OPTIMIZED GAUSS AND CHOLESKY ALGORITHMS FOR USING THE LMMSE DECODER IN MIMO/BLAST SYSTEMS WITH FREQUENCY-SELECTIVE CHANNELS

Reduced-complexity Equalization

João Carlos Silva, Nuno Souto, Francisco Cercas, António Rodrigues
Instituto Superior Técnico/IT, Torre Norte 11-11, Av. Rovisco Pais 1, 1049-001 Lisboa, Portugal

Rui Dinis, Sérgio Jesus
CAPS, Av. Rovisco Pais 1,1049-001 Lisboa, Portugal

Keywords: MMSE Equalizer, Gauss, Cholesky, MIMO, W-CDMA.

Abstract: The LMMSE (Linear Minimum Mean Square Error) algorithm is one of the best linear receivers for DS-CDMA (Direct Sequence-Code Division Multiple Access). However, for the case of MIMO/BLAST (Multiple Input, Multiple Output / Bell Laboratories Layered Space Time), the perceived complexity of the LMMSE receiver is taken as too big, and thus other types of receivers are employed, yielding worse results. In this paper, we investigate the complexity of the solution to the LMMSE and the Zero-Forcing (LMMSE without noise estimation) receiver's equations. It will be shown that the equation can be solved with optimized Gauss or Cholesky algorithms. Some of those solutions are very computationally efficient and thus, allow for the usage of the LMMSE in fully-loaded MIMO systems.

1 INTRODUCTION

Digital communication using MIMO, sometimes called a "volume-to volume" wireless link, has recently emerged as one of the most significant technical breakthroughs in modern communications. Just a few years after its invention the technology is already part of the standards for wireless local area networks (WLAN), third-generation (3G) networks and beyond.

MIMO schemes are used in order to push the capacity and throughput limits as high as possible without an increase in spectrum bandwidth, although there is an obvious increase in complexity. For N transmit and M receive antennas, we have the capacity equation (Foschini, 1998), (Telatar, 1999)

$$C_{EP} = \log_2\left(\det\left(\mathrm{I}_M + \frac{\rho}{N} HH'\right)\right) \text{ b/s/Hz} \qquad (1)$$

where $\boldsymbol{H}$ is the channel matrix, $\boldsymbol{H'}$ is the transpose-conjugate of $\boldsymbol{H}$ and ρ is the SNR at any receive antenna. (Foschini, 1998) and (Telatar, 1999) both demonstrated that the capacity grows linearly with m=min(M,N), for uncorrelated channels.

Therefore, it is possible to augment the capacity/throughput by any factor, depending on the number of transmit and receive antennas. The downside to this is the receiver complexity, sensitivity to interference and correlation between antennas, which is more significant as the antennas are closer together. For a 3G system, for instance, it is inadequate to consider more than 2 or 4 antennas at the UE (User Equipment)/ mobile receiver.

Note that, unlike in CDMA where user's signatures are quasi-orthogonal by design, the separability of the MIMO channel relies on the presence of rich multipath which is needed to make the channel spatially selective. Therefore, MIMO can be said to effectively exploit multipath.

The receiver for such a scheme is obviously complex; due to the number of antennas, users and multipath components, the performance of a simple RAKE/ MF (Matched Filter) receiver (or enhanced schemes based on the MF) always introduces a

Carlos Silva J., Souto N., Cercas F., Rodrigues A., Dinis R. and Jesus S. (2006).
OPTIMIZED GAUSS AND CHOLESKY ALGORITHMS FOR USING THE LMMSE DECODER IN MIMO/BLAST SYSTEMS WITH FREQUENCY-SELECTIVE CHANNELS - Reduced-complexity Equalization.
In *Proceedings of the International Conference on Wireless Information Networks and Systems*, pages 200-208

significant amount of noise, that doesn't allow for the system to perform at full capacity. Thus being, the LMMSE receiver was considered for such cases, acting as an equalizer.

The structure of the paper is as follows. In Section II, the LMMSE receiver for MIMO with multipath is introduced. Section III presents the proposed optimizations to the standard methods of Gauss and Cholesky. In section IV an approximate method based in the Cholesky way is discussed, and conclusions are drawn in Section V.

2 LMMSE RECEIVER

A standard model for a DS-CDMA system with K users (assuming 1 user per physical channel) and L propagation paths is considered. The modulated symbols are spread by a Walsh-Hadamard code with length equal to the Spreading Factor (*SF)*. The signal on a MIMO-BLAST system with N_{TX} transmit and N_{RX} receive antennas, at one of the receiver's antennas, can be expressed as:

A standard model for a DS-CDMA system with K users (assuming 1 user per physical channel) and L propagation paths is considered. The symbols (QPSK/16QAM) are spread by a Walsh-Hadamard code with length equal to the Spreading Factor (*SF)*. The received signal on a MIMO system with N_{TX} transmit and N_{RX} receive antennas on one of the receiver's antennas can be expressed as:

$$r_v(t)_{RX=1} = \sum_{n=1}^{N}\sum_{tx=1}^{N_{TX}}\sum_{k=1}^{K} A_{k,tx} b_{k,tx}^{(n)} s_k(t-nT) * c_{k,tx,rx}(t) + n(t) \quad (2)$$

where N is the number of received symbols, $A_{k,tx} = \sqrt{E_k}$, E_k is the energy per symbol, $b_{k,tx}^{(n)}$ is the n-th transmitted data symbol of user k and transmit antenna tx, $s_k(t)$ is the k-th user's signature signal (equal for all antennas), T denotes the symbol interval, *n(t)* is a complex zero-mean AWGN with variance σ^2, $c_{k,tx,rx}(t) = \sum_{l=1}^{L} c_{k,tx,rx,l}^{(n)} \delta(t-\tau_{k,l})$ is the impulse response of the k-th user's radio channel, $c_{k,tx,rx,l}$ is the complex attenuation factor of the k-ths user's l-th path of the link between the tx-th and rx-th antenna, $\tau_{k,l}$ is the propagation delay (assumed equal for all antennas) and (*) denotes convolution. The received signal on can also be expressed as:

$$r_v(t)_{RX=1} = \sum_{n=1}^{N}\sum_{tx=1}^{N_{TX}}\sum_{k=1}^{K}\sum_{l=1}^{L} A_{k,tx} b_{k,tx}^{(n)} c_{k,tx,rx}(t) s_k(t-nT-\tau_{k,l}) + n(t) \quad (3)$$

Using matrix algebra, $r_v = SCAb + n$, where ***S, C*** and ***A*** are the spreading, channel and amplitude matrices respectively. The spreading matrix S has dimensions $(SF \cdot N \cdot N_{RX} + \tau_{MAX} \cdot N_{RX}) \times (K \cdot L \cdot N \cdot N_{RX})$ (τ_{max} is the maximum delay of the channel's impulse response, normalized to number of chips), and is composed of sub-matrices S_{RX} in its diagonal for each receive antenna $S=\text{diag}(S_{RX=1},\ldots,S_{RX=N_{RX}})$. Each of these sub-matrices has dimensions $(SF \cdot N + \tau_{MAX}) \times (K \cdot L \cdot N)$, and are further composed by smaller matrices S^L_n, one for each bit position, with size $(SF + \tau_{MAX}) \times (K \cdot L)$. The S_{RX} matrix structure is made of

$$S_{RX,n} = \left[0_{(SF\cdot(n-1))\times(K\cdot L)}, S_n^L, 0_{(SF\cdot(N-n))\times(K\cdot L)}\right]^T$$

$$S_{RX} = \left[S_{RX,1},\ldots,S_{RX,N}\right]$$

The S^L matrices are made of $K \cdot L$ columns; $S_n^L = \left[S_{col(k=1,l=1),n},\ldots,S_{col(k=1,l=L),n},\ldots,S_{col(k=K,l=L),n}\right]$. Each of these columns is composed of $S_{col(KL),n} = \left[0_{(1\times delay(L))}, spread_n(K)_{1\times SF}, 0_{(1\times(\tau_{MAX}-delay(L)))}\right]^T$, where $spread_n(K)$ is the combined spreading & scrambling for the bit n of user K.

These S^L matrices are either all alike if no long scrambling code is used, or different if the scrambling sequence is longer than the SF. The S_L matrices represent the combined spreading and scrambling sequences, conjugated with the channel delays. The shifted spreading vectors for the multipath components are all equal to the original sequence of the specific user.

$$S_n^L = \begin{bmatrix} S_{1,1,1,n} & \cdots & \cdots & S_{K,1,1,n} & & \\ \vdots & \ddots & S_{1,1,L,n} & \cdots & \vdots & \ddots & S_{K,1,L,n} \\ S_{1,SF,1,n} & & \vdots & \cdots & S_{K,SF,1,n} & & \vdots \\ & \ddots & S_{1,SF,L,n} & \cdots & \cdots & \ddots & S_{K,SF,L,n} \end{bmatrix}$$

Note that, in order to correctly model the multipath interference between symbols, there is an overlap between the S^L matrices, of τ_{MAX}.

The channel matrix ***C*** is a $(K \cdot L \cdot N \cdot N_{RX}) \times (K \cdot N_{TX} \cdot N)$ matrix, and is composed of RX sub-matrices, each one for a RX antenna $C=\left[C^R_{RX=1},\ldots,C^R_{RX=N_{RX}}\right]^T$. The diagonals of each C^R matrix are composed of N C^{KT} matrices.

$$C = \begin{bmatrix} \begin{bmatrix} C^{KT}_{1,1} & & \\ & \ddots & \\ & & C^{KT}_{N,1} \end{bmatrix} \\ \vdots \\ \begin{bmatrix} C^{KT}_{1,N_{RX}} & & \\ & \ddots & \\ & & C^{KT}_{N,N_{RX}} \end{bmatrix} \end{bmatrix}$$

Each C^{KT} matrix is $(K \cdot L) \times (K \cdot N_{TX})$, and represents the fading coefficients for each path, user and TX antenna, for the current symbol and RX

antenna. The matrix structure is made up of further smaller matrices along the diagonal $C^{KT}=\text{diag}\left(C^{T}_{K=1},\ldots,C^{T}_{K=K}\right)$, with C^{T} of dimensions $L \times N_{TX}$, representing the fading coefficients for the user's multipath and tx-th antenna component.

