**Energy Efficient Resource Allocation in Wireless Energy Harvesting Sensor Networks**

**ABSTRACT**

One of the most crucial difficulties in the widespread adoption of Wireless Sensor Networks is extending the sensor life (WSNs). In recent years, Energy Harvesting (EH) sensors have been suggested as a solution to the aforementioned issue. These sensors may obtain the energy they need from the environment in a variety of ways, extending their useful lives. We consider a Wireless Energy Harvesting Sensor Network (WEHSN) based on TDMA in which the time slot consists of two time intervals, the first of which is used to transmit data from the sensors and the second of which is used to absorb energy. Extending the sensor life is one of the most significant obstacles to the broad adoption of Wireless Sensor Networks (WSNs). Energy Harvesting (EH) sensors have been proposed as a remedy for the aforementioned problem in recent years. These sensors could extend their useful lifetimes by obtaining the energy they require from the environment in a number of different ways. We take into account a Wireless Energy Harvesting Sensor Network (WEHSN) based on TDMA where the time slot consists of two time intervals, the first of which is utilized for sensor data transmission and the second of which is utilized for energy absorption.

**Keywords:** Energy Harvesting, Resource Allocation, Energy Efficiency, Wireless Sensor Networks.

**CHAPTER 1**

**INTRODUCTION**

In recent years, there has been a lot of interest in the rapidly developing fields of Wireless Sensor Networks (WSN) and Internet of Things (IoT) applications such the smart home, smart factory, and others. The efficient use of resources, such as electricity and energy harvesting technology, would increase the lifespan of the sensors and significantly improve system performance.

Sultana et al. in, has considered a cognitive D2D communication system to improve the resource allocation efficiency as well as spectral efficiency. Nobar et al has studied a cognitive Wireless Powered Communication Networks (WPCN) with green power beacon, -in which the secondary network is wirelessly powered by an energy harvesting power beacon- to provide an efficient energy and spectrum performance, simultaneously. The authors of the paper have investigated two spectrum access schemes for the proposed model, i.e., a random spectrum access and a spectrum-sensing based spectrum access, which results in deriving the closed form expressions for service rate of the Secondary User (SU) and the Primary User (PU). Also, a modified model has been discussed by Nobar et al. in to meet the required criteria of resource-limited cognitive WSNs and to maximize the performance of the secondary network under some QoS constraints. They have applied an infinite battery status for PU but limited battery life time in wireless sensor nodes. Likewise, Ding et al. has studied an iterative joint resource management and time allocation in to maximize the energy efficiency whereas Yang et al. in has tried to maximize the energy efficiency via minimizing the total consumed energy in a cluster-based IoT network with energy harvesting property.

Another approach by Pei et al. in, proposes a joint resource block and transmission power control scheme for the energy harvesting D2D communications which applies an underlaying Non-Orthogonal Multiple Access (NOMA) scheme to a cellular network under signal-to-interference-and noise ratio constraint of the Cellular Users (CU). Furthermore, maximizing the netwok throughput in TDMA and NOMA for uplink wireless powered IoT networks, has studied in by Wu et al, where the spectral and energy efficiency are limited to the circuit energy consumption.

In this letter, we consider an energy efficient resource allocation in a TDMA based WEHSN. Unlike, which only has optimized the network throughput, our target is to maximize the energy efficiency by decreasing the total energy consumption in the sensors. We derive the closed form expressions for the optimization problem defined for energy efficiency and then, we apply Dinkelbach algorithm to convert the optimization problem to parametric form and find the optimal resource allocation in the network. Using the mentioned algorithm, leads into much decrease in the energy consumption in the network, consequently, yielding better performance.

Wireless Sensor Network (WSN) is an infrastructure-less wireless network that is deployed in a large number of wireless sensors in an ad-hoc manner that is used to monitor the system, physical or environmental conditions. Sensor nodes are used in WSN with the onboard processor that manages and monitors the environment in a particular area. They are connected to the Base Station which acts as a processing unit in the WSN System. Base Station in a WSN System is connected through the Internet to share data.

