P-2552

ELIMINATING NOISY SENSOR DATA IN WIRELESS SENSOR NETWORKS

By

J.GANESH

Reg. No: 0720108005

of

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna university, Coimbatore)

COIMBATORE – 641 006

MAHATMA GANDHI CENTRAL LIBRARY KCT CBE-6

A PROJECT REPORT

Submitted to the

FACULTY OF INFORMATION AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements for the award of the degree of

MASTER OF ENGINEERING IN

COMPUTER SCIENCE AND ENNGINEERING

MAY, 2009

BONAFIDE CERTIFICATE

Certified that this project report titled "**ELIMINATING NOISY SENSOR DATA IN WIRELESS SENSOR NETWORKS**" is the bonafide work of **Mr.J.Ganesh (0720108005)** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

25/4/09

GUIDE

(Mrs.L.S.JAYASHREE)

HEAD OF THE DEPARTMENT

(Dr.S.THANGASAMY)

The candidate with **University Register No. 0720108005** was examined by us in Project Viva-Voce examination held on 24/05/09

4/5/09

Internal Examiner

4/5/09

External Examiner

Departments of Computer Science and Engineering & Information Technology

CSI COLLEGE OF ENGINEERING

CET'09

Kotti, The Nilgiris- 643 215

THIRD NATIONAL CONFERENCE ON EMERGING TECHNOLOGIES - 2009

CERTIFICATE

This is to certify that ~~Dr./Prof.~~/Mr./~~Ms.~~ GANESH . J .....of KUMARAGURU COLLEGE OF TECHNOLOGY ..... has participated in CET '09 and presented a paper entitled ELIMINATING NOISY SENSOR DATA IN WIRELESS SENSOR NETWORKS .....

at the conference held on 11th March 2009.

Authors: Mr. J. Ganesh, Mrs. L. S. Jayashree .....

C.S.I.C.E

CO-ORDINATOR

CONVENER

PRINCIPAL

ABSTRACT

Wireless sensor networks have been widely used in many monitoring applications. Sensor data is subject to several sources of errors such as noise from external sources, hardware noise, inaccuracies, imprecision, and various environmental effects. However, it is well known that the collected sensor data are noisy. Before using collected sensor data, eliminate or completely reduce effect that brought about by noise. Otherwise the result may go wrong or incorrect. Various detection methods are to be used to eliminate the noisy data from the sensor data. The control chart technique, Rosner test, Kalman filter, Linear Regression Technique, Huber test and modified z-score Technique are to be implemented and the performance are compared with each other method. The performance graph shows that the modified z-score technique and Huber Method detects noisy data better than the other methods.

ஆய்வுச் சுருக்கம்

பல பயன்பாடுகளை மேற்பார்வை கொள்ள கம்பியில்லா உணரி பிணையத்தை பயன்படுத்துகிறோம். உணரி பிணையத்திலிருந்து பெறபட்ட தரவுகளில் பல காரணங்களால் பிழைகள் ஏற்படுகின்றன. அந்த காரணங்களாவன முறையே வெளி ஆதாரங்கள், வன்பொருளில் பிழை, தவறான கணக்கிடும் முறை, மற்றும் சுழ்நிலை அபாயங்கள். எனவே நாம் உணரி தரவுகளை பயன்படுத்துவதற்கு முன்னதாக அவற்றில் உள்ள பிழைகளை நீக்க வேண்டும். இந்த ஆய்வில் பல முறைகளை பயன்படுத்தி பிழைகளை நீக்குகிறோம். அந்த முறைகளாவன கொன்ற்றொல் சர்ட் முறை, ரொஸ்னெர் முறை, ஹுபெர் முறை, கல்மன் ஃபில்டெர், லினியர் ரெக்ரெச்சியோன் முறை மற்றும் மொடிஃபிஎட் ஃஜ்-ஸ்கொரே முறை. மேற்கூறிய முறைகளை அமுல்படுத்தி அவற்றை ஒன்றொடு ஒன்று ஒப்பிடப்படுகிறது. செயல்திறன் வரைபடத்திலிந்து ஹுபெர் முறையும் மற்றும் மொடிஃபிஎட் ஃஜ்-ஸ்கொரே முறையும் சிறப்பாக பிழைகளை நிக்குகிறது என்று இந்த ஆய்வின் மூலம் அறிகிறோம்.

ACKNOWLEDGEMENT

I express my profound gratitude to our Chairman **Padmabhusan Arutselver Dr. N. Mahalingam B.Sc, F.I.E.,** for giving this great opportunity to pursue this course.

I would like to begin by thanking to Dr.**Joseph V. Thanikal., PhD**, *principal* and prof. **R.Annamalai,.,** *Vice Principal* for providing the necessary facilities to complete my thesis.

I take this opportunity to thank **Dr. S.Thangasamy, Ph.D.,** Dean ,*Head of the Department*, Computer Science and Engineering, for his precious suggestions.

I register my hearty appreciation to **Mrs. L.S.Jayashree M.E. (PhD),** *Assoc. Professor,* my thesis advisor. I thank for her support, encouragement and ideas. I thank her for the countless hours she has spent with me, discussing everything from research to academic choices.

I thank all project committee members for their comments and advice during the reviews. Special thanks to **Mrs.V.Vanitha M.E.,** *Assistant professor*, Department of Computer science and Engineering, for arranging the brain storming project review sessions.

I would like to convey my honest thanks to all **Teaching** staff members and **Non Teaching** staffs of the department for their support. I would like to thank all my classmates who gave me a proper light moments and study breaks apart from extending some technical support whenever I needed them most.

I dedicate this project work to my **parents** for no reasons but feeling from bottom of my heart, without their love this work wouldn't possible.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

WSN	Wireless Sensor Networks
CCT	Control Chart Technique
LRT	Linear Regression Technique
ADC	Analog to Digital Converters
IQR	InterQuartile Range
ESD	Extreme Studentized Deviate
LCL	Lower Control Limit
UCL	Upper Control Limit
SEE	Standard Error of Estimation

LIST OF SYMBOLS

x	Sample data
$\bar{x}$	Mean of the given sample data
σ	Standard deviation of given sample data
n	Number of sample data
R	Test statistics value calculated in Rosner test
$\hat{\mu}$	Median
$\hat{\sigma}$	Mean Absolute Deviation
e	Error estimation
Q	Process Noise
R	Measurement Noise

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW OF WIRELESS SENSOR NETWORKS

1.1.1. Sensor network

A wireless sensor network is a collection of nodes organized in a network. Each node consists of one or more microcontrollers, CPUs or DSP chips, a memory and a RF transceiver, a power source such as batteries and accommodates various sensors and actuators. The nodes communicate wirelessly and often self-organize after being deployed in an ad hoc fashion.

1.1.2. Components of sensor networks

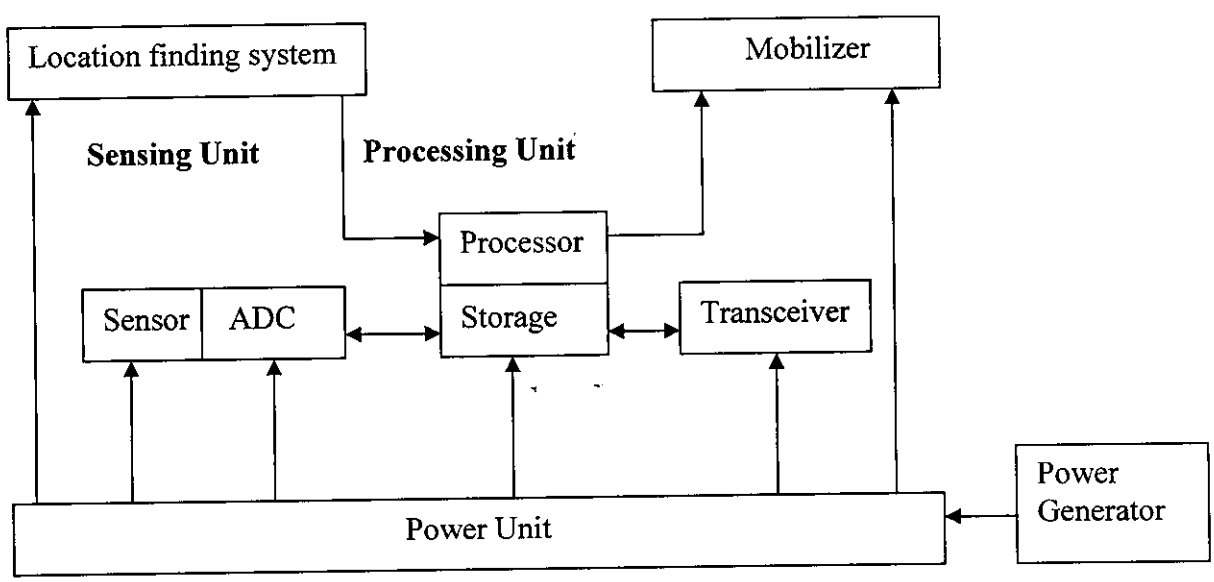

Figure 1.1.Components of sensor networks [1]

- **Sensing unit**

Sensing units are usually composed of two subunits: sensors and analog to digital converters (ADCs).The analog signals produced by the sensors are converted to digital signals by the ADC, and then fed into the processing unit.

