Hardware Efficient Approximative Matrix Inversion for Linear Pre-coding in Massive MIMO

Hemanth Prabhu, Ove Edfors, Joachim Rodrigues, Liang Liu, and Fredrik Rusek
Department of Electrical and Information Technology, Lund University, Sweden
{Hemanth.Prabhu, Ove.Edfors, Joachim.Rodrigues, Liang.Liu, Fredrik.Rusek}@eit.lth.se

***Abstract*—This paper describes a hardware efficient linear precoder for Massive MIMO Base Stations (BSs) comprising a very large number of antennas, say, in the order of 100s, serving multiple users simultaneously. To avoid hardware demanding direct matrix inversions required for the Zero-Forcing (ZF) precoder, we use low complexity Neumann series based approximations. Furthermore, we propose a method to speed-up the convergence of the Neumann series by using tri-diagonal precondition matrices, which lowers the complexity even further. As a proof of concept a flexible VLSI architecture is presented with an implementation supporting matrix inversion of sizes up-to 16×16. In 65 nm CMOS, a throughput of 0.5M matrix inversions per sec is achieved at clock frequency of 420 MHz with a 104K gate count.**

I. Introduction

Conventional Multiple-Input Multiple-Output (MIMO) systems incorporated in most modern standards such as 3GPP LTE, LTE-Advanced, IEEE 802.11n, are already starting to approach theoretical throughput limits. To go beyond, Massive (or Very Large) MIMO is a promising candidate, where the Base Station (BS) is equipped with a *very large* number of antennas (M) compared to previously considered systems, while serving a relatively low number of users or Mobile Stations (MSs). Basically, by equipping the BS with a large array of antennas ($M \rightarrow \infty$), it has been shown that in favourable propagation conditions, all effects of uncorrelated noise and fast fading disappear, as does the Multi-User Interference (MUI) [1].

To reach the full potential of massive MIMO [2], it is critical to perform efficient pre-coding. In [3], measurements for massive MIMO systems were performed, showing Zero-Forcing (ZF) linear pre-coding sum rates of up to 98% of those achieved by (optimal) Dirty Paper Coding (DPC), for BS to MS antenna ratios as low as 10. However, even linear pre-coding such as ZF has high computational and hardware complexity, particularly when inverting a $K \times K$ matrix, where K is the number of MSs. In addition to having to deal with large matrices, the pre-coding matrix may have to be updated frequently, depending on the Doppler spread of the channels.

In this paper we tackle this by approximating the matrix inversion using the Neumann Series, wherein the inversion is computed by performing a sum of powers (multiplication) of matrices. In hardware, matrix multiplication is much preferred over matrix inversion, since it has a simpler data-flow and can be highly parallelized. In [4], it was shown that using the Neumann series leads to a more energy efficient low complexity implementation than traditional direct inversion methods. This is mainly due to the fact that the $K \times K$ matrix becomes diagonally dominant as the ratio $\beta = M/K$, *i.e.* the ratio of number of antennas at BS to number of MSs increases, which can be used as starting point (inverse of diagonal elements) in the Neumann series. This approach of inversion was implemented in FPGA for uplink data detection [5], where a reasonable detection performance was achieved by a first order Neumann series. However, for scenarios with low β values and for high correlation between antennas (for instance when MSs are closely located) the matrix is not strongly diagonally dominant. Thus, choosing the diagonally dominant elements as initial approximation of the matrix inverse will lead to slow convergence of the Neumann series, in-turn requiring more iterations to achieve a certain accuracy. In addition to high throughput, the accuracy of the inversion is an important parameter which defines the suppressing of MUI in downlink.

To simultaneously handle high throughputs and accuracy (more iterations) with reasonable hardware cost, we develop a method to improve the convergence of the Neumann series. The underlying idea is to choose some off-diagonal elements along with the diagonal elements as initial approximation. This, in turn, increases the complexity of setting up the initial approximation, since it is required to invert the initial matrix before using in the Neumann Series. However, we see that by choosing tri-diagonal matrix as initial approximation the overall complexity is reduced significantly. Furthermore, we describe an efficient VLSI architecture for handling the tri-diagonal initial matrix with negligible additional latency. Finally, we present a reference implementation in 65 nm capable of handling matrix sizes up-to 16×16 with high throughput.

