**A REAL TIME SPEECH TO TEXT CONVERSION**

**SYSTEM USING BIDIRECTIONAL KALMAN**

**FILTER IN MATLAB**

**ABSTRACT**

A real-time speech to text conversion system accurately reproduces the user's pronunciation of the uttered words in the text form. A real-time speech was made by us a real-time, noisy environment was used to test the recognition system. We enhanced the functionality of this real-time voice recognition system by using the design of a bidirectional nonstationary Kalman filter. In nonstationary noisy environments, bidirectional Kalman filter has been shown to be the best noise estimator. The real-time speech to text conversion system converts the spoken words as soon as they are said. The goal of this study was to develop a new speech recognition system that is computationally straightforward and more noise-resistant than the HMM-based system. Our own custom-built database was chosen for its adaptability, and the TIDIGIT database was used for a comparison of accuracy with an HMM-based voice recognition system. The MFCC features of the speech sample were calculated, and words were identified based on how well each sampled word matched the features. When the system was evaluated in various noise environments, we achieved an overall word accuracy of 90%.

**CHAPTER 1**

**INTRODUCTION**

Natural language processing has historically been a promising subject for research. Natural Language Processing has a wide range of uses. One of the most significant uses of natural language processing is speech recognition. We have traditionally valued speech as the most crucial component of daily communication. We use a particular language to communicate our views. By using speech recognition, computers can comprehend our language (natural language). Voice recognition, often known as word-by-word recognition, is the process of taking speech characteristics and categorizing those using datasets that have already been recorded.

A word needs to be sent to more advanced software for syntactic and semantic analysis in order to be recognized. This pattern-matching method evaluates audio signals by framing them into phonetics (the amount of words, phrases, and sentences). To complete this activity, one must first record a voice sample and then convert it to wav format. When a word is recognized, parameters based on the spectrum are acquired.

Several statistical techniques are used to analyse words and determine their specific worth. Words vary within their defined range of occurrence, finding the most useful speech signal characteristics is one of the crucial jobs in improving the word recognition process. Linear Prediction Coefficients (LPC) and Mel Frequency Cepstrum Coefficients (MFCC) are two methodologies that have been utilized to carry out similar jobs. These methods produce a new spectrum that is distinct from the prior spectrum of spoken words.

Speech intensification is the process of improving a speech sample's overall comprehensibility or quality. Speech improvement involves de-reverberating and separating the unconstrained signals in addition to reducing noise in a speech sample. It is desirable to enhance the speech since when speech is processed through any of the instrument in \slab it get impacted with the noise (background noise or otherwise) and individuality of the voice varies with times which influences the complete recognition process.

Finding devices that actually work in a variety of real-world settings has thus become a very difficult problem for the scientists. Nonetheless, this criterion is crucial in defending the algorithm's effectiveness in terms of quality and understandability.

Kalman Filter is a state estimator that produces an optimal estimate and minimizes the mean square error. Kalman filtering is an effective approach to remove nonstationary noise form background or otherwise. It is a state space model that always distillates the adorable information from the signal which is going to be processed. In its contrivance a system model is first selected and model parameters are estimated from its previous state. The parameters of real time model are selected first and there are so many unknown parameters that are really hard to select. There are so many proposed algorithms on Kalman filtering that shows that it a best parameter estimator in the world have used the Kalman filter for speech enhancement purpose. The uses the Bidirectional Kalman filter for a robust speech recognition system. In this paper we have used the Bidirectional Kalman filter for the intensification of our speech recognition system in background noise.

**CHAPTER 2**

**LITERATURE SURVEY**

**[1] Takao Suzuki, Yasuo Shoji, IEEE,Digital Communications Laboratories, Oki Electric Industry Co. Ltd., Japan, pp. 1515-1519, 1989:**

From the viewpoint of speech communication services for the asynchronous transfer mode (ATM) network and in order to introduce the necessary conditions for speech processing over an ATM network, the authors have developed a novel speech-processing scheme applied at the end of the ATM network. For this speech processing, speech signals are processed basically by two techniques: silence deletion for speech compression and low-bit coding for 32-kb/s adaptive differential PCM (ADPCM). In order to reduce speech quality degradation caused by lost ATM cells in network congestion conditions, the authors propose a cell-reconstruction algorithm using waveform substitution for ADPCM-coded speech based on the pitch estimation method. In addition, to maintain good speech quality, some new algorithms for speech processing are introduced. It was confirmed through subjective evaluation tests that the proposed speech-processing scheme for the ATM network could provide good speech quality up to a cell loss rate of about 3%. Two kinds of custom LSIs for implementing these speech-processing algorithms are described.