- **Processing Unit**

The processing unit which is generally associated with a small storage unit manages the procedures that make the sensor nodes collaborate with the other nodes to carry out the assigned sensing tasks.

- **Transceiver Unit**

A transceiver unit connects the nodes to the networks.

- **Power Unit**

One of the most important components of a sensor node is the power unit. Power units may be supported by a power scanning unit such as solar cells.

1.2 APPLICATIONS OF SENSOR NETWORKS

Wireless Sensor Networks have a wide range of applications such as,

1. Military applications
    i. Monitoring friendly forces and equipment.
    ii. Battlefield surveillance.
    iii. Nuclear, biological and chemical attack detection.
2. Environmental Applications
    i. Forest fire detection.
    ii. Bio-complexity mapping of the environment.
    iii. Flood detection.
3. Health applications
    i. Tele-monitoring of human physiological data.
    ii. Tracking and monitoring doctors and patients inside a hospital.
    iii. Drug administration in hospitals.
4. Home application
    i. Home automation.
    ii. Smart environment

In order to enable reliable and efficient observation and initiate right actions, physical phenomenon features should be reliably detected/estimated from the collective information provided by sensor nodes. Moreover, instead of sending the raw data to the nodes responsible for the fusion, sensor nodes use their processing abilities to locally carry out simple computations and transmit only the required and partially processed data. Hence, these properties of WSN impose unique challenges for development of communication protocols in such architecture. The intrinsic properties of individual sensor nodes, pose additional challenges to the communication protocols in terms of energy consumption.

1.3 CHALLENGES OF WIRELESS SENSOR NETWORKS

A sensor network design is influenced by many factors, which include,

- **Energy efficiency/system lifetime**

As sensor nodes are battery-operated, protocols must be energy-efficient to maximize system life time. System life time can be measured such as the time until half of the nodes die or by application-directed metrics, such as when the network stops providing the application with the desired information about the phenomena.

- **Fault Tolerance**

Some sensor nodes may fail or be blocked due to lack of power, have physical damage or environmental interference. The failure of sensor nodes should not affect the overall task of the sensor network. This is the reliability or fault tolerance issue. Fault tolerance is the ability to sustain sensor network functionalities without any interruption due to sensor node failures.

- **Scalability**

The number of sensor nodes deployed in studying a phenomenon may be in the order of hundreds or thousands. Depending on the application, the number may reach an extreme value of millions. The new schemes must be able to work with this number of nodes.

- **Production Costs**

Since the sensor networks consist of a large number of sensor nodes, the cost of a single node is very important to justify the overall cost of the networks. If the cost of the network is more expensive than deploying traditional sensors, then the sensor network is not cost-justified. As a result, the cost of each sensor node has to be kept low.

- **Environment**

Sensor nodes are densely deployed either very close or directly inside the phenomenon to be observed. Therefore, they usually work unattended in remote geographic areas. They may be working in busy intersections, in the interior of large machinery, at the bottom of an ocean, inside a twister, on the surface of an ocean. They work under high pressure in the bottom of an ocean, in harsh environments such as debris or a battlefield, under extreme heat and cold such as in the nozzle of an aircraft engine or in arctic regions, and in an extremely noisy environment such as under intentional jamming.

- **Hardware Constraints**

A sensor node is made up of four basic components: a sensing unit, a processing unit, a transceiver unit and a power unit. Sensing units are usually composed of two subunits: sensors and analog to digital converters (ADCs). The analog signals produced by the sensors based on the observed phenomenon are converted to digital signals by the ADC, and then fed into the processing unit. The processing unit, which is generally associated with a small storage unit, manages the procedures that enable the sensor node collaborate with the other nodes to carry out the assigned sensing tasks. A transceiver unit connects the node to the network. One of the most important components of a sensor node is the power unit. Power units may be supported by a power scavenging unit such as solar cells. There are also other subunits, which are application dependent. Most of the sensor network routing techniques and sensing tasks require the knowledge of location with high accuracy.

- **Sensor Network Topology**

Sheer numbers of inaccessible and unattended sensor nodes, which are prone to frequent failures, make topology maintenance a challenging task. Hundreds to several thousands of nodes are deployed throughout the sensor field.

- **Pre-Deployment Phase**

Sensor nodes can be either thrown in mass or placed one by one in the sensor field. They can be deployed by dropping from a plane, delivering in an artillery shell,

rocket or missile, throwing by a catapult , placing in factory, and placing one by one either by a human or a robot. Although the sheer number of sensors and their unattended deployment usually preclude placing them according to a carefully engineered deployment plan, the schemes for initial deployment must reduce the installation cost, eliminate the need for any pre-organization and preplanning, increase the flexibility of arrangement, and promote self-organization and fault tolerance.

- **Post-Deployment Phase**

After deployment, topology changes are due to change in sensor nodes position, reach ability (due to jamming, noise, moving obstacles, etc.), available energy, malfunctioning, and task details. Sensor nodes may be statically deployed. However, device failure is a regular or common event due to energy depletion or destruction. It is also possible to have sensor networks with highly mobile nodes. Besides, sensor nodes and the network experience varying task dynamics, and they may be a target for deliberate jamming. Therefore, sensor network topologies are prone to frequent changes after deployment.

- **Re-Deployment of Additional Nodes Phase**

Additional sensor nodes can be re-deployed at any time to replace the malfunctioning nodes or due to changes in task dynamics. Addition of new nodes poses a need to re-organize the network. Coping with frequent topology changes in an ad hoc network that has myriads of nodes and very stringent power consumption constraints requires special routing protocols.

- **Transmission Media**

In a multi-hop sensor network, communicating nodes are linked by a wireless medium. These links can be formed by radio, infrared or optical media. To enable global operation of these networks, the chosen transmission medium must be available worldwide. One option for radio links is the use of *Industrial, Scientific and Medical* (ISM) bands, which offer license free communication in most countries.

- **Power Consumption**

The wireless sensor node, being a microelectronic device, can only be equipped with a limited power source. In some application scenarios, replenishment of power resources might be impossible. Sensor node lifetime, therefore, shows a strong dependence on battery lifetime. The malfunctioning of few nodes can cause significant topological changes and might require rerouting of packets and re-organization of the network. Hence, power conservation and power management take on additional importance. It is for these reasons that researchers are currently focusing on the design of power aware protocols and algorithms for sensor networks. In sensor networks, power efficiency is an important performance metric, directly influencing the network lifetime. Application specific protocols can be designed by appropriately trading off other performance metrics such as delay and throughput with power efficiency. The main task of a sensor node in a sensor field is to detect events, perform quick local data processing, and then transmit the data. Power consumption can hence be divided into three domains: *sensing*, *communication*, and *data processing*.

CHAPTER 2

LITERATURE REVIEW

2.1 Cleaning and Querying Noisy Sensors [3].

Sensor data is subject to several sources of errors such as noise from external sources, hardware noise, inaccuracies and imprecision, and various environmental effects. Noise cleaning performed two side such as sensor side and sink side. This paper focus on noise cleaning performed either sink side or sensor side.

