Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/6/20230491

Iterative algorithms and architectures in massive MIMO detection

Liu Qirui

School of Electrical and Electronic Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798

LIUQ0050@e.ntu.edu.sg

Abstract. Massive multiple-input multiple-output (MIMO) wireless systems play an important role in the 5G networks. The complexity of signal detection in massive MIMO is increasing rapidly due to the growth of the number of transmitting antennas. In this paper, we introduced different iterative algorithms to decrease the computational cost of the approximate minimum mean-square error (MMSE) algorithm, including the Neumann Series algorithm, Jacobi method, Gauss-Seidel method, Successive Over Relaxation method and the Conjugate Gradient method. In addition, the VLSI architecture implementations of algorithms mentioned above are also discussed in the article.

Keywords: massive MIMO, detection, architecture, Neumann series, Iterations, Conjugate Gradients.

1. Introduction

The drastic increase of communication data calls new communication technologies. Large-scale MIMO or massive MIMO is an emerging technology, whose prototype, MIMO, has already been used in 3G. Massive MIMO goes a step further with deploying 64, 128 or more antennas, serving users simultaneously, which improves transmission efficiency and spectrum utilization by beam forming and space division multiplexing [1].

Massive MIMO deploys a large amount of antennas, which increases the data throughput at a cost of computational complexity. MIMO computation consists of encoding on the downlink and detection on the uplink, with the latter takes up a majority of the computational complexity [2]. Many algorithms have been proposed to reduce or bypass this computational complexity. When the number of antennas at the base station (BS) is much larger than the number of user devices, many iterative methods based on approximate matrix inversion can greatly reduce the computational complexity due to their linear properties, while their accuracy trade-off is acceptable. Most of these algorithms have been proposed in the field of numerical computation.

The idea of these linear detectors is to replace the large matrix inversion with series of sums and products of components of the matrix. Neumann series approximation (NSA) algorithm was applied in reference [3]. It converts the matrix inversion into a polynomial form, which is only consist of matrix multiplications. Another iterative algorithm called Jacobi method was proposed in reference [4]. Researchers also introduced Gauss-Seidel (GS) Methods in reference [5] and [6], which computes the

DOI: 10.54254/2755-2721/6/20230491

linear equations iteratively at the premise of the Hermitian positive definite property of matrix. Scholars also came up with successive over-relaxation (SOR) [7-9], which is also an iterative method. It can achieve a high performance if a factor ω called relaxation parameter is finely adjusted. The Conjugate Gradient method (CG) goes further to optimize the iteration [10].

To achieve all kinds of these different algorithms, a number of VLSI architectures and techniques are proposed. These designs aim to diminish the computational complexity and resource consumption, such as systolic array deployed in reference [3] and [8] to calculate the gram matrix, Look-Up Tables (LUT) used to improve performance, unit reusing and multiplexing. In this article, we are going to summarize these techniques and compare them, discuss their advantages and draw a conclusion.

The rest of the paper is organized as follows. Section II briefly introduces the NS algorithm. Section III compares the difference between the Jacobi method, the GS method and the SOR method. Section IV described the procedure of the CG method. Finally, conclusions are drawn in Section V.

2. Neumann series approximation

2.1. Uplink system model

Considering a massive MIMO system with N antennas at BS and M user terminals, each UT has a single antenna. Transmitted signal $\boldsymbol{s} = [s_1, ..., s_M]^T$ and received signal $\boldsymbol{y} = [y_1, ...y_N]^T$ are M×1 and N×1 vectors. The wireless transmission channel can be modeled as $\mathbf{y} = \mathbf{Hs} + \mathbf{n}$, where the $\mathbf{H}$ is the uplink channel matrix, and the $\mathbf{n}$ corresponds to the Gaussian additive noise vector.