II. System Description

The system model and the pre-coding in this section is in line with the corresponding description in [2], where the channel gain between the i-th BS antenna and the k-th user is denoted by h_{ki}. The channel matrix to all users is denoted as $\boldsymbol{H} \in \mathbb{C}^{K \times M}$, where h_{ki} is the (k,i)-th entry. Let $\boldsymbol{x} = [x_1, x_2, ..., x_M]^T$ denote the transmitted vector from the M BS antennas, which is normalized to satisfy $\mathbb{E}[\boldsymbol{x}^H \boldsymbol{x}] = 1$, where $()^H$ is the Hermitian transpose. The overall symbol vector received by the K autonomous users is

$$\boldsymbol{y} = \sqrt{P_T}\boldsymbol{H}\boldsymbol{x} + \boldsymbol{w}\,, \qquad (1)$$

where P_T is the total transmit power, and $\boldsymbol{w} \sim \mathcal{CN}(0, \sigma^2 \boldsymbol{I}_K)$, where $\boldsymbol{I}_K$ is an $K \times K$ identity matrix, is complex Gaussian white noise. The pre-coding process at the transmit side is specified as

$$\boldsymbol{x} = \boldsymbol{F}\boldsymbol{s}\,, \qquad (2)$$

978-1-4799-3432-4/14/$31.00 ©2014 IEEE

where $\boldsymbol{F}$ is an $M \times K$ pre-coding matrix, and $\boldsymbol{s}$ is a $K \times 1$ vector containing the symbols intended for the K users, as described in [3]. Although the Massive MU-MIMO model is similar to a standard MIMO model, the increased number of BS antennas has several consequences. Things that were random before, now start to look deterministic. For example, with increasing BS antennas, the Gram matrix $\boldsymbol{H}\boldsymbol{H}^H/M$, asymptotically tends to a diagonal matrix for certain "nice enough" channel conditions.

A. *Linear Pre-coding Schemes*

The Matched Filter (MF) is a simple linear pre-coding scheme given as $\boldsymbol{F}_{\text{MF}} \propto \boldsymbol{H}^H$. MF, although simple, requires much more antennas at the BS compared to ZF to attain close to optimal performance [2]. On the other hand, ZF requires a central processing unit and higher processing cost when computing the pre-coding matrix

$$\boldsymbol{F}_{\text{ZF}} \propto \boldsymbol{H}^H(\boldsymbol{H}\boldsymbol{H}^H)^{-1}\,. \tag{3}$$

The multiplication with $\boldsymbol{H}^H$ can be performed in a distributed fashion at the remote antenna units (as in MF), whereas the Gram matrix inversion is performed in the central processing unit. In hardware, high throughput matrix inversion is expensive both in terms of area and power, especially when the number of active users (K) is varying and large. In the next section we will look into different methods to perform matrix inversion which are hardware friendly.

III. Low Complexity Matrix Inversion

To reduce the hardware cost of matrix inversion, we propose the use of Neumann series in line with [2] and [4]. For an invertible matrix $\boldsymbol{Z}$, if a matrix $\boldsymbol{X}$ satisfies

$$\lim_{n\to\infty} \left(\boldsymbol{I}_K - \boldsymbol{X}^{-1}\boldsymbol{Z}\right)^n \simeq 0_K\,, \tag{4}$$

