Exploiting the Structure of the Joint Detection Problem with Decision Feedback

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

1. Siemens AG, ICN CA CTO 71
Communication on Air
Hofmannstr. 51
D-81359 Munich, Germany
Martin.Haardt@icn.siemens.de
Marius.Vollmer@icn.siemens.de

2. Information Processing Lab
University of Dortmund
Otto-Hahn-Str. 4
D-44221 Dortmund, Germany
goetze@dt.e-technik.uni-dortmund.de
mvo@dt.e-technik.uni-dortmund.de

Performing joint detection in a TD-CDMA system corresponds to the solution of a large system of linear equations, whose matrix exhibits a strong band and Toeplitz structure. Taking advantage of the structure allows a cheap solution by standard algorithms (Cholesky, QR-decomposition). Including decision feedback into the solution process improves the performance of the system. This, however, leads to a loss of the beneficial band structure. Furthermore, the known approximation schemes can no longer be applied. In this paper, we present a Schur-type algorithm that is able to exploit the remaining Toeplitz structure of the decision feedback system. A new approximation scheme is developed that can successfully reduce the computational complexity of this algorithm.

Keywords: TD-CDMA, joint detection, decision feedback, Toeplitz structure, Schur-type algorithm, approximate solution

Topics: 11. Migration to 3rd generation, 19. Multiuser detection

1 Introduction

Third generation mobile radio systems will incorporate a CDMA component to share the available radio resources among participating users. To overcome the near/far problem of traditional CDMA systems, receiver structures are proposed that perform joint (or multi-user) detection [5, 6]. A joint detector combines the knowledge about all active users 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. The system of equations can be very large and thus algorithms must be developed that can exploit its special structural characteristics.

Figure 1 shows how a CDMA radio system can be modelled. Both the spreading and the channel are treated as time-invariant convolutions of the up-sampled data symbols. For a TD-CDMA system with burst structure, this leads to a mathematical representation of the form

$$\boldsymbol{T}\boldsymbol{d} + \boldsymbol{n} = \boldsymbol{e}, \tag{1}$$

where the system matrix $\boldsymbol{T}$ contains information about the known spreading codes $\boldsymbol{c}^{(k)}$ and the estimated channel impulse responses $\boldsymbol{h}^{(k)}$. The data vector $\boldsymbol{d}$ contains the unknown data symbols of all users, $\boldsymbol{n}$ represents noise, and $\boldsymbol{e}$ contains the antenna samples that have been received.

This system of linear equations is usually overdetermined and is thus solved with some kind of least squares method, for example, with a QR decomposition $\boldsymbol{T} = \boldsymbol{Q}\boldsymbol{R}$ that leads to

$$\boldsymbol{R}\boldsymbol{d} = \boldsymbol{Q}^H\boldsymbol{e} \tag{2}$$

or with a Cholesky decomposition of $\boldsymbol{T}^H\boldsymbol{T} = \boldsymbol{R}^H\boldsymbol{R}$ for the normal equation that leads to

$$\boldsymbol{R}^H\boldsymbol{R}\boldsymbol{d} = \boldsymbol{T}^H\boldsymbol{e}, \tag{3}$$

where $\boldsymbol{R}$ is an upper triangular matrix. Therefore, the resulting equations can be solved with one or two back substitution steps, respectively. The matrix $\boldsymbol{R}$ is called the Cholesky factor of $\boldsymbol{T}^H\boldsymbol{T}$.

The system matrix $\boldsymbol{T}$ and thus $\boldsymbol{R}$ are quite large for the system parameters and burst sizes proposed for the third generation mobile radio systems [4]. Fortunately, $\boldsymbol{T}$ can be arranged so that it exhibits a strong band structure and is block-Toeplitz. Figure 2 shows a typical arrangement of $\boldsymbol{T}$ and related matrices. The band structure is present in all three matrices. This band structure can be easily exploited while computing $\boldsymbol{R}$ and thus the computational complexity of the algorithms is reduced considerably. Furthermore, both $\boldsymbol{T}$ and $\boldsymbol{T}^H\boldsymbol{T}$ are block-Toeplitz. Together with the strong band structure, this leads to an approximate block-Toeplitz structure of $\boldsymbol{R}$. It is possible to exploit this by only computing a small part of $\boldsymbol{R}$ and filling the rest with copies of elements from this part [8].