BS needs to recover the transmitted signal $\mathbf{s}$ from the received signal $\mathbf{y}$ by solving linear equations $\mathbf{y} = \mathbf{Hs} + \mathbf{n}$:

$$\mathbf{y} = \mathbf{Hs} + \mathbf{n}\#(\mathbf{1}) \tag{1}$$

$$\mathbf{H}^{\mathrm{H}}\mathbf{y} = \mathbf{H}^{\mathrm{H}}\mathbf{Hs} + \mathbf{H}^{\mathrm{H}}\mathbf{n} \tag{2}$$

$$\mathbf{y}^{\mathrm{MF}} = \mathbf{Gs} + \mathbf{H}^{\mathrm{H}}\mathbf{n} \tag{3}$$

In MMSE method [2],

$$\mathbf{A} = \mathbf{G}E_s + N_0\mathbf{I}_{\mathrm{M}} \tag{4}$$

And then s is estimated as

$$\hat{\mathbf{s}} = \mathbf{A}^{-1}\mathbf{y}^{\mathrm{MF}} \tag{5}$$

2.2. Neumann series approximation

Neumann Series Approximation is a direct method to reduce the computing complexity of the matrix inversion [11]. Consider this expansion:

$$\boldsymbol{I} - \boldsymbol{B^n} = (\boldsymbol{I} - \boldsymbol{B})\left(\boldsymbol{I} + \boldsymbol{B} + \boldsymbol{B^2} + ... + \boldsymbol{B^{n-1}}\right) \tag{6}$$

For B that $\lim_{n\to\infty} \boldsymbol{B}^n = \mathbf{0}$, let $n \to \infty$, then

$$(\mathbf{I} - \mathbf{B})^{-1} = \sum_{\mathrm{i}=0}^{\mathrm{n}-1} \mathbf{B}^{\mathrm{i}} \tag{7}$$

Let $\mathbf{B} = \mathbf{I} - \mathbf{X^{-1}A}$, above can be also written as

$$\mathbf{A}^{-1} = \sum_{\mathrm{n}=0}^{\infty} \left(\mathbf{X}^{-1}(\mathbf{X} - \mathbf{A})\right)^{\mathrm{n}} \mathbf{X}^{-1} \tag{8}$$

Regularized matrix A in massive MIMO is diagonal dominant. If A is decomposed into $\mathbf{A} = \mathbf{D} + \mathbf{E}$ where D is its main diagonal matrix and E is the remainder, then D is close enough to A that $\lim_{\mathrm{n}\to\infty} \left(\mathbf{I} - \mathbf{D}^{-1}\mathbf{A}\right)^{\mathrm{n}} = \mathbf{0}$. Let $\mathbf{X} = \mathbf{D}$, we can imply that

DOI: 10.54254/2755-2721/6/20230491

$$\mathbf{A}^{-1} = \sum_{n=0}^{\infty} \left(\mathbf{D}^{-1}(-\mathbf{E})\right)^{n} \mathbf{D}^{-1} \tag{9}$$

Since E is a minor part of the matrix, we can keep the first k terms of this polynomial as an approximation:

$$\widetilde{\boldsymbol{A}}_k^{-1} = \sum_{n=0}^{k-1} \left(-\mathbf{D}^{-1}\mathbf{E}\right)^{n} \mathbf{D}^{-1} \tag{10}$$

2.3. Implementation

A VLSI architecture is proposed in reference [3]. The task of the calculation can be divided in to following parts: (1) **G** and **A**; (2) $\mathbf{D}^{-1}$ and **E**; (3) $\widetilde{\mathbf{A}}_{\mathrm{k}}^{-1}$ and $\hat{\mathbf{s}}$.

2.3.1. Gram matrix. The gram matrix can be proved as symmetric; thus, we can compute it with a lower-triangular systolic array. The architecture is shown in Fig.1. We can regard the channel matrix H as a vector of column vectors $\mathbf{H} = [\mathbf{A}_0, \mathbf{A}_1, ..., \mathbf{A}_{n-1}]$, where $\mathbf{A}_i = [\mathbf{H}_{i,0}^{T}, \mathbf{H}_{i,1}^{T}, ..., \mathbf{H}_{i,n-1}^{T}]$, then $\mathbf{G} = \mathbf{H}^{H}\mathbf{H} = [\mathbf{A}_0, \mathbf{A}_1, ..., \mathbf{A}_{n-1}]^{T} \cdot [\mathbf{A}_0, \mathbf{A}_1, ..., \mathbf{A}_{n-1}]$. The transposed matrix $\mathbf{H}^{T}$ is shifted sequentially into the systolic array. Every PE does a multiply-and-accumulate (MAC) process with both inputs and then pass them on. When the input arrives at the diagonal PE, it is conjugated and passed to the lower column.

