**COMPARATIVE STUDY OF LINEAR PRECODING TECHNIQUES**

**ABSTRACT**

This paper mainly studies the large-scale MIMO downlink precoding algorithm in 5G key technology. Firstly, the key technology of 5G and precoding technology are summarized. Then, the maximum ratio emission (MRT), zero forcing (ZF), minimum mean square error (MMSE) and other precoding methods are analyzed in detail. Finally, the capacity and bit error rate of the main algorithms are compared by simulation.

**Keywords:** Linear pre-coding Algorithm, Block Diagonalization (BD), Maximum Ratio Transmission (MRT), Zero Forcing (ZF) Minimum Mean Square Error (MMSE), 5G mobile communication technology, MIMO.

**CHAPTER 1**

**INTRODUCTION**

1. 5G mobile communication technology At present, with the development of technology, people begin to put forward higher requirements for the mobile system: compared with the 4G system's 10-100 times of user speed requirements, 10-100 times of the number of wireless device connections, etc., 5G has made great technical evolution and performance upgrade in speed, delay, power consumption, connection, terminal, etc. At the same time, 5G will also have the following key technologies:(1)large scale dense network (large scale distributed MIMO): this technology provides flexible 5G intensive cells. It is a transmission point composed of many inexpensive antennas, which can serve multiple users at the same time. With a large-scale MIMO system, multiple messages of several terminals can be transmitted on the same time and frequency resources, and get the maximum beamforming gain and the minimum interference, so it can improve the system capacity and network coverage, and then maximize the utilization of frequency resources.(2)Cognitive Radio: in the case of no interference to other devices, mobile phones can adaptively find unused frequency bands by continuously detecting the frequency, and allow different radio technologies to effectively share the same spectrum by changing the transmission scheme. This kind of perception is all-round, and should have a comprehensive understanding of the comprehensive information based on the geographical environment and climate conditions of the location. The above dynamic management of wireless resources is realized by distributed and software defined radio.(3)ultra wideband spectrum: because the capacity of channel increases with the increase of bandwidth, in order to achieve the Gbps order of magnitude communication rate required by 5G mobile communication, it should also have the continuous bandwidth of high frequency band. However, due to the strong low-frequency penetration, in addition to the highfrequency band, 5G also needs to use the low-frequency band. B. Precoding technology Precoding is a process of signal preprocessing based on known channel state information at the transmitter of downlink. The precoder can be regarded as a multi-mode beamformer, which forms the spatial orthogonal characteristic beam after the transmission signal is decoupled, so that the interference between each user and antenna will be minimized, and according to the current channel condition, more energy will be allocated in the channel with better channel condition and relatively stable channel, and less or no energy will be allocated to the poor channel, so as to improve the spectrum efficiency and channel capacity of the large-scale antenna system, simplify the algorithm complexity of the receiver, reduce the bit error rate, obtain a better signal-to-noise ratio, and ultimately optimize the performance of the system. And after the mobile station receives the signal, it only needs to process the signal more simply. According to the design scheme, the precoding technology can be divided into linear and nonlinear precoding technology. Linear Precoding schemes include: maximum ratio emission precoding scheme, zero forcing precoding scheme, block diagonalization precoding scheme, minimum mean square error precoding scheme and other precoding schemes. The processing methods of nonlinear precoding mainly include Tomlinson-Harashima precoding and vector precoding. In the conventional scenario, because of the high complexity of nonlinear precoding and the ideal performance of Linear Precoding, Linear Precoding is more suitable for practical scenarios.
2. Precoding technology

Precoding is a process of signal preprocessing based on known channel state information at the transmitter of downlink. The precoder can be regarded as a multi-mode beamformer, which forms the spatial orthogonal characteristic beam after the transmission signal is decoupled, so that the interference between each user and antenna will be minimized, and according to the current channel condition, more energy will be allocated in the channel with better channel condition and relatively stable channel, and less or no energy will be allocated to the poor channel, so as to improve the spectrum efficiency and channel capacity of the large-scale antenna system, simplify the algorithm complexity of the receiver, reduce the bit error rate, obtain a better signal-to-noise ratio, and ultimately optimize the performance of the system. And after the mobile station receives the signal, it only needs to process the signal more simply

According to the design scheme, the precoding technology can be divided into linear and nonlinear precoding technology. Linear Precoding schemes include: maximum ratio emission precoding scheme, zero forcing precoding scheme, block diagonalization precoding scheme, minimum mean square error precoding scheme and other precoding schemes. The processing methods of nonlinear precoding mainly include Tomlinson-Harashima precoding and vector precoding. In the conventional scenario, because of the high complexity of nonlinear precoding and the ideal performance of Linear Precoding, Linear Precoding is more suitable for practical scenarios.

**CHAPTER 2**

**LITERATURE SURVEY**

**[1] Rusek F, Persson D, Lau B K, et al. Scaling up MIMO: Opportunities and challenges with very large arrays[J]. Signal Processing Magazine, IEEE, 2013, 30(1): 40-60.**

Multiple-input multiple-output (MIMO) technology is maturing and is being incorporated into emerging wireless broadband standards like long-term evolution (LTE) [1]. For example, the LTE standard allows for up to eight antenna ports at the base station. Basically, the more antennas the transmitter/receiver is equipped with, and the more degrees of freedom that the propagation channel can provide, the better the performance in terms of data rate or link reliability. More precisely, on a quasi static channel where a code word spans across only one time and frequency coherence interval, the reliability of a point-to-point MIMO link scales according to Prob(link outage) ` SNR-ntnr where nt and nr are the numbers of transmit and receive antennas, respectively, and signal-to-noise ratio is denoted by SNR. On a channel that varies rapidly as a function of time and frequency, and where circumstances permit coding across many channel coherence intervals, the achievable rate scales as min(nt, nr) log(1 + SNR). The gains in multiuser systems are even more impressive, because such systems offer the possibility to transmit simultaneously to several users and the flexibility to select what users to schedule for reception at any given point in time [2]..

