**AUTOMATIC MODULATION CLASSIFICATION USING PRINICIPLE COMPOSITION ANALYSIS BASED FEATURES SELECTION**

**ABSTRACT**

In both military and civilian applications, Automatic Modulation Classification (AMC) is crucial. In this paper, feature-based AMC is employed. A technique called Principle Component Analysis (PCA) is used to minimize the feature vector's dimensionality. To examine the accurate classification rate for test signals at various SNRs, two classifiers—primarily k-nearest neighbor (KNN) and support vector machine (SVM)—are utilized. Data trained at two distinct SNRs, 15dB and 3dB, respectively, are used in the experiments. According to the results, KNN classifier performs better when data is trained at high SNRs. However, when data is learned at low SNR, both classifiers perform nearly identically.

**Keywords:** Automatic Modulation Classification (AMC); Principle Component Analysis; Feature based; classifier; Support Vector Machine; k-Nearest Neighbor.

**CHAPTER 1**

**INTRODUCTION**

The subject of Automatic Modulation Classification (AMC) has been extensively studied. Both military and civilian applications place a high value on it. It is used in electronic warfare (EW) systems, which include signal interception, jamming, and identification, in military applications. However, for many civilian applications, modulation classification in link adaption systems is seen as being of utmost significance.

Transmitters modulate a data signal in digital communications, while receivers demodulate it to recover the original data. The identification of the modulation scheme at the receiving end is known as automatic modulation categorization (AMC). A multi-standard communication system must have an automated modulation classification (AMC), which enables blind recognition of the modulation schemes contained in the received signal.

The likelihood-based and feature-based AMC techniques are the two primary categories. A threshold is compared with the likelihood function of the received signals in likelihood-based categorization.

For modulation classification, feature-based classifiers take advantage of the distinctive traits of retrieved features. Higher order statistics, neural networks, the zero crossing approach, instantaneous frequency or amplitude, and cyclic spectrum features are a few typical aspects that can be found in literature. In a unique feature subset for modulation classification was created using the ambiguity function domain.

The aim of this paper is to investigate the performance of two machine learning based classifiers using Principle Component Analysis (PCA) algorithm. Features have been generated by taking advantage of distinct characteristics of modulated signals in ambiguity domain. The two classifiers that has been used are k-nearest neighbor and multi class Support Vector Machine (SVM). The rest of the paper is organized as follows: Section introduces the system model used in this paper. Section elaborates on the feature selection algorithm that has been used. Section defines the designing parameter of the two classifiers. Section concludes the paper.

AMC using pattern recognition methods is usually not optimal but is easy to implement with close to optimal performance if designed properly. The two main components of pattern recognition based classification are feature extraction and classification. Normally classifiers performance depends on the proper design of the signal features. Therefore, it is extremely important to choose feature size with maximum information. So a feature selection algorithm is used for reducing the pattern vector to a lower dimension, which comprises of the most significant information from the original vector.

**Principal component analysis (PCA)**

Principal component analysis (PCA) is a popular technique for analyzing large datasets containing a high number of dimensions/features per observation, increasing the interpretability of data while preserving the maximum amount of information, and enabling the visualization of multidimensional data. Formally, PCA is a statistical technique for reducing the dimensionality of a dataset. This is accomplished by linearly transforming the data into a new coordinate system where (most of) the variation in the data can be described with fewer dimensions than the initial data. Many studies use the first two principal components in order to plot the data in two dimensions and to visually identify clusters of closely related data points. Principal component analysis has applications in many fields such as Population Genetics, Micro-biome studies, Atmospheric Science etc.

**K-Nearest Neighbor (KNN)**

* KNN classifier is both simple and fast. It is designed using k nearest neighbor (k=16). Distance metric used in the designing is Euclidean. Sample are classified based on ‘nearest’ rule where decision is based on majority rule with nearest point tie-break.
* K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
* K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
* K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
* K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
* K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
* It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

**Support Vector Machine (SVM)**

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyper-plane. SVM chooses the extreme points/vectors that help in creating the hyper-plane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyper-plane.

One vs all (OVA) technique has been used for multiclass SVM. Kernel function has been modified to Gaussian radial basis function (rbf) kernel with scaling factor σ =1. Least square (‘LS’) method has been used to find the separating hyperplanes.

**CHAPTER 2**

**LITERATURE SURVEY**

**[1] Víctor Iglesias, Jesós Grajal, Omar Yeste-Ojeda, “Automatic modulation classifier for military applications,” 19th European Signal Processing Conference, 29 Aug.-2 Sept. 2011:**

Automatic modulation recognition plays an important role in several military and civilian applications. Depending on the application, latency can be the bottleneck for designing an automatic modulation classifier (AMC). In this paper, an AMC based on low complexity signal features to improve latency and percentage of real-time operation is designed for broad-band military applications.