**Summary:** Studied about a new speech processing scheme for ATM switching systems.

**[2] J. S. Lim, IEEE Trans. Acoustics, Speech and Signal Processing, vol. ASSP-26, no. 5, pp.471 -472 1978:**

An intelligibility test was performed to evaluate a correlation subtraction method for enhancement of degraded speech due to additive white noise. Results indicate that such a scheme does not significantly increase speech intelligibility at the S/N ratios where the intelligibility scores of unprocessed speech range between 20 and 70 percent.

**Summary:** Studied about the evaluation of a correlated subtraction method for enhancing speech degraded by additive white noise.

**[3] R. E. Kalman and R. S. Bucy, Trans. ASME Series D, J. Basic Engineering, pp. 95-108, 1961:**

A nonlinear differential equation of the Riccati type is derived for the covariance matrix of the optimal filtering error. The solution of this “variance equation” completely specifies the optimal filter for either finite or infinite smoothing intervals and stationary or nonstationary statistics. The variance equation is closely related to the Hamiltonian (canonical) differential equations of the calculus of variations. Analytic solutions are available in some cases. The significance of the variance equation is illustrated by examples which duplicate, simplify, or extend earlier results in this field. The Duality Principle relating stochastic estimation and deterministic control problems plays an important role in the proof of theoretical results. In several examples, the estimation problem and its dual are discussed side-by-side. Properties of the variance equation are of great interest in the theory of adaptive systems. Some aspects of this are considered briefly.

**Summary:** Studied about new results in linear filtering and prediction theory.

**[4] Gabrea, M. IEEE Canadian Conf. on Electrical and Computer Engineering, 2001, vol. 1, pp. 521–526:**

This paper deals with the problem of speech enhancement when only a corrupted speech signal is available for processing. Kalman filtering is known as an effective speech enhancement technique, in which the speech signal is usually modeled as an autoregressive (AR) model and represented in the state-space domain. Various approaches based on the Kalman filter are presented in the literature. They usually operate in two steps: first, additive noise and driving process variances and speech model parameters are estimated and second, the speech signal is estimated by using Kalman filtering. In this paper sequential estimators are used for suboptimal adaptive estimation of the unknown a priori driving process and additive noise statistics simultaneously with the system state. The estimation algorithm provides improved state estimates at little computational expenses. **Summary:** Studied about adaptive kalman filtering-based speech enhancement algorithm.

**[5] Jeong, S., Hahn, M., Electron. Lett., 2001, 37, (12), pp. 800–802:**

In car noise environments, speech quality is severely degraded and speech recognition systems fail to produce high recognition rates. Highpass filtering and Kalman filtering with a whitening process are applied to car noise corrupted speech signals and improvement both in speech quality and in speech recognition is achieved**.**

**SUMMARY:** Studied about speech quality and recognition rate improvement in car noise environments.

**CHAPTER 3**

**EXISTING METHOD**

The complex problem of speech to text conversion of Kannada Language. We propose a novel Kannada Automated Speech to Text conversion System (ASTC). We train and test the Speech Processing System using CMU Sphinx framework. CMU Sphinx is dynamic in nature with support for other languages along with English. We train the Acoustic model for Kannada speech with 1000 general spoken sentences and tested 150 sentences. We build our system utilizing features available in CMU Sphinx, thus showcasing the conceivable flexibility of this framework for Kannada voice to text conversion. In this paper, Kannada sentences with four to ten word length is researched. The speech conversion system permits ordinary people to speak to the computer in order to retrieve information in textual form. The number of alphabets in Kannada are 52. The system investigates extensibility of recognizing all letters and morphological variants of spoken Kannada words.