**Sensors:**

Sensors in WSN are used to capture the environmental variables and which is used for data acquisition. Sensor signals are converted into electrical signals.

**Radio Nodes:**

It is used to receive the data produced by the Sensors and sends it to the WLAN access point. It consists of a microcontroller, transceiver, external memory, and power source.

**WLAN Access Point:**

It receives the data which is sent by the Radio nodes wirelessly, generally through the internet.

**Evaluation Software:**

The data received by the WLAN Access Point is processed by a software called as Evaluation Software for presenting the report to the users for further processing of the data which can be used for processing, analysis, storage, and mining of the data.

**CHAPTER 2**

**LITERATURE SURVEY**

**[1] J. Huang, C. Xing, and C. Wang, “Simultaneous wireless information and power transfer: Technologies, applications, and research challenges,” IEEE Communications Magazine, vol. 55, no. 11, pp. 26–32, Nov 2017:**

Energy efficiency will play a crucial role in future communication systems and has become a main design target for all 5G radio access networks. The high operational costs and impossibility of replacing or recharging wireless device batteries in multiple scenarios, such as wireless medical sensors inside the human body, call for a new technology by which wireless devices can harvest energy from the environment via capturing ambient RF signals. SWIPT has emerged as a powerful means to address this issue. In this article, we survey the current architectures and enabling technologies for SWIPT and identify technical challenges to implement SWIPT. Following an overview of enabling technologies for SWIPT and SWIPT-assisted wireless systems, we showcase a novel SWIPT-supported power allocation mechanism for D2D communications to illustrate the importance of the application of SWIPT. As an ending note, we point out some future research directions to encourage and motivate more research efforts on SWIPT.

**Summary:** Studied about the Simultaneous wireless information and power transfer: Technologies, applications, and research challenges.

**[2] K. Kang, R. Ye, Z. Pan, J. Liu, and S. Shimamoto, “Full-duplex wireless powered iot networks,” IEEE Access, vol. 6, pp. 53 546–53 556, 2018:**

This paper studies the emerging wireless power transfer for the Internet-of-Things (IoT) network, where one hybrid access point (H-AP) with constant power supply communicates with a set of IoT devices. This H-AP is assumed to work in a full-duplex mode, which transmits/receives signals to/from these IoT devices simultaneously during the whole frame. The IoT devices are capable of harvesting energy from the received signals broadcast by the H-AP. And the harvested energy is used to support the uplink transmission. Since time-division multiple access is used in uplink transmission, one IoT device keeps harvesting energy till its own uplink time slot. The objective of this paper is to maximize the total surplus energy, which is defined as the gap between available energy and consumed energy for uplink transmissions, by exploiting the optimal time allocation scheme for each device. A distributed non-cooperative and a bargaining cooperative game-based algorithms are proposed to solve this problem. In addition, the well-known KKT condition approach is adopted as a comparison. The numerical results show that the bargaining cooperative algorithm outperforms the distributed non-cooperative algorithm (DNCA) and KKT algorithm (KKTA) in terms of total surplus energy and fairness index. The performance of DNCA is better than that of KKTA in terms of total surplus energy while KKTA is fairer than DNCA.

**Summary:** Studied about the Full-duplex wireless powered iot networks.

**[3] Z. Chu, F. Zhou, Z. Zhu, R. Q. Hu, and P. Xiao, “Wireless powered sensor networks for internet of things: Maximum throughput and optimal power allocation,” IEEE Internet of Things Journal, vol. 5, no. 1, pp. 310–321, Feb 2018:**