**Summary:** Studied about the Automatic Modulation Classifier (AMC) in military applications.

**[2] Zhechen Zhu and Asoke K. Nandi, “Modulation Classification for Civilian Applications,” Wiley, 2014:**

Automatic Modulation Classification (AMC) has been a key technology in many military, security, and civilian telecommunication applications for decades. In military and security applications, modulation often serves as another level of encryption; in modern civilian applications, multiple modulation types can be employed by a signal transmitter to control the data rate and link reliability.

This book offers comprehensive documentation of AMC models, algorithms and implementations for successful modulation recognition. It provides an invaluable theoretical and numerical comparison of AMC algorithms, as well as guidance on state-of-the-art classification designs with specific military and civilian applications in mind.

**Summary:** Studied about the Modulation Classification for Civilian Applications.

**[3] A. Swami and B. M. Sadler, “Hierarchical digital modulation classification using cumulants,” IEEE Trans. Commun., vol.48, pp. 416- 429, 2000:**

A simple method, based on elementary fourth-order cumulants, is proposed for the classification of digital modulation schemes. These statistics are natural in this setting as they characterize the shape of the distribution of the noisy baseband I and Q samples. It is shown that cumulant-based classification is particularly effective when used in a hierarchical scheme, enabling separation into subclasses at low signal-to-noise ratio with small sample size. Thus, the method can be used as a preliminary classifier if desired. Computational complexity is order N, where N is the number of complex baseband data samples. This method is robust in the presence of carrier phase and frequency offsets and can be implemented recursively. Theoretical arguments are verified via extensive simulations and comparisons with existing approaches.

**Summary:** Studied about Hierarchical digital modulation classification using cumulants.

**[4** **A. K. Nandi and E. E. Azzouz, “Modulation recognition using artificial neural networks,” Signal Processing, pp. 165-175, 1997:**

This paper presents artificial neural networks (ANNs) for the recognition of either analogue or digital modulation types. Computer simulations of different types of band-limited, modulated signals corrupted by band-limited Gaussian noise sequence have been carried out to measure the performance of the ANN approach. The threshold SNR for the recognition of either analogue or digitally modulated signals with average success rate ⩾98% is found to be about 10 dB. Comparisons of results from the ANN approaches and the decision-tree methods are presented. **Summary:** Studied about Modulation recognition using artificial neural networks.

**[5] S. Z. Hsue and S. S. Soliman, “Automatic modulation classification using zero crossing,” IEE Radar and Signal Processing, vol. 137, pp. 459-464, 1990:**

A modulation recogniser that automatically reports modulation types of constant-envelope modulated signals is developed using zero-crossing techniques. The zero-crossing sampler, as a signal conditioner, has the advantage of providing accurate phase transition information over a wide dynamic frequency range. Signal parameters such as zero-crossing variance carrier-to-noise ratio (CNR) and carrier frequency are estimated. Phase difference and zero-crossing interval histograms play the role of features for modulation recognition. The classifier performance is given in the form of a confusion matrix. The simulation results obtained demonstrate that a reasonable average probability of correct classification is achievable for CNR ages; 15 dB**.**

**SUMMARY:** Studied about Automatic modulation classification using zero crossing.

**CHAPTER 3**

**EXISTING METHOD**

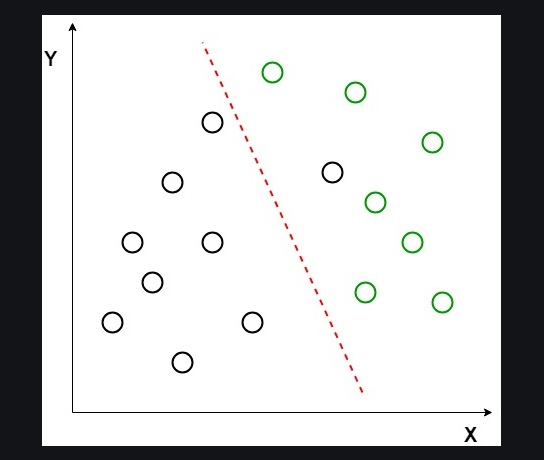
The first proposed Automatic Modulation Classification using Linear Discriminant Analysis (LDA) for extracting features that are used for classification using KNN and SVM. LDA was used for extracting features from the modulation that are used for classification using the classifiers. Later, the SVM and KNN are trained and tested feeding the same features generated by LDA. SVM and KNN showed better results at both training and testing.