To improve the bit error rate performance of the transmission system, it has been proposed to include decision feedback into the process [5, 6]. This decision feedback is to be performed in the last back-substitution step of solving (2) or (3), i.e., while solving

$$\boldsymbol{R}\boldsymbol{d} = \boldsymbol{z} \tag{4}$$

where either $\boldsymbol{z} = \boldsymbol{Q}^H\boldsymbol{e}$ or $\boldsymbol{R}^H\boldsymbol{z} = \boldsymbol{T}^H\boldsymbol{e}$. Figure 3 shows the procedure used for solving (4). As can be

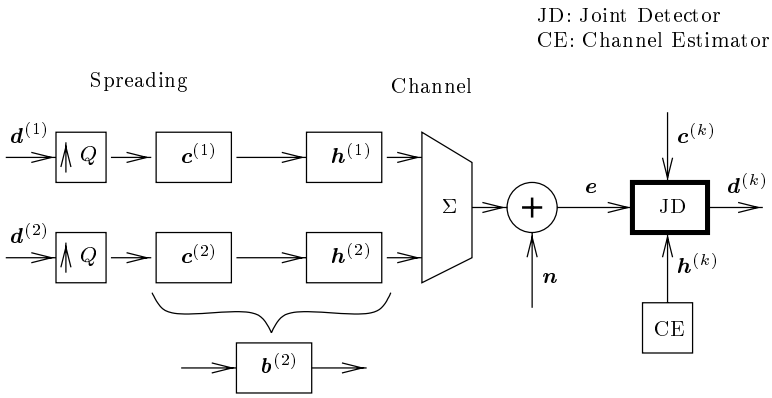
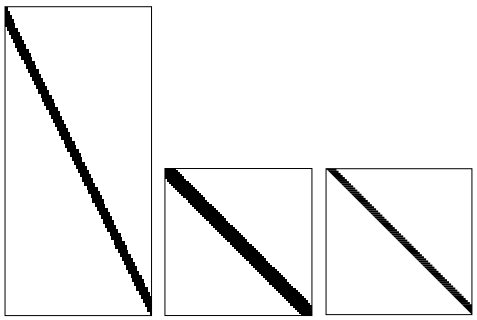


Figure 1: Model of a CDMA transmission system for two users



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

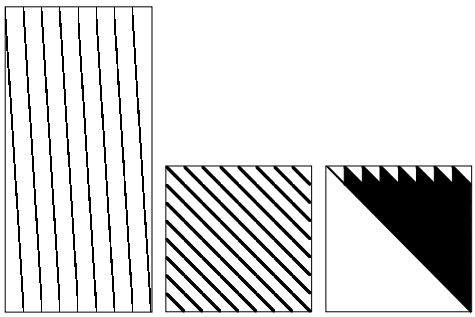
Figure 2: The sparseness of typical matrices. The black regions represent non-zero matrix elements. $N = 69$, $K = 8$, $Q = 16$, $W = 60$

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 $\boldsymbol{d}$.

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

Figure 3: Performing decision feedback while back substituting. The function $\mathtt{map}(x)$ maps x to the nearest member of the symbol alphabet, that is, it implements the decision.

This in turn dictates a different way of arranging the spreading codes and channel impulse responses in $\boldsymbol{T}$. Figure 4 shows this new arrangement for the same system parameters as in figure 2. It can be seen that the Cholesky factor $\boldsymbol{R}$ looses its band structure. Additionally, its approximate Toeplitz structure is much more complicated. This is due to the fact that $\boldsymbol{T}$ and $\boldsymbol{T}^H\boldsymbol{T}$ are no longer block-Toeplitz matrices but Toeplitz-block matrices. Consequently, the computational complexity explodes when applying a band-structure optimised algorithm to this variant of the joint detection problem.



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

Figure 4: The sparseness of the matrices for decision feedback. The black regions represent non-zero matrix elements. $N = 69$, $K = 8$, $Q = 16$, $W = 60$

After detailing the system model of the decision feedback system, this paper shows how to derive a generalised Schur algorithm that can cope with this specific kind of Toeplitz-block structure of the decision feedback problem (section 3) and how to generalise the approximation techniques to it (section 4). In section 5, we present comparisons of the computational complexity of two algorithms, and then conclude.