$$C^{KT} = \begin{bmatrix} C_{1,1,1} & \cdots & C_{N_{TX},1,1} & & & \\ \vdots & & \vdots & & & \\ C_{1,L,1} & \cdots & C_{N_{TX},L,1} & & & \\ & & & \ddots & & \\ & & & & C_{1,1,K} & \cdots & C_{N_{TX},1,K} \\ & & & & \vdots & & \vdots \\ & & & & C_{1,L,K} & \cdots & C_{N_{TX},L,K} \end{bmatrix}$$

The ***A*** matrix is a diagonal matrix of dimension $\left(K \cdot N_{TX} \cdot N\right)$, and represents the amplitude of each user per transmission antenna and symbol,

$$A=\text{diag}\left(A_{1,1,1},\ldots,A_{N_{TX},1,1},\ldots,A_{N_{TX},K,1},\ldots,A_{N_{TX},K,N}\right).$$

Vector ***b*** represents the information symbols. It has length $\left(K \cdot N_{TX} \cdot N\right)$, and has the following structure $b=\left[b_{1,1,1},\ldots,b_{N_{TX},1,1},\ldots,b_{1,K,1},\ldots,b_{N_{TX},K,1},\ldots,b_{N_{TX},K,N}\right]^{T}$.

Note that the bits of each *TX* antenna are grouped together in the first level, and the bits of other interferers in the second level. This is to guarantee that the resulting matrix to be inverted has all its non-zeros values as close to the diagonal as possible. Also note that there is usually a higher correlation between bits from different antennas using the same spreading code, than between bits with different spreading codes.

Finally, the ***n*** vector is a $\left(N \cdot SF \cdot N_{RX} + N_{RX} \cdot \tau_{MAX}\right)$ vector with noise components to be added to the received vector r_v, which is partitioned by N_{RX} antennas,

$$r_v=\left[R_{1,1,1},\ldots,R_{1,SF,1},\ldots,R_{N,1,1},\ldots,R_{N,SF+\tau_{MAX},1},\ldots,R_{N,1,N_{RX}},\ldots,R_{N,SF+\tau_{MAX},N_{RX}}\right]^{T}$$

The MMSE algorithm yields the symbol estimates, y_{MMSE}, which should be compared to vector ***b***,

$$y_{MF} = \left(SCA\right)^{H} rv \quad y_{MMSE} = \left(EM\right)^{-1} y_{MF}$$

$$R = S^{H} \cdot S \quad , EM = AC^{H} RCA + \sigma^{2} I \qquad (4)$$

where σ^2 is the noise variance of *n*, y_{MF} is the matched filter output and *EM* is the Equalization Matrix (cross-correlation matrix of the users' signature sequences after matched filtering, at the receiver).

The expected main problem associated with such scheme is the size of the matrices, which assume huge proportions. Due to the multipath causing Inter-Symbolic Interference (ISI), the whole information block has to be simulated at once, requiring the use of a significant amount of memory and some computing power for the algebraic operations, with emphasis on the inversion of the *EM* in the MMSE algorithm.

Matrix Reordering

Matrix reordering is used in order to simplify the solving of the MMSE equation. While in the original version the ***SCA*** matrix is devised in such a way as to make the received vector is divided per receive antenna in order to make the system matrices more perceivable, the reordering of the structure of the ***SCA*** matrix is done solely to simplify processing.

Replacing: $\mathbf{T} = reord\left(\mathbf{SCA}\right)$; $\mathbf{N} = \sigma^{2}\mathbf{I}$; $\hat{\mathbf{d}} = \mathbf{y}_{MMSE}$ and $\mathbf{e} = reord(\mathbf{r}_{V})$ in the MMSE equation we obtain a simpler version:

$$\hat{\mathbf{d}} = (\mathbf{T}^{H}\mathbf{T} + \mathbf{N})^{-1}\mathbf{T}^{H}\mathbf{e}$$

where $reord\left(\mathbf{x}\right)$ represents a line-reordering of vector or matrix ***x*** where the lines of each antenna were intercalated with the propose of making a more compact and almost block-circulant matrix.

$$reord\ (\mathbf{x}) = \begin{bmatrix} \mathbf{x}_{TX=1}(1,:) \\ \mathbf{x}_{TX=2}(1,:) \\ \vdots \\ \mathbf{x}_{TX=N_{TX}}(1,:) \\ \vdots \\ \mathbf{x}_{TX=1}(N,:) \\ \mathbf{x}_{TX=2}(N,:) \\ \vdots \\ \mathbf{x}_{TX=N_{TX}}(N,:) \end{bmatrix} \qquad (6)$$

Figure 1(a) shows the reordering result for a two antennas matrix. For high SNRs equation (5) becomes the Zero-Forcing detector equation:

$$\hat{\mathbf{d}} = (\mathbf{T}^{H}\mathbf{T})^{-1}\mathbf{T}^{H}\mathbf{e} \qquad (7)$$

Since usually $\tau_{MAX} \le SF$, the $\mathbf{T}^{H}\mathbf{T}$ product results in a square matrix with the structure presented in Figure 1(b) with: $a = 2KN_{TX}$ and $n = KN_{TX}N$. It can be shown that $\mathbf{T}^{H}\mathbf{T}$ is a positive definite Hermitian matrix. Earlier works (Vollmer, 2001), (Machauer, 2001), dealt only with the Zero-Forcing detector equation for constant channel situations. Here the validity of those algorithms for unsteady channels situations will be evaluated. New algorithms for unsteady channel situations will be proposed and some optimizations will be also introduced and presented in pseudo-code form. Finally, all the algorithms will be adapted to the LMMSE detector.