**Summary:** Studied about the Opportunities and challenges with very large arrays.

**[2] Zarei S, Gerstacker W, Schober R.** **A low-complexity linear precoding and power allocation scheme for downlink massive MIMO systems[C]//Signals, Systems and Computers, 2013 Asilomar Conference on.IEEE, 2013: 285-290 .**

Cell-Free Massive multiple-input multiple-output (MIMO) comprises a large number of distributed, low cost, low power single antenna access points (APs), connected to a network controller. The number of AP antennas is significantly larger than the number of users. The system is not partitioned into cells and each user is served by all APs simultaneously. The simplest linear precoding schemes are conjugate beamforming and zeroforcing. Max-min power control provides equal throughput to all users and is considered in this work. Surprisingly, under maxmin power control, most APs are found to transmit at less than full power. The zero-forcing precoder significantly outperforms conjugate beamforming. For zero-forcing, a near-optimal power control algorithm is developed that is considerably simpler than exact max-min power control. An alternative to Cell-Free systems is small-cell operation in which each user is served by only one AP for which power optimization algorithms are also developed. Cell-Free Massive MIMO is shown to provide five- to ten-fold improvement in 95%-likely per-user throughput over small-cell operation.

**Summary:** Studied about the A low-complexity linear precoding and power allocation scheme for downlink massive MIMO systems.

**[3] Ngo H Q, Larsson E G, Marzetta T L. Energy and spectral efficiency of very large multiuser MIMO systems[J]. Communications, IEEE Transactions on, 2013, 61(4): 1436-1449**

A multiplicity of autonomous terminals simultaneously transmits data streams to a compact array of antennas. The array uses imperfect channel-state information derived from transmitted pilots to extract the individual data streams. The power radiated by the terminals can be made inversely proportional to the square-root of the number of base station antennas with no reduction in performance. In contrast if perfect channel-state information were available the power could be made inversely proportional to the number of antennas. Lower capacity bounds for maximum-ratio combining (MRC), zero-forcing (ZF) and minimum mean-square error (MMSE) detection are derived. An MRC receiver normally performs worse than ZF and MMSE. However as power levels are reduced, the cross-talk introduced by the inferior maximum-ratio receiver eventually falls below the noise level and this simple receiver becomes a viable option. The tradeoff between the energy efficiency (as measured in bits/J) and spectral efficiency (as measured in bits/channel use/terminal) is quantified for a channel model that includes small-scale fading but not large-scale fading. It is shown that the use of moderately large antenna arrays can improve the spectral and energy efficiency with orders of magnitude compared to a single-antenna system.

**[4] Zarei S, Gerstacker W, Muller R R, et al. Low-complexity linear precoding for downlink large-scale MIMO systems[C]//Personal Indoor and Mobile Radio Communications (PIMRC), 2013 IEEE 24th International Symposium on. IEEE, 2013: 1119-1124.**

In this work, we present a low-complexity linear precoding scheme for downlink large-scale multiple-input multiple-output (MIMO) systems. The proposed scheme can achieve near minimum mean square error (MMSE) precoding performance in terms of the sum rate and is based on a matrix polynomial instead of matrix inversion. Simulation results show that matrix polynomials consisting of only a few terms are sufficient to closely approach the sum rate of the classical MMSE precoder and to perform orders of magnitude better than the simple conjugate beamforming (BF) precoder. We derive exact expressions for the computational complexity of the proposed scheme in terms of the number of additions and multiplications and compare it to the complexity of the BF and MMSE precoders. Our complexity analysis shows that for large number of base station antennas N compared to the number of generated transmit symbols τ per channel estimate and large number of users K, the proposed polynomial precoder has a lower complexity than the classical MMSE precoder.

**Summary:** Studied about low complexity linear pre coding for downlink large scale mimo

**CHAPTER 3**

1. Cell-Free System Model Unlike cellular wireless networks, in a cell-free system we do not partition the network into cells and do not assign users to particular base stations. Instead we assume that a geographical area is covered by M randomly distributed single antenna APs. We assume that in this area there are K single antenna users, and that M ≫ K. An example of a cellfree system is shown in Figure 1. In contrast to a standard cellular network, in a cell-free system each user is served not by one base station, but by all APs simultaneously. All APs are connected to a NC (not shown in Figure 1). We use a flat fading channel model for each OFDM subcarrier. The OFDM subcarrier index is omitted for simplicity. Size of the overall area is considered small enough that the largest difference between propagation time from any two APs to a user is smaller than the duration of the OFDM cyclic prefix. The channel coefficient between AP m and user k is given by

gmk = p βmkhmk, (1)

where βmk is the large scale fading coefficient which accounts for path loss and shadowing effects. This coefficient changes slowly and hence can be accurately estimated and tracked. Throughout the text we assume the NC knows the coefficients βmk, ∀m, k. The second factor hmk ∼ CN (0, 1) is the small scale fading coefficient. We assume that these coefficients are i.i.d. random variables that stay constant during a coherent 3 interval and are independent in different coherent intervals. For a wide-band OFDM system βmk is independent of frequency, while hmk has frequency dependence and a Nyquist sampling interval in frequency that is equal to the reciprocal of the channel delay-spread. We denote by G ∈ CM×K, [G]mk = gmk the channel matrix between all APs and users. We further assume channel reciprocity, i.e., that the uplink and downlink channel coefficients are the same. We focus on the scenario of users with mobility less than 10km/h. In other words we assume that most of our users are pedestrians, which is typically the case in real life scenarios. B. Channel Estimation The main idea of cell-free systems is that each user is served by all APs. In order to reduce interference between signals intended for different users the APs should form their transmitted signals by taking into account the channel coefficients. Thus, estimation of these coefficients is an important part of the communication protocol. In this work we assume that the TDD protocol described in [23, Figure 3], is used. At the first step of this protocol all users simultaneously and synchronously transmit pilot sequences ψ1 , · · · , ψK ∈ C τ , which propagate to all M APs. At the second step all APs get estimates gˆmk of gmk and use these estimates to beamform data to all users. We assume that pilot sequences ψi , i = 1, · · · , K, assigned to the corresponding users are orthonormal, i.e., ψ H i ψj = δij . As we mentioned before we assume the mobility of users less than 10km/h. For such speeds and carrier frequency of 1.9 GHz, the coherence interval is large, which enables using a large number of orthogonal pilots for channel estimation. Hence it is reasonable to assume that these pilots are assigned to users in such way that users with the same pilot are located far away from each other and the pilot contamination (coherent interference resulting from two or more users sharing the same pilot sequence) is negligible Let g˜mk = gmk − gˆmk be the channel estimation error. It is well known that gˆmk and g˜mk are uncorrelated and (see [24, Chapter 11]) gˆmk ∼ CN (0, αmk), g˜mk ∼ CN (0, βmk − αmk), (2) where αmk = ρrτβ2 mk 1+ρrτβmk . In the following sections we analyze two main linear precoding schemes in downlink transmission: conjugate beamforming precoding and zero-forcing precoding. As mentioned in section I, throughout this paper we consider the max-min optimization problems.