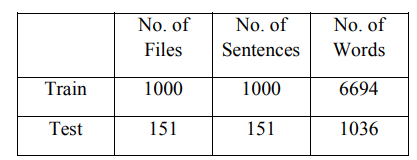
The applications to encode and decode the speech data for English are supported by huge database of language models. Availability of this database enhanced the performance of speech recognition accuracy for English. This is not the same for other Asian or Indian Languages. To generate an Acoustic model for Indian languages is a challenging task. Choosing a tool and configuring the system to work for Indian language requires a series of configuration steps with open source tools such as SIP and CMU Sphinx. Little work has been done to work with these tools for recognizing Indian Language speech compared to English language.

**An ASR System like Sphinx-4 uses three types of language-dependent models:**

Acoustic model for Kannada represents statistical range of possible audio representations for Kannada Language phonemes. A pronunciation dictionary specifying how each word is pronounced in terms of the phonemes in the acoustic model. A language model (LM) models pattern probability of occurrence of words. This is normally customized for domain specific application. Every word in the language model must be in the pronunciation dictionary. In our Acoustic model training we created above models according to CMU Sphinx framework specifications for new languages.

**Data Preparation**

We used publicly available speech corpora for Kannada created by IIIT Hyderabad. The corpus contains 1000 sentences related to general information about Karnataka. The corpus files were of single speaker. We created speech sample of the same sentences and converted into two user speech data. The corpus comprise 5000 words in which 2112 are unique words. These words are span through the usage of 36 phonemes. The recorded samples were rechecked to ensure the utterances were recorded efficiently.



**Fig: Speech Corpus Statistics**

**Language model**

The aim of the language model is to produce accurate value of probability of a word. A language model represents the structural constraints available in the language to generate the probabilities. In language model, two different words have similar sounding phone. The LM also specifies what are the valid words in the language and their arrival sequence in the speech data. The generated model for Kannada plays a vital role in generating context-dependent and context independent acoustic model generation for Kannada speech sample.

**Phoneme set**

Kannada spoken language uses 52 alphabets. These alphabets are transformed into 36 phonemes which are written using English alphabets according to Sphinx-4 specifications to represent phonemes for new language. Based on these phonemes Kannada word dictionary is prepared.

**Dictionary Preparation**

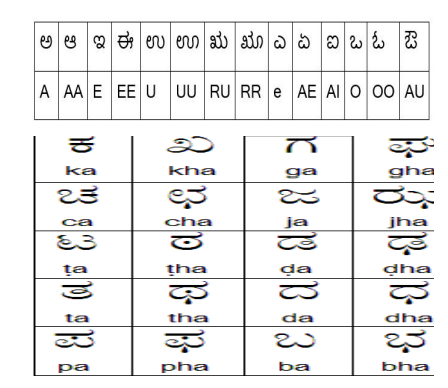
Dictionary should comprise the words which are constituents of available phoneme set. Every word should be a subset of available phoneme group of the word. For each token T should be a subset group of basic phonemes. We are using 36 phonemes for Kannada Speech Processing System. T ⊆ S {p1, p2….. pn} Where S is a subset of phonemes of a given language. T is any word or token of pronunciation dictionary. We created 2112 Kannada tokens in our dictionary.

**Feature Extraction**

The main features of our system are frequency of the waveform of the speech data. The system found the upper frequency as 6800 and lower frequency as 130 for the Kannada speech. It uses Nfilt parameter value of 52 and discrete cosine transformation (DCT) to extract transformations. The speech sample features also considers the age of the speaker as a parameter. Speaker-dependent information exists both in the spectral envelope and in the supra-segmental features of speech. This individual information can be further classified into temporal and dynamic features. Speaker identification/verification methods can be divided into text-dependent and text-independent methods. We are using text dependent feature extraction.

**Acoustic model**

Acoustic model creation for the given speech data in Sphinx framework is done by sphinx train module. It uses built in HMM and Viterbi algorithms and modules to train the system by itself. The output of this phase is written in configuration file. This configuration file describes the components and their respective values to be used by the speech processing system. The parameters written in configuration file are used by the system to run the decoder which generates the acoustic model for a given language.

****

**Fig: Kannada Letters**

**Acoustic-Phonetic features of Kannada**

The acoustic-phonetic of Kannada differs from the European languages. The Kannada alphabet consists of 14 vowels, 25 stop consonants and 9 non-stop consonants. The stop consonants are ordered in a systematic manner in most of the Indian language and this order may suggest ideas for developing a recognition system. The pronunciation of Kannada alphabets and words are to be modelled with the help of English letter based phoneme set.