2 System Model

The mathematical description of the transmission system is based on the model depicted in figure 1. Since we are dealing with a burst structured system where individual data blocks are separated by a guard period, we can treat each block separately. Furthermore, it is assumed that the channel can be modelled by a time-invariant FIR filter. Likewise, each symbol will

be spread with the same spreading code so that this spreading can also be modelled by a time-invariant convolution, combined with a suitable up-sampling of the data sequence. The two convolutions can be combined into one with the impulse response vector

$$\boldsymbol{b}^{(k)} = \boldsymbol{c}^{(k)} \star \boldsymbol{h}^{(k)}$$

where $\boldsymbol{c}^{(k)} \in \mathbb{C}^{Q}$, $\boldsymbol{h}^{(k)} \in \mathbb{C}^{W}$, $\boldsymbol{b}^{(k)} \in \mathbb{C}^{Q+W-1}$ and $k = 1 \ldots K$. The $\star$ denotes the convolution of two finite sequences. The spreading factor is denoted by Q, the channel impulse response length is W, and K is the number of active users.

Let $\boldsymbol{d}^{(k)}$ be the data symbol sequence of user k and $\mathcal{S}_Q(\boldsymbol{x})$ denote the result of up-sampling $\boldsymbol{x}$ by a factor Q but without the last $Q-1$ zeros. Then we can express the vector of the received samples $\boldsymbol{e}$ as

$$\boldsymbol{e} = \sum_{k=1}^{K} \left(\boldsymbol{b}^{(k)} \star \mathcal{S}_Q(\boldsymbol{d}^{(k)})\right) + \boldsymbol{n}. \tag{5}$$

Since we drop the last $Q-1$ zeros after the up-sampling, we have $\boldsymbol{e} \in \mathbb{C}^{NQ+W-1}$. Likewise, $\boldsymbol{n} \in \mathbb{C}^{NQ+W-1}$. Here, N is the number of symbols in $\boldsymbol{d}^{(k)}$.

Equation (5) needs to be rewritten into the matrix form of (1). There are many ways to do this, two of which are outlined in figures 2 and 4.

The desire to perform decisions for the strongest user first leads to the following arrangement of the vector $\boldsymbol{d}$:

$$\boldsymbol{d} = \begin{bmatrix} \boldsymbol{d}^{(1)T} & \boldsymbol{d}^{(2)T} & \cdots & \boldsymbol{d}^{(K)T} \end{bmatrix}^T$$

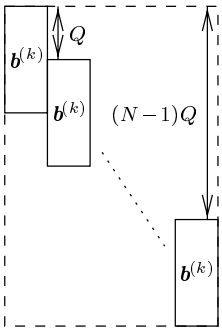
Here, $\boldsymbol{d}^{(1)}$ are the symbols of the weakest user and $\boldsymbol{d}^{(K)}$ belongs to the strongest user. One can use the channel impulse responses to estimate the relative powers of the received signals.

This particular arrangement of $\boldsymbol{d}$ in turn leads to the arrangement of $\boldsymbol{T}$ pictured in figure 4:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{V}^{(1)} & \boldsymbol{V}^{(2)} & \cdots & \boldsymbol{V}^{(K)} \end{bmatrix} \tag{6}$$

with

$$\boldsymbol{V}^{(k)} = \tag{7}$$



Note that $\boldsymbol{V}^{(k)}$ contains multiple copies of $\boldsymbol{b}^{(k)}$. This is due to the time-invariance of both the channel and the spreading codes and leads to the already mentioned Toeplitz-block structure of $\boldsymbol{T}^H\boldsymbol{T}$.

When the receiver has K_A antennae, we can combine the system equations for them into a single system of equations with the same structure as presented in (6) and (7).

3 Schur algorithm for decision feedback

There exists a very general way to describe various forms and degrees of the Toeplitz structure of a matrix: the displacement representation [7, 1].

The displacement representation consists of finding a way to remove as much redundancy as possible from a matrix. For the positive definite, Hermitian matrix $\boldsymbol{S} = \boldsymbol{T}^H\boldsymbol{T}$, the displacement representation can be expressed as

$$\boldsymbol{S} - \boldsymbol{Z}\boldsymbol{S}\boldsymbol{Z}^T = \sum_{i=1}^{K} \boldsymbol{\alpha}_i^H \boldsymbol{\alpha}_i - \sum_{i=1}^{K} \boldsymbol{\beta}_i^H \boldsymbol{\beta}_i. \tag{8}$$

The vectors $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ are $1 \times NK$ row vectors, i.e., the product $\boldsymbol{\alpha}_i^H\boldsymbol{\alpha}_i$ is a $NK \times NK$ matrix.