III. CONJUGATE BEAMFORMING

In this section, we consider CB precoding combined with transmit power optimization. Conjugate Beamforming with Power Optimization • AP m estimates βmk, k = 1, · · · , K and sends them to the NC. • NC computes power coefficients ηmk, ∀m, ∀k (defined later in this section) as a function of large scale fading coefficients (βmk) and sends them to corresponding APs. • Users synchronously transmit pilot sequences ψi , i = 1, · · · , K. • AP m gets estimates gˆmk, k = 1, · · · , K. • With conjugate beamforming precoding, the m-th AP transmits the signal xm = √ ρf X K i=1 √ ηmigˆ ∗ misi , (3) where si is data signal intended to user i, with E |si | 2 = 1, and ρf is the transmit power limit of each AP. The quantity ηmi is the power coefficient used by AP m for transmission to user i. By optimizing coefficients ηmi we hope to significantly increase the system performance. The signal received by the k-th user is yk = X M m=1 gmkxm + wk, (4) where wk ∼ CN (0, 1) is additive noise. We assume that user k is only aware of the statistics of the estimated channel coefficients E |gˆmk| 2 = αmk, ∀m, which is a result of channel hardening in Massive MIMO systems [2]. A general capacity lower bound for Massive MIMO systems has been derived in [18] and a more specific bound for cell-free systems is given in [22]. With our notations, the downlink achievable rate of user k for CB is Rk = log2 (1 + SINRk), where SINRk = ρf PM m=1 √ηmkαmk2 1 + ρf PK i=1 PM m=1 ηmiβmkαmi . (5) For the sake of completeness, we present a proof of this bound in appendix A. Note that the achievable rate using the SINR expression in (5) is obtained under the assumption that users are only aware of statistics of channel coefficients. In [22, Figure 2] it is shown that in Massive MIMO systems the achievable ra 4 (Rk) obtained by this assumption is close to the achievable rate in the case where the users know the instantaneous channel gain. Also note that the achievable rate using (5) is a function of only the large scale fading coefficients and not of the small scale fading coefficients. Therefore, for conducting transmit power optimization it is enough for APs to transmit (by backhaul network) only coefficients βmk to the NC. Using these coefficients, the NC finds optimal, or near optimal, power coefficients ηmk and conveys them to the corresponding APs. Note that coefficients βmk do not depend on OFDM subcarrier index and change slowly (about 40 times slower than small scale fading coefficients [25]). Thus, the overall needed backhaul traffic is quite small. In a wide-band system the SINR has no frequency dependence, and power coefficients are independent of frequency as well. In the following subsections we present optimal and suboptimal power optimization algorithms

**Disadvantages:**

* High Complexity: Precoding and power optimization algorithms can be complex and computationally demanding, particularly in large-scale systems. This can lead to increased processing time and system overheads.
* Limited Scalability: Precoding and power optimization techniques may have limited scalability, particularly as the number of users or antennas in the system increases. This can limit the potential gains that can be achieved by these techniques.
* Channel Estimation: Precoding and power optimization algorithms rely on accurate channel state information (CSI) to optimize system performance. However, obtaining accurate CSI can be challenging, particularly in dynamic environments or when using low-complexity algorithms.
* Interference Management: Precoding and power optimization algorithms may not be effective at managing interference in all scenarios. In some cases, interference may be difficult to mitigate, particularly when operating in crowded or congested environments.

**CHAPTER 4**

**PROPOSED METHOD**

We studied the performance of several main Linear Precoding algorithms in large-scale MIMO systems. For the convenience of expression, in a single cell system, we use indicates the downlink channel matrix from base station to user. Based on the theoretical analysis, we studied the performance of the main Linear Precoding algorithms, and made performance simulation under the actual scene conditions, and compared with the theoretical results.





In the single cell large-scale MIMO transmitter block diagram shown in Fig. 1, the base station precodes the signal and sends the signal vector to the user. S represents the original signal, and X represents the information vector sent by the sender to the user after precoding.

**A. Maximum Ratio Emission Precoding Algorithm**

In downlink, the MRT precoding technology is actually the matched filter precoding algorithm. The performance of MRT precoding in large-scale MIMO system depends on the channel transmission environment to a great extent, and the ideal environment is that the channels from the base station to different user terminals are as independent as possible. The expression of MRT precoding for the kth user in the cell is formula 3。MRT precoding is a very simple precoding technology, which can maximize the SNR of each user, but does not consider the interference between users.



**B. Zero Forcing Precoding Algorithm**

Zero forcing linear precoding scheme was originally proposed by Freescale Semiconductor Company. Different from the MRT precoding technology, ZERO FORCING precoding can completely remove the interference among users. It requires that all the signals received by users in the system do not contain the interference generated by other users, that is, make the precoding vector wk of user K in the channel matrix of other users in the zero space of, that is, the interference items of other users in the signals received by user K.



**C. Minimum Mean Square Error Precoding Algorithm**

Zero forcing precoding scheme reduces the interference of other users to zero, and does not consider the impact of noise on the system. When the noise causes the channel matrix to become ill conditioned or near ill conditioned, and the coefficient will be close to zero, the received signal component will be weakened, and the received signal to interference plus noise ratio (SINR) will be reduced. Meanwhile, the system will be affected. The difference is that in 2005, Christian B. peel, Bertrand M. hochwald and A. Lee swindlehurst proposed the minimum mean square error precoding scheme. It aims to maximize the SINR of the receiver, comprehensively considers the performance of the whole system, and makes MMSE precoding scheme perform better in the environment of low SNR by introducing parameters. The precoding matrix can be expressed as follows.