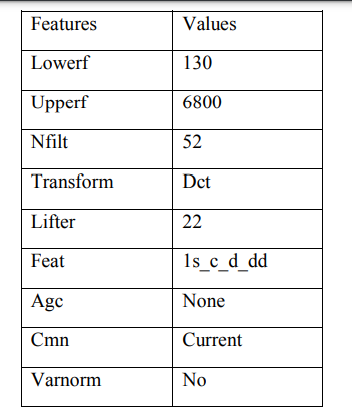
**Fig: Kannada Letters**

**Sphinx Train**

Sphinx-4 is used to convert speech recordings into Text. It also helps to identify the speakers, adapt models, and align existing transcription to audio. It is a speech recognition library in Java. Sphinx Train creates an acoustic model which is manipulated by Sphinx-4. This acoustic model is important for any language for speech recognition. Once data is trained, the database will be created. The next step is to create speech recognition files by running the sphinx train.

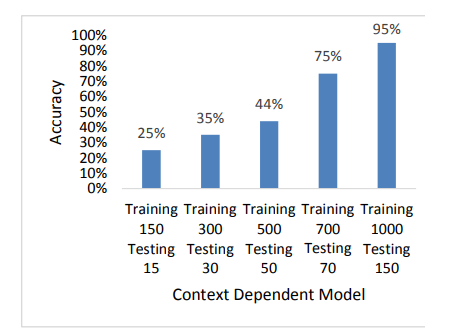
**Feature Extraction Parameters**

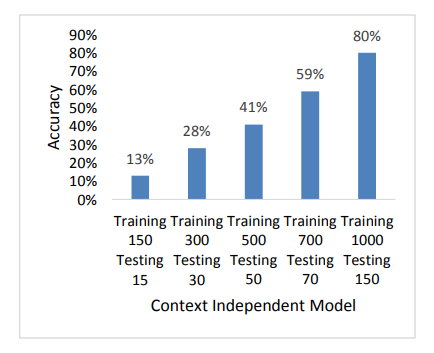
The objective of feature extraction is to encode each and every utterance and group them based on their characteristic features. To classify a speech signal into its respective group we take certain set of features and these play a major role in grouping the acoustics to their respective classes.



**Fig: Feature Extraction Details**

Experiments with Different Number of Trainings We started our training and testing with 150(training)/15(testing), 300/30,500/50,700/70 and 1000/150 sentences audio files. The experiment shows increase in accuracy level as the number of training files increased. Initially Thousand sentences of Kannada language are recorded and trained with different. Testing of randomly chosen one fifty sentences are made and the results are given in figures.





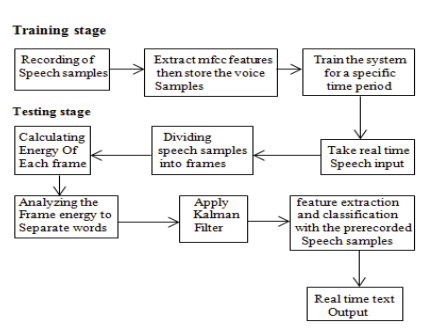
**Disadvantages:**

* Computationally complex.
* Takes more time for implementation.
* Training the classifier with less features will result in less accurate.

**CHAPTER 4**

**PROPOSED METHOD**

The process of speech recognition system is divided in two stages, first is training stage and second is the testing stage.



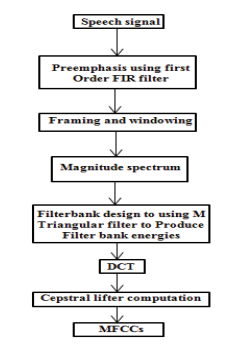
**Fig: Speech Recognition Process**

**TRAINING STAGE**

A database is generated in this stage by the user recording certain voice samples. The captured speech samples are then saved in Matlab's.wav format. Following this step, the speech recognition system needs to be trained.