then the inverse of $\boldsymbol{Z}$ can be expressed as

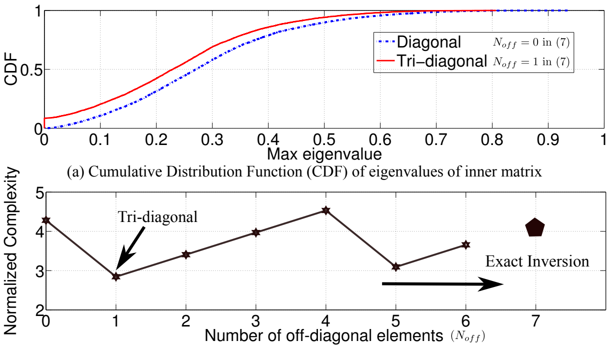
$$\boldsymbol{Z}^{-1} \approx \sum_{n=0}^{L} \left(\boldsymbol{I}_K - \boldsymbol{X}^{-1}\boldsymbol{Z}\right)^n \boldsymbol{X}^{-1}\,, \tag{5}$$

with equality when L grows to infinity. The matrix $\boldsymbol{X}$ is an initial approximation of $\boldsymbol{Z}$, which must be much easier to invert. Computing the Neumann series (5) requires $L-1$ matrix multiplications when the computation of matrix powers is performed iteratively. This can be accelerated exponentially, in line with [4], by re-writing (5) as

$$\boldsymbol{Z}^{-1} \approx \left(\prod_{n=0}^{P-1} (\boldsymbol{I} + (\boldsymbol{I} - \boldsymbol{X}^{-1}\boldsymbol{Z})^{2^n})\right)\boldsymbol{X}^{-1}, \tag{6}$$

where a certain P corresponds to $L = 2^P - 1$ in (5), hence reducing matrix multiplications logarithmically.

For a guaranteed convergence of (5) we must have $|\lambda_{\max}| < 1$, where $\lambda_{\max}$ is the eigenvalue of the inner matrix $(\boldsymbol{I}_K - \boldsymbol{X}^{-1}\boldsymbol{Z})$ with largest magnitude. Furthermore, the choice for the pre-conditioner matrix $\boldsymbol{X}$ is critical for the convergence speed of (5), *i.e.* the smaller $|\lambda_{\max}|$, the faster the convergence. In ZF pre-coders for massive MIMO, the matrix to be inverted is $\boldsymbol{Z} = \boldsymbol{H}\boldsymbol{H}^H$, from (3), which tends to be a diagonally dominant matrix. Hence a simple choice of $\boldsymbol{X}$ is a matrix containing only the diagonal elements of $\boldsymbol{Z}$, *i.e.* $\boldsymbol{X}_{\text{d}} = \text{diag}(\boldsymbol{Z})$. However, there are scenarios (for low β values or highly correlated channels) when the matrix $\boldsymbol{Z}$ is not strongly diagonally dominant or perhaps not at all. For these scenarios choosing only the diagonal elements would result in a slow convergence, which in-turn would mean more terms (P) required in the Neumann series.



(a) Cumulative Distribution Function (CDF) of eigenvalues of inner matrix

(b) Complexity to achive 16-bit accuracy with different pre-condition matrices. Normalized to the complexity of performing a matrix multiplication ($i.e$ K^3).

Fig. 1. Convergence and Complexity analysis of Neumann series.

A. *Convergence Speed-Up*

For improving the convergence speed of (5), we would like to have as small $\lambda_{\max}$ as possible. We propose a method based on choosing off-diagonal elements along with diagonal elements for the initial matrix $\boldsymbol{X}$. This can be expressed as

$$\boldsymbol{X} = \text{diag}_0(\boldsymbol{Z}) + \sum_{\ell=1}^{N_{\text{off}}} \text{diag}_\ell(\boldsymbol{Z})\,, \tag{7}$$

where N_{off} is the number of off-diagonal elements including the main diagonal elements and $\text{diag}_\ell(\boldsymbol{Z})$ has the same size as $\boldsymbol{Z}$ with

$$[\text{diag}_\ell(\boldsymbol{Z})]_{i,j} = \begin{cases} \boldsymbol{Z}_{i,j} & \text{if } |i-j| = \ell \\ 0 & \text{otherwise}. \end{cases}$$