Next, the performance of several representative Linear Precoding algorithms described above is simulated. Assuming that the channel is Rayleigh fading, the channel matrix is made up of elements randomly generated by CN (0,1) with independent and same distribution, and normalized. The total transmit power of each base station is fixed, and the power is evenly distributed. In the scenario, it is assumed that the number of transmit antennas is m = 20, the number of receive users is k = 20, and the number of receive antennas for each user is 1. Here, we use Matlab software to write simulation program for simulation.

In this paper, the main performance indexes of MRT, ZF and MMSE Linear Precoding schemes are analyzed under the model of single cell MIMO. At the same time, the rate and bit error rate of MRT, ZF and MMSE precoding are compared by simulation, and MMSE is obtained The precoding technology is obviously better than the other two precoding technologies, especially in the environment of low SNR, which verifies the conclusion of theoretical derivation.

****

**CHAPTER 5**

**ADVANTAGES AND APPLICATIONS**

**Advantages:**

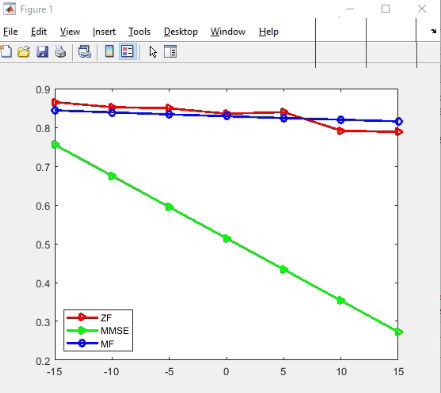
* Improved Signal Quality.
* Increased Capacity.
* Better Spectral Efficiency.
* Enhanced Robustness
* Lower Power Consumption

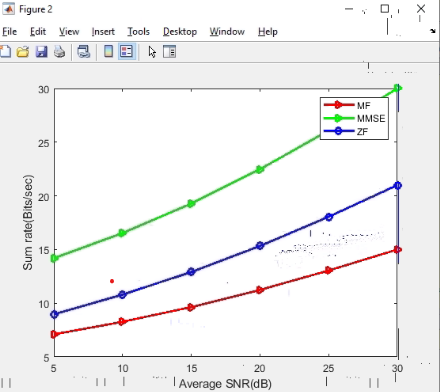
**Applications:**

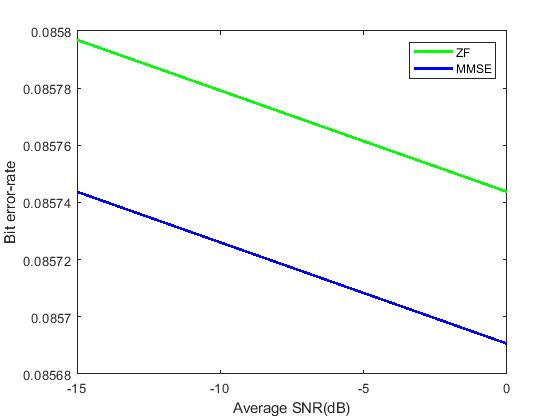
* Multiple-Input Multiple-Output (MIMO) Systems.
* Wireless LANs
* 5G Networks
* Satellite Communications
* Radar Systems

**CHAPTER 6**

**RESULTS**

****

****

****

**CHAPTER 7**

**CONCLUSION**

In this paper, the main performance indexes of MRT, ZF and MMSE Linear Precoding schemes are analyzed under the model of single cell MIMO. At the same time, the rate and bit error rate of MRT, ZF and MMSE precoding are compared by simulation, and MMSE is obtained The precoding technology is obviously better than the other two precoding technologies, especially in the environment of low SNR, which verifies the conclusion of theoretical derivation

**REFERENCES**

[1] Rusek F, Persson D, Lau B K, et al. Scaling up MIMO: Opportunities and challenges with very large arrays[J]. Signal Processing Magazine, IEEE, 2013, 30(1): 40-60.

[2] Zarei S, Gerstacker W, Schober R. A low-complexity linear precoding and power allocation scheme for downlink massive MIMO systems[C]//Signals, Systems and Computers, 2013 Asilomar Conference on.IEEE, 2013: 285-290.

[3] Ngo H Q, Larsson E G, Marzetta T L. Energy and spectral efficiency of very large multiuser MIMO systems[J]. Communications, IEEE Transactions on, 2013, 61(4): 1436-1449.

[4] Zarei S, Gerstacker W, Muller R R, et al. Low-complexity linear precoding for downlink large-scale MIMO systems[C]//Personal Indoor and Mobile Radio Communications (PIMRC), 2013 IEEE 24th International Symposium on. IEEE, 2013: 1119-1124.

[5] Wei S, Goeckel D, Janaswamy R. On the asymptotic capacity of MIMO systems with antenna arrays of fixed length[J]. Wireless Communications, IEEE Transactions on, 2005, 4(4): 1608-1621.

[6] Rusek F, Persson D, Lau B K, et al. Scaling up MIMO: Opportunities and challenges with very large arrays[J]. Signal Processing Magazine, IEEE, 2013, 30(1): 40-60.

**BIBLIOGRAPHY**

**Introduction To Matlab**

What Is MATLAB?

The name MATLAB stands for Matrix Laboratory. The software is built up around vectors and matrices. This makes the software particularly useful for linear algebra but MATLAB is also a great tool for solving algebraic and differential equations and for numerical integration. MATLAB has powerful graphic tools and can produce nice pictures in both 2D and 3D. It is also a programming language, and is one of the easiest programming languages for writing mathematical programs. These factors make MATLAB an excellent tool for teaching and research.

MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects. It integrates computation, visualization, and programming environment. Furthermore, MATLAB is a modern programming language environment: it has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming. MATLAB has many advantages compared to conventional computer languages (e.g., C, FORTRAN) for solving technical problems.

MATLAB abilities a family of add-on software program utility software application software program software utility software-unique solutions called toolboxes. Very essential to maximum customers of MATLAB, toolboxes assist you to studies and observe specialized technology. Toolboxes are entire collections of MATLAB abilities (M-files) that increase the MATLAB surroundings to remedy precise schooling of problems. Areas in which toolboxes are to be had embody signal processing, manipulate systems, neural networks, fuzzy correct judgment, wavelets, simulation, and hundreds of others.

