Comparative Study of Joint-Detection Techniques for TD-CDMA based Mobile Radio Systems

Marius Vollmer[1,2], Martin Haardt[1], Jürgen Götze[2]

1. Siemens AG
Information and Communication Mobile
Hofmannstr. 51
D-81359 Munich, Germany
haardt@ieee.org
Marius.Vollmer@icn.siemens.de

2. University of Dortmund
Information Processing Lab
Otto-Hahn-Str. 4
D-44221 Dortmund, Germany
juergen.goetze@uni-dortmund.de
marius.vollmer@uni-dortmund.de

Submitted: December 15, 2000; Revised: April 24, 2001.

Abstract

Third generation mobile radio systems use TD-CDMA in their TDD mode. Due to the TDMA component of TD-CDMA, joint (or multi-user) detection techniques can be implemented with a reasonable complexity. Therefore, joint detection will already be implemented in the first phase of the system deployment to eliminate the intra-cell interference. In a TD-CDMA mobile radio system, joint-detection is performed by solving a least squares problem, where the system matrix has a block-Sylvester structure. In this paper, we present and compare several techniques that reduce the computational complexity of the joint detection task even further by exploiting this block-Sylvester structure and by incorporating different approximations. These techniques are based on the Cholesky factorization, the Levinson algorithm, the Schur algorithm, and on Fourier techniques, respectively. The focus of this paper is on Fourier techniques since they have the smallest computational complexity and achieve the same performance as the joint detection algorithm that does not use any approximations. Similar to the well-known implementation of fast convolutions, the resulting Fourier-based joint detection scheme also uses a sequence of FFTs and overlapping. It is well suited for the implementation on parallel hardware architectures.

Keywords

TD-CDMA, joint detection, Toeplitz structure, Cholesky factor, Schur algorithm, Levinson Algorithm, Fast Fourier Transform, Overlap Save.

I. Introduction

THE UMTS Terrestrial Radio Access (UTRA) that has been specified in the Third Generation Partnership Project (3GPP) consists of an FDD (Frequency Division Duplex) and a TDD (Time Division Duplex) mode. WCDMA (Wideband CDMA) has been chosen for UTRA FDD, whereas UTRA TDD uses TD-CDMA (Time Division CDMA) [1].

In a CDMA-based system, several co-channel users (that transmit at the same time and on the same frequency) can be separated in the code domain, i.e., by exploiting their different spreading codes. Due to the additional separation in the time domain, the number of co-channel users that are active at the same time in

UTRA TDD is smaller than in a pure CDMA system, cf. Figure 1. Therefore, joint (multi user) detection techniques can be implemented with a reasonable complexity. Joint detection eliminates the intra-cell multiple access interference that is caused by users that are located in the same cell and transmit at the same time and on the same frequency (co-channel users) [2], [3].

A joint detector combines the knowledge about all users that share one burst into one large system of equations. This knowledge consists of the channel impulse responses that have been estimated from training sequences, the spreading codes, and the received antenna samples from all antenna elements. An efficient algorithm should exploit the special structural characteristics of the system matrix, i.e., its block-Sylvester structure (see Figure 3 for a preview of this structure). Sylvester matrices are Toeplitz matrices with a band structure. The Sylvester structure is the manifestation of the fact that the TD-CDMA system uses spreading codes that are periodic with the spreading factor and that the channel is assumed to be constant for one burst.

There are several algorithms that have been specifically developed to work efficiently with Toeplitz matrices. We will briefly look at both the Levinson algorithm, which can directly solve a system of equations involving a positive-definite system matrix with Toeplitz structure, and at the Schur algorithm, which efficiently computes the upper triangular factor $\boldsymbol{R}$ of the QR factorization of the Toeplitz matrix. However, the band structure of the system matrix together with the Toeplitz structure enables the use of far reaching approximations in the solution process. These approximations reduce the computational requirements of all involved algorithms, and in fact, make it possible that an algorithm such as the Cholesky factorization — that does not exploit the Toeplitz structure explicitly — remains competitive.

A different route consists of modifying the system matrix to be block-circulant in addition to being block-Toeplitz structured. This allows the use of Fast Fourier Transforms to efficiently transform the joint-detection problem into the frequency domain. The transformed problem is easier to solve than the original problem analogous to the way that a convolution is easier to perform in the frequency domain than in the time domain.

An issue common to all described algorithms is that they need to be applied to a *block*-Toeplitz matrix. While Toeplitz matrices arise in the context of single-user systems, multi-user systems lead to block-Toeplitz matrices. We will see that this adaptation is easy to perform for the Cholesky and Fourier algorithms, but not entirely straightforward for the Levinson and Schur algorithms.

The paper is organized as follows: Section II explains the data model used to derive the system equation. including spatial-temporal processing; sections III, IV, and V briefly describe the joint-detection algorithms based on the Cholesky factorization, the Levinson algorithm and the Schur algorithm, respectively; section VI explains the Fourier algorithm; and section VII presents simulation results and computational requirement figures for all joint detection algorithms.

II. System Model

A. TD-CDMA

TD-CDMA is the radio access scheme for the UTRA TDD mode. It is based on a TDMA scheme which is extended by a supplementary CDMA component [4], [3], [5]. In this TD-CDMA system, K CDMA codes (users) 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 [5]. 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. In this paper, we assume that all users use the same spreading factor. However, it is straightforward to extend the algorithms to variable spreading factors.

The joint detection in a TD-CDMA system in the space-time domain eliminates the intra-cell interference (interference from users within the same cell). Moreover, we can achieve an inter-cell interference cancellation by taking into account the space-time covariance matrix of the inter-cell-interference-plus-noise.

B. Data Model

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 part of the received measurement vector that is exclusively determined by the midamble [6].

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

$$\boldsymbol{d}^{(k)} = \left[\begin{array}{cccc} d_1^{(k)} & d_2^{(k)} & \cdots & d_N^{(k)} \end{array} \right]^T \in \mathbb{C}^N, \quad 1 \leq k \leq K. \tag{1}$$

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

$$\boldsymbol{c}^{(k)} = \left[\begin{array}{cccc} c_1^{(k)} & c_2^{(k)} & \cdots & c_Q^{(k)} \end{array} \right]^T \in \mathbb{C}^Q, \quad 1 \leq k \leq K.$$

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

as

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

where $\otimes$ denotes the Kronecker product. Assume that K spreading codes are transmitted at the same time. After eliminating the influence of the midamble, the received measurements at the mth antenna obey the following linear model:

$$\boldsymbol{x}^{(m)} = \begin{bmatrix} x_1^{(m)} \\ x_2^{(m)} \\ \vdots \\ x_{NQ+W-1}^{(m)} \end{bmatrix} = \sum_{k=1}^{K} \begin{bmatrix} \boldsymbol{h}^{(k,m)} & & & \\ & \boldsymbol{h}^{(k,m)} & & \\ & & \ddots & \\ & & & \boldsymbol{h}^{(k,m)} \end{bmatrix} \cdot \boldsymbol{C}^{(k)} \cdot \boldsymbol{d}^{(k)} + \begin{bmatrix} n_1^{(m)} \\ n_2^{(m)} \\ \vdots \\ n_{NQ+W-1}^{(m)} \end{bmatrix}$$

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

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.

Using the definition of $\boldsymbol{B}^{(k,m)} \in \mathbb{C}^{(NQ+W-1)\times N}$ in (3), $\boldsymbol{x}^{(m)}$ may be expressed as

$$\boldsymbol{x}^{(m)} = \sum_{k=1}^{K} \boldsymbol{B}^{(k,m)} \cdot \boldsymbol{d}^{(k)} + \boldsymbol{n}^{(m)}, \quad 1 \le m \le M, \tag{4}$$

$$= \sum_{k=1}^{K} \begin{bmatrix} \boldsymbol{b}^{(k,m)} & & & \\ & \boldsymbol{b}^{(k,m)} & & \\ & & \ddots & \\ & & & \boldsymbol{b}^{(k,m)} \end{bmatrix} \cdot \boldsymbol{d}^{(k)} + \boldsymbol{n}^{(m)},$$

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.,

$$\boldsymbol{b}^{(k,m)} = \left[\begin{array}{cccc} b_1^{(k,m)} & b_2^{(k,m)} & \cdots & b_{Q+W-1}^{(k,m)} \end{array} \right]^T = \boldsymbol{h}^{(k,m)} * \boldsymbol{c}^{(k)} \in \mathbb{C}^{Q+W-1}, \tag{5}$$

$$1 \leq k \leq K, \quad 1 \leq m \leq M.$$

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,

$$\boldsymbol{d} = \mathrm{vec}\left\{ \left[\begin{array}{cccc} \boldsymbol{d}^{(1)} & \boldsymbol{d}^{(2)} & \cdots & \boldsymbol{d}^{(K)} \end{array} \right]^T \right\} = \left[\begin{array}{cccc} \boldsymbol{d}_1^T & \boldsymbol{d}_2^T & \cdots & \boldsymbol{d}_N^T \end{array} \right]^T \in \mathbb{C}^{NK}. \tag{6}$$