**FEATURE EXTRACTION**

Each signal consists of a few characteristics or properties. We can categorize the signal qualities based on their characteristics. We take some of speech's characteristics out. The extraction of attributes is one of the simplest methods for speech recognition. Speech is a time-varying signal, and managing such a signal is a challenging undertaking. Hence, speech characteristics are crucial to recognition. Short time qualities are used to deal with a lengthy sequence of speech (melody of speech). Thus, we choose to extract MFCC-based short time speech features.



**Fig: MFCC Feature Extraction**

With this system, we have employed MFCC characteristics to recognize each word. The word "Mel" in the MFCCs refers to a speech signal's melody. The essential bandwidth frequencies filters of the human ear, which are the basis for MFCC characteristics, spaced linearity between the high frequency and low frequency of each word uttered by the user. The human understanding for different frequency ranges of the uttered word is shown on a nonlinear scale. Pitch period of every word is measured with a Mel scale.

**CREATING THE DATABASE AND TRAINING OF VOICE SAMPLES**

To recognize the uttered word of the speaker, a database is created to resemble the pronounced word. To create such database, we first recorded some speech samples. We have trained this system for 100 words in English Language and three separate samples of each word were taken.

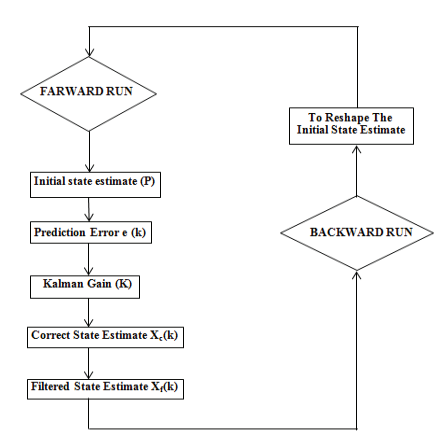
**APPLYING BIDIRECTIONAL NONSTATIONARY KALMAN FILTER ALGORITHM**

After the process of word separation Kalman filter removes the unwanted noise and gives the filtered output. The proposed bidirectional Kalman filter algorithm is given in the figure. Real time speech‘s (k)’ get interrupted with some unwanted noise v(k) while processing.



As we know the nature of noise is nonstationary and cannot be measured. In that case Kalman filter is applied to minimize the mean square error of the clean speech signal and the noiseous speech signal to enhance the speech quality. Equations of the bidirectional Kalman filter algorithm are divided into two parts: forward run and backward run. In the forward run first we set the initial state estimate P to ‘0’. Then the second stage is the error prediction e(k) that can be calculated with the Y(k) (signal + noise) and a predicted measurement variable (assumed) Yp(k).





**Fig: Bi-Directional Non-Stationary Kalman Filter Loop Operation**

As we know the nature of noise is nonstationary and cannot be measured. In that case Kalman filter is applied to minimize the mean square error of the clean speech signal and the noiseous speech signal to enhance the speech quality. Equations of the bidirectional Kalman filter algorithm are divided into two parts: forward run and backward run. In the forward run first we set the initial state estimate P to ‘0’. Then the second stage is the error prediction e(k) that can be calculated with the Y(k) (signal+noise) and a predicted measurement variable(assumed) Yp(k).



Next step is to calculate the Kalman gain K. Its value can be calculated with the following parameters as,



‘C’ is the measurement gain matrix and ‘Pp(k)’ is the Autocavriance matrix of the predicted state variable ‘R’ is the Autocovariance matrix of the measurement noise. The correct state estimate is enumerated with Kalman gain and the predicted state estimate Xp(k) as follows,



The filtered state estimate Xf(k) is equal to the correct state estimate. After this the filtered state estimate is applied to reshape the initial state estimate. Loop is executed again and to achieve the best estimate.

**CHAPTER 5**

**ADVANTAGES AND APPLICATIONS**

**Advantages:**

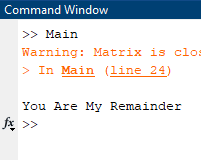
* The advantage of Bi-Directional Kalman Filter achieves a greater accuracy in estimation.
* The Kalman filter at different levels produces different estimates which will be compared and the noisy features will be minimized at a greater level.
* Bi-Directional Kalman Filter unlike other methods or techniques is somewhat easy and will takes less time.