This paper investigates a wireless powered sensor network, where multiple sensor nodes are deployed to monitor a certain external environment. A multiantenna power station (PS) provides the power to these sensor nodes during wireless energy transfer phase, and consequently the sensor nodes employ the harvested energy to transmit their own monitoring information to a fusion center during wireless information transfer (WIT) phase. The goal is to maximize the system sum throughput of the sensor network, where two different scenarios are considered, i.e., PS and the sensor nodes belong to the same or different service operator(s). For the first scenario, we propose a global optimal solution to jointly design the energy beamforming and time allocation. We further develop a closed-form solution for the proposed sum throughput maximization. For the second scenario in which the PS and the sensor nodes belong to different service operators, energy incentives are required for the PS to assist the sensor network. Specifically, the sensor network needs to pay in order to purchase the energy services released from the PS to support WIT. In this case, this paper exploits this hierarchical energy interaction, which is known as energy trading. We propose a quadratic energy trading-based Stackelberg game, linear energy trading-based Stackelberg game, and social welfare scheme, in which we derive the Stackelberg equilibrium for the formulated games, and the optimal solution for the social welfare scheme. Finally, numerical results are provided to validate the performance of our proposed schemes.

**Summary:** Studied about Wireless powered sensor networks for internet of things: Maximum throughput and optimal power allocation.

**[4]** **A. Sultana, L. Zhao, and X. Fernando, “Efficient resource allocation in device-to-device communication using cognitive radio technology,” IEEE Transactions on Vehicular Technology, vol. 66, no. 11, pp. 10 024–10 034, Nov 2017:**

Device-to-device (D2D) communication is developed as a new paradigm to enhance network performance according to LTE and WiMAX advanced standards. The D2D communication may have dedicated spectrum (overlay) or shared spectrum (underlay). However, the allocated dedicated spectrum may not be effectively used in the overlay mode, while interference between the D2D users and cellular users cause impairments in the underlay mode. Can the resource allocation of a D2D system be optimized using the cognitive approach where the D2D users opportunistically access the underutilized radio spectrum? That is the focus of this paper. In this paper, the transmission rate of the D2D users is optimized while simultaneously satisfying five sets of constraints related to power, interference, and data rate, modeling D2D users as cognitive secondary users. Furthermore, a two-stage approach is considered to allocate the radio resources efficiently. A new adaptive subcarrier allocation scheme is designed first, and then, a novel power allocation scheme is developed utilizing geometric water-filling approach that provides optimal solution with low computation complexity for this nonlinear problem. Numerical results show that the proposed approach achieved significant performance enhancement than the existing schemes. **Summary:** Studied about Efficient resource allocation in device-to-device communication using cognitive radio technology.

**[5] S. K. Nobar, K. A. Mehr, and J. M. Niya, “Rf-powered green cognitive radio networks: Architecture and performance analysis,” IEEE Communications Letters, vol. 20, no. 2, pp. 296–299, Feb 2016:**

In this letter, we consider an RF-powered green cognitive radio network (RF-GCRN), where a central node, called a power beacon (PB), harvests green energy from ambient sources and wirelessly delivers random harvested energy to cognitive users. Random in-band energy transmission by PB is the only energy source of cognitive users. Performance of this network, with a single pair of secondary users, is analyzed under two spectrum access schemes, i.e., random access and spectrum sensing-based access schemes. Results show feasibility of the RF-GCRN model, if the energy transmission rate is below a certain threshold. This threshold is determined according to maximum tolerable delay of primary user and parameters of spectrum access scheme. Finding a closed form expression for this threshold results in a quite complicated formula, which cannot be helpful in determining feasibility region, due to excessive complexity. Instead, we numerically calculated feasibility regions of both access schemes to facilitate parameter selection process.

**SUMMARY:** Studied about Rf-powered green cognitive radio networks: Architecture and performance analysis.

**CHAPTER 3**

**EXISTING METHOD**

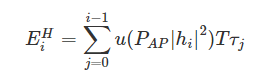
In the Internet of Things (IoT) network, where a single hybrid access point (H-AP) with a consistent power source communicates with a variety of IoT devices, wireless power transfer is a new technology that is being studied in this research. It is expected that this H-AP operates in full-duplex mode, which sends and receives signals to and from these Internet of Things devices simultaneously during the entire frame. The H-received AP's signals can be used to generate energy by the IoT devices. Additionally, the uplink transmission is supported by the energy that was captured. One Internet of Things device continues to collect energy until its own uplink time slot because uplink transmission uses time-divisio multiple access by utilizing the most effective time allocation strategy for each device, or uplink transmissions. To address this issue, distributed non-cooperative and bargaining cooperative game-based algorithms are developed. Additionally, the popular KKT condition technique is used as a benchmark. The numerical findings demonstrate that in terms of total surplus energy and the fairness index, the bargaining cooperative algorithm outperforms the distributed non-cooperative algorithm (DNCA) and the KKT algorithm (KKTA). In terms of overall surplus energy, DNCA performs better than KKTA, although KKTA is more equitable than DNCA.