Here and in the sequel, the vec operator is defined according to

$$\mathrm{vec}\left\{ \begin{bmatrix} a_1 & a_2 & \cdots \\ b_1 & b_2 & \cdots \end{bmatrix} \right\} = \begin{bmatrix} a_1 & b_1 & a_2 & b_2 & \cdots \end{bmatrix}^T, \tag{7}$$

that is, the vec operator forms a column vector from the elements of its argument matrix by concatenating the columns of that matrix, starting at the left. The combined CIR vectors $\boldsymbol{b}^{(k,m)}$ of the kth user at all M antennas can also be simplified to a single combined CIR vector in the space-time domain

$$\boldsymbol{b}^{(k)} = \mathrm{vec}\left\{ \left[\begin{array}{cccc} \boldsymbol{b}^{(k,1)} & \boldsymbol{b}^{(k,2)} & \cdots & \boldsymbol{b}^{(k,M)} \end{array} \right]^T \right\} \in \mathbb{C}^{M(Q+W-1)}, \quad 1 \leq k \leq K. \tag{8}$$

Moreover, let us define the space-time array measurement vector (during one half burst) as

$$\boldsymbol{x} = \mathrm{vec}\left\{ \left[\begin{array}{cccc} \boldsymbol{x}^{(1)} & \boldsymbol{x}^{(2)} & \cdots & \boldsymbol{x}^{(M)} \end{array} \right]^T \right\} \in \mathbb{C}^{M(NQ+W-1)}. \tag{9}$$

Using equation (4), the vector $\boldsymbol{x}$ may then be expressed as

$$\boldsymbol{x} = \begin{array}{|c|} \hline \boldsymbol{V} \\ \hline \end{array} \begin{array}{|c|} \hline \\ \boldsymbol{V} \\ \hline \end{array} \begin{array}{|c|} \hline \\ \\ \\ \boldsymbol{V} \\ \hline \end{array} \cdot \begin{array}{|c|} \hline \boldsymbol{d}_1 \\ \hline \boldsymbol{d}_2 \\ \hline \vdots \\ \hline \boldsymbol{d}_N \\ \hline \end{array} + \boldsymbol{n} = \boldsymbol{T} \cdot \boldsymbol{d} + \boldsymbol{n}, \tag{10}$$

where the matrix

$$\boldsymbol{V} = \begin{array}{|c|c|c|c|} \hline \boldsymbol{b}^{(1)} & \boldsymbol{b}^{(2)} & \cdots & \boldsymbol{b}^{(K)} \\ \hline \end{array} \in \mathbb{C}^{M(Q+W-1)\times K} \tag{11}$$

contains the combined CIR vectors of all K users in the space-time domain. In equation (10), the space-time vector

$$\boldsymbol{n} = \text{vec}\left\{ \left[\begin{array}{cccc} \boldsymbol{n}^{(1)} & \boldsymbol{n}^{(2)} & \cdots & \boldsymbol{n}^{(M)} \end{array} \right]^T \right\} \in \mathbb{C}^{M(NQ+W-1)}. \tag{12}$$

models inter-cell interference, i.e., dominant interferers from adjacent cells, and additive (thermal) noise. Notice that the matrix $\boldsymbol{T} \in \mathbb{C}^{M(NQ+W-1)\times(NK)}$ in equation (10) has a block-Toeplitz structure.

C. Joint Data Detection via Block Linear Equalization

Given the linear space-time data model in equation (10), 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

$$\hat{\boldsymbol{d}} = \left[\begin{array}{cccc} \hat{\boldsymbol{d}}_1^T & \hat{\boldsymbol{d}}_2^T & \cdots & \hat{\boldsymbol{d}}_N^T \end{array} \right]^T = \boldsymbol{W}^H \boldsymbol{x} = \left[\begin{array}{cccc} \boldsymbol{w}_1 & \boldsymbol{w}_2 & \cdots & \boldsymbol{w}_{NK} \end{array} \right]^H \boldsymbol{x} \in \mathbb{C}^{NK}. \tag{13}$$

In the sequel, two alternative solutions are presented.

C.1 LS 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 (13) such that the Euclidean norm of the error

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

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.,

$$\hat{\boldsymbol{d}} = \boldsymbol{W}^H \boldsymbol{x} = \boldsymbol{T}^+ \boldsymbol{x} = \left(\boldsymbol{T}^H \boldsymbol{T} \right)^{-1} \boldsymbol{T}^H \cdot \boldsymbol{x}. \tag{14}$$

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

C.2 MVDR 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

$$\boldsymbol{W}^H \boldsymbol{T} = \left[\begin{array}{cccc} \boldsymbol{w}_1 & \boldsymbol{w}_2 & \cdots & \boldsymbol{w}_{NK} \end{array} \right]^H \boldsymbol{T} = \boldsymbol{I}_{NK}. \tag{15}$$

In the literature, the minimum variance distortionless response (MVDR) solution is also called zero-forcing block linear equalizer, linear minimum variance unbiased estimate, weighted least squares estimate, or Gauss-Markov estimate [7]. The solution of this constrained optimization problem is given by

$$\hat{\boldsymbol{d}} = \boldsymbol{W}^H \boldsymbol{x} = \left(\boldsymbol{T}^H \boldsymbol{R}_{nn}^{-1} \boldsymbol{T}\right)^{-1} \boldsymbol{T}^H \boldsymbol{R}_{nn}^{-1} \cdot \boldsymbol{x}, \tag{16}$$

where we assume that the space-time covariance matrix of the inter-cell-interference-plus-noise (IC-IN) $\boldsymbol{R}_{nn} = \mathrm{E}\{\boldsymbol{n}\boldsymbol{n}^H\}$ is non-singular.[1] It contains contributions from the dominant interferers form adjacent cells and from the additive noise. Notice that the MVDR solution (16) simplifies to the standard least squares solution (14) if $\boldsymbol{R}_{nn} = \sigma_N^2 \boldsymbol{I}_{M(NQ+W-1)}$.

In order to reduce the computational complexity and to facilitate the estimation of $\boldsymbol{R}_{nn}$, the space-time covariance matrix of the IC-IN is approximated as the Kronecker product of the temporal covariance matrix of the IC-IN $\boldsymbol{R}_{\mathrm{I}_{(\mathrm{t})}}$ with the spatial covariance matrix of the IC-IN $\boldsymbol{R}_{\mathrm{I}}$, i.e., $\boldsymbol{R}_{nn} \approx \boldsymbol{R}_{\mathrm{I}_{(\mathrm{t})}} \otimes \boldsymbol{R}_{\mathrm{I}}$. Measurements have shown that for UTRA-TDD the temporal covariance matrix $\boldsymbol{R}_{\mathrm{I}_{(\mathrm{t})}}$ can, furthermore, be approximated with the identity matrix. Hence, we only have to estimate the spatial covariance matrix of the IC-IN $\boldsymbol{R}_{\mathrm{I}}$.

To actually compute (16), one has to consider the approximate inverse of $\boldsymbol{R}_{nn}$,

$$\boldsymbol{R}_{nn}^{-1} \approx \boldsymbol{I} \otimes \boldsymbol{R}_{\mathrm{I}}^{-1}. \tag{17}$$

Equation (17) indicates that we only need to be concerned with the inverse of the small matrix $\boldsymbol{R}_{\mathrm{I}} \in \mathbb{C}^{M \times M}$. The obvious way to proceed is then to find the Cholesky factor $\boldsymbol{L}_{\mathrm{I}}$ so that $\boldsymbol{L}_{\mathrm{I}}^H \boldsymbol{L}_{\mathrm{I}} = \boldsymbol{R}_{\mathrm{I}}$ and to calculate the weighted system matrix $\tilde{\boldsymbol{T}}$ and the weighted right-hand side $\tilde{\boldsymbol{x}}$ via back substitution according to

$$\tilde{\boldsymbol{T}} = (\boldsymbol{I} \otimes \boldsymbol{L}_{\mathrm{I}})^{-1} \boldsymbol{T},$$
$$\tilde{\boldsymbol{x}} = (\boldsymbol{I} \otimes \boldsymbol{L}_{\mathrm{I}})^{-1} \boldsymbol{x}.$$

Equation (16) can then be expressed as

$$\hat{\boldsymbol{d}} = \left(\tilde{\boldsymbol{T}}^H \tilde{\boldsymbol{T}}\right)^{-1} \tilde{\boldsymbol{T}}^H \tilde{\boldsymbol{x}}. \tag{18}$$

This equation has the form of equation (14) and we can therefore concentrate on the LS solution in the sequel.

The processing requirements for computing $\tilde{\boldsymbol{T}}$ and $\tilde{\boldsymbol{x}}$ are modest compared to the rest of the joint detection process. The number of real multiplications n is

$$\begin{aligned} n = \quad & \tfrac{2}{3}(M^3 - M) && \text{(Cholesky factorization)} \\ + \quad & 2(vQK + NQ + W - 1)(M^2 - M) && \text{(back substitution),} \end{aligned}$$

[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 (15), 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 $v = \lceil (Q+W-1)/Q \rceil$. For example, a typical scenario with $Q = 16, K = 8, W = 57, N = 69, M = 8$ needs 201936 real multiplications.