Noise cleaning performed by using Bayesian approach, which combines prior knowledge of the true sensor readings, the noise characteristics of this sensor and the observed noisy readings. However, it is often not possible or at least difficult to know the noise characteristics of a sensor.

2.2 Model Based error correction for wireless sensor networks [6].

This paper eliminating noise data by using model based approach. The model approach to correct the transient errors of sensor data. Specifically, an off-line process selects the type and order of the models first and then, based on the collected sampling sensor data, the parameters of this model are estimated to correct the data.

2.3 Declarative support for sensor data cleaning [7].

This paper proposed a general framework for building sensor data cleaning infrastructures in pervasive applications. The main idea is to utilize the spatial-temporal correlation among the sensor data to recover lost readings and remove outliers.

2.4 In-network outlier cleaning for data collection in sensor networks [9].

A sensor network is equipped with thousands of in-expensive, low fidelity motes, which can easily generate sensing errors. The abnormal unreal sensor readings generated in a temporally or permanently failed sensor is called outliers.

The limited battery power and costly data transmission have introduced a new challenge for outlier cleaning in a sensor network: it must be done in network to avoid spending energy on transmitting outliers. In this paper, we propose an in network outlier cleaning approach, including wavelet based outlier correction and neighboring

DTW (Dynamic Time Warping) distance-based outlier removal. The noise cleaning can be performed at sensor side only.

2.5 Simple Moving Average approach [5].

Simple moving average of period (x) is series of successive averages (arithmetic mean) of m terms at time.

Starting with 1st, 2nd, 3rd terms, etc.

For example

The first average mean of 1st to m term

$$\text{Average} = \frac{1+2+3+\ldots+x}{x} \qquad \text{-------- 2.1}$$

The second average is mean of 2nd to m+1 term

$$\text{Average} = \frac{2+3+4+\ldots+(x+1)}{x} \qquad \text{-------- 2.2}$$

And so on.

If x is odd (i.e.) 2k+1, say moving average is placed against mid value of the time interval.

If m is even, then moving average is placed between x=k and x=k+1.

The SMA is completely eliminates the oscillatory movements affecting the series.

The fluctuations are regular and periodic then the SMA completely eliminates the oscillatory movement.

Moving average method is very flexible in the sense, the addition of a few more figures to the data. Simply results in some more trend value.

The previous calculations are not affected.

The popular approach to remove noise in random samples and compute the monitoring values is to use a moving average. Unlike a moving average that is usually used for a one-dimensional time series, moving average in sensor networks has two dimensions. Sensor data 0are averaged temporally within one sensor, and also spatially among neighboring sensors.

The SMA algorithm for sensor networks is as follows

At any time t, the algorithm first averages a sequence of samples $x_{i,t-k+1},\ldots,x_{i,t}$ at each sensor i, and gets

$$\overline{X}_i = (x_{i,t-k+1}+\ldots+x_{i,t})/k. \qquad \text{--------2.3}$$

It then averages values of neighboring sensors,

$$\overline{X}_j = \Sigma_{j\varepsilon R(i)} x_j / \left|R(i)\right|, \qquad \text{--------2.4}$$

Where R(i) is a set of neighboring sensors of sensor *i*.

SMA is not suitable for sensor network applications, when taking into account the two important evaluation criteria: energy efficiency and query response time. In the SMA method, both criteria cannot be met at the same time. In order to improve energy efficiency, sampling rates should be lowered (i.e. the interval between two consecutive samples are lengthened). The consequence of low sampling rates is that it takes a long time to reflect a change in the moving average. On the other hand, if the sampling rates are high, the response to a change can be quick. However, more samples need to be taken and we know that sampling is one of the costly operations in sensors.

CHAPTER 3

TECHNIQUES FOR IDENTIFYING THE NOISY SENSOR DATA

3.1. Boxplot test [12].

A boxplot also gives a picture of the symmetry of a dataset, and shows outliers very clearly. Both of these features are important when deciding which summary statistics would best describe the dataset. A condition of many hypothesis tests is that the data is approximately normally distributed and a boxplot can assist in determining this. Prior to conducting a hypothesis test, a statistician looks at the data, and histograms and a boxplot would be the displays most often chosen.

The boxplot rule is a visual test to inspect for outliers, see Figure 5 for example of a box plot. The interquartile range is included into a box and the 5% and 95% confidence intervals are indicated with error bars outside of the box. Values that lie outside of the confidence interval are possible outliers.

Steps for boxplot rule

Step 1:

Calculate the median of sample data

Step 2:

First the arrange the sample data in ascending order and Split the data value into two half way,

If the number of data is even, split the data into two half.

If the number of data is odd, split the data into two half with median value.

Step 3:

Find the median of each half.

Step 4:

Find the difference between the two median.

3.1.1 How to Draw a Boxplot

There is a commonly accepted method of drawing the whiskers on a boxplot. However, there is a plethora of methods for drawing the box. Some of these methods

were developed because they extend nicely to percentiles other than 25% and 75%, others were chosen because of theoretical considerations and some were developed for simplicity. Bob Hayden chose his method partly for metaphysical reasons! In this paper I will opt for simplicity, with a bit of metaphysics thrown in for good measure.

Drawing the Box

We have to start with the box, as constructing the whiskers requires we use the box to determine if any data values are outliers. I will describe two methods, both simple, from which you can choose. The first is historically correct and has Bob's nice metaphysical property, while the second is the method used by the TI-82 and TI-83 graphing calculator. If your students use this technology and you want them to get the same answers as their calculator then you should opt for this method.

Tukey's Method

The method recommended by Tukey, who invented the boxplot, is as follows:

Find the median. Then find the median of the data values whose ranks are LESS THAN OR EQUAL TO the rank of the median. This will be a data value or it will be half way between two data values.

With a dataset with an odd number of values, include the median in each of the two halves of the dataset and then find the median of each half. This gives the first and third quartiles. If the dataset has an even number of values, just split the data into two halves, and find the median of each half.

Here is an example using a small dataset, which contains an *odd* number of values:

35 47 48 50 51 53 54 70 75

Split the data into two halves, each including the median:

35 47 48 50 51 and 51 53 54 70 75

Find the median of each half. In this example, the first quartile is 48 and the third quartile is 54. Hence the interquartile range is 54-48 = 6.

I'll add a number to the above dataset to illustrate how to find the quartiles for an even number of values (what the heck, the data is bogus anyway):

35 47 48 50 51 53 54 60 70 75

Split the data into two halves:

35 47 48 50 51 and 53 54 60 70 75

Now find the median of each half. In this example, the first quartile is 48 and the third quartile is 60. Hence the IQR is 60-48 = 12.

For example, take the dataset 1 4 78 81 345. The minimum is 1, the maximum is 345, and the median is 78. Splitting the dataset into two halves each containing the median gives Q1 as 4 and Q3 as 81. This is the metaphysical property that he has noted.

Alternative Method

Calculate the median of the sample data. Then find the median of the data values whose ranks are LESS THAN the rank of the median. This will be a data value or it will be half way between two data values.

Here is the same example using the above numbers, which contains an *odd* number of values:

35 47 48 50 51 53 54 70 75

Split the data into two halves, *not* including the median:

35 47 48 50 and 53 54 70 75

Calculate the median of each half. In this example, the first quartile is 47.5 and the third quartile is 62. Hence the interquartile range is 62-47.5 = 14.5.

Note the difference in the answers between the two methods! It is not really surprising when you consider that we are doing a 5 number summary on a set of only 9 numbers.

Drawing the Whiskers

The maximum length of each whisker is 1.5 times the interquartile range (IQR). To draw the whisker above the 3rd quartile, draw it to the largest data value that is less than or equal to the value that is 1.5 IQRs above the 3rd quartile. Any data value larger than that should be marked as an outlier. Some statisticians differentiate between 'mild' outliers and 'severe' outliers. Mild outliers lie between 1.5 and 3 IQRs above the 3rd quartile while severe outliers are more than 3 IQRs above the 3rd quartile (Q3). I don't believe we need to make the distinction between mild and severe outliers in our courses.