In this paper considers a wireless powered IoT network consisting of one H-AP communicating with multiple IoT devices denoted by IoT Di, i=1,⋯,N . It is assumed that the H-AP and all devices operate in full-duplex and half-duplex, respectively. Therefore, the H-AP is able to broadcast energy signal and receive information simultaneously. In the downlink direction, each device harvests energy from the received signal broadcast by the H-AP with practical non-linear energy harvesting model introduced in. Since TDMA is adopted in uplink transmission, each individual IoT device keeps harvesting till its allocated time slot defined by Tτi,i=1,⋯,N , which is illustrated in Fig. 2. T is defined as the frame length and τi is a ratio varying from 0 and 1. Note that in order to ensure the initial operation of the IoT devices, an original available energy is defined as EOi and an initial energy harvesting time slot is defined as Tτ0 . The downlink, uplink, and interference channels are assumed to be quasi-static flat-fading with hi , gi and gji , respectively.

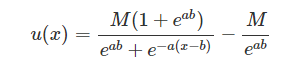
he H-AP broadcasts energy signal to all IoT devices during the whole frame and the uplink transmission of one device interferes with other devices which are harvesting energy. Hence, the signal received by the IoTDi is expressed as



where Ij(τj)=Pj−−√gjixj,j=0⋯i−1 . Note that I0(τ0)=0 . PAP is the transmission power of the H-AP. Pj is the uplink transmission power of IoT Dj and x and xj are the transmitted signals with E[|x|2]=E[|xi|2]=1 . ni∼CN(0,σ2i) is the additive Gaussian noise introduced by the received antenna at the IoT Di . Note that although the received signal of (1) consists of the interference from the uplink transmission device, the harvested energy from interference is neglectable because the interference power is pretty small compared with that from the downlink transmission and it is too small to trigger the energy harvesting functionality since there exists a receiver sensitivity threshold for the non-linear harvesting model. Therefore, the harvested energy of IoT Di during the whole frame is written as



Where, u(⋅) is defined as follows



Where, a, b and M are positive parameters which capture the joint effects of different non-linear phenomena caused by hardware constraints. Therefore, the total available energy of IoT Di can be expressed as



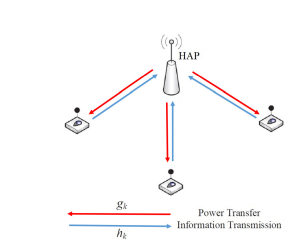
**Disadvantages:**

* Computationally complex.
* Not reaching desired level of energy constraint.

**CHAPTER 4**

**PROPOSED METHOD**

We consider a WEHSN, which consists of one Hybrid Access Point (HAP) plugged to an infinite power supply and M sensors capable of energy harvesting. We use “harvest-and-then-transmit” protocol proposed in. At first, sensors harvest energy in downlink (DL) from a Wireless Energy Transferring (WET), then, they transmit information in uplink (UL) towards a Wireless Information Transmission (WIT). The total time interval for energy harvesting and information transmission is denoted by Tmax. We consider a TDMA-based WEHSN in which whole sensors harvest energy.



**Fig. 1. System Model of Wireless Energy Harvesting Sensor Networks**

During DL WET and transmit information in duration of UL WIT. The second interval is divided into M slots belonging to each sensor. The perfect Channel State Information (CSI) is assumed to be available in each sensor for resource allocation. The DL channel gain between the HAP and sensor i, and the UL channel gain between sensor i and HAP are denoted by gi and hi, respectively. During the downlink period, HAP broadcasts the energy signal with a constant power P0, during τ0 omnidirectionally to all sensor. Thus, the amount of harvested energy at sensor i can be expressed as



Where, ηi ∈ (0, 1] is the constant energy conversion coefficient of sensor i and Pcri is consumed power in the circuit in energy harvesting period. We assume the amount of harvested energy in each sensor is positive (fi > 0). if fi < 0 the corresponding sensor is prohibited to participate in transmission due to lack of enough energy. During the uplink period, due to TDMA-based WEHSN, each sensor transmits information in allocated time slot τi.