D. Common preprocessing steps

The remainder of this text will describe four different approaches to compute (14). Although these approaches differ in their core features, they use two common preparatory operations: the computation of the *correlation matrix*

$$\boldsymbol{S} = \boldsymbol{T}^H \boldsymbol{T} \tag{19}$$

and the *matched filter outputs*

$$\boldsymbol{y} = \boldsymbol{T}^H \boldsymbol{x}. \tag{20}$$

It is straightforward to take the structural characteristics of $\boldsymbol{T}$ into account when computing $\boldsymbol{S}$ and $\boldsymbol{y}$. For example, $\boldsymbol{S}$ is a Hermitian block-Toeplitz matrix with band structure and it is only necessary to compute a small part of it, see Figure 4.

With the definitions in (19) and (20), equation (14) can be rewritten as

$$\hat{\boldsymbol{d}} = \boldsymbol{S}^{-1} \boldsymbol{y}. \tag{21}$$

This is the formulation of the joint detection problem that we concentrate on in the following sections. The inverse of $\boldsymbol{S}$ exists if $\boldsymbol{T}$ has full rank. In the following we assume that this is the case.

III. Cholesky Algorithm

A straightforward approach to solving (14) is to compute the Cholesky factor $\boldsymbol{R}$ of $\boldsymbol{S}$ (as, for example, explained in [8]) such that

$$\boldsymbol{S} = \boldsymbol{R}^H \boldsymbol{R}$$

and consequently

$$\hat{\boldsymbol{d}} = \boldsymbol{R}^{-1} \boldsymbol{R}^{-H} \boldsymbol{y}.$$

The matrix $\boldsymbol{R}$ is a upper triangular matrix which enables us to multiply by its inverse via back substitutions. It is easy to exploit the band structure of $\boldsymbol{S}$ while computing $\boldsymbol{R}$ and the fact that $\boldsymbol{R}$ inherits this band structure from $\boldsymbol{S}$ reduces the computational requirements of the back substitutions significantly.

However, much more savings can be realized when computing $\boldsymbol{R}$ only approximately. This is possible because $\boldsymbol{R}$ does not only inherit the band structure of $\boldsymbol{S}$ but it also shows an approximate block-Toeplitz structure (with the same block size as $\boldsymbol{S}$). Figure 5 (a) depicts how this fact can be exploited when $\boldsymbol{R}$ is

computed: it suffices to compute the first few block-rows of $\boldsymbol{R}$ and then we assume that the remaining block-rows are identical to the last computed block-row. See [9] for a theoretical justification of this approach. Also, see [10] for a discussion of the row approximation in the context of a sliding-window equalizer.

A different approach is to consider block-columns instead of block-rows and stop after computing a small number of block-columns, see Figure 5 (b). This leads to the same approximations as in [11]. However, the row based approach leads to better results, see section VII.

In summary it should be noted that although the Cholesky algorithm can not directly exploit the Toeplitz structure of $\boldsymbol{T}$ and $\boldsymbol{S}$, it can still indirectly take advantage of it if $\boldsymbol{R}$ is only computed approximately. These approximations are very effective and are the reason why the Cholesky algorithm still remains competitive with the other, more optimized and more complicated algorithms. However, the Block-Fourier algorithm achieves the same performance with dramatically reduced computational requirements.

IV. Block-Levinson Algorithm

The Levinson algorithm [8] is a direct way to compute the solution to a system of linear equations

$$\boldsymbol{S}\boldsymbol{d} = \boldsymbol{y}$$

where $\boldsymbol{S}$ is Hermitian, Toeplitz and positive definite while needing only $O(n^2)$ arithmetical operations for a matrix of size $n \times n$, compared to $O(n^3)$ for the unapproximated Cholesky algorithm.

However, the matrix $\boldsymbol{S}$ in our case is *block*-Toeplitz so that the original Levinson algorithm needs to be extended [12]. Although the matrix $\boldsymbol{S}$ is Hermitian, it does not have the right kind of symmetry for the Levinson algorithm when working with blocks. The Block-Levinson algorithm is therefore based on the Levinson algorithm for non-Hermitian matrices.

Appendix A contains more details about the Block-Levinson algorithm. Reference [13] contains a derivation of the Block-Levinson algorithm using Matrix Levinson Polynomials.

The Block-Levinson algorithm has several opportunities for introducing approximations. The internal parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\nu}$ (see Appendix A) can be observed to converge to zero quite rapidly and thus we can skip a significant amount of operations after a small number of iterations by assuming that they are in fact zero. An additional approximation can be introduced by observing that the elements of the block-vectors $\boldsymbol{Y}$ and $\boldsymbol{W}$ also tend to zero for higher indices. Thus, we can save operations by shortening the length of the vector operations involving $\boldsymbol{Y}$ and $\boldsymbol{W}$.

V. Block-Schur Algorithm

The Schur algorithm [14], [15] is another algorithm that is able to compute the least squares solution for Toeplitz structured systems of equations with $O(n^2)$ operations. It achieves this by efficiently finding the

triangular factor $\boldsymbol{R}$ of the system matrix such that

$$\boldsymbol{T} = \boldsymbol{Q}\boldsymbol{R}, \quad \text{and} \quad \hat{\boldsymbol{d}} = \boldsymbol{R}^{-1}\boldsymbol{Q}^H\boldsymbol{x}.$$

Here, $\boldsymbol{R}$ is again the Cholesky factor of $\boldsymbol{S}$ that has already appeared in section III. Thus, it is a band structured, approximately block-Toeplitz, upper triangular matrix. The matrix $\boldsymbol{Q}$ is a unitary matrix, i.e., $\boldsymbol{Q}^H\boldsymbol{Q} = \boldsymbol{I}$.

The Block-Schur algorithm gains its efficiency from the fact that it works with a *displacement representation* of $\boldsymbol{S}$ that contains much less redundancy than $\boldsymbol{S}$ itself. It then proceeds to transform this representation into $\boldsymbol{R}$ using local transformations. Simultaneously, the right-hand side $\boldsymbol{Q}^H\boldsymbol{x}$ is computed from a representation of $\boldsymbol{y}$ using the same transformations.

The Block-Schur algorithm works in such a way that the matrix $\boldsymbol{R}$ becomes available row by row, starting from the top. This allows for the same row-wise approximation that has been discussed for the Cholesky algorithm simply by stopping the Block-Schur algorithm after enough rows have been computed. However, since the right-hand side is affected by this approximation as well, the resulting performance is slightly worse.

Using the Block-Schur algorithm for joint detection is discussed in detail in [16]. Reference [17] presents the use of the displacement representation in the context of cyclic reduction. Their algorithm is also amendable to approximations with the additional benefit that the approximations carry over to the back substitution step. The algorithm relies on matrix operations instead of local transformations, however.

VI. Block-Fourier Algorithm

The Block-Fourier algorithm exploits the fact that digital signals can be transformed efficiently to and from the frequency domain using Fast Fourier Transforms and that performing signal processing in the frequency domain is often more efficient than calculating the same task in the time domain. For example, it is well known that the convolution of two signals can be computed much more efficiently by transforming them to the frequency domain and performing element-wise multiplications there instead of a series of scalar products in the time domain.

The Block-Fourier algorithm for performing Joint Detection [18], [19] is based on the same ideas as the fast convolution method: the received samples and the system matrix are transformed and the least squares solution is computed in the frequency domain. The final solution is then found by transforming the frequency solution back into the time domain.

Data Detection in the presence of a channel that is modeled by an FIR filter can be seen as the inverse problem to convolution: as a deconvolution. Simply worded, instead of multiplying the spectra element-wise in the frequency domain, we need to perform element-wise divisions.

Just as in the convolution case, we must account for the fact that using a discrete Fourier transform leads to a circular deconvolution. This can be realized by applying a suitable form of *zero-padding* and *overlapping*

when constructing the solution from smaller parts, just as zero-padding and overlap-save or overlap-add is used in the fast convolution method. In the context of multi-user data detection, we face the additional problem that we need to extend the method to cope with the block-structure of the problem. This can be achieved by using *block-Fourier* transforms instead of the usual single signal transforms.

In the sequel, we will explain the Block-Fourier algorithm in detail. We will first show how to express fast deconvolution for a single user in linear algebra terms; then, we will extend this to multiple users and explain block-Fourier transforms; finally, we will apply the results to the TD-CDMA system that has been in the focus of this paper.

A. Diagonalizing Block-Circulant Matrices

Describing circular convolutions and deconvolutions as matrix operations leads to *circulant* matrices. A circulant matrix is a Toeplitz matrix with the additional restriction that it must be square and each column is a rotated version of the column to the left of it [8]. For example,

$$\begin{bmatrix} c_1 & c_4 & c_3 & c_2 \\ c_2 & c_1 & c_4 & c_3 \\ c_3 & c_2 & c_1 & c_4 \\ c_4 & c_3 & c_2 & c_1 \end{bmatrix}$$

is a circulant matrix.