Here is an example, using the first set of numbers above using Tukey's method of determining Q1 and Q3:

35 47 48 50 51 53 54 70 75

The IQR is 6. Now 1.5 times 6 equals 9. This is the maximum length of the whisker. Subtract 9 from the first quartile: 48 - 9 = 39. Note that 35 is an outlier, and the whisker should be drawn to 47, which is the smallest value that is not an outlier.

Add 9 to the third quartile: 54 + 9 = 63. Any value larger than 63 is an outlier, so in this instance both 70 and 75 are outliers. Draw the whisker to the largest value in the dataset that is not an outlier, in this case 54. Since this value *is* the 3rd quartile, we draw no whisker at all! Mark 70 and 75 as outliers. The boxplot is given below:

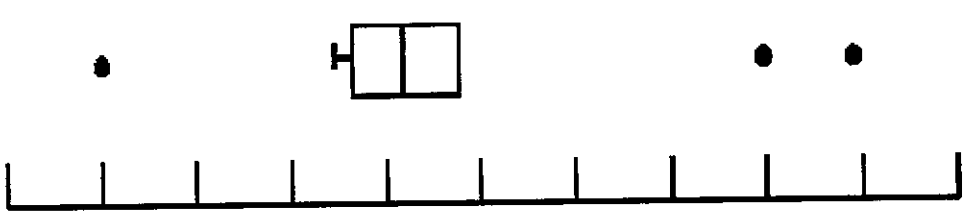

Fig 3.1 Boxplot Technique with smaller limit.

If we use the alternative method with the IQR of 14.5 then we have no outliers, and the boxplot looks like this:

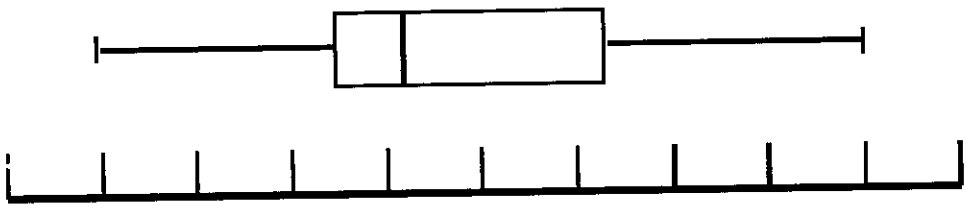

Fig 3.2 Boxplot Technique with larger limit.

3.2 Dixon test [11].

Dixon's test is generally used for detecting a small number of outliers. This test can be used when the sample size is between 3 and 25 observations. The data is ranked in ascending order, and then based on the sample size; the tau statistic for the highest value or lowest value is computed.

Observations	Highest value suspect	Lowest value suspect
3 to 7	$\tau = \frac{X_n - X_{n-1}}{X_n - X_1}$	$\tau = \frac{X_2 - X_1}{X_n - X_1}$
8 to 10	$\tau = \frac{X_n - X_{n-1}}{X_n - X_2}$	$\tau = \frac{X_2 - X_1}{X_{n-1} - X_1}$
11 to 13	$\tau = \frac{X_n - X_{n-2}}{X_n - X_2}$	$\tau = \frac{X_3 - X_1}{X_{n-1} - X_1}$
14 to 20-30	$\tau = \frac{X_n - X_{n-2}}{X_n - X_3}$	$\tau = \frac{X_3 - X_1}{X_{n-2} - X_1}$

Table 3.2.1 Dixon method formula.

The tau statistic is compared to a critical value at a chosen value of alpha. If the tau statistic is less than the critical value, the null hypothesis is not rejected, and the conclusion is that no outliers are present. If the tau statistic is greater than the critical value, the null hypothesis is rejected, and the conclusion is the most extreme value is an outlier. To check for other outliers, the Dixon test can be repeated

Drawback

- The power of this test decreases as the number of repetitions increases.
- This test can be used when the sample size is between 3 and 25 observations.
- Small number of sample will reduce the accuracy of identifying the outliers.

3.3 GRUBB'S TEST [11]

Statisticians have devised several ways to detect outliers. Grubbs' test is particularly easy to follow. This method is also called the ESD method (extreme studentized deviate).

The first step is to quantify how far the outlier is from the others? Calculate the ratio Z as the difference between the outlier and the mean divided by the SD. If Z is large, the value is far from the others. Note that you calculate the mean and SD from all values, including the outlier.

$$z = \frac{|mean - value|}{SD} \qquad \text{--------3.1}$$

Since 5% of the values in a Gaussian population are more than 1.96 standard deviations from the mean, your first thought might be to conclude that the outlier comes from a different population if Z is greater than 1.96. This approach only works if you know the *population* mean and SD from other data. Although this is rarely the case in experimental science, it is often the case in quality control. You know the overall mean and SD from historical data, and want to know whether the latest value matches the others. This is the basis for quality control charts.

When analyzing experimental data, you don't know the SD of the population. Instead, you calculate the SD from the data. The presence of an outlier increases the calculated SD. Since the presence of an outlier increases both the numerator (difference between the value and the mean) and denominator (SD of all values), Z does not get very large. In fact, no matter how the data are distributed, Z cannot get larger than $N-1/\sqrt{N}$,

where N is the number of values. For example, if N=3, Z cannot be larger than 1.555 for any set of values.

If your calculated value of Z is greater than the critical value in the table, then the P value is less than 0.05. This means that there is less than a 5% chance that you'd encounter an outlier so far from the others (in either direction) by chance alone, if all the data were really sampled from a single Gaussian distribution. Note that the method only works for testing the most extreme value in the sample (if in doubt, calculate Z for all values, but only calculate a P value for Grubbs' test from the largest value of Z.

Once you've identified an outlier, you may choose to exclude that value from your analyses. Or you may choose to keep the outlier, but use robust analysis techniques that do not assume that data are sampled from Gaussian populations.

Steps involved in Grubbs' test

Step 1:

The EPA suggests taking the logarithms of sensor data, which are often log normally distributed.

Step 2:

The sensor data are ranked in ascending order and the mean and standard deviation are calculated

Step 3:

The lowest or highest data point can be tested as an outlier.

The τ statistics fro largest value $\tau = \frac{x_i - \bar{x}}{s}$ --------3.2

The τ statistics for smallest value $\tau = \frac{\bar{x} - x_i}{s}$ --------3.3

The τ statistics is compared with a critical value for sample size and selected

If τ > critical value (the critical value is get from t z test table)

The null hypothesis is rejected and the conclusion is that the datum under consideration is an outlier.

3.4 CONTROL CHART TECHNIQUE [10].

In this section, we study control chart technique for outlier data detection. Usually, CCT is used to determine whether your process is operating in statistical control. The purpose of a control chart is to detect any unwanted changes in the process. These changes will be signaled by abnormal (outlier) points on the graph. Basically, control chart consists of three basic components:

- A centre line, usually the mathematical average of all the samples plotted.
- Upper and lower control limits that define the constraints of common cause variations.
- Performance data plotted over time.

Steps involved in control chart technique

Step 1:

Calculate the average for sample data to get a centerline of a control chart, the formula is

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \qquad \text{--------3.4}$$

Where $\bar{x}$ =mean/average value of sample data.

x_i =every data value ($x_1 ... x_n$)

n=total number of data, which is 100.

Step 2:

Calculate the standard deviation of sample data

$$\sigma = \left[\frac{\sum (x_i - \bar{x})^2}{n-1} \right]^{1/2} \qquad \text{--------3.5}$$

Where σ =standard deviation.

Step 3:

Calculate the upper control (UCL) and lower Control Limit (LCL) by using formula below is,

$$UCL=\bar{x}+Z\sigma_x \qquad \text{--------3.6}$$

$$LCL=\bar{x}-Z\sigma_x \qquad \text{--------3.7}$$

$$\text{Where } \sigma_x=\frac{\sigma}{\sqrt{n}}$$

Step 4:

If the sample data x_i falls between the Lower Control Limit (LCL) and Upper Control Limit (UCL), then the sample data x_i is not outlier.

If the sample data x_i falls outside the Lower Control Limit (LCL) and Upper Control Limit (UCL), then the sample data x_i is outlier.

In a 3-sigma system, Z is equal to 3. The reason that 3-sigma control limits balance the risk of error is that, for normally distributed data, data points will fall inside 3-sigma limits 99.7% of the time when a process is in control. This makes the witch hunts infrequent but still makes it likely that unusual causes of variation will be detected.