The improvement in convergence can be seen in Fig. 1(a), where the probability of having a smaller $|\lambda_{\max}|$ when choosing off-diagonal elements is always higher than when choosing diagonal elements. Consequently, setting up the inner matrix $(\boldsymbol{I} - \boldsymbol{X}^{-1}\boldsymbol{Z})$ requires more computation as compared to choosing only diagonal elements. However, by doing this we expect the overall complexity to reduce due to faster convergence, which in-turn would mean fewer terms (P) required in the Neumann series. To elaborate on this, an approximate overall complexity in terms of multiplications is divided into two parts (computing powers of matrix, and inner matrix setup) as

$$C_{\text{Total}} = \underbrace{2(P-1)K^3}_{\text{Iterations}} + \underbrace{(2(N_{\text{off}}) + 1) * (2 * K^2 + 2K)}_{\text{Inner matrix setup}}\,.$$

The number of iterations reduces as N_{off} increases, since the initial matrix ($\boldsymbol{X}$) has more and more elements of $\boldsymbol{Z}$. Hence, based on required accuracy, the properties of the matrix to be inverted (diagonal dominance, size) the optimal number of off-diagonal elements can be decided. In Fig. 1(b) we can see that for a 16-bit precision the optimal $N_{\text{off}} = 1$, *i.e.* choosing tri-diagonal matrix ($\boldsymbol{X}_{\text{t}} = \text{diag}_0(\boldsymbol{Z}) + \text{diag}_1(\boldsymbol{Z})$). Additionally, efficient hardware is developed to handle tri-diagonal matrices,

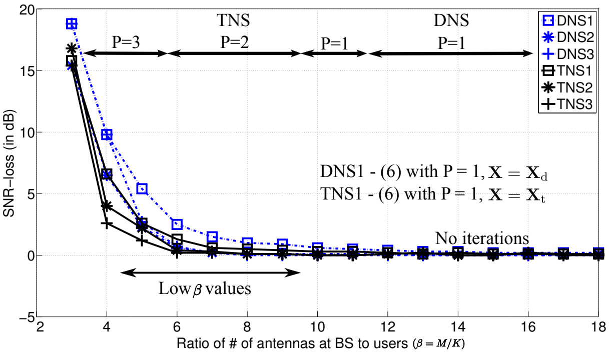


Fig. 2. SNR loss relative to exact-ZF to achieve BER of 10^{-3}, for $K = 10$, 4-QAM modulation, IID channel for different antenna configurations at BS.

which may not be the case for $N_\mathrm{off} > 1$, which tend to exact inversion.

B. Performance Analysis

We now demonstrate the performance of the proposed Tri-diagonal Neumann series (TNS) by evaluating the SNR-loss compared to exact-ZF to achieve a Bit Error Rate (BER) of 10^{-3}, as shown in Fig. 2. The performance of the Neumann series get closer to exact ZF as β increases. For β below 12 there is an improvement with the proposed TNS compared to Diagonal Neumann Series (DNS). Basically, the proposed TNS requires 1 (or 2) iterations less than DNS, to achieve the same performance. This results in lower overall complexity for TNS, since the initial tri-diagonal computation is of complexity $\mathcal{O}(6K + 3K^2)$ whereas each iteration requires $\mathcal{O}(K^3)$ computations. Additionally, we also expect TNS to have better performance for a wider range of channel conditions, since it can cope with matrices that need not to be strongly diagonally dominant.

IV. VLSI Architecture

This section describes the VLSI architecture that realizes the proposed matrix inversion algorithm with a high throughput and hardware efficiency. The architecture is highly flexible to support different matrix sizes (K), iterations (P), for both TNS and DNS method. Fig. 3 depicts the top level block diagram of the architecture, which consists of four major blocks following the data flow of (6). The initial matrix inversion ($\boldsymbol{X}^{-1}$) is performed in the Tri-diag inverse block. If the DNS method is used, only the division unit is required to invert the diagonal elements. The inverted initial matrix is then multiplied with $\boldsymbol{Z}$ in the Tri-diag multiplication block. The inner matrix module performs the subtraction with an identity matrix ($\boldsymbol{I} - \boldsymbol{X}^{-1}\boldsymbol{Z}$), which is a trivial operation. Finally, the powers of the matrix are computed in the generic matrix multiplication module.