Systems of equations like

$$\boldsymbol{C}\boldsymbol{x} = \boldsymbol{b}, \tag{22}$$

where $\boldsymbol{C}$ is a circulant matrix, can be solved efficiently since $\boldsymbol{C}$ can be *diagonalized* with discrete Fourier transforms, implemented by *Fast Fourier Transforms* (FFT) [20]. It is an inherent property of circulant matrices that their eigenvectors are the columns of the Fourier transform matrix $\boldsymbol{F}$. This means that we can write every circulant matrix $\boldsymbol{C}$ as [21]

$$\boldsymbol{C} = \boldsymbol{F}^{-1}\boldsymbol{\Lambda}\boldsymbol{F} \tag{23}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix that contains the eigenvalues of $\boldsymbol{C}$ on its diagonal. Substituting this into (22) yields

$$\boldsymbol{\Lambda}\boldsymbol{F}\boldsymbol{x} = \boldsymbol{F}\boldsymbol{b}. \tag{24}$$

The matrix $\boldsymbol{\Lambda}$ itself can also be computed with a Fourier transform according to

$$\mathrm{diag}(\boldsymbol{\Lambda}) = \boldsymbol{F}\boldsymbol{C}(:,1) \tag{25}$$

where $\mathrm{diag}(\boldsymbol{\Lambda})$ denotes the diagonal elements of $\boldsymbol{\Lambda}$ rearranged as a vector. The *colon notation* as used in $\boldsymbol{C}(:,1)$ is explained in Appendix A.

We can therefore solve (22) in the frequency domain as long as $\boldsymbol{C}$ is not singular (i. e., it has only non-zero eigenvalues):

$$\boldsymbol{x} = \text{IFFT}(\text{FFT}(\boldsymbol{b})./\text{FFT}(\boldsymbol{C}(:,1))), \tag{26}$$

where $\text{FFT}(\cdot)$ and $\text{IFFT}(\cdot)$ denote the discrete Fourier transform of a vector and its inverse, respectively, and $./$ denotes element-wise division.

A *block*-circulant matrix $\boldsymbol{C}$ of size $DP \times DK$ with blocks of size $P \times K$ is a matrix that fulfils the conditions for being a circulant matrix when we consider its blocks instead of the individual elements. Thus, it has the property

$$\boldsymbol{C}_{i,j} = \boldsymbol{C}_{\tilde{i},\tilde{j}} \quad \text{with} \quad \begin{aligned} \tilde{i} &= ((i+P-1) \bmod DP) + 1, \\ \tilde{j} &= ((j+K-1) \bmod DK) + 1. \end{aligned} \tag{27}$$

Therefore, each element of $\boldsymbol{C}$ has the same value as the element P rows below and K columns to the right of it. Indices in this calculation that exceed the size of $\boldsymbol{C}$ wrap around and correspond to indices in the first block-row and block-column of $\boldsymbol{C}$.

When dealing with block matrices we also need to apply block-Fourier transforms. Suppose $\boldsymbol{C}_{(P,K)} \in \mathbb{C}^{DP \times DK}$ is block-circulant with a block size of $P \times K$. Then we can find a block-diagonal matrix $\boldsymbol{\Lambda}_{(P,K)}$ with the same block size such that

$$\boldsymbol{C}_{(P,K)} = \boldsymbol{F}_{(P)}^{-1} \boldsymbol{\Lambda}_{(P,K)} \boldsymbol{F}_{(K)} \tag{28}$$

where $\boldsymbol{F}_{(P)} \in \mathbb{C}^{DP \times DP}$ and $\boldsymbol{F}_{(K)} \in \mathbb{C}^{DK \times DK}$ are block-Fourier transforms with a block size of $P \times P$ and $K \times K$, respectively. They are defined as

$$\boldsymbol{F}_{(n)} = \boldsymbol{F} \otimes \boldsymbol{I}_n \tag{29}$$

where $\boldsymbol{I}_n$ is the identity matrix of size $n \times n$ and $\boldsymbol{F}$ is the Fourier matrix of size $D \times D$.

As before, the matrix $\boldsymbol{\Lambda}_{(P,K)}$ can be computed with a block-Fourier transform of the first block-column of $\boldsymbol{C}_{(P,K)}$. Let $\text{diag}_{(P,K)}(\boldsymbol{\Lambda}_{(P,K)})$ denote the blocks of size $P \times K$ on the block-diagonal of $\boldsymbol{\Lambda}_{(P,K)}$ rearranged as a $DP \times K$ matrix. Then we have

$$\text{diag}_{(P,K)}(\boldsymbol{\Lambda}_{(P,K)}) = \boldsymbol{F}_{(P)} \boldsymbol{C}_{(P,K)}(:, 1:K). \tag{30}$$

The block version of equation (22) thus reduces to

$$\boldsymbol{\Lambda}_{(P,K)} \boldsymbol{F}_{(K)} \boldsymbol{x} = \boldsymbol{F}_{(P)} \boldsymbol{b}. \tag{31}$$

The block-Fourier transforms denoted by $\boldsymbol{F}_{(P)}$ (and $\boldsymbol{F}_{(K)}$) and their inverses can be performed with P (K) non-block discrete (inverse) Fourier transforms of length D. Next we have to invert $\boldsymbol{\Lambda}_{(P,K)}$. Since the blocks on the diagonal of $\boldsymbol{\Lambda}_{(P,K)}$ are essentially unstructured, this can, for example, be achieved by some standard method like the Cholesky decomposition of $\boldsymbol{\Lambda}_{(P,K)}^H \boldsymbol{\Lambda}_{(P,K)}$ and two back-substitutions.

B. Application to TD-CDMA

Although $\boldsymbol{T}$ in (14) (and $\tilde{\boldsymbol{T}}$ in (18)) is not block-square, its block-columns are already rotated versions of the first block-column. Therefore, we can simply add block columns to it until it is block-square. The number of block-columns that must be added depends on the degree of the block-band structure of $\boldsymbol{T}$, which is the same as the inter-symbol interference in the original transmission system.

After $\boldsymbol{T}$ has been extended to be block-circulant, it has $D \times D$ blocks of size $P \times K$, where $D = N + \lceil (Q + W - 1)/Q \rceil - 1$ and $P = MQ$.[2] The vector $\boldsymbol{x}$ needs to be zero-padded at its end so that it has length DMQ. Likewise, the new solution vector contains the desired results in its first NK elements.

Simulations (section VII) have shown that the error made by solving for $\hat{\boldsymbol{d}}$ with this extended version of $\boldsymbol{T}$ is insignificant. This is due to the fact the the distortions introduced by making $\boldsymbol{T}$ block-square affect mainly the guard periods between bursts.

A slight variation of the method outlined so far leads to reduced computational requirements. Instead of transforming $\boldsymbol{T}$ into the frequency domain and computing the correlation matrix there, we can compute $\boldsymbol{S}$ in the time domain and transform the result. This is generally advantageous because computing $\boldsymbol{S}$ requires few operations due to its rich structure.

C. Overlapping

The convolution matrix $\boldsymbol{T}$ with its strong band structure offers possibilities to reduce the computational demands of the joint detector even further. Just as it is possible to perform the convolution of a long signal with a much shorter filter impulse response with the well known overlap-save technique [22], it is possible to use this technique for equalizing such a filter.

The idea is to reduce the size of the involved matrices and solve the whole problem by solving multiple smaller ones instead. The reduction in size is expressed by forcing D to smaller values when deriving a block-circulant matrix from $\boldsymbol{T}$. With such a smaller matrix only a smaller part of the data vector can be estimated of course; thus, we need to partition the data vector into slices of length D. But if D is smaller than its ideal value $N + \lceil (Q + W - 1)/Q \rceil - 1$, the distortions mentioned in the previous section do no longer fall into the guard period, leading to unacceptable errors in the estimated data vector slices.

Figure 6 depicts this effect. The dashed line shows the relative error of the data symbols of one user where D has been set to 32 and the full vector of $N = 69$ symbols has been calculated by carrying out the frequency domain detection three times on successive blocks of 32 symbols each. Obviously, each run of $D = 32$ symbols has large errors at the beginning and the end, but not in its middle part.

As a remedy, one discards a certain number of symbols at the start of the data vector slice (the *prelap*) and

[2] The notation $\lceil x \rceil$ denotes the *ceiling* of x, i. e., the smallest integer that is greater than or equal to x.

at the end of it (the *postlap*). The computation of the complete data vector needs to be arranged in such a way that the discarded symbols from the previous slice can be found in the middle of the next one as depicted in Figure 7. Figure 6 also shows that the relative error for such an overlapping computation has been reduced to a lower level for all symbols (solid line).

D. Parallelism

The execution of the algorithm can be divided into four stages, with a lot of parallelism in each stage. Furthermore, if dedicated hardware is available for each stage, they can be pipelined so that all stages can be performed in parallel, too. The following calculations take into account that one burst consists of two half bursts before and after the training sequence (midamble).

Let $L = \lceil N/(D - \mathit{prelap} - \mathit{postlap}) \rceil$ be the number of vector slices needed to cover all desired data symbols, and let $\boldsymbol{x}^{(\ell)}$ and $\boldsymbol{d}^{(\ell)}$ denote the actual slices of $\boldsymbol{x}$ and $\hat{\boldsymbol{d}}$, respectively.

The *first* stage computes $\hat{\boldsymbol{S}} = \hat{\boldsymbol{T}}^H \hat{\boldsymbol{T}}$ where $\hat{\boldsymbol{T}}$ is the circularized system matrix.