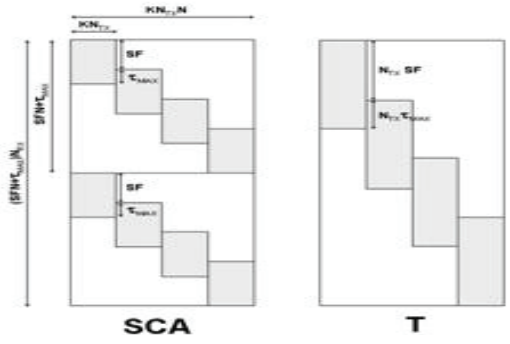

Figure 1: Line reordering sample for $N_{TX} = 2$.

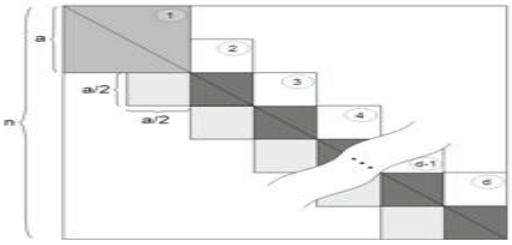

Figure 2: Typical correlation matrix.

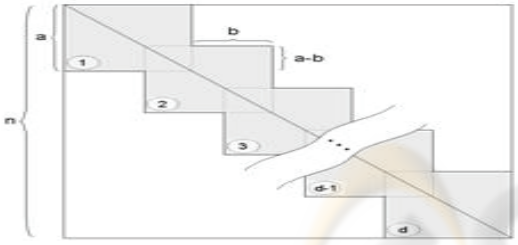

Figure 3: Generalized equalization matrix.

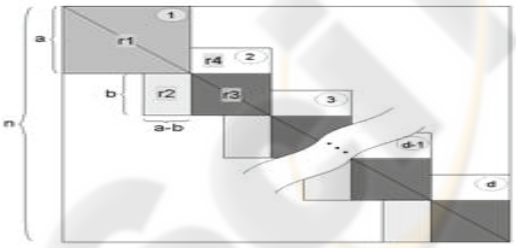

Figure 4: Optimized Gauss algorithm.

3 GAUSS AND CHOLESKI ALGORITHMS

Equation (7) can be written as an ***Ax=b*** system with ***A*** being a positive definite Hermitian matrix, where $A = \mathbf{T}^H\mathbf{T}$, $x = \hat{\mathbf{d}}$ and $b = \mathbf{T}^H\mathbf{e}$. It can be solved for ***x*** using the Gauss elimination or the Cholesky algorithm.

We are interested in solving the ***Ax=b*** system for a particular ***b*** vector only, so there is no need to invert the ***A*** matrix. The Gauss elimination can be used to transform the ***Ax=b*** system in a ***Ux=b'*** where ***U*** is an upper triangular matrix and then **x** can be obtained by direct substitution. The Cholesky method is a little bit more complex: first ***A*** is factorized in $\boldsymbol{A}=\boldsymbol{U}^{\boldsymbol{H}}\boldsymbol{U}$ by the Cholesky algorithm, then $\boldsymbol{U}^{\boldsymbol{H}}\boldsymbol{Ux}=\boldsymbol{b}$ can be decomposed in $\boldsymbol{U}^{\boldsymbol{H}}\boldsymbol{c}=\boldsymbol{b}$ and $\boldsymbol{Ux}=\boldsymbol{c}$; these two systems can be solved by direct substitution. The Cholesky algorithm can save almost half of the floating point operations needed in the Gauss elimination because it takes advantage of the symmetry of the ***A*** matrix, but the Gauss elimination is less complex and requires no square roots to be calculated. Previous works (Noguet, 2004) have successfully designed and implemented a real-time hardware structure of the regular Cholesky algorithm for the ZF joint-detection algorithm in the UMTS/TDD context based on a SIMD structure (Flynn, 1972) (systolic array). With the algorithms presented in this paper a significant complexity reduction can be expected, thereby reducing by a great amount the size of the hardware structure, alongside the cost and processing time. Table 1(a) shows the number of floating point operations required by both methods. The additions are separated into real and complex (R+ and C+, respectively). The extra operations wasted by the Gauss algorithm are partially recovered in the substitution phase, where the Cholesky method requires the solution of two triangular systems and hence twice the operations of the Gauss algorithm. The number of operations required in this phase is also included in Table 1(a).

The *Order* column presents the highest power of the total number of operations considering each multiplication-addition pair as a single operation.

Table 1: Number of floating operations needed to solve the ***A***x=***b*** system with standard Gauss algorithm.