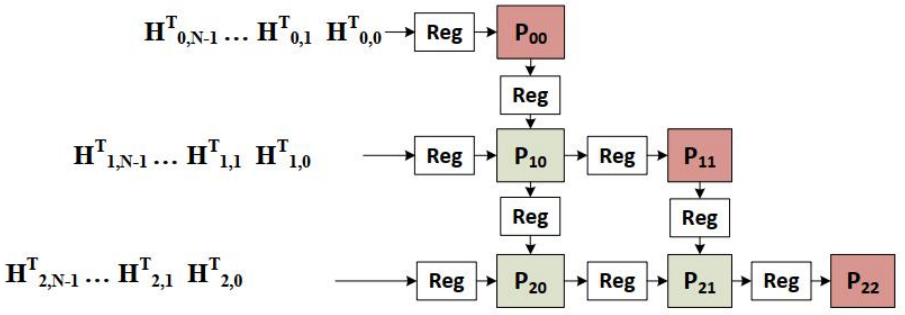

Figure 1. Systolic array architecture [3].

2.3.2. Diagonal matrix inversion. As **D** is a diagonal matrix and diagonal elements of **A** are all real, its reversion only needs to calculate the reciprocal of each diagonal term. This computation can be implemented by using Look-Up Tables (LUT) efficiently.

2.3.3. Neumann series. The implementation of the NSA method usually reserves the first 2 terms of the series, so that the complexity is relatively low. The 2-term approximation of $\mathbf{A}^{-1}$ is

$$\widetilde{\mathbf{A}}_2^{-1} = \mathbf{D}^{-1} - \mathbf{D}^{-1}\mathbf{E}\mathbf{D}^{-1} \tag{11}$$

The approximation needs to wait for the result of $\mathbf{D}^{-1}$, and then traverse through the strictly lower triangle matrix **E** to get the final estimation of the transmission vector. This process can also be pipelined as the systolic array.

3. Iterative solution of linear equations

In MMSE method, the mainly request is to solve the linear matrix equation:

$$\hat{\boldsymbol{s}} = \mathbf{A}^{-1}\boldsymbol{y}^{MF} \tag{12}$$

However, the computation complexity becomes extremely large when antenna amount in MIMO system increases, which requires methods to avoid the matrix inversing. Realizing the equation has the form of $\mathbf{A}\boldsymbol{s} = \boldsymbol{y}$, we can exploit the iterative method of solving the linear equations. Decomposing

DOI: 10.54254/2755-2721/6/20230491

matrix **A** into $\mathbf{A} = \mathbf{M} - \mathbf{N}$,where the matrix **M** is nonsingular, the iteration equation can be derived as below:

$$\mathbf{M}\boldsymbol{s}^{\mathrm{k}+1} = \mathbf{N}\boldsymbol{s}^{\mathrm{k}} + \boldsymbol{y} \tag{13}$$

$$\boldsymbol{s}^{\mathrm{k}+1} = \mathbf{M}^{-1}\mathbf{N}\boldsymbol{s}^{\mathrm{k}} + \mathbf{M}^{-1}\boldsymbol{y} \tag{14}$$

To ensure the iteration converges, the spectral radius of iteration matrix $\mathbf{M}^{-1}\mathbf{N}$ should satisfies$\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$. Usually, the diagonal elements of **A** are chosen to be the matrix **M**, in order to turn the matrix inversion into simple elements reciprocal. From different **M** can derive different iterative methods, including Jacobi iteration, Gauss-Seidel iteration, SOR method and so on. Details of these iteration methods will be introduced below.

3.1. Jacobi iteration

A Jacobi iteration method and implementation is proposed in article [4].

Decomposing the matrix **A** into

$$\mathbf{A} = \mathbf{D} - \mathbf{L} - \mathbf{U} \tag{15}$$

where **D**, **L**, **U** are respectively diagonal matrix, strictly lower triangle matrix and strictly upper triangle matrix of **A**. Thus for $\mathbf{A}\boldsymbol{s} = \boldsymbol{y}$, we can derive:

$$\boldsymbol{s} = \mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})\boldsymbol{s} + \mathbf{D}^{-1}\boldsymbol{y} \tag{16}$$

Turn that into iteration, we can get

$$\boldsymbol{s}^{(k+1)} = \mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})\boldsymbol{s}^{(k)} + \mathbf{D}^{-1}\boldsymbol{y} \tag{17}$$

where $\mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})$ is the iteration matrix. Since **D** is diagonal, its inverse matrix can be computed by simply taking the reciprocal of each element.