The *second* stage consists of all block-FFT operations, namely the computation of

$$\begin{aligned} \boldsymbol{\Lambda}_{(P,K)} &= \boldsymbol{F}_{(P)} \hat{\boldsymbol{T}}, \\ \boldsymbol{\Lambda}_{(P,K)}^H \boldsymbol{\Lambda}_{(P,K)} &= \boldsymbol{F}_{(K)} \hat{\boldsymbol{S}}(:, 1:K), \\ \boldsymbol{\chi}^{(\ell)} &= \boldsymbol{F}_{(P)} \boldsymbol{x}^{(\ell)}, \quad 1 \le \ell \le L. \end{aligned}$$

Clearly, all transformations are independent and can thus be computed in parallel. Moreover, it can be seen from the definition of $\boldsymbol{F}_{(P)}$ that there also exists parallelism within each transformation.

Given

$$\begin{gathered} \boldsymbol{y} \in \mathbb{C}^{DP}, \quad \boldsymbol{F} \in \mathbb{C}^{D \times D}, \quad \boldsymbol{I}_{(P)} \in \mathbb{C}^{P \times P} \\ \boldsymbol{F}_{(P)} = \boldsymbol{F} \otimes \boldsymbol{I}_{(P)}, \quad \boldsymbol{z} = \boldsymbol{F}_{(P)} \boldsymbol{y} \end{gathered}$$

we have

$$\boldsymbol{z}(i : P : n - P + i) = \boldsymbol{F}\boldsymbol{y}(i : P : n - P + i), \quad 1 \le i \le P.$$

Hence, a block-FFT of block-size P decomposes into P parallel non-block-FFTs. Therefore, the first stage consists of $PK + 2PL + K^2$ parallel FFTs of length D.

The *third* stage has to compute

$$\boldsymbol{\delta}^{(\ell)} = (\boldsymbol{\Lambda}_{(P,K)}^H \boldsymbol{\Lambda}_{(P,K)})^{-1} \boldsymbol{\Lambda}_{(P,K)}^H \boldsymbol{\chi}^{(\ell)}.$$

Since $\boldsymbol{\Lambda}_{(P,K)} \in \mathbb{C}^{DP \times DK}$ is block-diagonal, this problem decomposes into D independent systems of linear equations, with $2L$ right hand sides each.

The *fourth* stage finally has to apply block-IFFTs to $\boldsymbol{\delta}^{(\ell)}$ to yield $\boldsymbol{d}^{(\ell)}$. This can be done with $2KL$ independent inverse FFTs of length D, analogous to the first stage.

See Appendix B for more details.

VII. Simulation Results and Computational Complexity

Since the presented algorithms rely on approximations to achieve their low computational requirements, it is important to verify that the resulting performance loss is acceptable. Therefore, detailed simulations of an uplink TD-CDMA system have been performed. The simulations model a uplink scenario for speech service with $K = 8$ active users, $M = 2$ antennas for diversity reception, forward error correction using a convolutional coder, and radio transmission in a vehicular environment (Type A) with a mobile speed of 120 km/h [23], [24].

The convolutional coder has rate 1/3 and is followed by puncturing with rate 168/160. It has been included in the simulation to achieve more realistic statistical properties of the transmitted symbols.

The parameters of the simulated bursts are $N = 60$ symbols per data block, a spreading factor of $Q = 16$, a midamble length of $L = 539$ chips and a guard period of 101 chips. Each data point has been obtained by simulating 5000 bursts.

The channel model is realized as a tapped delay line with a fixed delay, Rayleigh fading and a Jakes-type Doppler spectrum for each tap. The delays and relative powers of the taps are given in Table I.

Tap	Delay (ns)	Power (dB)
1	0	0
2	310	-1
3	710	-9
4	1090	-10
5	1730	-15
6	2510	-20

TABLE I

Tap delays and relative average powers of the Vehicular A channel model.

The channel noise is modeled as being white.

The scenario also includes a severe near/far constellation: 4 of the users are received at 20 dB above the remaining 4 users. The shown bit error rates are the mean value of the four weaker users. Table II lists the simulated E_b/N_0 values and corresponding SNR values for all users.

	weak		strong	
	E_b/N_0	SNR	E_b/N_0	SNR
Simulation run 1	4.7	-8.1	24	12
Simulation run 2	5.7	-7.1	25	13
Simulation run 3	6.6	-6.2	26	14
Simulation run 4	7.7	-5.1	27	15
Simulation run 5	8.7	-4.1	28	16

TABLE II

E_b/N_0 AND SNR VALUES OF ALL $K = 8$ USERS IN DB.

The receiver employs two diversity antennas, i.e, the signals impinging on the antennas are assumed to be uncorrelated. This is reflected in the simulation setup by modelling the path from the transmitter to each receiver antenna by an independently fading, point-to-point channel model.

The channel impulse responses are estimated using the method described in [25] with a assumed maximum impulse length of $W = 60$ chips.

The resulting bit error rates as a function of the ratio of the bit energy and the spectral noise density are shown in Figure 8. The approximation parameters in this figure were chosen so as to yield the maximum reduction in computational requirements while still yielding an acceptable bit error rate as compared to the unapproximated solution.

The legends of the figures refer to the approximation parameters used in the algorithms as follows:

- For the Cholesky algorithm, "n rows" means that only n block-rows of $\boldsymbol{R}$ have been computed and the rest has been filled from block-row n. Likewise, "n columns" refers to the column-wise approximation. See also Figure 5.
- For the Block-Levinson algorithm, "n iterations" means that $\boldsymbol{\alpha}$ and $\boldsymbol{\nu}$ are set to zero after $n - 1$ iterations of the main loop of the algorithm.
- For the Block-Schur algorithm, "n rows" means that only the first n block-rows of $\boldsymbol{R}$ have been computed and the right-hand side has only been transformed correspondingly.
- For the Block-Fourier algorithm, "*D*/*prelap*/*postlap*" means that overlapping was performed with a FFT length of D, using the given prelap and postlap. Refer to Figure 7. Note that for "128/0/0" no overlapping is done.

Figures 9, 10, 11, and 12 show the corresponding computational requirements in the form of required real valued multiplications to perform joint detection for one burst, depending on the number of active users, K.

They clearly show the significant reduction in computational requirements by using the discussed approximations. Figures 13, 14, and 15 compare the algorithms also shown in Figure 8 in terms of their computational requirements, depending on the number of active codes, K, the number of symbols in one data block N, and the number of antennas at the receiver, M.

The computational requirements as shown in the figures are computed by a suit of programs available in [26].

It can be seen that the row-wise approximation method for the Cholesky algorithm is superior to the column-wise method when both the bit error performance and computational requirements are taken into account. The Block-Levinson algorithm is not able to achieve lower computational requirements than the Cholesky algorithm for a comparable bit error performance, although it could be expected from the difference in computational complexity of the exact algorithms. The Block-Schur algorithm is computationally more expensive than both the Cholesky and Block-Levinson algorithms, but it should be better suited to a fine grained parallel implementation.

Finally, the Block-Fourier algorithm clearly has the lowest computational requirements while showing the same bit error performance as the exact Cholesky algorithm.

VIII. Conclusions

In this paper, we have presented four algorithms that can be used to implement joint data detection in a TD-CDMA system. We have shown how to introduce approximations into the algorithms and thus have achieved significant reductions of their computational requirements.

It can be seen that all algorithms except the Block-Fourier algorithm need about the same amount of multiplications for approximation parameters that lead to an acceptable bit error ratio. The Block-Fourier algorithm is significantly cheaper. This low computational complexity has been achieved by using overlapping techniques for the deconvolution. Overlapping was performed both at the beginning and at the end of the considered data vector. Both the block-FFTs and the inversion of the diagonalized system matrix consist of independent subproblems that can be solved in parallel.

Thus, the Block-Fourier algorithm is the method of choice for implementing joint detection. However, the Block-Levinson or Block-Schur algorithm are also interesting from different points of view. The Block-Schur algorithm allows a parallel implementation on fine grained architectures (e.g. CORDIC) [27]. The Block-Levinson algorithm permits to add stages to an existing architecture to increase the accuracy of the solution (such as stages are added to a lattice filter) [28].

APPENDICES

A. DETAILS OF THE BLOCK-LEVINSON ALGORITHM

Because a block-Toeplitz matrix does not have the right kind of symmetry for the original Levinson algorithm, the Block-Levinson algorithm is based on an extended version of the Levinson algorithm that can work with non-symmetric matrices.

This variant of the Levinson algorithm solves two related prediction problems simultaneously and derives the desired solution from them, whereas the original Levinson algorithm needs to solve only one underlying prediction problem [8]. The block algorithm can then be derived by replacing the scalar operations of the non-block algorithm with matrix operations on blocks.

To present the resulting block algorithm, we will use *colon notation* and *block notation*, which are defined as follows for vectors.