It has powerful built-in routines that enable a very wide variety of computations. It also has easy to use graphics commands that make the visualization of results immediately available. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. The software package has been commercially available since 1984 and is now considered as a standard tool at most universities and industries worldwide.

**Brief History of MATLAB:**

Cleve Moler, the chairman of the computer science department at the University of New Mexico, started developing MATLAB in the late 1970s. The first MATLAB® was not a programming language; it was a simple interactive matrix calculator. There were no programs, no toolboxes, no graphics and no ODEs or FFTs. He designed it to give his student’s access to LINPACK and EISPACK without them having to learn FORTRAN. It soon spread to other universities and found a strong audience within the applied mathematics community. The mathematical basis for the first version of MATLAB was a series of research papers by J. H. Wilkinson and 18 of his colleagues, published between 1965 and 1970 and later collected in Handbook for Automatic Computation, Volume II, Linear Algebra*,* edited by Wilkinson and C. Reinsch. These papers present algorithms, implemented in Algol 60, for solving matrix linear equation and Eigen value problems.

In the 1970s and early 1980s, I was teaching Linear Algebra and Numerical Analysis at the University of New Mexico and wanted my students to have easy access to LINPACK and EISPACK without writing FORTRAN programs. By “easy access,” I meant not going through the remote batch processing and the repeated edit-compile-link-load-execute process that was ordinarily required on the campus central mainframe computer. Jack little, an engineer, was exposed to it during a visit Moler made to Stanford University in 1983. Recognizing its commercial potential, he joined with Moler and Steve Bangert. They rewrote MATLAB in C and founded Math Works in 1984 to continue its development. These rewritten libraries were known as JACKPAC. In 2000, MATLAB was rewritten to use a newer set of libraries for matrix manipulation, LAPACK. MATLAB was first adopted by researchers and practitioners in control engineering, Little's specialty, but quickly spread to many other domains. It is now also used in education, in particular the teaching of linear algebra and numerical analysis, and is popular amongst scientists involved in video processing**.**

## **EISPACK and LINPACK**:

In 1970, a group of researchers at Argonne National Laboratory proposed to the U.S. National Science Foundation (NSF) to “explore the methodology, costs, and resources required to produce, test, and disseminate high-quality mathematical software and to test, certify, disseminate, and support packages of mathematical software in certain problem areas.” The group developed EISPACK (Matrix Eigen system Package) by translating the Algol procedures for Eigen value problems in the handbook into FORTRAN and working extensively on testing and portability. The first version of EISPACK was released in 1971 and the second in 1976.

In 1975, four of us Jack Dongarra, Pete Stewart, Jim Bunch, and myself proposed to the NSF another research project that would investigate methods for the development of mathematical software. A byproduct would be the software itself, dubbed LINPACK, for Linear Equation Package. This project was also centered at Argonne. LINPACK originated in FORTRAN; it did not involve translation from Algol. The package contained 44 subroutines in each of four numeric precisions. In a sense, the LINPACK and EISPACK projects were failures. We had proposed research projects to the NSF to “explore the methodology, costs, and resources required to produce, test, and disseminate high-quality mathematical software.” We never wrote a report or paper addressing those objectives. We only produced software.

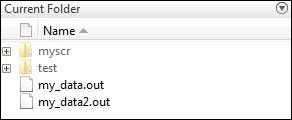
So, I studied Niklaus Wirth’s book Algorithms + Data Structures *=* Programs and learned how to parse programming languages. I wrote the first MATLAB an acronym for Matrix Laboratory in FORTRAN, with matrix as the only data type. The project was a kind of hobby, a new aspect of programming for me to learn and something for my students to use. There was never any formal outside support, and certainly no business plan. This first MATLAB was just an interactive matrix calculator. This snapshot of the start-up screen shows all the reserved words and functions. There are only 71. To add another function, you had to get the source code from me, write a FORTRAN subroutine, add your function name to the parse table, and recompile MATLAB.

**Starting MATLAB:**

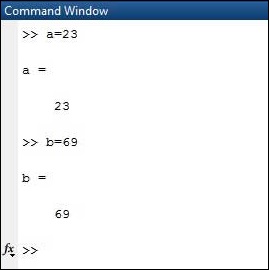
After logging into your account, you can enter MATLAB by double-clicking on the MATLAB shortcut icon (MATLAB 7.0.4) on your Windows desktop. When you start MATLAB, a special window called the MATLAB desktop appears. The desktop is a window that contains other windows. The major tools within or accessible from the desktop are:

* The Command Window
* The Command History
* The Workspace
* The Current Directory
* The Help Browser

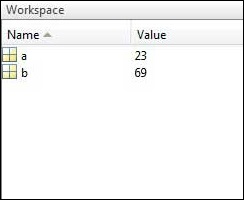
**Current Folder:** This panel allows you to access the project folders and files.



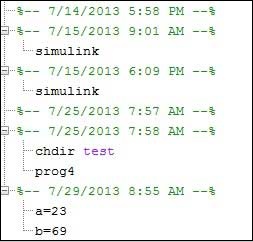
**Command Window:** This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).



**Workspace:**  The workspace shows all the variables created and/or imported from files.



**Command History:** This panel shows or return commands that are entered at the command line.



**Help Browser:**

The critical way to get assist online is to use the MATLAB help browser, opened as a separate window every through clicking at the question mark photograph (?) on the computing tool toolbar, or through manner of typing assist browser on the spark off in the command window. The assist Browser is an internet browser blanketed into the MATLAB computing tool that shows a Hypertext Markup Language (HTML) files. The Help Browser consists of panes, the help navigator pane, used to find out information, and the show pane, used to view the information. Self-explanatory tabs apart from navigator pane are used to performs are searching out.

**MATLAB language:**

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs.

**MATLAB working environment:**

This is the set of tools and facilities that you work with as the MATLAB user or programmer. It includes facilities for managing the variables in your workspace and importing and exporting data. It also includes tools for developing, managing, debugging, and profiling M-files, MATLAB's applications.

**MATLAB mathematical function library:**

This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.