The matrix $\boldsymbol{S} = \boldsymbol{T}^H\boldsymbol{T}$ consists of $K \times K$ blocks that are each of size $N \times N$ and internally Toeplitz. The matrix $\boldsymbol{S}$ is always Hermitian and positive definite if $\boldsymbol{T}$ is not rank deficient. The chances of $\boldsymbol{T}$ being rank-deficient in realistic scenarios are negligible, however.

The advantage of the Schur-type algorithms for structured matrices is that it is sufficient to work with the row vectors $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ (instead of the full matrix $\boldsymbol{S}$) to compute $\boldsymbol{R}$.

Finding a displacement representation The first step is to find a shift matrix $\boldsymbol{Z}$ that removes as much redundancy as possible from $\boldsymbol{S}$. For a Toeplitz-block structured $\boldsymbol{S}$ this is the block-lower-right-shift matrix

$$\boldsymbol{Z} = \boldsymbol{I}_K \otimes \boldsymbol{L}_N \tag{9}$$

where $\boldsymbol{I}_K$ is the $K \times K$ identity matrix, $\otimes$ is the Kronecker product, and $\boldsymbol{L}_N$ is the $N \times N$ lower-right shift matrix defined as

$$(\boldsymbol{L}_N)_{ij} = \begin{cases} 1 & \text{for } i-j=1, \\ 0 & \text{else.} \end{cases}$$

Computing $\boldsymbol{S} - \boldsymbol{Z}\boldsymbol{S}\boldsymbol{Z}^T$ with this definition simply replaces all elements of $\boldsymbol{S}$ with zero that do not lie on the first row or column of one of its $N \times N$ blocks, i.e., it removes the redundancy due to the Toeplitz-block structure. Figure 5 shows the result. This grid needs to be expressed with $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ according to (8). We can do this by using one $(\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i)$ pair for each pair of a column and a row that cross on the diagonal.

Algorithm GEN in figure 6 computes these pairs step by step. After one pair has been computed, its corresponding column/row pair is removed from the grid to prepare it for the next step.

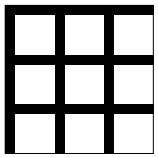


Figure 5: $\boldsymbol{S} - \boldsymbol{Z}\boldsymbol{S}\boldsymbol{Z}^T$. The black regions represent non-zero matrix elements.

$\boldsymbol{D} = \boldsymbol{S} - \boldsymbol{Z}\boldsymbol{S}\boldsymbol{Z}^T$
for i **from** 1 **to** 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

Figure 6: Algorithm GEN.

Schur algorithm for displacement representation The displacement representation (8) can be rewritten to directly express $\boldsymbol{S}$. We do this by introducing *Krylov* matrices. The Krylov matrix $\mathcal{U}(\boldsymbol{\xi})$ of a row vector $\boldsymbol{\xi}$ with respect to $\boldsymbol{Z}$ is for our purposes a $NK \times NK$ matrix defined as

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

Note that $\boldsymbol{\mathcal{U}}(\boldsymbol{\xi})$ is an upper triangular matrix. With this definition, (8) can be rewritten into

$$\boldsymbol{S} = \sum_{i=1}^{K} \mathcal{U}(\boldsymbol{\alpha}_i)^H \mathcal{U}(\boldsymbol{\alpha}_i) - \sum_{i=1}^{K} \mathcal{U}(\boldsymbol{\beta}_i)^H \mathcal{U}(\boldsymbol{\beta}_i). \qquad (10)$$

The goal of the Schur-type algorithm is now to gradually eliminate the contributions of $\mathcal{U}(\boldsymbol{\alpha}_2)$, $\mathcal{U}(\boldsymbol{\alpha}_3)$, ..., $\mathcal{U}(\boldsymbol{\alpha}_K)$ and $\mathcal{U}(\boldsymbol{\beta}_1), \mathcal{U}(\boldsymbol{\beta}_2), \ldots, \mathcal{U}(\boldsymbol{\beta}_K)$ so that only the first term of the first sum remains. The Schur algorithm preserves the upper triangular structure of the matrices. Once the elimination is complete, it is easy to see that the only remaining matrix must be the desired Cholesky factor $\boldsymbol{R}$:

$$\boldsymbol{S} = \boldsymbol{R}^H \boldsymbol{R}. \qquad (11)$$

The gradual transformation must preserve the validity of (10) in each step. This leads to the use of unitary and complex hyperbolic elementary rotations [2, 3]. For the 2×2 case, they are defined as

$$\mathcal{G}(a,b) = \frac{1}{\sqrt{1+|\rho|^2}} \begin{bmatrix} 1 & \rho^* \\ -\rho & 1 \end{bmatrix}, \qquad \rho = \frac{b}{a},$$