Let

$$\boldsymbol{a} = \begin{bmatrix} a_1 & a_2 & a_3 & \cdots & a_n \end{bmatrix}$$

then

$$\begin{aligned}
\boldsymbol{a}(i:j:k) &:= \begin{bmatrix} a_i & a_{i+j} & a_{i+2j} & \cdots & a_k \end{bmatrix} \\
\boldsymbol{a}(i:k) &:= \boldsymbol{a}(i:1:k) \\
\boldsymbol{a}(i) &:= \boldsymbol{a}(i:i) \\
\boldsymbol{a}(:) &:= \boldsymbol{a}(1:n) \\
\boldsymbol{a}_{[n]}(i) &:= \boldsymbol{a}((i-1)n+1:in) \\
\boldsymbol{a}_{[n]}(i:j:k) &:= \begin{bmatrix} \boldsymbol{a}_{[n]}(i) & \boldsymbol{a}_{[n]}(i+j) & \boldsymbol{a}_{[n]}(i+2j) & \cdots & \boldsymbol{a}_{[n]}(k) \end{bmatrix} \\
\boldsymbol{a}_{[n]}(i:k) &:= \boldsymbol{a}_{[n]}(i:1:k)
\end{aligned}$$

This notation is extended to matrices in the obvious way, by treating row and column indices separately. We also use the additional rule $\boldsymbol{A}_{[n]}(i,j) := \boldsymbol{A}_{[n,n]}(i,j)$, i.e., when only one block size is specified, it is used for both rows and columns.

Block-Levinson algorithm. Starting from $\boldsymbol{S}$ and $\boldsymbol{y}$, the Block-Levinson algorithm computes $\hat{\boldsymbol{d}} = \boldsymbol{S}^{-1}\boldsymbol{y}$ according to the following procedure.

$\boldsymbol{\beta} \leftarrow \boldsymbol{S}_{[K]}(1,1)$
$\boldsymbol{\gamma} \leftarrow \boldsymbol{\beta}$
$\boldsymbol{\alpha} \leftarrow -\boldsymbol{\beta}^{-1}\boldsymbol{S}_{[K]}(2,1)^H$
$\boldsymbol{\nu} \leftarrow -\boldsymbol{\gamma}^{-1}\boldsymbol{S}_{[K]}(2,1)$

$\hat{\boldsymbol{d}} \leftarrow \boldsymbol{\beta}^{-1}\boldsymbol{y}_{[K]}(1)$
$\boldsymbol{Y} \leftarrow \boldsymbol{\alpha}$
$\boldsymbol{W} \leftarrow \boldsymbol{\nu}$

for $k = 1 : N-1$
  $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta}(\boldsymbol{I} - \boldsymbol{\alpha}\boldsymbol{\nu})$
  $\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma}(\boldsymbol{I} - \boldsymbol{\nu}\boldsymbol{\alpha})$
  $\boldsymbol{\mu} \leftarrow \boldsymbol{\gamma}^{-1}(\boldsymbol{y}_{[K]}(k+1) - \sum_{i=2}^{k+1} \boldsymbol{S}_{[K]}(i,1)\hat{\boldsymbol{d}}_{[K]}(k-i+2))$
  $\hat{\boldsymbol{d}} \leftarrow \begin{bmatrix} \hat{\boldsymbol{d}} + \boldsymbol{Y}_{[K]}(k:-1:1,1)\boldsymbol{\mu} \\ \boldsymbol{\mu} \end{bmatrix}$
  if $k < N-1$
    $\tilde{\boldsymbol{Y}} \leftarrow \boldsymbol{Y}$
    $\boldsymbol{\alpha} \leftarrow \boldsymbol{\beta}^{-1}(\boldsymbol{S}_{[K]}(k+2,1)^H + \sum_{i=2}^{k+1} \boldsymbol{S}_{[K]}(i,1)^H \boldsymbol{Y}_{[K]}(k-i+2))$
    $\boldsymbol{Y} \leftarrow \begin{bmatrix} \boldsymbol{Y} + \boldsymbol{W}_{[K]}(k:-1:1)\boldsymbol{\alpha} \\ \boldsymbol{\alpha} \end{bmatrix}$
    $\boldsymbol{\nu} \leftarrow \boldsymbol{\gamma}^{-1}(\boldsymbol{S}_{[K]}(k+2,1) + \sum_{i=2}^{k+1} \boldsymbol{S}_{[K]}(i,1)\tilde{\boldsymbol{Y}}_{[K]}(k-i+2))$
    $\boldsymbol{W} \leftarrow \begin{bmatrix} \boldsymbol{W} + \tilde{\boldsymbol{Y}}_{[K]}(k:-1:1)\boldsymbol{\nu} \\ \boldsymbol{\nu} \end{bmatrix}$
  end
end

B. Details of the Block-Fourier Algorithm

In addition to the colon and block notation from the previous appendix, let $\mathbf{0}_{(n)}$ denote the zero vector of length n and $\mathbf{0}_{(n,m)}$ the zero matrix of size $n \times m$. We can then describe the Block-Fourier algorithm with overlapping as follows.

Block-Fourier algorithm. Starting from $\hat{\boldsymbol{T}}$ and $\boldsymbol{x}$, the Block-Fourier algorithm computes $\hat{\boldsymbol{d}} = (\hat{\boldsymbol{T}}^H \hat{\boldsymbol{T}})^{-1} \hat{\boldsymbol{T}}^H \boldsymbol{x}$ according to the following procedure. It will use FFTs of length D and use a prelap of p^- and a postlap of p^+.

$p \leftarrow D - p^- - p^+$
$L \leftarrow \lceil N/p \rceil$
$\hat{\boldsymbol{S}} \leftarrow \hat{\boldsymbol{T}}^H \hat{\boldsymbol{T}}$
$\hat{\boldsymbol{x}} \leftarrow \begin{bmatrix} \mathbf{0}_{(p^-)} & \boldsymbol{x}^T & \mathbf{0}_{(Lp+p^+-M(NQ+W-1))} \end{bmatrix}^T$
$\boldsymbol{\Lambda} \leftarrow \text{diag}_{(P,K)} \boldsymbol{F}_{(P)} \hat{\boldsymbol{T}}(:, 1:K)$
$\boldsymbol{\Sigma} \leftarrow \text{diag}_{(K,K)} \boldsymbol{F}_{(K)} \hat{\boldsymbol{S}}(:, 1:K)$
$\boldsymbol{R} \leftarrow \text{chol}(\boldsymbol{\Sigma})$
for $\ell = 1:L$
  $\boldsymbol{x}^{(\ell)} \leftarrow \hat{\boldsymbol{x}}((\ell-1)pP + 1 : ((\ell-1)p + D)P)$
  $\boldsymbol{d}^{(\ell)} \leftarrow \boldsymbol{F}_{(K)}^{-1} \boldsymbol{R}^{-1} \boldsymbol{R}^{-H} \boldsymbol{\Lambda}^H \boldsymbol{F}_{(P)} \boldsymbol{x}^{(\ell)}$
  $\hat{\boldsymbol{d}}((\ell-1)pK + 1 : \ell pK) \leftarrow \boldsymbol{d}^{(\ell)}(p^- : p^- + p)$
end

REFERENCES

[1] M. Haardt, A. Klein, R. Koehn, S. Oestreich, M. Purat, V. Sommer, and T. Ulrich, "The TD-CDMA based UTRA TDD mode," *IEEE J. Select. Areas Commun.*, vol. 18, pp. 1375–1386, August 2000, special issue on "Wideband CDMA".

[2] S. Verdu, *Multiuser Detection*, Cambridge University Press, 1998.

[3] 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.

[4] 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.

[5] 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.

[6] Bernd Steiner and Paul Walter Baier, "Low Cost Channel Estimation in the Uplink Receiver of CDMA Mobile Radio Systems," *FREQUENZ*, vol. 47, 1993.

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

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

[9] J. Rissanen and L. Barbosa, "Properties of Infinite Covariance Matrices and Stability of Optimum Predictors," *Information Sciences*, vol. 1, pp. 221–236, 1969.

[10] Y. Pigeonnat, "Alternative solutions for joint detection in TD/CDMA multiple access scheme for UMTS," in *1999 2nd IEEE Workshop in Signal Processing Advances in Wireless Communications*, 1999, pp. 329–332.

[11] 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.

[12] Benjamin Bauer, "Block-Levinson Algorithms for TD-CDMA," Tech. Rep., University of Dortmund, 2000, unpublished.

[13] W. Lee, J. Nam, and C. Ryu, "Joint Beamformer-RAKE and Decorrelating Multiuser Detector Using Matrix Levinson Polynomials," *IEICE Trans. on Communications*, vol. E83-B, no. 8, August 2000, special issue on Personal, Indoor and Mobile Radio Communications.

[14] 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.

[15] 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.

[16] M. Vollmer, M. Haardt, and J. Götze, "Schur algorithms for Joint Detection in TD-CDMA based mobile radio systems," *Annals of Telecommunications (special issue on multi user detection)*, vol. 54, no. 7-8, pp. 365–378, July-August 1999.

[17] W. Yang, S. Y. Kim, G. Xu, and A. Kavak, "Fast joint detection with cyclic reduction exploiting displacement structures," in *ICASSP*, 2000, vol. 5, pp. 2857–2860.