Finally, data are plotted on the chart and data that are out from UCL and LCL and are detected as outlier data. Figure 1 shows an example of control chart that has one data outside UCL. This data is known as outlier data.

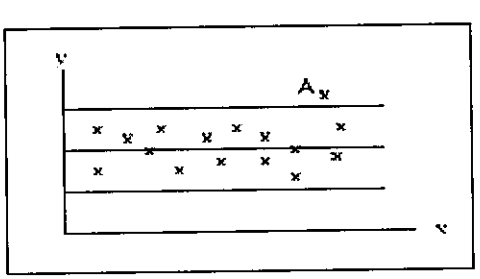

Fig 3.3 Control Chart Technique

3.5 ROSNER TEST [11].

Rosner Test for detecting up to k outliers can be used when the number of data points is 25 or more. This test identifies outliers that are both high and low; it is therefore always two-tailed. The data are ranked in ascending order and the mean and standard deviation are determined. The procedure entails removing from the data set the observation, x, that is farthest from the mean. Then a test statistic, R, is calculated

$$R_{i+1} = \frac{|x_i - \bar{x}|}{s} \qquad \text{--------3.8}$$

The R statistic is then compared with a critical value. The null hypothesis, stating that the data fits a normal distribution, is then tested.

If R is less than the critical value, the null hypothesis cannot be rejected, and therefore there are no outliers.

If R is greater than the critical value, the null hypothesis is rejected and the presence of k outliers is accepted. This test can also be used with log-normally distributed data, when the logarithms of the data are used for computation.

3.6 HUBER'S METHOD [11].

Huber method makes more use of the information provided by the data.

In this method, we progressively transform the original data by a process called winsortism. The Huber method consist of following steps

Step 1:

Calculate the $\hat{\mu}, \hat{\sigma}$ (median, MAD estimates) of the sample data.

MAD is the difference between actual data and median of the sample data of given, and take the median of that difference. The MAD formula is

Median $(x_i - \hat{\mu})$

Step 2:

Calculate the lower limit and upper limit

$$\text{LOW} = \hat{\mu} - 3 * \hat{\sigma} \qquad \text{--------3.8}$$

$$\text{UP} = \hat{\mu} + 3 * \hat{\sigma} \qquad \text{--------3.9}$$

Step 3:

If the sample data x_i falls between the LOW and UP, then the sample data x_i is not outlier, otherwise the sample data is outlier.

3.7 LINEAR REGRESSION TECHNIQUE (LRT) [10].

The linear regression analysis is concerned with the derivation of an equation which defines a best fitting regression line. The principles of regression analysis are explained here in the context of straight lines involving only two variables. But the principles are readily extended to cases involving non-linear relations and to cases involving more than two variables.

Equation of a straight line

Y=A+B*x.

It is conventional to assume that y varies as x varies rather than the other way round i.e. that depends on x. hence, y is known as the dependent variable and x as the independent (or explanatory) variable. The interpretation of this equation is that, as x changes, y changes by b times the change in x and therefore b measures the slope or gradient of the line. It follows that when x takes the value of zero, y=A (a is referred to as the intercept). Given the equation for a straight, it is possible to predict values of y for any given values of x.

Deriving a best fitting regression line.

The best fitting regression line is determined using the criterion of least squares. Consider the fig. This shows a scatter of points indicating the expenditure y of eight households in relation to their incomes x and regression line fitted to these data denoted by

Y=A +B*x. --------3.10

Where y stands for the value of variable y computed from the relationship for a given value of the variable x (y is therefore distinguished from y which represented actual, observed values). The line y=A + B*x express the average relationship between the two variables. It is called the linear regression of y on x and the slope coefficient b is referred to as the regression coefficient. The problem remaining is to determine the values of A and B. this will then give us the equation for the best fitting line.

Principles of the least square method.

Minimize $\sum (y_i - y)^2 \equiv$ minimize $\sum e_i^2$ --------3.11

Where y_i is the observed value of the dependent variable for the i^{th} observation and y is the computed value of the dependent variable for the i^{th} observation.

The procedure for finding the values of A and B from first principles, using the method of least squares, involves differential calculus. Fortunately, its demonstration is not essential to understanding the principles of regression analysis and hence it is not included here, it leads to the following expressions for determining the values of A and B.

Formulae for regression coefficients

$$B = \frac{\sum_{i=1}^{n} x_i * y_i - n * \bar{x} * \bar{y}}{\sum_{i=1}^{n} x_i^2 - n * \bar{x}^2} \qquad \text{--------3.12}$$

$$A = \bar{y} - b * \bar{x} \qquad \text{--------3.13}$$

Once a and b have been determined, we can then express the equation for the best fitting line, i.e. y=A + B*x. note that the coefficient A and B can have negative as well as positive values. A negative value for b indicates an inverse relationship between x and y, so that y decreases as x increases and vice versa. A negative value for A indicates, of course, a negative intercept on the y axis. The application of these formulae is simple, especially for small data sets, merely involving the summation of x_i^2 , $x_i * y_i$ and calculation of $\bar{x}$ and $\bar{y}$.

Testing the statistical significance of the regression model.

The calculation of the regression coefficients A and B from a sample of data naturally provides only estimates of the corresponding intercept and slope coefficients for the population from which the sample has been drawn (we denote these population coefficients by A and B respectively). As with the calculation of other statistics from samples, these are generally only of interest as estimates of the corresponding population parameters. In the case of regression, the population coefficients are. Of course, unknown and the sample coefficients represent only one result amongst a number of possible results that could have been obtained depending on the sample. In

other words, the sample coefficients are subject to sampling error. Thus, the fact that a sample regression coefficient is calculated as having a positive or a negative value does not necessarily indicate that the corresponding population parameter is also positive or negative.

A significance test is required to rest the probability of obtaining such positive or negative (i.e. non-zero) sample coefficients even though the population parameters are really zero. Most commonly, we are concerned with testing the null hypothesis H_0:B=0, i.e. that the slope of the population regression line is zero. Acceptance of this hypothesis means that the population regression line is a horizontal line so that the value of y does not vary as x varies. In this case, information about x would be of no value in helping us to predict values of y. on the other hand. If we accept one of the possible alternative hypothesis that we may wish to test (e.g. $B \neq 0$, $B>0$ or $B<0$), then the population regression line must slope upwards (or downwards) such that information about x will help us to predict values of y.

To test these hypotheses we must turn to the only information available, namely the sample regression equation, and test H_0 against the sample result. It is possible to test both A and B, but we focus only on B here as this is normally of most interest. The test is therefore concerned with comparing the sample coefficient b with the hypothesized value of the population parameter.

Standard error of B

$$SEE=\sqrt{\left(\frac{\sum e_i^2}{n-2}\right)}=\sqrt{\left(\frac{\sum y_i^2 - A*\sum y_i - B*\sum x_i * y_i}{n-2}\right)} \qquad \text{--------3.14}$$

Where SEE denotes the Standard Error of Estimation

Estimation and Prediction using the regression model.

The regression equation can be used, of course, to predict values of the dependent variable y for given values of x. it will be appreciated, however, that such single point estimates are subject to error and, as with estimates of population means, it is possible to construct an interval around such estimates to provide a given degree of confidence for the prediction, example 95percent confidence that the true population value would lie within the specified interval.

There are two situations to consider

- Interval estimation of the average value of y for a given value of x.
- Interval estimation of a single value of y for a given value of x.

The value of y may refer to the mean expenditure of a group of households or the expenditure of an individual household. In both cases the point estimate of y (i.e. y=A + B*x) would be the same but an interval estimate of y would differ in each case. This is because the variation for a mean value is less than that for individual values within the group. These interval are referred to respectively as

- Confidence intervals
- Prediction intervals

The regression model to estimate average values or to predict individual values of the dependent variable y with specified levels of confidence.