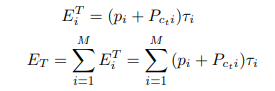
Therefore, the consumed energy in each sensor during the information transmission will be equal to (pi+Pcti)τi, where pi denotes the power allocated for sensor i in WIT and Pcti is the circuit power consumption in information transmission period. Then, the achievable throughput (normalized by bandwidth) for sensor i can be expressed as

****

Where, σ2 is the additive white Gaussian noise power at the HAP. Therefore, the system throughput would be obtained as

****

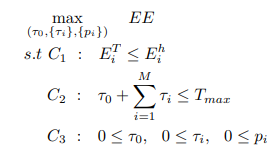
Then, the consumed energy of each sensor and total energy consumption in the network will be as follows

****

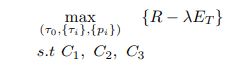
Also, the Energy Efficiency (EE) is defined as



We can formulate the EE maximization as

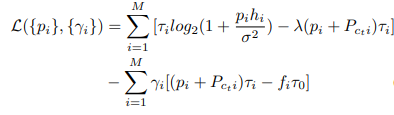


Where, C1 constraint assures that the consumed energy in WIT duration is less than the harvested energy in each sensors. No information will be transmitted, if the first constraint, C1, does not hold for sensor i. The problem defined in is known as fractional programming (FP). Thus, we could convert the optimization problem given in to parametric form

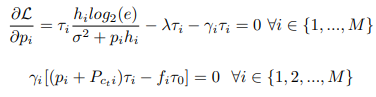


Where, λ is a non-negative parameter. We define F(λ) = max{R − λPT }. Since F(λ) is continuous and strictly decreasing, the equation F(λ) = 0 will yield a unique answer. Therefore, the optimal point of problem and converted problem will be the same. To maximize EE, we have to find the proper λ, which is calculated according to Dinkelbach Algorithm.

First, we assume that τ0 and {τi} are fixed. In this case, problem is a convex optimization problem related to {pi} and also satisfies the Slater’s condition. Thus, the optimal solution can be obtained efficiently by applying the Lagrange dual method. To this end, we need the Lagrangian function of problem which can be written as



Where, {γi} is Lagrangian coefficient. Due to complementary slackness, the optimal Lagrange multipliers are zero in constraints C3, so the corresponding lagrangian multipliers in constraints C3 are omitted in equation. To satisfy Karush-Kuhn-Tucker (KKT) conditions we have



**CHAPTER 5**

**ADVANTAGES AND APPLICATIONS**

**Advantages:**

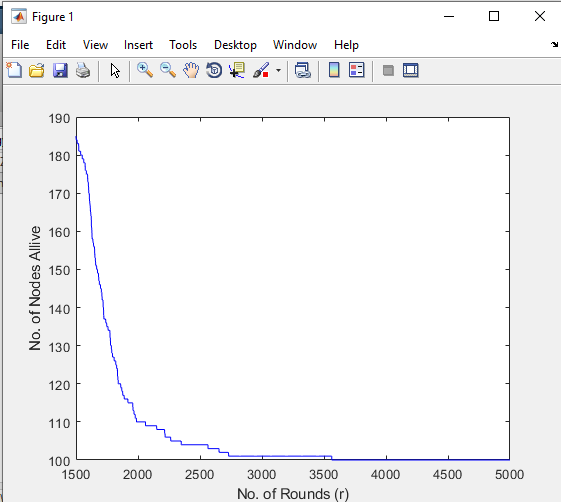
* The advantage of our proposed method is it can maximize the energy for each single sensor.
* The proposed method makes the nodes live longer than the existing methods.
* Energy allocation increases throughput in the network.
* Energy allocation minimizes the HAP transmit power.