	÷	×	C+	R+	$\sqrt{\ }$	Subs÷	Subs×/C+	Order
Gauss	$\frac{n^2-n}{2}$	$\frac{n^3-n}{3}$	$\frac{n^3-n}{3}$	0	0	n	$\frac{n^2-n}{2}$	$\frac{n^3}{3}$
Cholesky	$\frac{n^2-n}{2}$	$\frac{n^3-n}{6}$	$\frac{n^3-3n^2+2n}{6}$	$\frac{n^2+n}{2}$	n	$2n$	n^2-n	$\frac{n^3}{6}$

Table 2: Number of floating operations needed to solve the ***A***x=***b*** system with optimized Gauss algorithm.

Phase	÷	× / C+
r1	$\frac{a^2-a}{2}$	$\frac{a^3-a}{3}$
r2	$b(a-b)$	$\frac{3a^3-6a^2}{16}$
r3	$\frac{b^2-b}{2}$	$\frac{b^3-b}{3}$

Table 3: Number of floating operations needed to solve the ***A***x=***b*** system with optimized Cholesky algorithm.

Phase	÷	×	C+	R+	√
r1	$\frac{a^2-a}{2}$	$\frac{a^3-a}{6}$	$\frac{a^3-3a^2+2a}{6}$	$\frac{a^2+a}{2}$	a
r2	$b(a-b)$	$\frac{b}{2}\left[(a-b)^2-a+b\right]$	$\frac{b}{2}\left[(a-b)^2-a+b\right]$	0	0
r3	$\frac{b^2-b}{2}$	$\frac{a}{2}(b^2-b)-\frac{b}{6}(2b^3-3b+1)$	$\frac{a}{2}(b^2-b)-\frac{b}{3}(b^2-1)$	$ab-\frac{b^2}{2}+\frac{b}{2}$	b

Table 4: Number of floating operations needed to solve the ***A***x=***b*** system with optimized Cholesky algorithm; special case when $b = a/2$.

	÷	×	C+	R+	√	Subs ÷	Subs ×	Subs C+
Gauss	$\frac{n^2-n}{2}$	$\frac{n^3-n}{3}$	$\frac{n^3-n}{3}$	0	0	n	$\frac{n^2-n}{2}$	$\frac{n^2-n}{2}$
Cholesky	$\frac{n^2-n}{2}$	$\frac{n^3-n}{6}$	$\frac{n^3-3n^2+2n}{6}$	$\frac{n^2+n}{2}$	n	$2n$	n^2-n	n^2-n
Gauss a_b	$n\left(\frac{3a}{4}-\frac{1}{2}\right)$	$\frac{n(11a^2-18a-8)}{24}$	$\frac{n(11a^2-18a-8)}{24}$	0	0	n	$n\left(\frac{3}{4}a-1\right)$	$n\left(\frac{3}{4}a-1\right)$
Cholesky a_b	$n\left(\frac{3}{4}a-\frac{1}{2}\right)$	$n\left(\frac{7}{24}a^2-\frac{a}{2}-\frac{1}{6}\right)$	$\frac{n}{24}(7a^2-18a+8)$	$n\left(\frac{3}{4}a+\frac{1}{2}\right)$	n	$2n$	$n\left(\frac{3}{2}a-2\right)$	$n\left(\frac{3}{2}a-2\right)$

Optimizations

A generic positive definite Hermitian matrix that is nonzero only in equally overlapped squares centred along the diagonal is represented in Figure 1(c). The number of overlapped squares is $d = \frac{n-a}{b}+1$. The Gauss algorithm can be optimized for this type of matrix by eliminating the operations involving zero elements. The idea is presented in Figure 1(d). First, the standard Gauss algorithm is applied to the *r1* square sub-matrix. There is no need to change the *r4* rectangle. Next step is the elimination of the *r2* block using the last *a-b* pivots of *r1* (the pivots are the diagonal elements after the elimination phase). During this phase *r3* is updated. Finally the standard algorithm is applied to *r3*. This process is repeated until all blocks are updated. During this process, as each line is updated, the correspondent element of vector ***b*** is simultaneously updated. Note that the matrix diagonal is fully contained in the diagonal squares.

Table 1(b) presents the number of floating point operations required in each of the tree phases described. The total number of operations can be calculated from:

$$N_{Gauss\,Opt} = N_{Gauss\,r1} + (d-1)\left(N_{Gauss\,r2} + N_{Gauss\,r3}\right) \quad (8)$$

Leading to:

$$N^{\div}_{Gauss\,Opt} = n\left(a - \frac{b}{2} - \frac{1}{2}\right) - \frac{a(a-b)}{2} \quad (9)$$

$$N^{\times}_{Gauss\,Opt} = N^{C+}_{Gauss\,Opt} = \\ = n\frac{9a^3 - 18a^2 + 16b\left(b^2-1\right)}{48b} - a\frac{9a^3 - 2a^2\left(8b+9\right) + 16b^3}{48b} \quad (10)$$