The initial solution of the iteration $s^{(0)}$ can be decided as

$$\boldsymbol{s}^{(0)} = \mathbf{D}^{-1}\boldsymbol{y} \tag{18}$$

which is the first term of the Neumann series.

The implementation in reference [4] introduced a decomposition of the approximation matrix so that it won't need to be computed any more, which reduced the computation complexity from $\mathcal{O}(B \times U^2)$ to $\mathcal{O}(B \times U)$.

3.2. Gauss-seidel iteration

Article [5] and [6] came up with the Gauss-Seidel iteration.

Still using the decomposition $\mathbf{A} = \mathbf{D} - \mathbf{L} - \mathbf{U}$, but this time when computing each row of the iteration except the first row, we substitute the result computed before. Then the iteration can be inferred as

$$\boldsymbol{s}^{(k+1)} = (\mathbf{D} + \mathbf{L})^{-1}\mathbf{U}\boldsymbol{s}^{(k)} + (\mathbf{D} + \mathbf{L})^{-1}\boldsymbol{y} \tag{19}$$

This equation is the Gauss-Seidel (GS) iteration, where $(\mathbf{D} + \mathbf{L})^{-1}\mathbf{U}$ is the iteration matrix. The inversion of the lower triangle matrix $\mathbf{D} + \mathbf{L}$ is a little bit more complex than **D**. Let **L'** denote the **D+L**, suppose matrix is the inversion of **L'**, it can be inferred that F is also a lower triangle matrix. from **AF = I** we can derive that the element of **F**, $\boldsymbol{f}_{ij}$, is determined by every **A** elements on its right and every **F** elements above. Thus we can get the inversion in a sequential way: first the diagonal elements $\boldsymbol{a}_{ii}$, then $\boldsymbol{a}_{i+1,i}$ and so on.

The GS method converges faster than Jacobi iteration, however it cannot be computed in a parallel way due to the sequential structure.

Reference [5] used the Neumann Series to provide the initial solution for the iteration instead of a zero vector, which makes the convergence much faster.

DOI: 10.54254/2755-2721/6/20230491

The Numerical simulation result of the GS algorithm shows that it has an improvement on convergence rate compared to the NSE-based algorithm, while the 2-term initial solution also improved the performance compares to NSE and Cholesky algorithm.

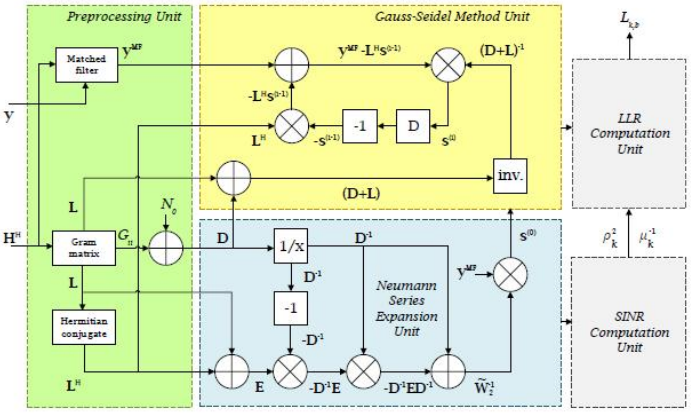

Figure 2. GS-based detection architecture [6].

Fig.2. above is the architecture of the implementation. The calculation is divided into 5 unit: Preprocessing unit, 2-terms NSE unit, GS method unit, SINR computation and LLR computation unit. A systolic array is deployed in GS method unit to compute the inverse of **D+L**.

3.3. Susscessive over-relaxation

Successive Over-Relaxation (SOR) method is a generalization of GS method. Reference [7], [8] and [9] raised the algorithm and architecture of the SOR iteration. The SOR method utilizes a weighted average of the current result and the last result with a weight ω, instead of the origin result:

$$\boldsymbol{s}^{(k+1)} = (1-\omega)\boldsymbol{x}^{(k)} + \omega\boldsymbol{x}^{'(k+1)} \tag{20}$$

$\boldsymbol{x}^{'(k+1)}$ here denotes iteration results from the GS method. Take the GS iteration equation:

$$\boldsymbol{s}^{(k+1)} = (\mathbf{D}-\omega\mathbf{L})^{-1}[(1-\omega)\mathbf{D}+\omega\mathbf{U}]\boldsymbol{s}^{(k)} + (\mathbf{D}-\omega\mathbf{L})^{-1}\omega\boldsymbol{y} \tag{21}$$

Only calculation related to relaxation factor ω is newly introduced in the SOR method. The performance of SOR varies with different ω, which means convergence of iteration could be faster under proper ω while improper ω may lead to a performance worse than the GS method.