Linear Discriminant Analysis or Normal Discriminant Analysis or Discriminant Function Analysis is a dimensionality reduction technique that is commonly used for supervised classification problems. It is used for modelling differences in groups i.e. separating two or more classes. It is used to project the features in higher dimension space into a lower dimension space.

For example, we have two classes and we need to separate them efficiently. Classes can have multiple features. Using only a single feature to classify them may result in some overlapping as shown in the below figure. So, we will keep on increasing the number of features for proper classification.

Suppose we have two sets of data points belonging to two different classes that we want to classify. As shown in the given 2D graph, when the data points are plotted on the 2D plane, there’s no straight line that can separate the two classes of the data points completely. Hence, in this case, LDA (Linear Discriminant Analysis) is used which reduces the 2D graph into a 1D graph in order to maximize the separability between the two classes.

Here, Linear Discriminant Analysis uses both the axes (X and Y) to create a new axis and projects data onto a new axis in a way to maximize the separation of the two categories and hence, reducing the 2D graph into a 1D graph. Two criteria are used by LDA to create a new axis:



1. Maximize the distance between means of the two classes.
2. Minimize the variation within each class.

In the above graph, it can be seen that a new axis (in red) is generated and plotted in the 2D graph such that it maximizes the distance between the means of the two classes and minimizes the variation within each class. In simple terms, this newly generated axis increases the separation between the data points of the two classes. After generating this new axis using the above-mentioned criteria, all the data points of the classes are plotted on this new axis and are shown in the figure given below.

**KNN Classifier**

Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.

**SVM Classifier**

SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat.

**Disadvantages:**

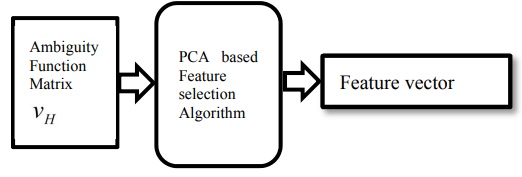
* Computationally complex.
* Outage probability will be high.

**CHAPTER 4**

**PROPOSED METHOD**

Principal component analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set. Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize and make analyzing data much easier and faster for machine learning algorithms without extraneous variables to process. In this paper, ambiguity function based frequency features at different time intervals have been used to generate a feature vector. Ambiguity function is a × NM matrix (M and N are frequency and time sizes respectively.

For each class I distinct AF matrix is computed. Optimum features are selected as follows: Connect column of AF matrix of each class to form a high dimensional vector Hv. Compute Principle Component Analysis on vector Hv. Sort the Eigen vectors in descending order. Get 5, 10 and 20 indices of the projected matrix in new sub space according to the sorted Eigen vectors. Check the classification rate with each number of indices. Indices here denotes number of features selected for classification. The above algorithm does not require any prior knowledge of the received signal like carrier frequency. It is also robust under low SNR. Moreover, ambiguity function used in this paper have a distinct energy distributions amongst different modulation classes and thus is a good feature subset.

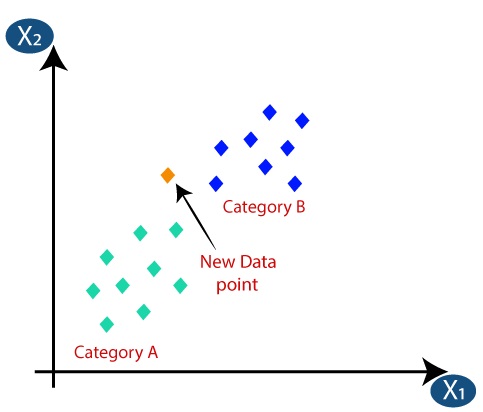


**KNN Classifier**

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset.

The K-NN working can be explained on the basis of the below algorithm:

1. Select the number K of the neighbors
2. Calculate the Euclidean distance of K number of neighbors
3. Take the K nearest neighbors as per the calculated Euclidean distance.
4. Among these k neighbors, count the number of the data points in each category.
5. Assign the new data points to that category for which the number of the neighbor is maximum.
6. Our model is ready.



1. Firstly, we will choose the number of neighbors, so we will choose the k=5.
2. Next, we will calculate the Euclidean distance between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:
3. By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B.
4. As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

Here, in this paper the feature vector generated by the Principal Composition Analysis (PCA) is fed to the KNN classifier. The classifier is trained on the features generated by PCA. The Classifier trained on the less but more accurate features which were generated in the previous stage. The KNN classifier takes more time to train, if the feature vector is large and have more features. So, the PCA is more efficient at achieving this particular thing. The training data will be both 3dB and 15dB respectively. The KNN classifier is trained on both for achieving a greater accuracy. The classifier is tested using the test data that were separated during the initial stage. KNN classifier is better at classifying the data at lower dBs but at higher dBs both classifiers showed almost similar results.