A similar adaptation can be developed for the Cholesky factorization algorithm. We will optimize the column-Cholesky algorithm (presented in Figure 5) although similar results could be achieved for the line version of that algorithm. Figure 6(a) sketches such approach. In this case only the upper triangle has to be accessed. First, the standard column-Cholesky algorithm is applied to the *r1* triangle. In a second step the rectangle *r2* is calculated accessing only the elements of *r2* and *r1*. In the next step, the triangle *r3* is computed using only elements of *r2* and *r3*. Last two steps are repeated for all remaining blocks, using always only elements of the last and current block. As in the optimized Gauss algorithm, the rectangular blocks do not contain the diagonal. Table 1(c) presents the number of floating point operations of the tree phases described. The total number of operations can be calculated from:

$$N_{Chol\,Opt} = N_{Chol\,r1} + (d-1)\left(N_{Chol\,r2} + N_{Chol\,r3}\right) \quad (11)$$

Leading to:

$$N^{\div}_{Chol\,Opt} = n\left(a - \frac{b}{2} - \frac{1}{2}\right) - \frac{a(a-b)}{2} \quad (12)$$

$$N^{\times}_{Chol\,Opt} = n\left(\frac{a^2}{2} - \frac{a}{2}(b+2) + \frac{b^2}{6} + b - \frac{1}{6}\right) - a\left(\frac{a^2}{3} - \frac{a}{2}(b+2) + b\left(\frac{b}{6}+1\right)\right) \quad (13)$$

$$N^{C+}_{Chol\,Opt} = n\left(\frac{a^2}{2} - \frac{a}{2}(b+2) + \frac{b^2}{6} + \frac{b}{2} + \frac{1}{3}\right) - a\left(\frac{a^2}{3} - \frac{a}{2}(b+1) + b\left(\frac{b}{6}+\frac{1}{2}\right)\right) \quad (14)$$

$$N^{R+}_{Chol\,Opt} = n\left(a - \frac{b}{2} + \frac{1}{2}\right) + \frac{a}{2}(b-a) \quad (15)$$

$$N^{\surd}_{Chol\,Opt} = n \quad (16)$$

```
for j = 1, n
    for k = 1, j - 1
        for i = j, n      {cmod(j,k)}
            a_ij = a_ij - a_ik * a_jk
    a_jj = sqrt(a_jj)
    for k = j + 1, n      {cdiv(j)}
        a_kj = a_kj / a_jj
```

Figure 5: Column oriented Cholesky factorization.

Both Gauss and Cholesky methods need final substitution phases. These substitutions can also be optimized since the resulting matrices have a structure similar to the original ***A*** matrix but with

nonzero elements only above (or below) the diagonal, like shown in Figure 6(b).

The solution of a system ***Ax=b*** with ***A*** having a structure similar to the structure presented in Figure 6(b) requires one division for each line of the matrix and one pair multiplication-addition for each nonzero element. Since there are

$$d\left(\frac{a^2}{2}-a\right)-(d-1)\left(\frac{(a-b)^2}{2}-(a-b)\right) \quad (17)$$

nonzero elements, the number of floating point operations needed can be written:

$$N^{\div}_{Subs\,Opt}=n \quad (18)$$

$$N^{\times}_{Subs\,Opt}=N^{C+}_{Subs\,Opt}=n\left(a-\frac{b}{2}-1\right)-\frac{a(a-b)}{2} \quad (19)$$

We are interested in a special type of block-diagonal positive definite Hermitian matrices, with $b=a/2$, like represented in Figure 6(b). Re-writing the above equations for this special case and keeping only the n-dependent terms ($n>>a$), results in Table 1(d) As can be seen, the optimized Cholesky algorithm can save almost 30% of the number of operations required for the optimized Gauss algorithm, despite it increased complexity and need of square root operations.

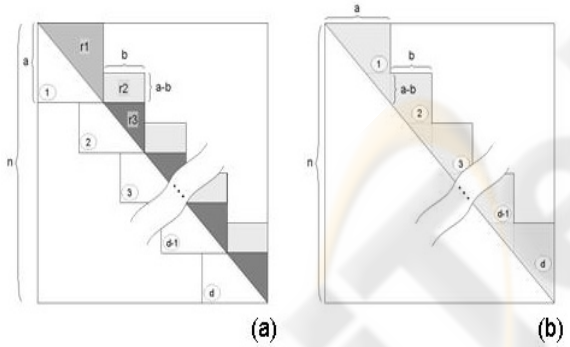

Figure 6: (a) Optimized Cholesky algorithm, (b) Gauss elimination / Cholesky factorization resulting matrix structure.

Partitioning

Partitioning the block-diagonal system Ax=b would be very useful to reduce the number of floating point operations needed to solve the system (if no overlap is used; i.e., in good channel conditions) and also could permit the introduction of parallelism is algorithms that are intrinsically sequential, like the algorithms presented in previous sections. In this section we will discuss different partitioning approaches.

Since A is block-diagonal and has generally decreasing values as we get farther from its diagonal, it is expected that it can be divided in smaller matrices that produce smaller systems whose combined solutions would approximate the solution of the original system. Figure 7(a) presents a sample solution of a system simply divided in 2 sub-systems as sketched in Figure 8(a). Note that there are two m × m blocks completely ignored at the middle of the A matrix. Surprisingly the obtained maximum error is only 12% of the exact solution. Figure 7(b) shows the same data when the **A** matrix is divided in four slices. The maximum error level is approximately the same as in the previous case, but now we have three high error regions.