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

The unitary rotation $\mathcal{G}(a,b)$ is used to perform the transformation when both transformed rows come from the matrices formed by $\boldsymbol{\alpha}_i$, or both come from those formed by $\boldsymbol{\beta}_i$. The complex hyperbolic transformation $\mathcal{H}(a,b)$ is used when one row comes from the $\boldsymbol{\alpha}_i$ region and the other comes from the $\boldsymbol{\beta}_i$ region [3].

The way the Schur algorithm exploits the Toeplitz-derived structure of $\boldsymbol{S}$ is by omitting the transformations in the redundant rows of $\mathcal{U}(\boldsymbol{\alpha}_i)$ and $\mathcal{U}(\boldsymbol{\beta}_i)$. Therefore it is sufficient to only work with one row of each of the matrices $\mathcal{U}(\boldsymbol{\alpha}_i)$ and $\mathcal{U}(\boldsymbol{\beta}_i)$ during the whole algorithm.

Computing $\boldsymbol{Q}^H\boldsymbol{e}$ Computing $\boldsymbol{R}$ is not enough to solve (1). We either need to compute $\boldsymbol{T}^H\boldsymbol{e}$ and then perform two back-substitutions, or we need to compute $\boldsymbol{Q}^H\boldsymbol{e}$ and can get by with only one back-substitution step. The Schur algorithm can be extended to compute $\boldsymbol{q}_e = \boldsymbol{Q}^H\boldsymbol{e}$ at the same time it computes $\boldsymbol{R}$.

To this end, we need to find vectors $\boldsymbol{y}_i$ that satisfy

$$\boldsymbol{e}^H\boldsymbol{T} = \sum_{i=1}^{K} \boldsymbol{y}_i^H \big(\mathcal{U}(\boldsymbol{\alpha}_i) - \mathcal{U}(\boldsymbol{\beta}_i)\big).$$

With $\boldsymbol{Z}$ defined as in (9), this equation can be solved by inspection:

$$\boldsymbol{y}_i = \big(\boldsymbol{T}(:, j : j+N-1)/\boldsymbol{\alpha}_i(j)\big)^H \boldsymbol{e},$$
$$j = (i-1)N + 1.$$

The transformations that are used to eliminate $\mathcal{U}(\boldsymbol{\alpha}_2), \ldots, \mathcal{U}(\boldsymbol{\beta}_K)$ can be combined into one transformation matrix $\boldsymbol{\Theta}$, i.e,

$$\boldsymbol{\Theta} \begin{bmatrix} \mathcal{U}(\boldsymbol{\alpha}_1) \\ \vdots \\ \mathcal{U}(\boldsymbol{\alpha}_K) \\ \mathcal{U}(\boldsymbol{\beta}_1) \\ \vdots \\ \mathcal{U}(\boldsymbol{\beta}_K) \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} \\ \vdots \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{bmatrix} \quad \text{such that } \boldsymbol{\Theta} \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_K \\ \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_K \end{bmatrix} = \begin{bmatrix} \boldsymbol{Q}^H\boldsymbol{e} \\ \vdots \\ * \\ * \\ \vdots \\ * \end{bmatrix}$$

where $*$ is used to denote vectors that are of no further interest. That is, we need to apply the same transformations that are used to turn (10) into (11) to the vectors $\boldsymbol{y}_i$. However, we can no longer omit the transformations that have been redundant while transforming the replacement representation alone. To account for this, we flip each $\boldsymbol{y}_i$ into a row vector, paste it onto the back of its corresponding $\boldsymbol{\alpha}_i$ or $\boldsymbol{\beta}_i$ and be careful to shift $\boldsymbol{y}_1$ when $\boldsymbol{\alpha}_1$ gets shifted.

Algorithm SCHUR in figure 7 contains all details.