**Applications:**

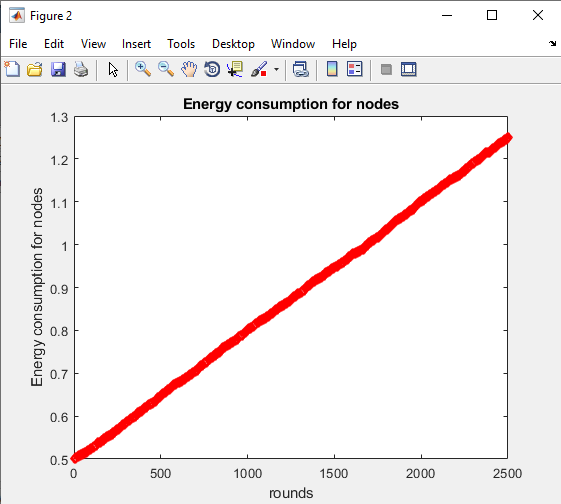
* Wireless Sensor Networks (WSN)
* Internet of Things (IoT)
* 5G MIMO
* 5G Cognitive Radio

**CHAPTER 6**

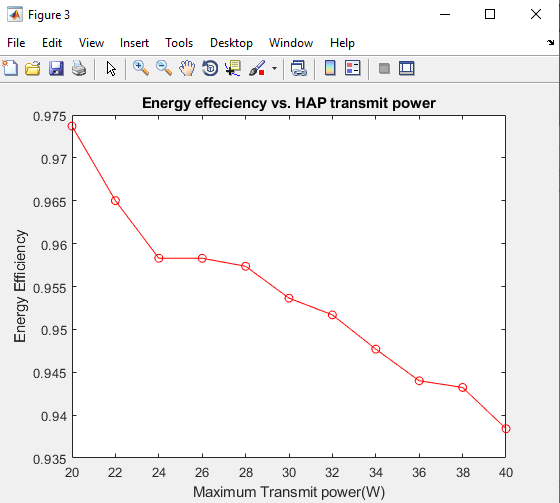
**RESULTS**



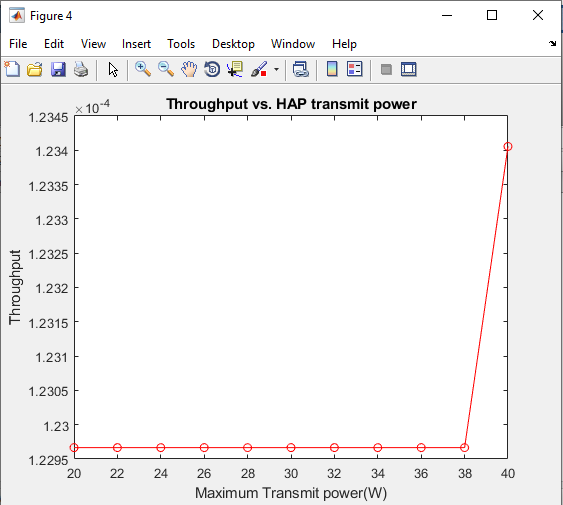
**Fig -1: No of Nodes Alive after 5000 Rounds**



**Fig -2: Energy Consumption for Nodes**



**Fig -3: Energy Efficiency Vs HAP Transmit Power**



**Fig -4: Throughput Vs HAP Transmit Power**

**CHAPTER 7**

**CONCLUSION**

Using the harvest-then-transmit protocol, we provide a novel system architecture where wireless sensors first gather the energy required for data transmission. Additionally, the sensors use TDMA throughout the remaining period to communicate with a hybrid access point. By imposing constraints on the time scheduling parameter and the transmission power for each sensor to the system performance, we address the energy efficiency optimization problem. The problem is solved and closed form expressions are obtained using the Dinkelbach technique. The numerical results show that even while throughput may slightly decline in comparison to other methods, energy consumption will decline much more, resulting in improved energy efficiency as the network performance.