Although the error obtained with the last partitioning method is not extremely high and appear only around the division lines, much better results can be attained if overlapping partitions are considered.

This proceeding is sketched in Figure 8(b). Each slice overlaps the last in 2 × lap blocks (a lap being the number of blocks that are discarded from each overlapping side of each computed slice). Note that the last slice can be smaller.

From each slice are obtained $(D-2lap)m$ values for the solution vector x, except in the case of the first and last slices, where $(D-lap)m$ and $M-((L-1)D+lap)m$ values are obtained respectively.

As seen in Figure 7(a) and (c), the error level rises at the beginning and end of each slice, so the overlapping method should discard that values. In each iteration are discarded lap × m values from the beginning and lap × m values from the end, except in the first end last slice, where are only discarded the last lap × *m* and first lap × *m* values respectively.

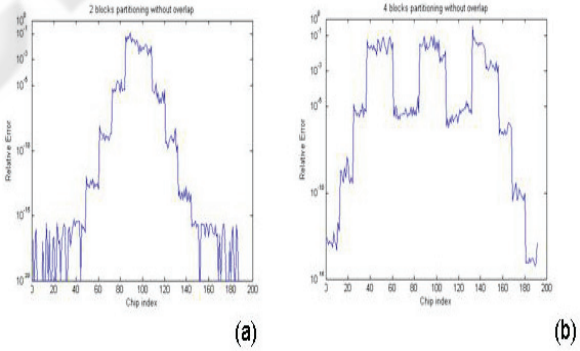

Figure 7: Relative error obtained solving a no-overlap partitioned system. (a) - *M*=192; *m*=12; *D*=8; *v*=10Km/h, (b) - *M*=192; *m*=12; *D*=4; *v*=10Km/h.

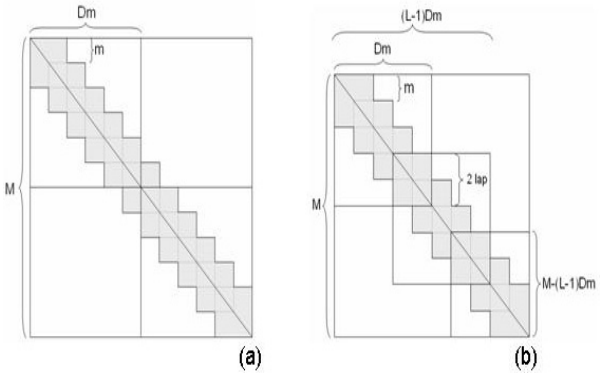

Figure 8: Partitioning (a) - without overlapping, (b) – with overlapping.

An alternative to have a final slice with size smaller than the early ones is to extend the b vector with zeros until $M/m-2lap$ becomes multiple of $D-2lap$. Several simulations were run for different channel changing speeds. Similar results were obtained for all matrices tested. Sample results are presented in Table 2, for a 10Km/h condition.

Table 5: Maximum relative error for the partitioning algorithm; *v*=10Km/h.

		lap							
		0	1	2	3	4	5	6	7
Factor:		1.E+00	1.E-04	1.E-07	1.E-11	1.E-15	1.E-15	1.E-15	1.E-15
D	1	0.506							
	2	0.459							
	3	0.152	0.911						
	4	0.221	0.494						
	5	0.127	0.494	0.090					
	6	0.082	0.494	0.123					
	7	0.459	0.105	0.123	0.247				
	8	0.221	0.494	0.021	0.063				
	9	0.083	0.640	0.042	0.038	0.898			
	10	0.099	0.194	0.123	0.063	0.637			
	11	0.079	0.190	0.037	0.222	0.817	0.505		
	12	0.033	0.082	0.020	0.038	0.646	0.623		
	13	0.159	0.059	0.015	0.036	0.642	0.552	0.416	
	14	0.459	0.494	0.018	0.047	0.572	0.432	0.504	
	15	0.127	0.845	0.090	0.033	0.420	0.284	0.252	0.378
	16		0.009	0.005	0.006	0.479	0.232	0.432	0.504

All the values presented should be multiplied by the corresponding column factor to obtain the maximum error of the partition algorithm relative to the exact solution of the original Ax=b system. The column for lap=0 corresponds to the lap less situation.

shows that the maximum error level depends almost exclusively from the overlapping level, so the proper overlapping can be easily selected just by knowing the maximum error allowed in the real system. The number of blocks D processed by each thread can be selected from the total number of threads L that can be executed simultaneously by the hardware, using the relation:

$$L = \left\lceil \frac{M/m - 2lap}{D - 2lap} \right\rceil, \quad (20)$$

where $\lceil x \rceil$ represents the smallest integer greater than or equal to x. Some results of the relative error obtained in an overlap system are portrayed in Figure 9.