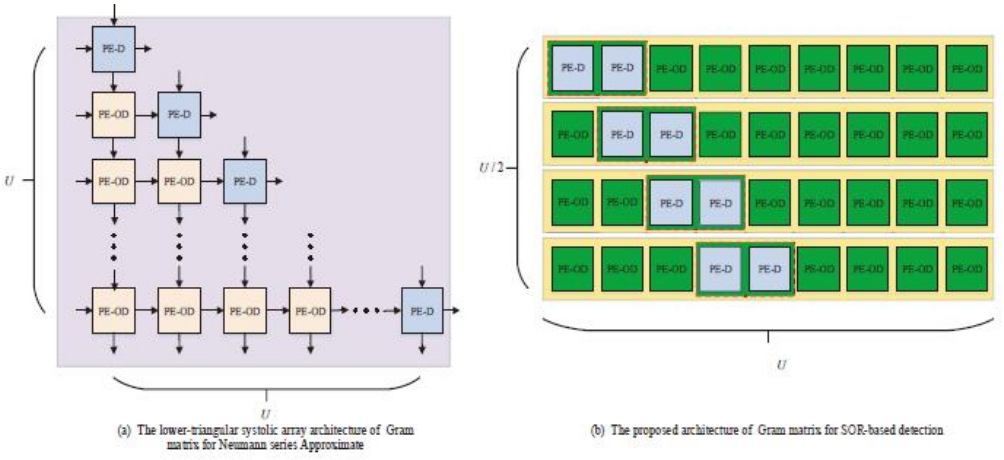

Figure 3. Architecture of systolic array in SOR-based detection [9].

DOI: 10.54254/2755-2721/6/20230491

The performance of the SOR algorithm against the relaxation factor has been verified with simulation in reference [7]. From the result it can be inferred that the optimal ω for the system is 1.05. The simulation also shows that the SOR method outperforms the conventional NSA methods. In reference [8], the gram matrix is computed in a triangular systolic array, where two diagonal processing unit (PE-D) can be combined as one off-diagonal processing unit (PE-OD), thus the implementation only includes one type of PE.

4. Conjugate gradient method

4.1. Coordinate descent method

All linear iterative methods above are acquired by simply decomposing the matrix, which may lead to a relatively slow convergence. Take a look again at the target equation

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b} \tag{22}$$

Where **A** is a real symmetric positive definite matrix. The idea is to transform it into the problem of finding the minimum point of a function in an n-dimensional vector space, which is the following n-ary quadratic function:

$$\varphi(x) = \frac{1}{2}\boldsymbol{x}^T\mathbf{A}\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x} \tag{23}$$

It can be proved that the minimum point of the function has a partial differential of 0, which is actually the original linear system of equations.

This property shows that this minimum point of the function is exactly the solution of the system of equations. This method of transforming a system of symmetric positive definite linear equations into the problem of finding the minimum of a multidimensional quadratic function is called the Calculus of variations.

In general, the problem of finding the minimum value of a multivariate function is an unconstrained optimization problem, which is often solved by the successive search of multidimensional vectors. The basic idea is as follows: given an arbitrary point, suppose we search for the next point along the direction vector $\boldsymbol{p}_0$

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \alpha_0\boldsymbol{p}_0 \tag{24}$$

Let $\boldsymbol{x}_1$ be the minimum point in this direction, and then select the search direction to repeat this process. The key problem of this method is to determine the search direction $\boldsymbol{p}$ and the search step length α.