**REFERENCES**

[1] J. Huang, C. Xing, and C. Wang, “Simultaneous wireless information and power transfer: Technologies, applications, and research challenges,” IEEE Communications Magazine, vol. 55, no. 11, pp. 26–32, Nov 2017.

[2] K. Kang, R. Ye, Z. Pan, J. Liu, and S. Shimamoto, “Full-duplex wireless powered iot networks,” IEEE Access, vol. 6, pp. 53 546–53 556, 2018.

[3] Z. Chu, F. Zhou, Z. Zhu, R. Q. Hu, and P. Xiao, “Wireless powered sensor networks for internet of things: Maximum throughput and optimal power allocation,” IEEE Internet of Things Journal, vol. 5, no. 1, pp. 310–321, Feb 2018.

[4] A. Sultana, L. Zhao, and X. Fernando, “Efficient resource allocation in device-to-device communication using cognitive radio technology,” IEEE Transactions on Vehicular Technology, vol. 66, no. 11, pp. 10 024– 10 034, Nov 2017.

[5] S. K. Nobar, K. A. Mehr, and J. M. Niya, “Rf-powered green cognitive radio networks: Architecture and performance analysis,” IEEE Commu nications Letters, vol. 20, no. 2, pp. 296–299, Feb 2016.

[6] S. K. Nobar, K. A. Mehr, J. M. Niya, and B. M. Tazehkand, “Cognitive radio sensor network with green power beacon,” IEEE Sensors Journal, vol. PP, no. 99, pp. 1–1, 2017.

[7] J. Ding, L. Jiang, and C. He, “User-centric energy-efficient resource management for time switching wireless powered communications,” IEEE Communications Letters, vol. 22, no. 1, pp. 165–168, Jan 2018.

[8] Z. Yang, W. Xu, Y. Pan, C. Pan, and M. Chen, “Energy efficient resource allocation in machine-to-machine communications with multiple access and energy harvesting for iot,” IEEE Internet of Things Journal, vol. 5, no. 1, pp. 229–245, Feb 2018.

[9] L. Pei, Z. Yang, C. Pan, W. Huang, M. Chen, M. Elkashlan, and A. Nallanathan, “Energy-efficient d2d communications underlaying noma-based networks with energy harvesting,” IEEE Communications Letters, vol. 22, no. 5, pp. 914–917, May 2018.

[10] Q. Wu, W. Chen, D. W. K. Ng, and R. Schober, “Spectral and energy-efficient wireless powered iot networks: Noma or tdma?” IEEE Transactions on Vehicular Technology, vol. 67, no. 7, pp. 6663–6667, July 2018.

**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.

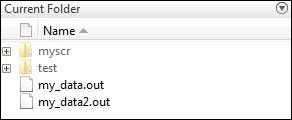
So, I studied Niklaus Wirth’s book Algorithms + Data Structures *=* Programs and learned how to parse programming languages. I wrote the first MATLAB an acronym for Matrix Laboratory in FORTRAN, with matrix as the only data type. The project was a kind of hobby, a new aspect of programming for me to 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:**

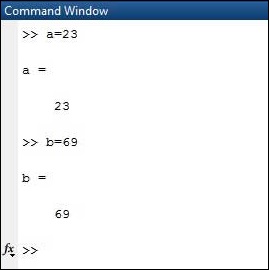
After logging into your account, you can enter MATLAB by double-clicking on the MATLAB shortcut icon (MATLAB 7.0.4) on your Windows desktop. When you start MATLAB, a special window called the MATLAB desktop appears. The desktop is a window that contains other windows. The major tools within or accessible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

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

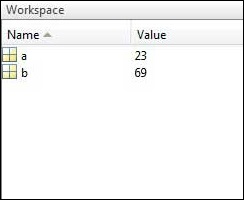
**Current Folder:** This panel allows you to access the project folders and files.



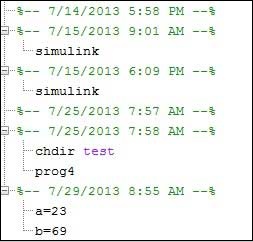
**Command Window:** This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).



**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.