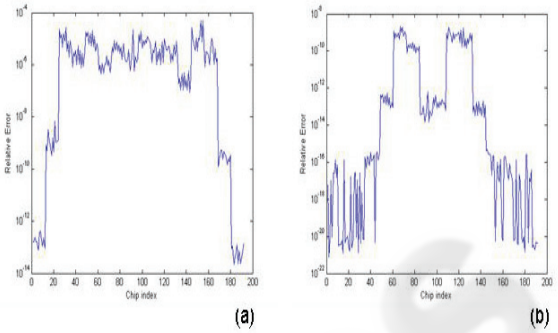

Figure 9: Relative error obtained solving an overlap partitioned system (a) - *D*=4; *lap*=1; *v*=10Km/h, (b) - *D*=8; *lap*=2; *v*=10Km/h.

4 PARTIAL CHOLESKI APPROXIMATION

The Cholesky decomposition of block-Toeplitz matrices is an upper (or lower) matrix approximately block-Toeplitz with the same block size as the original matrix. This means that the U matrix can be approximated calculating only the first L block-rows and assuming that the remaining block-rows are identical to the last calculated block-row. Figure 7 shows this approach. Only the dark shaded part is computed. The last computed block (marked as L) is then repeated until the full matrix is completed. This approximation is very effective when the channel is constant.

Figure 8 shows the maximum relative error of the system solution for each approximation level (L) i.e. the number of calculated blocks, for different speeds in a pedestrian A condition with 1 antenna.

As can be seen for a constant channel, calculating only the first one or two blocks allows approximations in the system solution with relative error below 10^{-4} or 10^{-9}. This can be used to greatly reduce the number of operations necessary to solve the system. If only 2 blocks are calculated the number of operations can be reduced approximately by a factor of a/n. However, when the channel changes, this approach can not be used regarding to the high error level obtained (unless the channel change is very slow).

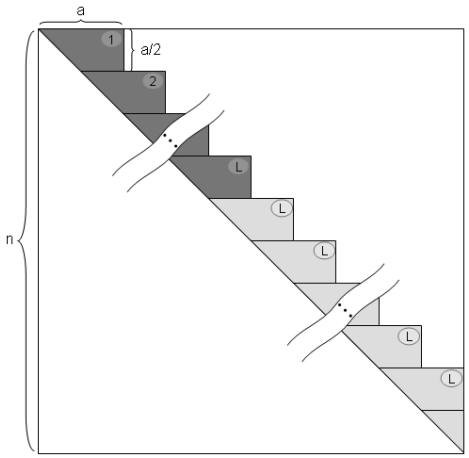

Figure 10: Partial Cholesky approximation.

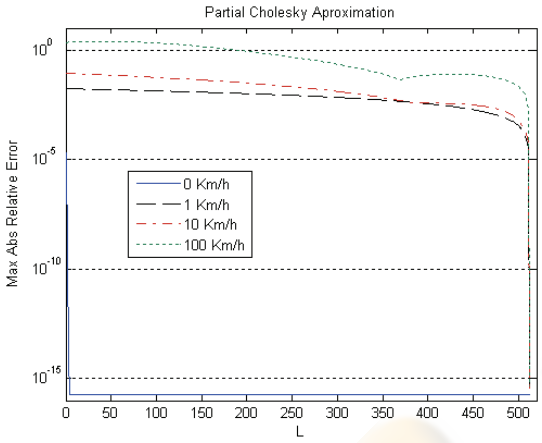

Figure 11: Partial Cholesky approximation; (pedestrianA – minimum load).

5 CONCLUSIONS

In this work where presented optimized versions of the Gauss and Cholesky algorithms that can be used in the solution of the equation of a zero-forcing or LMMSE detector in MIMO/BLAST systems. Those optimizations were based simply in the removal of the unnecessary operations regarding the structure of the involved matrices. It has the advantage of being a velocity independent solution, unlike other methods such as the Block-Fourier (Vollmer, 2001), (Machauer, 2001). The benefit of parallel processing can also be exploited for these methods, with the introduction of partitioning.

ACKNOWLEDGEMENTS

This work has been partially funded by the C-MOBILE (Advanced MBMS for the Future Mobile World) project, and by a grant of the Portuguese Science and Technology Foundation (FCT).

REFERENCES

G. J. Foschini and M. J. Gans, “On limits of wireless communications in a fading environment when using multiple antennas,” Wireless Pers. Commun., vol. 6, pp. 311–335, Mar. 1998.

I. E. Telatar, “Capacity of multiantenna Gaussian channels,” Eur. Trans. Commun., vol. 10, no. 6, pp. 585–595, 1999.

M. Vollmer, M. Haardt, J. Gotze, “Comparative Study of Joint-Detection Techniques for TD-CDMA Based Mobile Radio Systems”, IEEE Sel. Areas Comm., vol.19, no.8, August 2001.

R. Machauer, M. Iurascu, F. Jondral, “FFT Speed Multiuser Detection for High Rate Data Mode in UTRA-FDD”, IEEE VTS 54th, vol.1, 2001.

D. Noguet, “A reconfigurable systolic architecture for UMTS/TDD joint detection real time computation”, ISSSTA 2004, Sydney, Australia, 30 August-2 September 2004.

M. Flynn, “Some computer organizations and their effectiveness”, IEEE Transactions on Computers, vol c-21, 1972.