Confidence interval

To construct a 100(1-α) percent confidence interval for y for a given value of x_0, the following formulae can be used

$$\text{LOW}= y - cv * \left\{ SEE * \sqrt{\left[\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum (x_i - \bar{x})^2} \right]} \right\} \quad \text{--------3.15}$$

$$\text{UP}= y + cv * \left\{ SEE * \sqrt{\left[\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum (x_i - \bar{x})^2} \right]} \right\} \quad \text{--------3.16}$$

Where y=a + b* x_0 and cv has n-2 degrees of freedom

Prediction Interval

To construct a 100(1- α) percent prediction interval for the estimated individual value of y for a given value x_0. The following formulae can be used for calculate the LOW and UP limit.

$$\text{LOW}= y - cv * \left\{ SEE * \sqrt{1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum (x_i - \bar{x})^2}} \right\} \quad \text{--------3.17}$$

$$\text{UP}= y + cv * \left\{ SEE * \sqrt{1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum (x_i - \bar{x})^2}} \right\} \quad \text{--------3.18}$$

3.8 MODIFIED Z SCORE TECHNIQUE [4].

To detect the presence of outlier values in the observed dataset, the statistical test called modified Z-Score technique. In a modified Z-score test, the z-score is determined based on determined based an outlier resistant estimators. The median of absolute deviation about the median (MAD) in such an estimator.

The algorithm is given below:

Step 1:

Calculate the sample median (x_m).

Step 2:

Calculate the absolute value of the difference between the observations and the median $|x_i - x_m|$.

Step 3:

Calculate the Median of the Absolute Deviation (MAD) about the sample median.

Step 4:

Calculate the modified Z-Score for each observation, where $z_i=0.6745*(x_i-x_m)/MAD$.

Step 5:

An observation is labeled an outlier when $|z_i|$ is greater than 3.5.

A high modified Z-score indicates that an observation is more likely to be potential discordant outlier. The constant 0.6745 is needed because E (MAD) $=0.6745*\sigma$ for large n. An often- used rule for labeling potential discordant outliers is when $|z_i| > 3.5$.

3.9 KALMAN FILTER [8].

The Kalman filter has long been regarded as the optimal solution to many tracking and data prediction tasks, Its use in the analysis of visual motion has been documented frequently. The standard Kalman filter derivation is given here as a tutorial exercise in the practical use of some of the statistical techniques out lied in previous sections. The filter is constructed as a mean squared error minimiser, but an alternative derivation of the filter is also provided showing how the filter relates to

maximum likelihood statistics. Documenting this derivation furnishes the reader with further insight into the statistical constructs within the filter.

The purpose of filtering is to extract the required information from a signal, ignoring everything else. How well a filter performs this task can be measured using a cost or loss function. The goal of the filter to be the minimization of this loss function.

Kalman filter to remove noise from a signal, the process that we are measuring must be able to be described by a linear system. Many physical processes, such as a vehicle driving along a road, a satellite orbiting the earth, a motor shaft driven by winding currents, or a sinusoidal radio-frequency carrier signal, can be approximated as linear systems. A linear system is simply a process that can be described by the following two equations:

The system model can be represented as below

$$x_t = A * x_t + Q \qquad \text{--------3.19}$$

The measurement model can be represented as below

$$z_t = H * x_t + R \qquad \text{--------3.20}$$

A and H are model parameters, which is constant value. The variable Q is called the process noise, and R is called the measurement noise. Each of these quantities are (in general) vectors and therefore contain more than one element. The vector x contains all of the information about the present state of the system, but we cannot measure x directly. Instead we measure y, which is a function of x that is corrupted by the noise. We can use z to help us obtain an estimate of x, but we cannot necessarily take the information from z at face value because it is corrupted by noise. The measurement is like a politician. We can use the information that it presents to a certain extent, but we cannot afford to grant it our total trust.

CHAPTER 4

RESULTS AND DISCUSSIONS

The various outlier detection techniques are implemented. The result of each technique compared with one other technique.

The experiments are performed on a Pentium 2.4 GHz with 128MB of RAM on Windows XP professional edition using the MATLAB software.

The sample data can be taken from the Intel Berkley Lab data [12].

4.1. Control chart Technique.

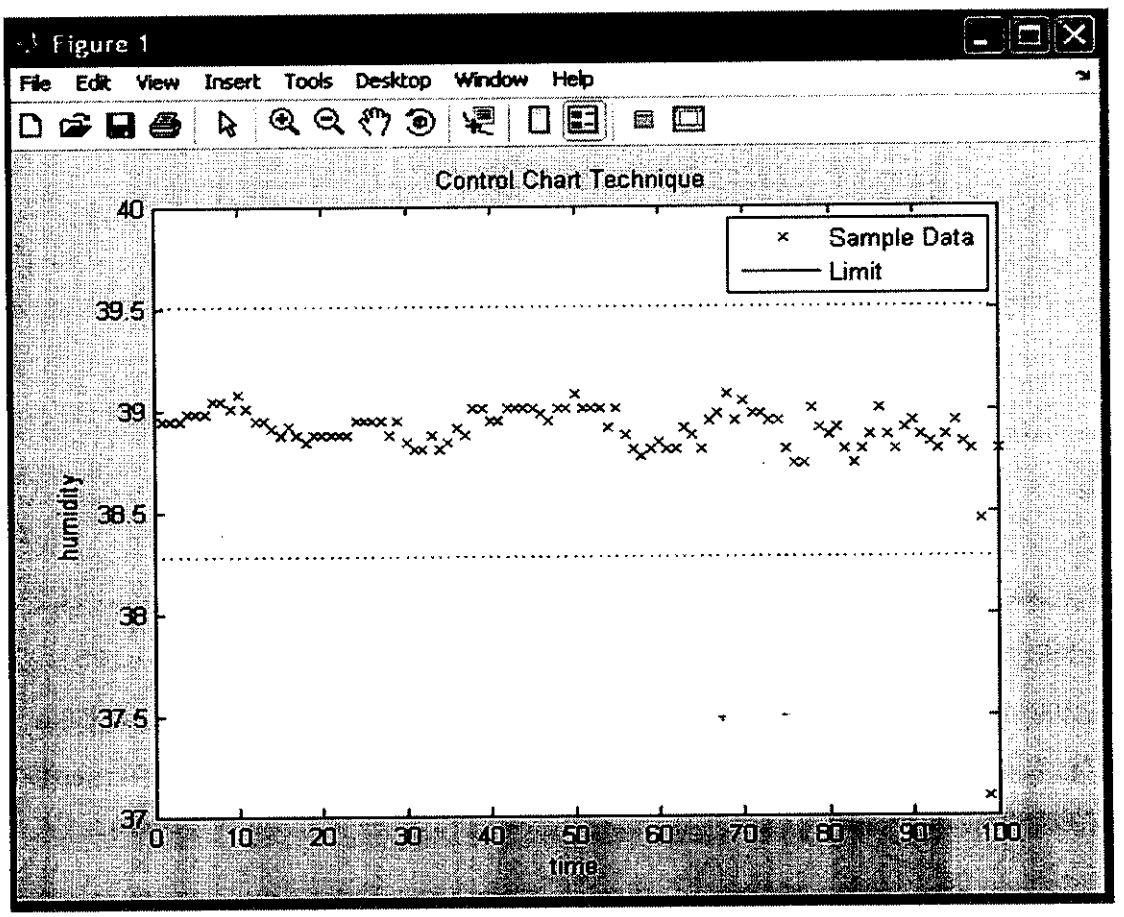

Fig 4.1.1 Control Chart Technique without outliers.

There are no outlier values introduced in original data. The Fig 4.1.1 shows that all values are original values.