**MATLAB Application Program Interface (API):**

This is a library that allows you to write C and FORTRAN programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

**MATLAB DESKTOP:**

MATLAB Desktop is the precept MATLAB utility window. The computing tool includes five sub home windows, the command window, the workspace browser, the modern-day-day list window, the command records window, and one or greater decide domestic windows, which is probably confirmed high-quality on the identical time due to the truth the client suggests a photo. The command window is in which the character types MATLAB instructions and expressions at the spark off (>>) and in which the output of these commands is displayed. MATLAB defines the workspace because the set of variables that the client creates in a bit consultation. The workspace browser suggests those variables and some facts about them. Double clicking on a variable within the workspace browser launches the Array Editor, which may be used to gain statistics and profits instances edit exceptional homes of the variable.

The modern-day-day-day Directory tab above the workspace tab suggests the contents of the cutting-edge list, whose path is shown inside the modern-day list window. For example, in the home windows on foot machine the path is probably as follows: C: MATLAB Work, indicating that listing “artwork” is a subdirectory of the number one list “MATLAB”; WHICH IS INSTALLED IN DRIVE C. Clicking on the arrow within the modern list window suggests a listing of these days used paths. Clicking at the button to the right of the window permits the individual to trade the present day listing. MATLAB uses a seeking out path to find out M-documents and one-of-a-type MATLAB associated documents, which can be put together in directories within the computer document tool. Any report run in MATLAB need to be dwelling in the modern-day-day listing or in a list that is on is looking for course. By default, the documents supplied with MATLAB and math works toolboxes are included inside the searching out direction. The first-rate manner to look which directories are on the searching out route. The satisfactory manner to appearance which directories are speedy the quest route, or to characteristic or regulate a searching for course, is to pick out outset path from the File menu the computing device, and then use the set course talk discipline. It is proper exercise to feature any generally used directories to the hunt route to avoid again and again having the exchange the cutting-edge-day listing.

The Command History Window contains a file of the instructions a person has entered in the command window, together with every contemporary-day and former MATLAB periods. Previously entered MATLAB instructions can be determined on and re-completed from the command statistics window thru proper clicking on a command or series of commands. This movement launches a menu from which to select numerous options similarly to executing the commands. This is useful to select out abilities options in addition to executing the instructions. This is a beneficial feature at the equal time as experimenting with numerous commands in a piece session.

**Using the MATLAB Editor to create M-Files:**

The MATLAB editorial manager is a literary substance proofreader particular for growing M-facts and a graphical MATLAB debugger. The supervisor can seem in a window through command facts technique for itself, or it is probably a right-clicking inside the PC. M-information this gadget signified through the use of the expansion .M, as in pixel up.M. The MATLAB editorial supervisor window has a few draws down menus for obligations collectively with sparing, seeing, and troubleshooting facts. Since it plays more than one easy test and furthermore affects utilization of shade to separate among exclusive variables of code, this article editorial supervisor is often supported due to reality the system of a need for composing and altering M-talents. To open the manager, type at enact opens the M-document filename. M in a supervisor window, sorted out for enhancing. As stated earlier than, the file should be inside the cutting-edge posting, or in a posting in the seeking out direction.

## **Features of MATLAB:**

Following are the basic features of MATLAB.

* It is a high-level language for numerical computation, visualization and application development.
* It also provides an interactive environment for iterative exploration, design and problem solving.
* It provides vast library of mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration and solving ordinary differential equations.
* It provides built-in graphics for visualizing data and tools for creating custom plots.
* MATLAB's programming interface gives development tools for improving code quality maintainability and maximizing performance.
* It provides tools for building applications with custom graphical interfaces.
* It provides functions for integrating MATLAB based algorithms with external applications and languages such as C, Java, .NET and Microsoft Excel.

## **Uses of MATLAB:**

MATLAB is widely used as a computational tool in science and engineering encompassing the fields of physics, chemistry, math and all engineering streams. It is used in a range of applications including

* Signal Processing and Communications
* Video and Video Processing
* Control Systems
* Test and Measurement
* Computational Finance
* Computational Biology

**Applications of MATLAB:**

MATLAB can be used as a tool for simulating various electrical networks but the recent developments in MATLAB make it a very competitive tool for Artificial Intelligence, Robotics, Video processing, Wireless communication, Machine learning, Data analytics and whatnot. Though it’s mostly used by circuit branches and mechanical in the engineering domain to solve a basic set of problems its application is vast. It is a tool that enables computation, programming and graphically visualizing the results. The basic data element of MATLAB as the name suggests is the Matrix or an array. MATLAB toolboxes are professionally built and enable you to turn your imaginations into reality. MATLAB programming is quite similar to C programming and just requires a little brush up of your basic programming skills to start working with.

Below are a few applications of MATLAB –

* **Statistics and machine learning (ML)**

This toolbox in MATLAB can be very handy for the programmers. Statistical methods such as descriptive or inferential can be easily implemented. So is the case with machine learning. Various models can be employed to solve modern-day problems. The algorithms used can also be used for big data applications.

* **Curve fitting**

The curve fitting toolbox helps to analyze the pattern of occurrence of data. After a particular trend which can be a curve or surface is obtained, its future trends can be predicted. Further plotting, calculating integrals, derivatives, interpolation, etc. can be done.

* **Control systems**

Systems nature can be obtained. Factors such as closed-loop, open-loop, its controllability and observability, Bode plot, NY Quist plot, etc. can be obtained. Various controlling techniques such as PD, PI and PID can be visualized. Analysis can be done in the time domain or frequency domain.

* **Signal Processing**

Signals and systems and digital signal processing are taught in various engineering streams. But MATLAB provides the opportunity for proper visualization of this. Various transforms such as Laplace, Z, etc. can be done on any given signal. Theorems can be validated. Analysis can be done in the time domain or frequency domain. There are multiple built-in functions that can be used.

* **Mapping**  
  Mapping has multiple applications in various domains. For example, in Big Data, the Map Reduce tool is quite important which has multiple applications in the real world. Theft analysis or financial fraud detection, regression models, contingency analysis, predicting techniques in social media, data monitoring, etc. can be done by data mapping.
* **Deep learning**

It’s a subclass of machine learning which can be used for speech recognition, financial fraud detection, and medical video analysis. Tools such as time-series, Artificial neural network (ANN), Fuzzy logic or combination of such tools can be employed.