[18] M. Vollmer, J. Götze, and M. Haardt, "Joint Detection in Mobile Radio Systems using FFT," in *Proc. International Conference on Telecommunications, ICT*, Cheju, Korea, 1999.

[19] M. Haardt and W. Mohr, "The complete solution for third-generation wireless communications: Two modes on air, one winning strategy," *IEEE Personal Communications Magazine*, pp. 6–12, Dec. 2000.

[20] C.F. Van Loan, *Computational Frameworks for the Fast Fourier Transform*, SIAM Publications, Philadelphia, PA, 1992.

[21] L. Eldén and E. Sjöström, "Fast computation of the principle singular vectors of Toeplitz matrices arising in exponential data modelling," *Signal Processing*, vol. 50, no. 1-2, pp. 151–164, April 1996.

[22] S. Haykin, *Adaptive Filter Theory*, Prentice Hall, third edition, 1996.

[23] Radiocommunication study groups, "Guidelines for evaluation of radio transmission technologies for FPLMTS," Draft, ITU, April 1996.

[24] Working Document towards ETR/SMG-50402, "Selection Procedures for the choice of radio transmission technologies of the Universal Mobile Telecommunications System (UMTS)," Version 0.9.4, ETSI, May 1997.

[25] 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.

[26] M. Vollmer, "Programs for computing computational requirements of four JD algorithms," available online at http://www-dt.e-technik.uni-dortmund.de/mitarbeiter/mvo/compreq.html.

[27] P. Dewilde and E.F. Deprettere, "The generalized Schur algorithm: Approximation and hierarchy," in *In Operator Theory: Advances and Applications*. 1988, pp. 97–116, Birkhäuser Verlag.

[28] P.A. Regalia and M.G. Bellanger, "On the Duality Between Fast QR Methods and Lattice Methods in Least Squares Adaptive Filtering," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 39, pp. 879–891, 1991.

[29] G. Ammar and W.B. Gragg, "Superfast Solution of Real Positive Definite Toeplitz Systems," *SIAM J. Matrix Anal. Appl.*, vol. 9, pp. 61–76, 1988.

LIST OF FIGURES

1 Frame structure of the TD-CDMA system. Here, 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. . . . 25
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. 25
3 The structure of $\boldsymbol{T}$. In this example, $N = 5$ and $\lceil P/Q \rceil = 3$. 26
4 The structure of $\boldsymbol{S}$. Only the dark shaded part needs to be computed. 26
5 Approximating $\boldsymbol{R}$. Only the dark shaded part is computed. 26
6 Estimation errors when using overlap-save techniques. $Q = 16$, $W = 57$, $N = 69$, $K = 4$, $D = 32$, no noise. 27
7 Overlapping Convolution Matrices 27
8 Coded Bit Error Rate for all four Algorithms. 27
9 Computational Requirements for the Cholesky Algorithm. 28
10 Computational Requirements for the Block-Levinson Algorithm. 28
11 Computational Requirements for the Block-Schur Algorithm. 29
12 Computational Requirements for the Block-Fourier Algorithm. 29
13 Computational Requirements for all four Algorithms. 30
14 Computational Requirements for all four Algorithms, against the number of symbols, N. . . . 30
15 Computational Requirements for all four Algorithms, against the number of antennas, M. . . . 31

LIST OF TABLES

I Tap delays and relative average powers of the Vehicular A channel model. 15

II E_b/N_0 and SNR values of all $K = 8$ users in dB. 16

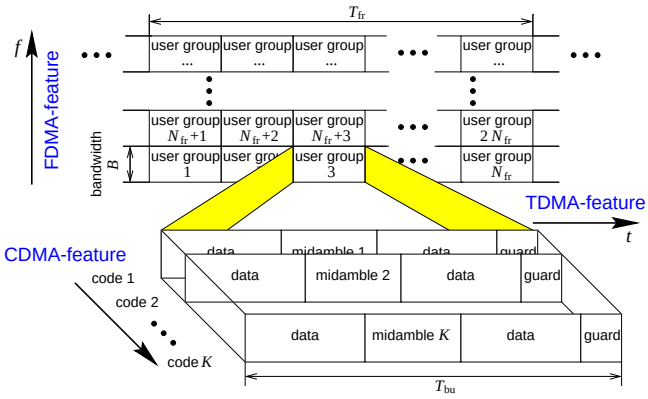


Fig. 1. Frame structure of the TD-CDMA system. Here, B, T_{fr}, N_{fr}, T_{bu}, and K denote the bandwidth of a frequency slot, the duration of a TDMA frame, the number of bursts per TDMA frame, the burst duration, and the number of CDMA codes per frequency and time slot, respectively.

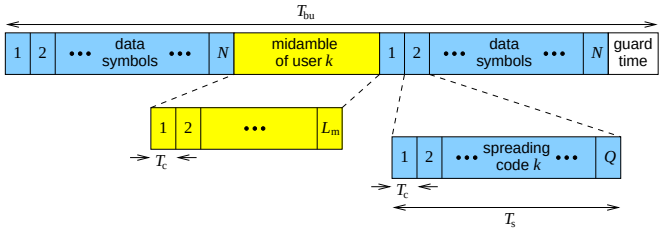


Fig. 2. Time slot structure of the TD-CDMA system. Here, T_{bu}, T_{s} T_{c}, and Q denote the burst duration, the symbol duration, the chip duration, and the spreading factor of the data symbols, respectively.

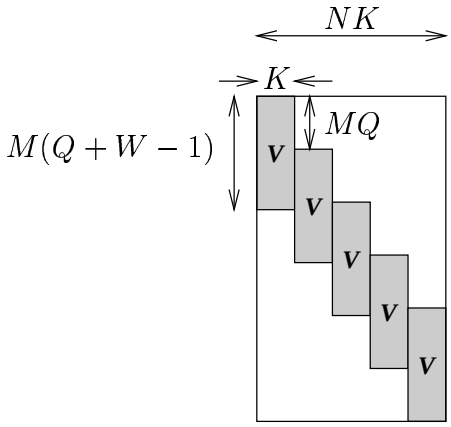


Fig. 3. The structure of $\boldsymbol{T}$. In this example, $N = 5$ and $\lceil P/Q \rceil = 3$.

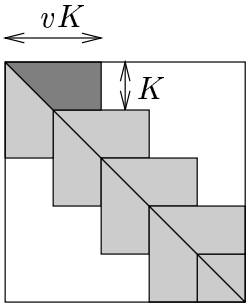
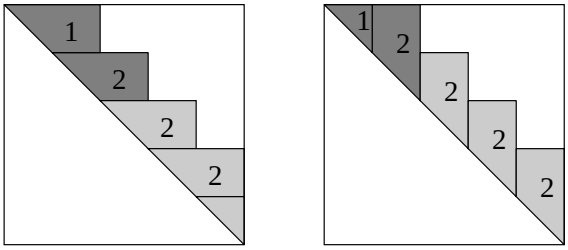


Fig. 4. The structure of $\boldsymbol{S}$. Only the dark shaded part needs to be computed.



(a) Row-wise (b) Column-wise

Fig. 5. Approximating $\boldsymbol{R}$. Only the dark shaded part is computed.

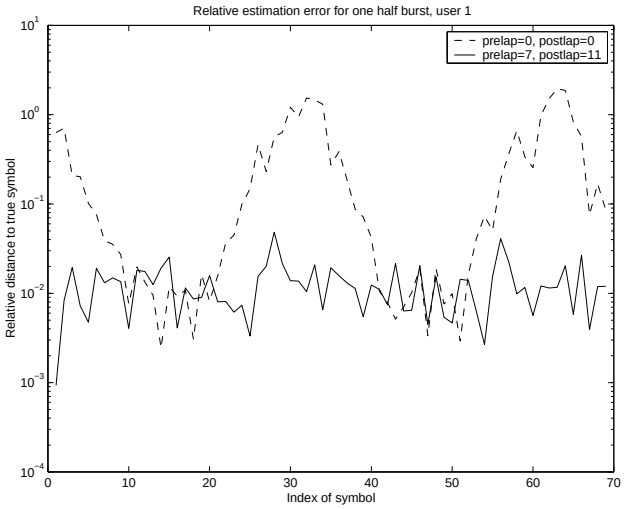


Fig. 6. Estimation errors when using overlap-save techniques. $Q = 16$, $W = 57$, $N = 69$, $K = 4$, $D = 32$, no noise.