4 Approximations

In general, the Cholesky factor of a Toeplitz matrix is not Toeplitz itself. However, the sparseness of $\boldsymbol{S}$ leads

for i **from** 1 **to** K
  $\boldsymbol{y}_i^{(a)} \leftarrow \boldsymbol{y}_i^{(b)} \leftarrow \boldsymbol{y}_i$
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
  $\begin{bmatrix} \boldsymbol{\alpha}_1 \ \boldsymbol{y}_1^{(a)T} \\ \boldsymbol{\beta}_1 \ \boldsymbol{y}_1^{(b)T} \end{bmatrix} \leftarrow \mathcal{H}(\boldsymbol{\alpha}_1(j), \boldsymbol{\beta}_1(j)) \begin{bmatrix} \boldsymbol{\alpha}_1 \ \boldsymbol{y}_1^{(a)T} \\ \boldsymbol{\beta}_1 \ \boldsymbol{y}_1^{(b)T} \end{bmatrix}$
  $\boldsymbol{R}(j,:) \leftarrow \boldsymbol{\alpha}_1$
  $\boldsymbol{q}_e(j) \leftarrow \boldsymbol{y}_1^{(a)}(1)$
  $\boldsymbol{\alpha}_1 \leftarrow \boldsymbol{\alpha}_1 \boldsymbol{Z}^T$
  $\boldsymbol{y}_1^{(a)} \leftarrow \begin{bmatrix} \boldsymbol{y}_1^{(a)}(2:N) & 0 \end{bmatrix}$
end

Figure 7: Algorithm SCHUR. The matrices $\boldsymbol{y}_i^{(a)}$ and $\boldsymbol{y}_i^{(b)}$ carry the right hand side through the algorithm.

to an approximate Toeplitz-block structure of $\boldsymbol{R}$ [8]. It is possible to exploit this fact by computing only the relevant part of $\boldsymbol{R}$ and fill the rest with copies from that part. Figure 8 shows how this can be done on a row-by-row basis which is well suited for Schur-type algorithms. 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 9 shows the bit error rate performance of a TD-CDMA system that uses Algorithm SCHUR in the receiver applying the approximations as shown in figure 8.

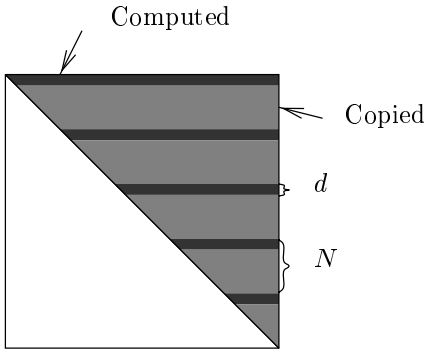


Figure 8: Row-by-row approximation

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

5 Computational Complexity

Given that the Cholesky algorithm cannot exploit the Toeplitz structure of $\boldsymbol{S}$ and that the previously used approximation methods cannot deal with the more complicated approximate Toeplitz-block structure, we get a computational complexity of $O(N^3K^3)$ for the known algorithms.[1]

The generalised 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 amount of inter symbol interference. However, the back-substitution remains of order $O(N^2K^2)$ because there is no band-structure it could exploit.

Figure 10 shows the number of multiplications necessary for performing joint-detection with decision feedback for two algorithms. Note that the figure uses 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 4. 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 bursts consists of two data blocks with N symbols each.

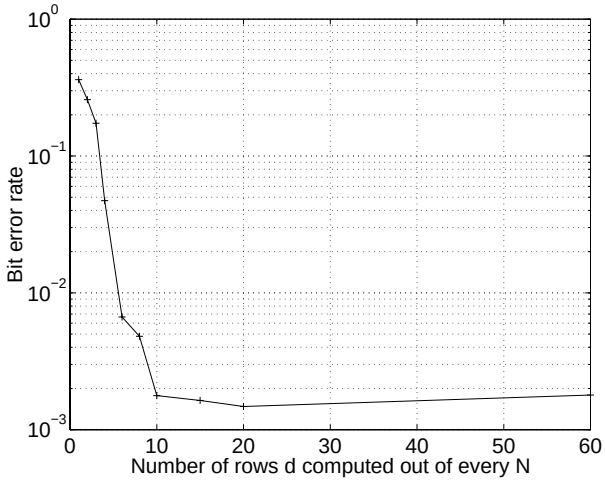


Figure 9: Performance degradation of approximations for a varying number of computed rows, d. $K = 4$, $Q = 16$, $W = 60$, $N = 69$, $K_A = 1$.