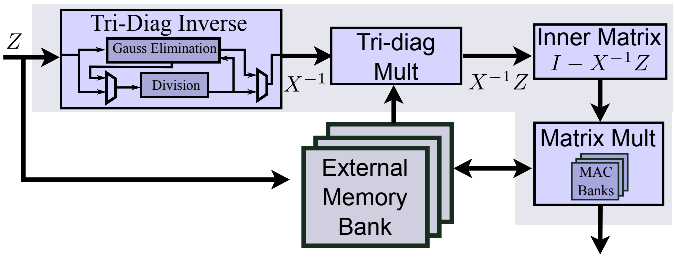


Fig. 3. Block diagram of Neumann series based linear pre-coder at BS.

Algorithm 1 Tri-band inversion, exploiting diagonal dominance and Hermitian symmetry.

```
for $i = 2 \to K$ do //Guass elimination
    $ratio = A(i, i-1)/A(i-1, i-1)$
    $A(i, i) = A(i, i) - ratio * A(i-1, i)$
    $A(i, i-1) = 0$, $A_\mathrm{inv}(i, i-1) = -ratio$
end for
for $i = 2 \to K$ do //compute inverse
    $A_\mathrm{inv}(i, i) = 1/A(i, i)$
    $A_\mathrm{inv}(i, i-1) = A(i, i-1)/A(i, i)$
    $A_\mathrm{inv}(i-1, i) = \mathrm{conj}(A(i, i-1))$ // conjugate
end for
$X_t^{-1} = A_\mathrm{inv}$
```

A. Detailed Functional Block

Hardware Efficient Tri-diag Inverse: The inverse of tri-diagonal matrix is a full matrix. However, the elements outside the three center diagonals of the inverted tri-diag matrix are typically small in magnitude, and therefore neglected. Furthermore, the Tri-diag matrix is Hermitian, hence we can use these properties to modify the Gauss elimination [6] to perform inversion, as shown in Alg. 1. This leads to a very low complexity and hardware friendly tri-diag inversion, using a division unit and a multiplier. The division unit is required for both the DNS and the TNS (see Alg. 1) methods. The diagonal elements have approximately a deterministic dynamic range depending on M and K. This is used to compute a first order curve fit ($\hat{x}_{est} = ax + b$, where x is divisor, a,b are constants) for initial estimate in a standard unrolled single-iteration Newton-Raphson method. The first order curve fitting requires one constant multiplier and one adder, hence avoiding large look-up-tables, making the division unit more hardware friendly.

Tri-diag Mult: The multiplication of the tri-diag inverse ($\boldsymbol{X}_\mathrm{t}^{-1}$) and $\boldsymbol{Z}$ has a complexity $\mathcal{O}(3K^2)$. An intuitive low latency way to perform the Tri-band matrix multiplication is shown in Fig. 4. The structure resembles an FIR filter, where the coefficients are the rows of $\boldsymbol{X}_\mathrm{t}^{-1}$ and the columns of $\boldsymbol{Z}$ are streaming input data. The circuit mainly requires $2K$ buffers to store diagonal and off-diagonal elements, three multipliers, and two adders. Furthermore, the circuit can easily be switched to diagonal matrix multiplication ($\boldsymbol{X}_\mathrm{d}^{-1}$) by setting *Mux2* and *Mux3* to zero.