**SVM can be of two types:**

* **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
* **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Hyperplane and Support Vectors in the SVM algorithm:

**Hyperplane:** There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

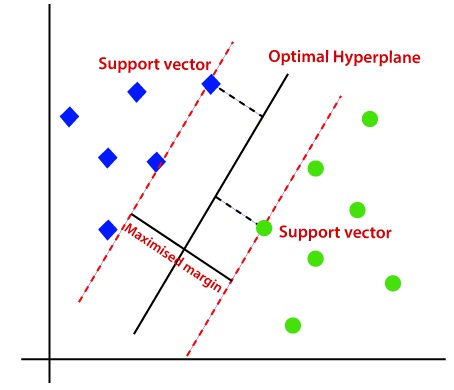
We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

**Support Vectors:**

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

**Linear SVM:**

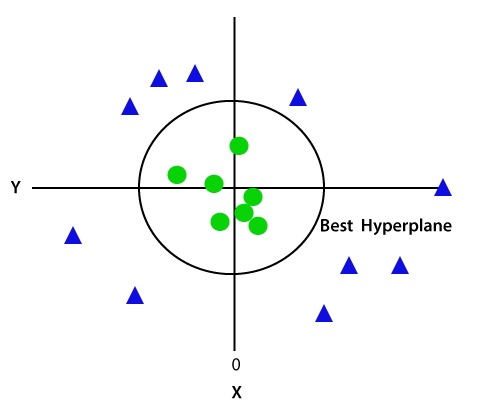
The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue.



Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a hyperplane. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called as margin. And the goal of SVM is to maximize this margin. The hyperplane with maximum margin is called the optimal hyperplane.

**Non-Linear SVM:**

If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the below image:



So to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y, so for non-linear data, we will add a third dimension z. It can be calculated as,

z=x2 +y2

Since we are in 3-d Space, hence it is looking like a plane parallel to the x-axis. If we convert it in 2d space with z=1, then it will become.

Here, in this paper, we are training the Support Vector Machine (SVM) with the features generated by PCA at both 3dB as well as 10dBs. The feature vector consists of less but very accurate features for a better training. The training is most complex process during the artificial intelligence environment. The training will take from minutes to as long as days. So, a feature vector of less size with an accurate details extraction is much more important. Here, the PCA is better at achieving the particular thing. Later, the classifier is tested using the test data that were separated at the initial stage. The SVM showed almost similar results at higher dBs as KNN classifier.

**CHAPTER 5**

**ADVANTAGES AND APPLICATIONS**

**Advantages:**

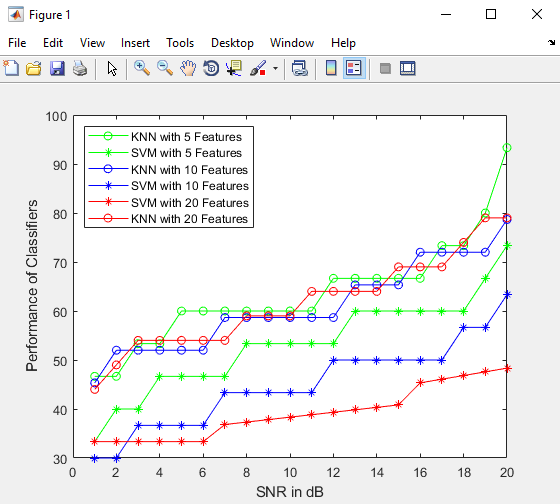
* The advantage of Principle Composition Analysis (PCA) is its feature vector which has less size compared to any other existing techniques.
* The PCA’s feature vector makes the training process consumes less time compared to existing techniques.
* KNN classifier as well as SVM classifier takes less time for a better training.

