 114–128, January 1994.

[11] J. Götze and H. Park, "Schur-type methods based on subspace criteria," in *Proc. IEEE Int. Symp. on Circuits and Systems*, Hong Kong, 1997, pp. 2661–2664.

[12] J. Chun, T. Kailath, and H. Lev-Ari, "Fast Parallel Algorithms for QR and Triangular Factorization," *SIAM J. Sci. Stat. Comput.*, vol. 8, no. 6, November 1987.