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

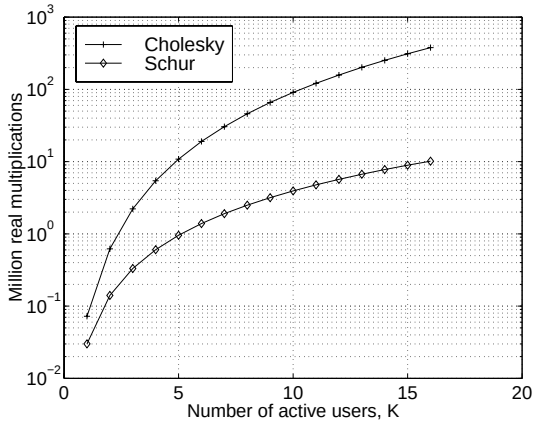


Figure 10: Computational complexity for a varying number of active codes, K. $Q = 16$, $W = 57$, $N = 69$, $d = 10$, $K_A = 1$.

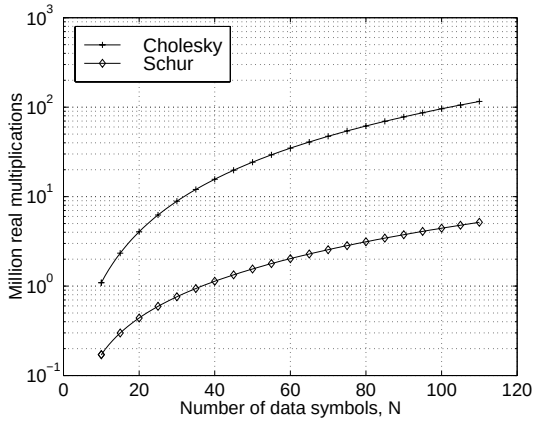


Figure 11: Computational complexity for a varying number of symbols per data block, N. $Q = 16$, $W = 57$, $K = 8$, $d = 10$, $K_A = 1$.

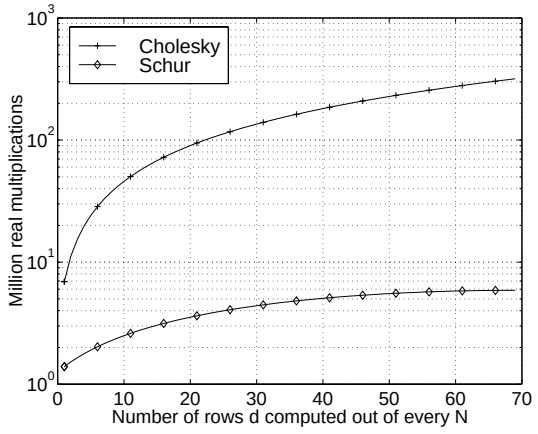


Figure 12: Computational complexity for a varying number of computed rows d out of every N. $Q = 16$, $W = 57$, $K = 8$, $N = 69$, $K_A = 1$.

6 Conclusions

By using a generalised 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 2, 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 4), such that $\boldsymbol{T}$ looses its band structure, the Schur-type algorithms can exploit the remaining Toeplitz-block structure and, therefore, allows 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 realised by generalising the known approximation schemes.

References

[1] J. Chun, T. Kailath, and H. Lev-Ari. Fast Parallel Algorithms for QR and Triangular Factorization. *SIAM J. Sci. Stat. Comput.*, 8(6), November 1987.

[2] Gene H. Golub and Charles F. van Loan. *Matrix Computations*. The John Hopkins University Press, third edition, 1996.

[3] Jürgen Götze and Haesun Park. Schur-type methods based on subspace criteria. In *Proc. IEEE Int. Symp. on Circuits and Systems*, pages 2661–2664, Hong Kong, 1997.

[4] M. Haardt, A. Klein, and J. Schindler. UTRA-TDD Physical Layer Description. In *Proc. 1st inter. symp. on wireless personal multi-media communications (WPMC'98)*, Yokosuka, Japan, Nov. 1998.

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

[6] 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)*, pages 488–492, Yokohama, Japan, September 1993.

[7] T. Kailath and J. Chun. Generalized Displacement Structure for Block-Toeplitz, Toeplitz-Block, and Toeplitz-Derived Matrices. *SIAM J. Matrix Anal. Appl*, 15(1):114–128, January 1994.

[8] J. Mayer, J. Schlee, and T. Weber. Realtime feasibility of joint detection CDMA. In *Proc. 2nd European Personal Mobile Communications Conference*, pages 245–252, Bonn, Germany, September 1997.