Determine the search step length first. Suppose that the search direction pk has been selected at point $\boldsymbol{x}_k$, then let

$$F(a) = \varphi\,(\boldsymbol{x}_k + \alpha\boldsymbol{p}_k) \tag{25}$$

Then the search step length is the value of α that minimizes the unary function. Take φ to get

$$\begin{aligned} f(\alpha) &= \frac{1}{2}(\boldsymbol{x}_k + \alpha\boldsymbol{p}_k)^T\mathbf{A}(\boldsymbol{x}_k + \alpha\boldsymbol{p}_k) - \boldsymbol{b}^T(\boldsymbol{x}_k + \alpha\boldsymbol{p}_k) \\ &= \frac{1}{2}\alpha^2\boldsymbol{p}_k^T\mathbf{A}\boldsymbol{p}_k - \alpha\boldsymbol{r}_k^T\boldsymbol{p}_k + \varphi(\boldsymbol{x}_k) \end{aligned} \tag{26}$$

Where $\boldsymbol{r}_k = \boldsymbol{b} - \mathbf{A}\boldsymbol{x}$ is the residual of the system of equations. Since the function is a simple unary quadratic function, the matrix **A** is symmetric and positive definite. It is easy to know that $f(\alpha)$ has a unique minimum.

Different methods adopt different strategies to determine the search direction. The strategy of steepest descent method is as follows.

Considering that the direction in which the multivariate function φ increases the fastest is its gradient direction, the negative gradient direction is the direction in which the function decreases the

DOI: 10.54254/2755-2721/6/20230491

fastest and most obviously and thus is selected as the search direction each time. $\varphi = \frac{1}{2}\boldsymbol{x}^T\mathbf{A}\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x}$, thus the negative gradient direction is also $\boldsymbol{b} - \mathbf{A}\boldsymbol{x}$, which is the residual $\boldsymbol{r}$.

Substituting the direction into $\varphi(\boldsymbol{x}_k + \alpha\boldsymbol{p}_k)$, we can get the derivative with respect to α, from which the α minimums the function turns out to be

$$\alpha = \frac{\boldsymbol{r}_k^T\boldsymbol{p}_k}{\boldsymbol{p}_k^T\mathbf{A}\boldsymbol{p}_k} \tag{27}$$

It is observed that at the end of each search, the current search direction is in the tangential direction of the isoline, and the next search direction is in the normal direction of the isoline, so the two adjacent steps are always perpendicular to each other.

5. Conjugate gradients method

In the steepest descent method, the negative gradient direction, although locally the best search direction, does not globally minimize φ. Conjugate gradients (CG) method is an improvement of it.

Given the initial vector x0, the negative gradient direction is still chosen as the search direction in the first step, $\boldsymbol{p}_0 = \boldsymbol{r}_0$; meanwhile for the rest of the iteration in the conjugate gradient method, $\boldsymbol{p}$ is no longer along the negative gradient direction, and the original $\alpha\boldsymbol{p}_1$ is replaced by $\xi\boldsymbol{r}_1 + \varphi\boldsymbol{p}_0$. The relation between ξ and φ is obtained by solving the equation with partial derivative of $f(\xi, \varphi)$ equal to 0, which is normalized to β:

$$\boldsymbol{\beta}_{k-1} = -\frac{\boldsymbol{r}_k^T\mathbf{A}\boldsymbol{p}_{k-1}}{\boldsymbol{p}_{k-1}^T\mathbf{A}\boldsymbol{p}_{k-1}} \tag{28}$$

$$\boldsymbol{p}_k = \boldsymbol{r}_k + \beta_{k-1}\boldsymbol{p}_{k-1} \tag{29}$$

As soon as the p_k is decided, the iteration solution can be computed as $\boldsymbol{x}_1 = \boldsymbol{x}_0 + \alpha_0\boldsymbol{p}_0$, where the computation of factor α is mentioned previously. Due to the properties of Krylov subspace iteration method [12], for N-dimensional matrices, $\boldsymbol{x}_n$ can minimize the equation in the whole space $\mathbf{R}^n$, which means $\boldsymbol{x}_n$ is the exact solution of the original equation.

A reconfigurable VLSI architecture has been proposed in reference [10]. With a reconfigurable processing element array which can perform the matrix-vector productions, vector dot products, scaled vector additions/subtractions and scalar divisions, an implementation of the CG-based algorithm is deployed.