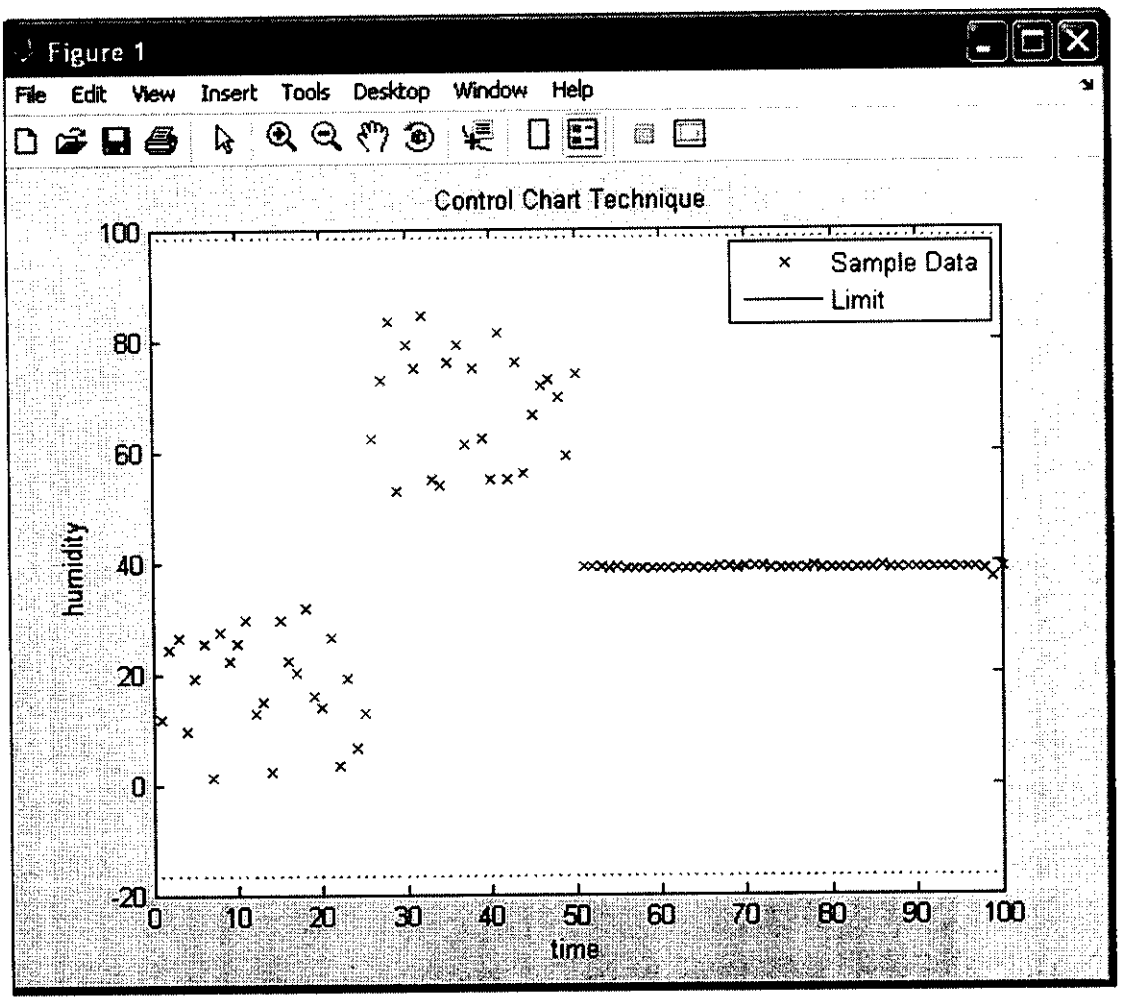

Fig 4.1.2 Control Chart Technique with outliers.

50 noisy data are introduced in the original data. CCT did not predict any outlier data.

Control chart technique

No of outlier Data	No of Outlier detected		Percentage	Percentage
	Range [20 30] [50 60]	Range [0 30] [50 80]		
4	3	3	75	75
10	4	5	40	50
14	5	4	35.71	28.57
20	2	3	10	15
26	Nil	Nil	0	0
32	Nil	Nil	0	0
38	Nil	Nil	0	0
44	Nil	Nil	0	0
50	Nil	Nil	0	0
56	Nil	Nil	0	0
64	Nil	Nil	0	0
70	Nil	Nil	0	0
76	Nil	Nil	0	0
80	Nil	Nil	0	0
84	Nil	Nil	0	0
90	Nil	Nil	0	0
100	Nil	Nil	0	0

Table 4.1.1 Performance of the Control Chart Technique

The different numbers of outlier values are introduced in the original data. The performance is tabulated.

4.2 Rosner Method

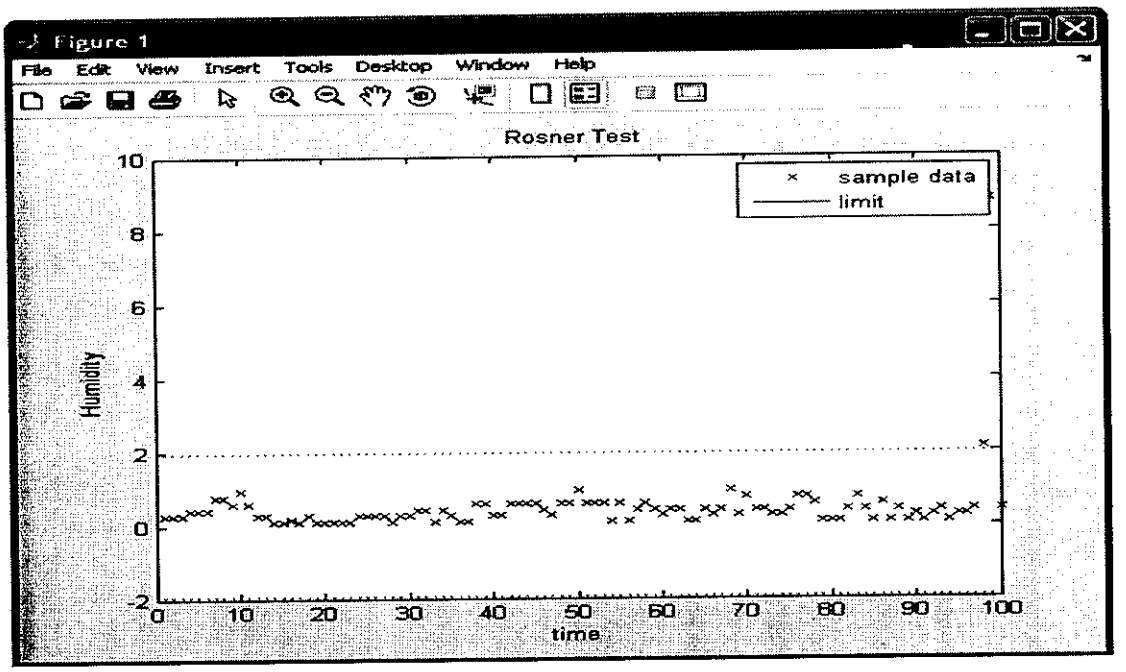

Fig 4.2.1 Rosner Method without outliers.

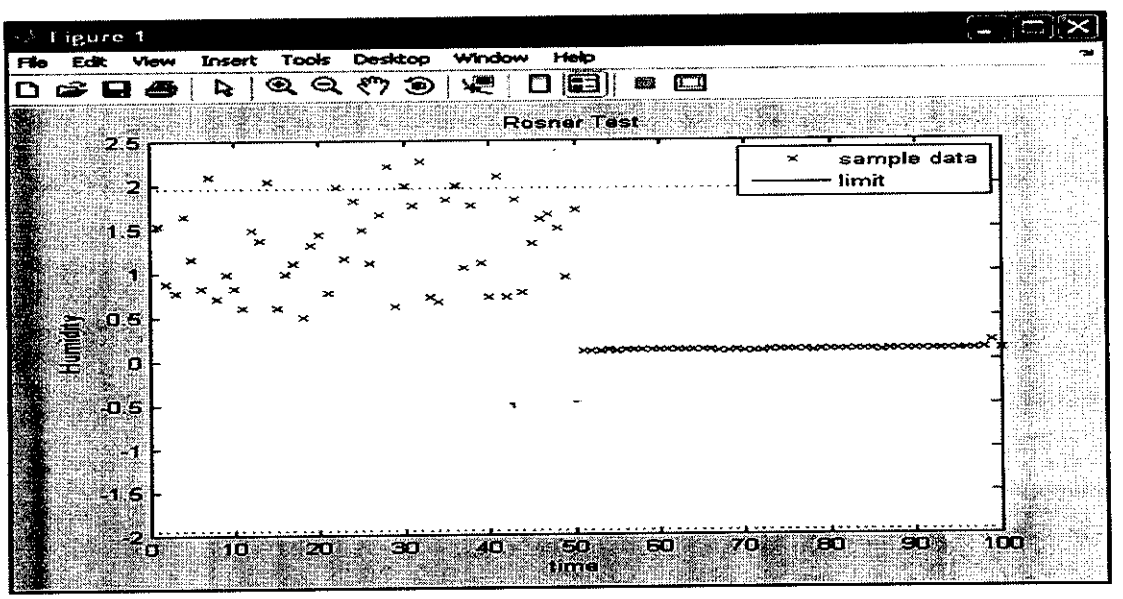

Fig 4.2.2 Rosner Method with outliers.