B. Timing Analysis

Based on the two methods DNS and TNS, the latency of tri-diag-inverse module is $6K$ and K clock cycles, respectively. The Tri-diag-mult and Inner-matrix module has a latency of

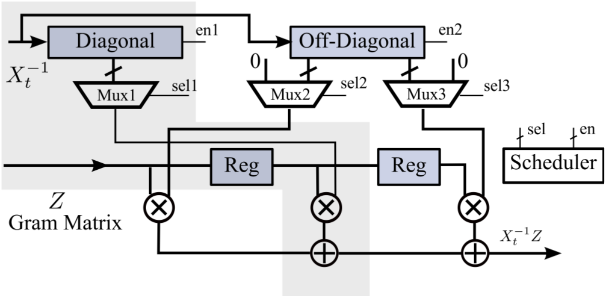
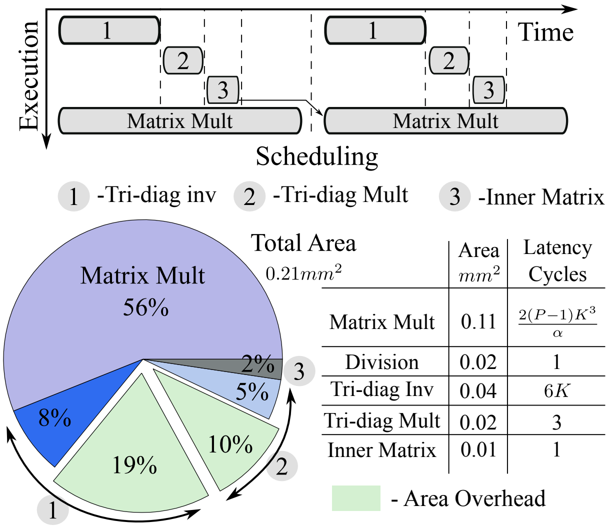


Fig. 4. Circuit description of Tri-diagonal matrix multiplication.



	Area mm^2	Latency Cycles
Matrix Mult	0.11	$\frac{2(P-1)K^3}{\alpha}$
Division	0.02	1
Tri-diag Inv	0.04	$6K$
Tri-diag Mult	0.02	3
Inner Matrix	0.01	1

Fig. 5. Neumann Series Implementation Result, pie-chart showing the area cost and table provides the latency used for scheduling.

three and one clock cycles, respectively. These modules are of streaming nature (non-iterative) and have a relatively low latency compared to the matrix-mult module. The Matrix-mult module is implemented using Multiply-Accumulate (MAC) banks, with the number of MAC units α as a parallelization factor. For a $K \times K$ matrix the latency is around $2(P-1)K^3/\alpha$ clock cycles. The α value can be decided based on maximum value of K and throughput requirements. During the execution of matrix-mult, the other modules perform pre-conditioning for the next matrix. This overlap of execution (Scheduling in Fig. 5) results in 100% hardware utilization of the Matrix-mult module. It is important to note that for best-case scenarios, for *e.g.*, large β values and low correlation, a first order ($P = 1$) Neumann series suffices, hence no matrix multiplication are required. For these scenarios the total latency is around $6K$ cycles dominated by the Tri-diag inverse module.

V. Implementation Results

A reference design of the matrix inversion based on Neumann series was synthesized in 65 nm technology for 16-bit internal precision. The maximum frequency is 420 MHz with an area cost of 0.21 mm^2 or 125K gates. The hardware cost of the proposed improvements in convergence (TNS method) has a small overhead of (10+19)% compared to the overall area as shown in Fig. 5. Moreover, the throughput improvement is remarkable, for *e.g.*, a system scenario with $\beta = 6$, $K = 16$ MSs, using TNS and DNS would require $P = 2$ and $P = 3$ iterations respectively to achieve same accuracy. This translates to a latency of 820 and 1650 cycles when $\alpha = 10$, operating at 420 MHz results in a throughput of 0.5M and 0.25M inversions per sec respectively.

To the best of our knowledge, there are no VLSI implementations for large matrix inversion of varying size. Table I shows results to give the reader a rough idea of the benefits of our proposed method. Furthermore, scaling up the existing implementations would not be fair comparison. This is mainly because the algorithms are not tuned to exploit the properties of matrices due to Massive MIMO. It is observed that the Neumann series approach is better than brute force computations both in terms of complexity and hardware efficiency. Additionally, with an almost similar gate count the implementation supports a range of matrix sizes from $K = 2, .., 16$. This is important in practice since the size varies based on number of MSs. Furthermore, the proposed approach has a high hardware re-usability, in-particular the generic matrix multiplication unit, which can be used for other purposes in BS when called for.