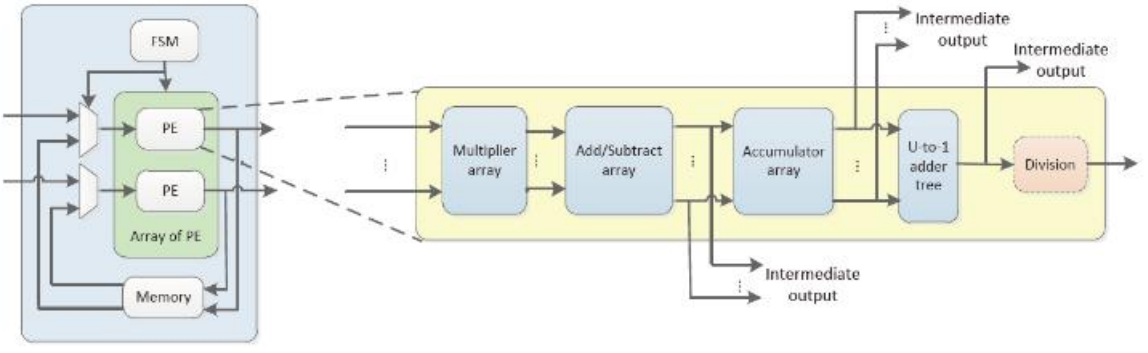

Figure 4. Reconfigurable processing elements (PEs) for CG-based detection [10].

6. Conclusions

We have discussed 5 kinds of iteration methods of massive MIMO detection with the implementation. The Neumann Series algorithm is the simplest method of approximate matrix inversion, which is often used as a preconditioner to initialize the other algorithm. The Jacobi, Gauss-Seidel and Successive

DOI: 10.54254/2755-2721/6/20230491

Over Relaxation method have a similar idea, while the Jacobi method is more suitable for parallel computation, and the SOR need an appropriate ω to maximum the convergence. The Conjugate Gradient method converges faster than the linear iteration method above theoretically, and a proper preconditioner can be helpful to accelerate the convergence.

There is a positive prospect in future research. Other linear iterative methods, projection methods and Krylov subspace methods can be tested, while preconditioning techniques and parallel implementations can be deployed in the research.

References

[1] Liu L, Peng G, Wei S. 2019 *Massive MIMO Detection Algorithm and VLSI Architecture*[M]. Springer Singapore.

[2] Albreem M A, Juntti M, Shahabuddin S. 2019Massive MIMO detection techniques: A survey[J]. *IEEE Communications Surveys & Tutorials,* 21(4): 3109-3132.

[3] Wu M, Yin B, Vosoughi A, et al. 2013 Approximate matrix inversion for high-throughput data detection in the large-scale MIMO uplink[C]//*2013 IEEE international symposium on circuits and systems (ISCAS).* IEEE, p2155-2158.

[4] Jiang F, Li C, Gong Z. 2017 A low complexity soft-output data detection scheme based on Jacobi method for massive MIMO uplink transmission[C]//*2017 IEEE international conference on communications (ICC).* IEEE, 2017: 1-5.

[5] Wu Z, Zhang C, Xue Y, et al. 2016 Efficient architecture for soft-output massive MIMO detection with Gauss-Seidel method[C]//*2016 IEEE international symposium on circuits and systems (ISCAS).* IEEE, 2016: 1886-1889.

[6] Zeng J, Lin J, Wang Z. 2018 An improved Gauss-Seidel algorithm and its efficient architecture for massive MIMO systems[J]. *IEEE Transactions on Circuits and Systems II: Express Briefs,* 65(9): 1194-1198.

[7] Gao X, Dai L, Hu Y, et al. 2014 Matrix inversion-less signal detection using SOR method for uplink large-scale MIMO systems[C]//*2014 IEEE Global Communications Conference. IEEE,* 2014: 3291-3295.

[8] Zhang P, Liu L, Peng G, et al. 2016 Large-scale MIMO detection design and FPGA implementations using SOR method[C]//*2016 8th IEEE International Conference on Communication Software and Networks (ICCSN).* IEEE, 2016: 206-210.

[9] Yu A, Zhang C, Zhang S, et al. 2016 Efficient SOR-based detection and architecture for large-scale MIMO uplink[C]//*2016 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS).* IEEE, 2016: 402-405.

[10] Yin B, Wu M, Cavallaro J R, et al. 2015 VLSI design of large-scale soft-output MIMO detection using conjugate gradients[C]//*2015 IEEE International Symposium on Circuits and Systems (ISCAS).* IEEE, 2015: 1498-1501.

[11] Meyer C D. 2000 *Matrix analysis and applied linear algebra*[M]. Siam.

[12] Saad Y. 2003 *Iterative methods for sparse linear systems*[M]. Society for Industrial and Applied Mathematics.