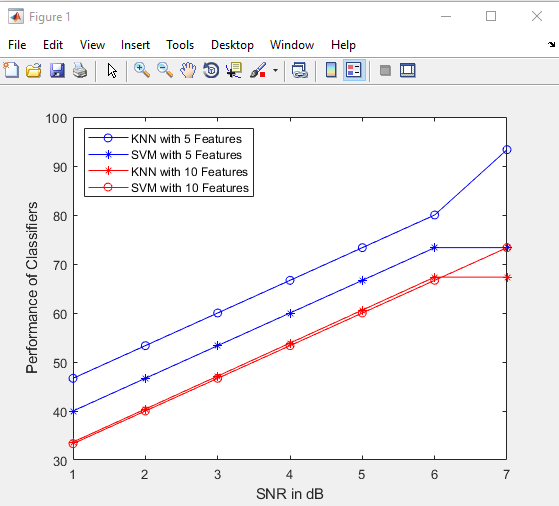
**Applications:**

* Machine Learning
* Deep Learning
* Digital Signal Processing
* Medical Signal Processing

**CHAPTER 6**

**RESULTS**

****

****

**CHAPTER 7**

**CONCLUSION**

Here, in this paper we implemented the classification of modulation dataset that contains three types of modulation techniques namely, Frequency Shift Keying (FSK), Phase Shift Keying (PSK) and Quadrature Amplitude Modulation (QAM). These are classified using both KNN classifier and SVM classifier. The features for training the classifiers were generated using the Principle Composition Analysis (PCA). We can finally conclude that, the PCA generated better features that is lesser in size and made the training process easier than any other existing techniques.

**REFERENCES**

[1] Víctor Iglesias, Jesós Grajal, Omar Yeste-Ojeda, “Automatic modulation classifier for military applications,” 19th European Signal Processing Conference, 29 Aug.-2 Sept. 2011.

[2] Zhechen Zhu and Asoke K. Nandi, “Modulation Classification for Civilian Applications,” Wiley, 2014.

[3] A. Swami and B. M. Sadler, “Hierarchical digital modulation classification using cumulants,” IEEE Trans. Commun., vol.48, pp. 416- 429, 2000.

[4] A. K. Nandi and E. E. Azzouz, “Modulation recognition using artificial neural networks,” Signal Processing, pp. 165-175, 1997.

[5] S. Z. Hsue and S. S. Soliman, “Automatic modulation classification using zero crossing,” IEE Radar and Signal Processing, vol. 137, pp. 459-464, 1990.

[6] H. Deng, M. Doroslovacki, H. Mustafa, J. Xu, and S. Koo, “Instantaneous feature based algorithm for HF digital modulation classification,” in Proc. CISS Conf., 2003, John Hopkins University, Baltimore, MD, US.

[7] William C. Headley, Jesse D. Reed, Claudio R. C. M. da Silva, “Distributed Cyclic Spectrum Feature-Based Modulation Classification,” Wireless Communications and Networking Conference, 31 March-3 April 2008.

[8] Barathram Ramkumar, “Automatic modulation classification for cognitive radios using cyclic feature detection,” IEEE Circuits and Systems Magazine , Volume: 9, Issue: 2, Second Quarter 2009.

[9] Haijian Zhang, Guoan Bi, Sirajudeen Gulam Razul, and Chong Meng Samson See, “Supervised Modulation Classification Based on Ambiguity Function Image and Invariant Moments,” 9th IEEE Conference on Industrial Electronics and Applications, pp.1461-1465, 2014.

[10] A. K. Nandi, and E. E Azzouz, “Algorithms for Automatic Modulation Recognition of Communication Signals,” IEEE Trans. Commun., vol. 46, pp. 431-436, 1998.

**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.

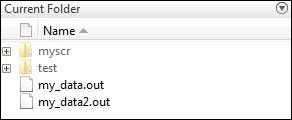
So, I studied Niklaus Wirth’s book Algorithms + Data Structures *=* Programs and learned how to parse programming languages. I wrote the first MATLAB an acronym for Matrix Laboratory in FORTRAN, with matrix as the only data type. The project was a kind of hobby, a new aspect of programming for me to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

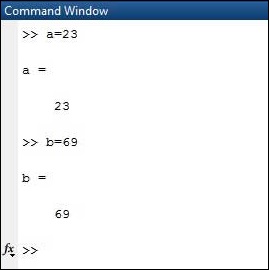
After logging into your account, you can enter MATLAB by double-clicking on the MATLAB shortcut icon (MATLAB 7.0.4) on your Windows desktop. When you start MATLAB, a special window called the MATLAB desktop appears. The desktop is a window that contains other windows. The major tools within or accessible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

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

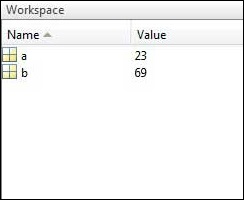
**Current Folder:** This panel allows you to access the project folders and files.



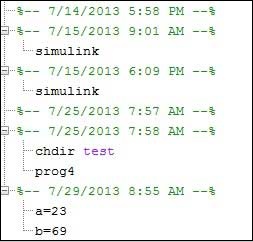
**Command Window:** This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).



**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.