**Applications:**

* Digital Signal Processing
* Speech Processing
* Audio Processing
* Machine Learning
* Deep Learning

**CHAPTER 6**

**RESULTS**

****

**Fig: Showing the Classification Results**

**CHAPTER 7**

**CONCLUSION**

Here, in this paper we can finally conclude that the system’s accuracy can be increased in more different noise scenarios. From the table it is concluded that the accuracy of the system is 100% up to 5dB SNR. After 5dB SNR the accuracy drops. By using this code the system can be trained for more words and paragraphs. Kalman filter removes background noise very efficiently. But this filter takes large time to filter out noise. In case or continuous acquisition of speech it can take more time to display text because of its word by word filtration process.

**REFERENCES**

[1] J. D. Tardelli, C. M. Walter, “Speech waveform analysis and recognition process based on non-Euclidean error minimization and matrix array processing techniques”. IEEE ICASSP, pp. 1237-1240, 1986.

[2] Takao Suzuki, Yasuo Shoji, “A new speech processing scheme for ATM switching systems”. IEEE,Digital Communications Laboratories, Oki Electric Industry Co. Ltd., Japan, pp. 1515-1519, 1989.

[3] J. S. Lim "Evaluation of a correlated subtraction method for enhancing speech degraded by additive white noise", IEEE Trans. Acoustics, Speech and Signal Processing, vol. ASSP-26, no. 5, pp.471 -472 1978.

[4] R. E. Kalman and R. S. Bucy, “New results in linear filtering and prediction theory,” Trans. ASME Series D, J. Basic Engineering, pp. 95-108, 1961.

[5] Gabrea, M.: ‘Adaptive Kalman filtering-based speech enhancement algorithm’. IEEE Canadian Conf. on Electrical and Computer Engineering, 2001, vol. 1, pp. 521–526.

[6] Jeong, S., Hahn, M.: ‘Speech quality and recognition rate improvement in car noise environments’, Electron. Lett., 2001, 37, (12), pp. 800–802.

[7] Ma, J., Deng, L.: ‘Efficient decoding strategies for conversational speech recognition using a constrained nonlinear state-space model’, IEEE Trans. Speech Audio Process., 2003, 11, (6), pp. 590–602.

[8] Mathe, M., Nandyala, S.P., Kishore Kumar, T.: ‘Speech enhancement using Kalman filter for white, random and color noise’. IEEE Int. Conf. on Devices, Circuits and Systems (ICDCS), 2012, pp. 195–198.

[9] Yeh Huann Goh, Paramesran Raveendran, Yann Ling Goh, “Robust speech recognition system using bidirectional Kalman filter”, IET Trans. Pp. 1751-9675, 2015.

[10] Tony Lacey, “Likelihood interpretation of Kalman filter”, tutorial lecture, April 2006.

**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.

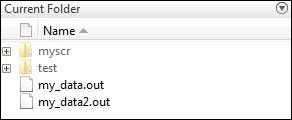
So, I studied Niklaus Wirth’s book Algorithms + Data Structures *=* Programs and learned how to parse programming languages. I wrote the first MATLAB an acronym for Matrix Laboratory in FORTRAN, with matrix as the only data type. The project was a kind of hobby, a new aspect of programming for me to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

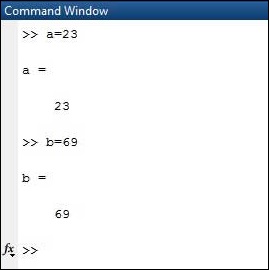
After logging into your account, you can enter MATLAB by double-clicking on the MATLAB shortcut icon (MATLAB 7.0.4) on your Windows desktop. When you start MATLAB, a special window called the MATLAB desktop appears. The desktop is a window that contains other windows. The major tools within or accessible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

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

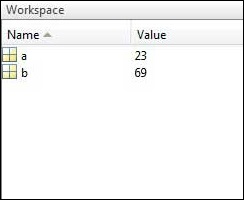
**Current Folder:** This panel allows you to access the project folders and files.



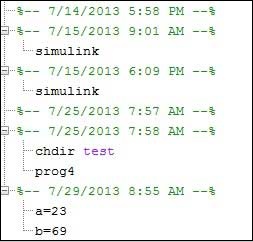
**Command Window:** This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).



**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.