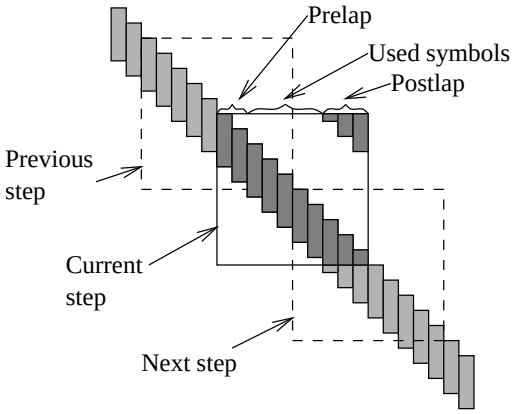


Fig. 7. Overlapping Convolution Matrices

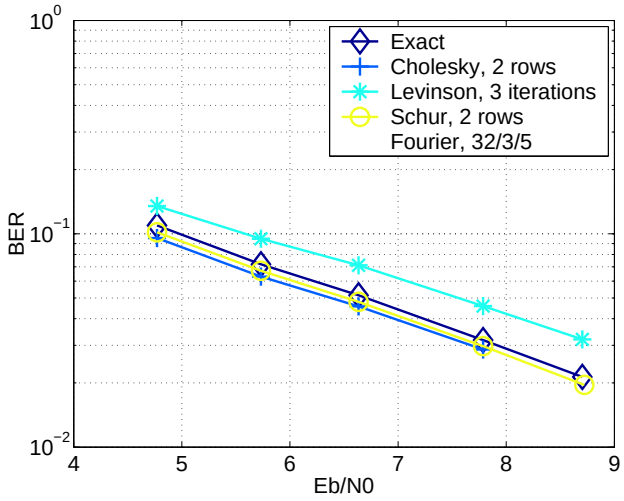


Fig. 8. Coded Bit Error Rate for all four Algorithms.

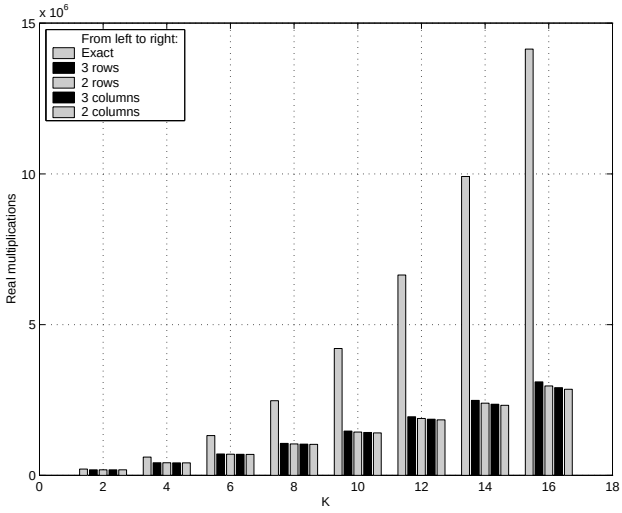


Fig. 9. Computational Requirements for the Cholesky Algorithm.

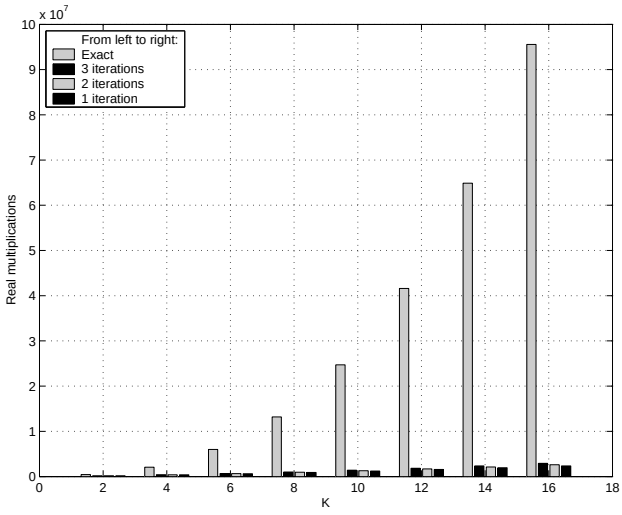


Fig. 10. Computational Requirements for the Block-Levinson Algorithm.

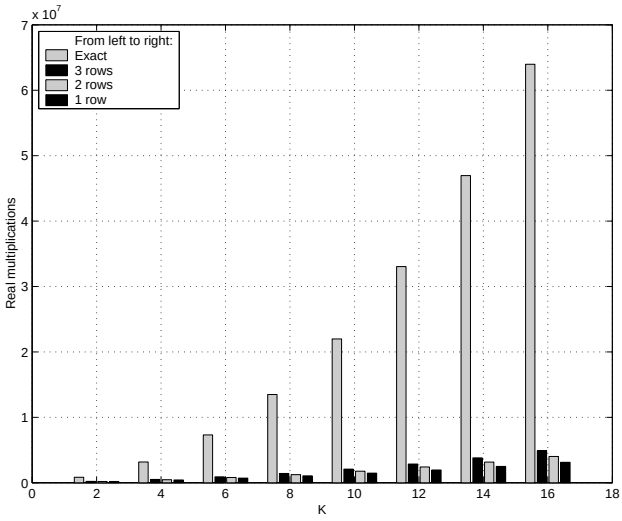


Fig. 11. Computational Requirements for the Block-Schur Algorithm.

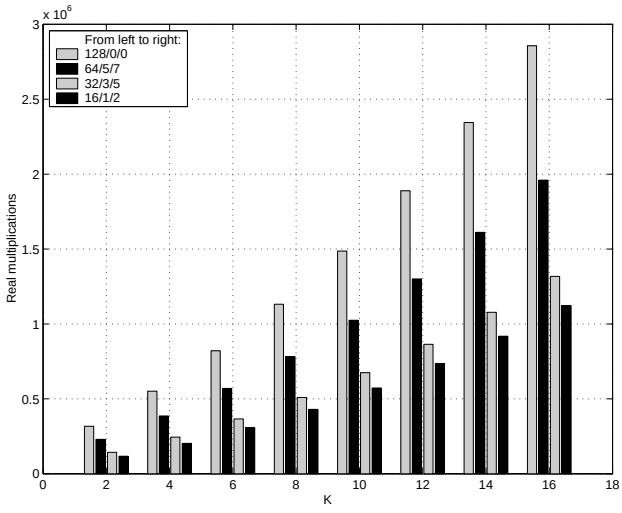


Fig. 12. Computational Requirements for the Block-Fourier Algorithm.

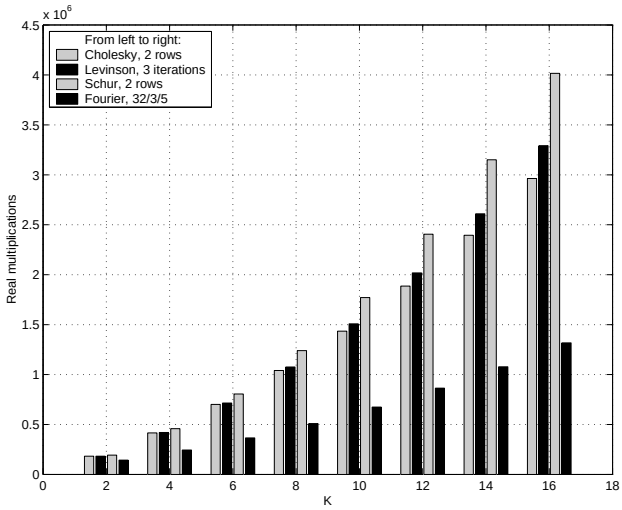


Fig. 13. Computational Requirements for all four Algorithms.

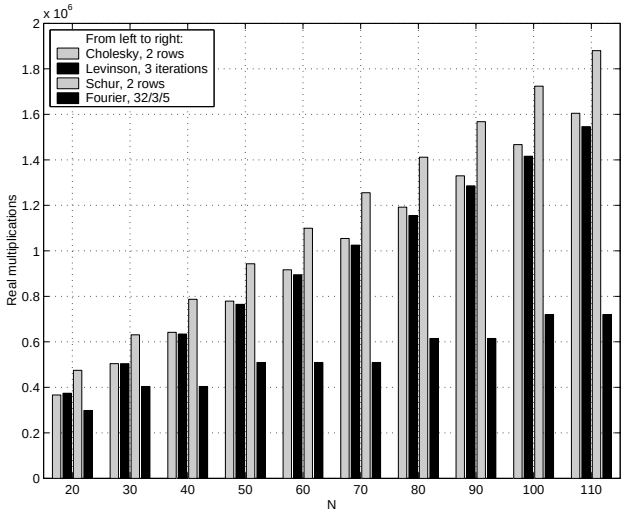


Fig. 14. Computational Requirements for all four Algorithms, against the number of symbols, N.

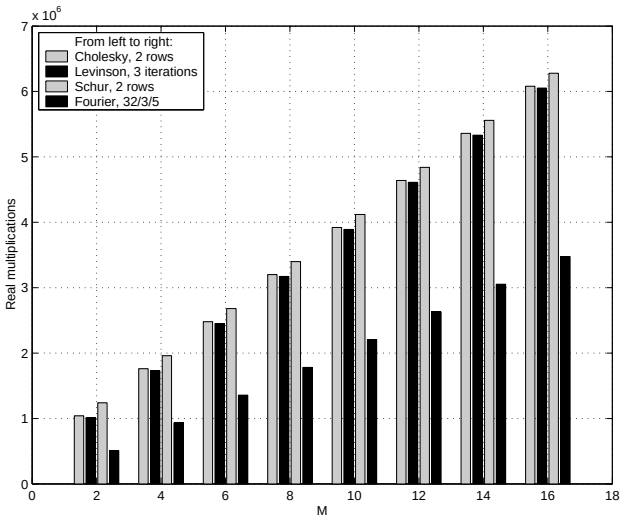


Fig. 15. Computational Requirements for all four Algorithms, against the number of antennas, M.