* **Financial analysis**

An entrepreneur before starting any endeavor needs to do a proper survey and the financial analysis in order to plan the course of action. The tools needed for this are all available in MATLAB. Elements such as profitability, solvency, liquidity, and stability can be identified. Business valuation, capital budgeting, cost of capital, etc. can be evaluated.

* **Video processing**

The most common application that we observe almost every day are bar code scanners, selfie (face beauty, blurring the background, face detection), video enhancement, etc. The digital video processing also plays quite an important role in transmitting data from far off satellites and receiving and decoding it in the same way. Algorithms to support all such applications are available.

* **Text analysis**

Based on the text, sentiment analysis can be done. Google gives millions of search results for any text entered within a few milliseconds. All this is possible because of text analysis. Handwriting comparison in forensics can be done. No limit to the application and just one software which can do this all.

* **Electric vehicles designing**

Used for modeling electric vehicles and analyze their performance with a change in system inputs. Speed torque comparison, designing and simulating of a vehicle, whatnot.

* **Aerospace**

This toolbox in MATLAB is used for analyzing the navigation and to visualize flight simulator.

* **Audio toolbox**

Provides tools for audio processing, speech analysis, and acoustic measurement. It also provides algorithms for audio and speech feature extraction and audio signal transformation.

**COMMUNICATION:**

Communications System Toolbox™ offers algorithms and gear for the layout, simulation, and analysis of communications systems. These capabilities are furnished as MATLAB ® features, MATLAB System gadgets™, and Simulink ® blocks. The machine toolbox includes algorithms for source coding, channel coding, interleaving, modulation, equalization, synchronization, and channel modeling. Tools are supplied for bit blunders charge evaluation, producing eye and constellation diagrams, and visualizing channel characteristics. The machine toolbox additionally provides adaptive algorithms that allow you to version dynamic communications structures that use OFDM, OFDMA, and MIMO techniques. Algorithms support fixed-point facts arithmetic and C or HDL code era.

**Key Features**

▪ Algorithms for designing the physical layer of communications systems, which includes supply coding, channel coding, interleaving, modulation, channel fashions, MIMO, equalization, and synchronization

▪ GPU-enabled System objects for computationally intensive algorithms together with Turbo, LDPC, and Viterbi decoders

▪ Interactive visualization equipment, consisting of eye diagrams, constellations, and channel scattering capabilities

▪ Graphical tool for evaluating the simulated bit mistakes rate of a machine with analytical outcomes

▪ Channel models, consisting of AWGN, Multipath Rayleigh Fading, Rician Fading, MIMO Multipath Fading, and

LTE MIMO Multipath Fading

▪ Basic RF impairments, along with nonlinearity, section noise, thermal noise, and section and frequency offsets

▪ Algorithms available as MATLAB features, MATLAB System objects, and Simulink blocks

▪ Support for fixed-point modeling and C and HDL code technology

**System Design, Characterization, and Visualization:**

The layout and simulation of a communications gadget requires analyzing its reaction to the noise and interference inherent in real-world environments, reading its behavior the usage of graphical and quantitative manner, and determining whether the resulting overall performance meets requirements of acceptability. Communications System Toolbox implements a selection of obligations for communications machine layout and simulation. Many of the functions, System objects™, and blocks inside the device toolbox perform computations associated with a specific thing of a communications gadget, consisting of a demodulator or equalizer. Other talents are designed for visualization or evaluation.

**System Characterization**

The system toolbox offers several standard methods for quantitatively characterizing system performance:

▪ Bit error rate (BER) computations

▪ Adjacent channel power ratio (ACPR) measurements

▪ Error vector magnitude (EVM) measurements

▪ Modulation error ratio (MER) measurements

Because BER computations are fundamental to the characterization of any communications system, the system toolbox provides the following tools and capabilities for configuring BER test scenarios and accelerating BER simulations:

**BER tool**— A graphical user interface that enables you to analyze BER performance of communications systems. You can analyze performance via a simulation-based, semi analytic, or theoretical approach.

**Error Rate Test Console** — A MATLAB object that runs simulations for communications systems to measure error rate performance. It supports user-specified test points and generation of parametric performance plots and surfaces. Accelerated performance can be realized when running on a multi core computing platform.

**Multi core and GPU acceleration** — A capability provided by Parallel Computing Toolbox™ that enables you to accelerate simulation performance using multi core and GPU hardware within your computer.

**Distributed computing and cloud computing support** — Capabilities provided by Parallel Computing Toolbox and MATLAB Distributed Computing Server™ that enable you to leverage the computing power of your server farms and the Amazon EC2 Web service. Performance Visualization. The system toolbox provides the following capabilities for visualizing system performance:

**Channel visualization tool** — For visualizing the characteristics of a fading channel

**Eye diagrams and signal constellation scatter plots** — for a qualitative, visual understanding of system behavior that enables you to make initial design decisions

**Signal trajectory plots** — for a continuous picture of the signal’s trajectory between decision points

**BER plots** — for visualizing quantitative BER performance of a design candidate, parameterized by metrics such as SNR and fixed-point word size

**Analog and Digital Modulation**

Analog and digital modulation strategies encode the facts circulation into a sign this is appropriate for transmission. Communications System Toolbox presents some of modulation and corresponding demodulation abilities. These talents are available as MATLAB features and gadgets, MATLAB System Modulation sorts provided by the toolbox are:

**Source and Channel Coding**

Communications System Toolbox affords source and channel coding talents that can help you develop and compare communications architectures fast, enabling you to discover what-if eventualities and avoid the need to create coding competencies from scratch.

**Source Coding**

Source coding, also referred to as quantization or signal formatting, is a manner of processing facts a good way to lessen redundancy or prepare it for later processing. The system toolbox offers a diffusion of styles of algorithms for imposing source coding and interpreting, inclusive of:

▪ Quantizing

▪ Companding (*µ*-law and A-law)

▪ Differential pulse code modulation (DPCM)

▪ Huffman coding

▪ Arithmetic coding

**Channel Coding**

▪ orthogonal area-time block code (OSTBC) (encoder and decoder for MIMO channels)

▪ Turbo encoder and decoder examples

The gadget toolbox offers application functions for developing your personal channel coding. You can create generator polynomials and coefficients and syndrome deciphering tables, in addition to product parity-take a look at and generator matrices.