TABLE I
ASIC Results of Matrix Inversion.

	QR-Decomposition		Direct Inv		This Work	
	GR [7]#	GS [8]	BMI [9]	DMI [10]	$\alpha = 5$	$\alpha = 10$
Order	4×4	4×4	8×8	4×4	$K \times K, K \in (2, 16)$	
Technology	0.18μm	90nm	90nm	0.25μm	65nm	
Gate Count	111K	334K	90K	73K	82K	104K
Max Freq.	100	300	500	170	420	420
Throughput	12.5M	50M	0.65M	1.72M	0.025M	0.51M
N.T*	0.19	0.26	0.016	0.016	0.06	0.12
N.H.E+	0.88	0.39	0.18	0.21	0.74	1.17

# Additional processing required to compute inverse, *i.e.* $R^{-1}Q^H$, which adds to the hardware cost.

* Normalized Throughput for $K = 16$ at 100 MHz = $\frac{\text{Throughput}*\text{order}^3*100}{\text{Freq}*K^3}$

+ Normalized Hardware Efficiency = N.T/(Gate Count)

VI. Conclusion

In this paper we develop a high throughput, hardware efficient matrix inversion required for the ZF linear pre-coder in Massive MIMO. We propose a method to improve the convergence of the Neumann series with tri-diagonal pre-condition matrices, which provides an overall improvement in performance. In particular, lowering the number of iterations in the Neumann series and hence increasing the throughput. Additionally, a VLSI architecture with low latency and cost was implemented to handle the overhead of the proposed improvement. The evaluations were performed for idealistic channels and more detailed studies are ongoing for different measured channel scenarios.

References

[1] T. Marzetta, "Noncooperative cellular wireless with unlimited numbers of base station antennas," *IEEE Trans. Wireless Commun.*, pp. 3590 – 3600, Nov. 2010.

[2] F. Rusek, D. Persson, B. Lau, E. Larsson, T. Marzetta, O. Edfors, and F. Tufvesson, "Scaling up MIMO: Opportunities and challenges with very large arrays," *IEEE Signal Proc.*, vol. 30, pp. 40–60, Jan. 2013.

[3] X. Gao, O. Edfors, F. Rusek, and F. Tufvesson, "Linear pre-coding performance in measured very-large MIMO channels," in *IEEE Veh. Technology Conference (VTC Fall)*, Sept. 2011.

[4] H. Prabhu, J. Rodrigues, O. Edfors, and F. Rusek, "Approximative matrix inverse computations for very-large MIMO and applications to linear pre-coding systems," in *IEEE WCNC*, Apr. 2013.

[5] M. Wu and *et al.*, "Approximate matrix inversion for high-throughput data detection in the large-scale MIMO uplink," in *IEEE Int. Symp. Circuits Syst., ISCAS*, May 2013.

[6] G. Golub and C. Van Loan, *Matrix Computations*. Johns Hopkins University Press, 1996.

[7] Z.-Y. Huang and P.-Y. Tsai, "Efficient implementation of QR decomposition for gigabit MIMO-OFDM systems," *IEEE Trans. Circuits Syst. I*, vol. 58, pp. 2531–2542, Oct. 2011.

[8] Y. Miyaoka, Y. Nagao, M. Kurosaki, and H. Ochi, "Sorted QR decomposition for high-speed MMSE MIMO detection based wireless communication systems," in *IEEE Int. Symp. Circuits Syst.,*, May 2012.

[9] D. Wu and *et al.*, "Fast complex valued matrix inversion for multi-user STBC-MIMO decoding," in *IEEE Computer Symp., ISVLSI*, Mar. 2007.

[10] A. Burg, S. Haene, D. Perels, P. Luethi, N. Felber, and W. Fichtner, "Algorithm and VLSI architecture for linear MMSE detection in MIMO-OFDM systems," in *IEEE Int. Symp. Circuits Syst., ISCAS*, May 2006.