The system toolbox additionally presents block and convolutional interleaving and deinters leaving functions to reduce facts errors as a result of burst mistakes in a conversation machine:

**Block,** including General block interleaver, algebraic interleaver, helical scan interleaver, matrix interleaver, and random interleaver.

**Convolutional,** including General multiplexed interleaver, convolutional interleaver, and helical interleaver

**Channel Modeling and RF Impairments**

Channel Modeling

Communications System Toolbox provides algorithms and tools for modeling noise, fading, interference, and different distortions which might be commonly found in communications channels. The system toolbox supports the subsequent styles of channels:

▪ Additive white Gaussian noise (AWGN)

▪ Multiple-enter multiple-output (MIMO) fading

▪ Single-enter single-output (SISO), Rayleigh, and Rician fading

▪ Binary symmetric

A MATLAB channel object provides a concise, configurable implementation of channel models, enabling you to

specify parameters such as:

▪ Path delays

▪ Average path gains

▪ Maximum Doppler shifts

▪ K-Factor for Rician fading channels

▪ Doppler spectrum parameters

For MIMO systems, the MATLAB MIMO channel object expands these parameters to also include:

▪ Number of transmit antennas (up to 8)

▪ Number of receive antennas (up to 8)

▪ Transmit correlation matrix

▪ Receive correlation matrix

To combat the effects noise and channel corruption, the system toolbox provides block and convolutional coding and decoding techniques to implement error detection and correction. For simple error detection with no inherent correction, a cyclic redundancy check capability is also available. Channel coding capabilities provided by the system toolbox include:

▪ BCH encoder and decoder

▪ Reed-Solomon encoder and decoder

▪ LDPC encoder and decoder

▪ Convolutional encoder and Viterbi decoder

****

**RF Impairments**

To model the effects of a non-ideal RF front end, you can introduce the following impairments into your communications system, enabling you to explore and characterize performance with real-world effects:

▪ Memory less nonlinearity

▪ Phase and frequency offset

▪ Phase noise

▪ Thermal noise

You can include more complex RF impairments and RF circuit models in your design using SimRF™.

****

**Equalization and Synchronization**

Communications System Toolbox lets you discover equalization and synchronization strategies. These techniques are usually adaptive in nature and tough to design and symbolize. The machine toolbox affords algorithms and tools that will let you swiftly select the proper approach on your communications machine. Equalization To compare one-of-a-kind techniques to equalization, the device toolbox offers you with adaptive algorithms which include:

▪ LMS

▪ Normalized LMS

▪ Variable step LMS

▪ Signed LMS

▪ MLSE (Viterbi)

▪ RLS

▪ CMA

These adaptive equalizers are available as nonlinear decision feedback equalizer (DFE) implementations and as

Linear (symbol or fractionally spaced) equalizer implementations.

**Synchronization**

The device toolbox provides algorithms for each service segment synchronization and timing phase synchronization. For timing section synchronization, the machine toolbox presents a MATLAB Timing Phase Synchronizer object that offers the following implementation techniques:

▪ Early-late gate timing method

▪ Gardner’s method

▪ Fourth-order nonlinearity method

**Stream Processing in MATLAB and Simulink**

Most verbal exchange structures cope with streaming and frame-primarily based statistics using a aggregate of temporal processing and simultaneous multi frequency and multichannel processing. This form of streaming multidimensional processing can be visible in superior communication architectures consisting of OFDM and MIMO. Communications System Toolbox enables the simulation of advanced communications structures via helping move processing and frame-based simulation in MATLAB and Simulink. In MATLAB, circulate processing is enabled by way of System items™, which use MATLAB objects to symbolize time-based and facts-driven algorithms, sources, and sinks. System objects implicitly manipulate many information of flow processing, including information indexing, buffering, and management of set of rules state. You can mix System gadgets with fashionable MATLAB functions and operators. Most System items have a corresponding Simulink block with the identical abilities. Simulink handles circulation processing implicitly with the aid of coping with the float of information thru the blocks that make up a Simulink model. Simulink is an interactive graphical environment for modeling and simulating dynamic systems that uses hierarchical diagrams to symbolize a machine version. It includes a library of widespread-reason, predefined blocks to represent algorithms, resources, sinks, and device hierarchy.

**Implementing a Communications System**

Fixed-Point Modeling Many communications systems use hardware that requires a fixed-point representation of your design.

Communications System Toolbox supports fixed-point modeling in all relevant blocks and System objects™ with tools that help you configure fixed-point attributes.

Fixed-point support in the system toolbox includes:

▪ Word sizes from 1 to 128 bits

▪ Arbitrary binary-point placement

▪ Overflow handling methods (wrap or saturation)

▪ Rounding methods: ceiling, convergent, floor, nearest, round, simplest, and zero

Fixed-Point Tool in Simulink Fixed Point™ facilitates the conversion of floating-point data types to fixed point. For configuration of fixed-point properties, the tool tracks overflows and maxima and minima.

**Code Generation**

Once you've got advanced your set of rules or communications device, you can robotically generate C code from it for verification, rapid prototyping, and implementation. Most System gadgets, functions, and blocks in Communications System Toolbox can generate ANSI/ISO C code the use of MATLAB Coder™, Simulink Coder™, or Embedded Coder™. A subset of System gadgets and Simulink blocks also can generate HDL code. To leverage present highbrow belongings, you can choose optimizations for specific processor architectures and integrate legacy C code with the generated code.

You can also generate C code for both floating-point and fixed-point data types.

DSP Proto typing DSPs are used in communication system implementation for verification, rapid prototyping, or final hardware implementation. Using the processor-in-the-loop (PIL) simulation capability found in Embedded Coder, you can verify generated source code and compiled code by running your algorithm’s implementation code on a target processor. FPGA Prototyping

FPGAs are used in communication systems for implementing high-speed signal processing algorithms. Using the FPGA-in-the-loop (FIL) capability found in HDL Verifier™, you can test RTL code in real hardware for any existing HDL code, either manually written or automatically generated HDL code.