There are no outlier values introduced in original data. The Fig 4.2.1 shows that all values are original values.

50 noisy data are introduced in the original data. Rosner test identifies few number outlier values, that can be shown in Fig 4.2.2.

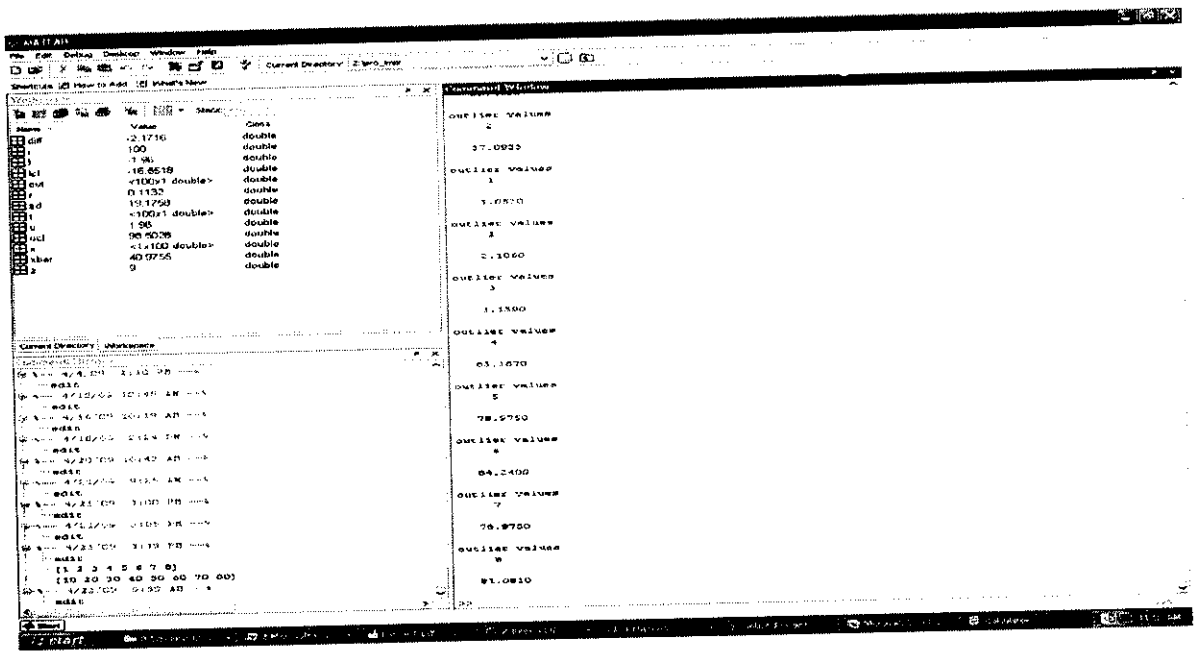

Fig 4.2.3 Rosner Method.

No of outlier Data	No. of Outlier detected		Percentage	Percentage
	Range [20 30] [50 60]	Range [0 30] [50 80]		
4	4	4	100	100
10	9	7	90	70
14	10	10	71.43	71.43
20	11	10	55	50
26	14	13	53.85	50
32	8	8	25	25
38	8	8	21.05	21.05
44	Nil	Nil	0	0
50	Nil	3	0	6
56	Nil	Nil	0	0
64	Nil	Nil	0	0
70	Nil	Nil	0	0
76	Nil	Nil	0	0
80	Nil	Nil	0	0
84	Nil	Nil	0	0
90	Nil	Nil	0	0
100	Nil	Nil	0	0

Table 4.2.1 Performance of Rosner Method

The different numbers of outlier values are introduced in the original data. The performance is tabulated.

4.3 Kalman Filter

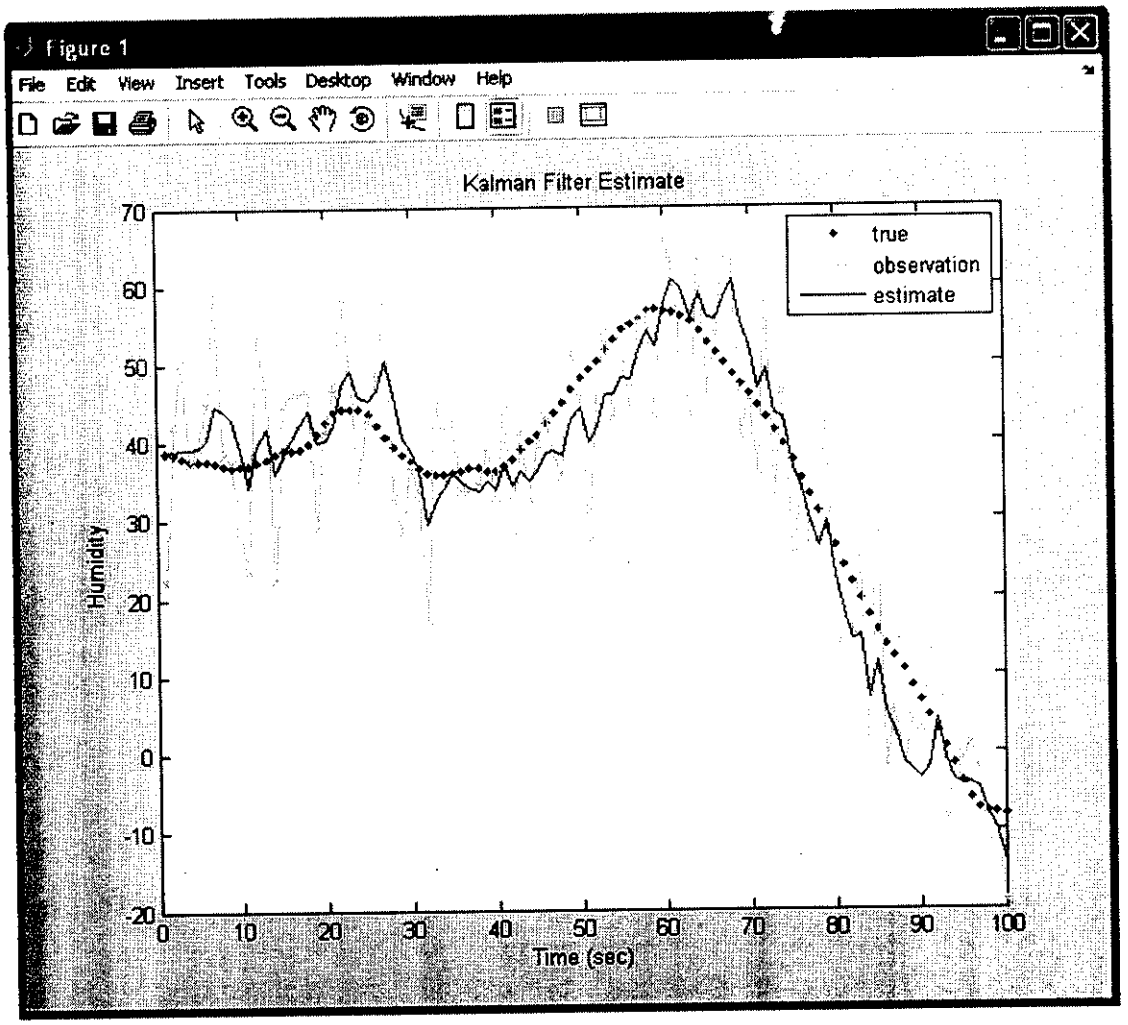

Fig 4.3.1 Kalman Filter Error Estimation.

The Kalman filter has used to predict next data from the previous data. The Fig 4.3.1 shows prediction is not exactly.

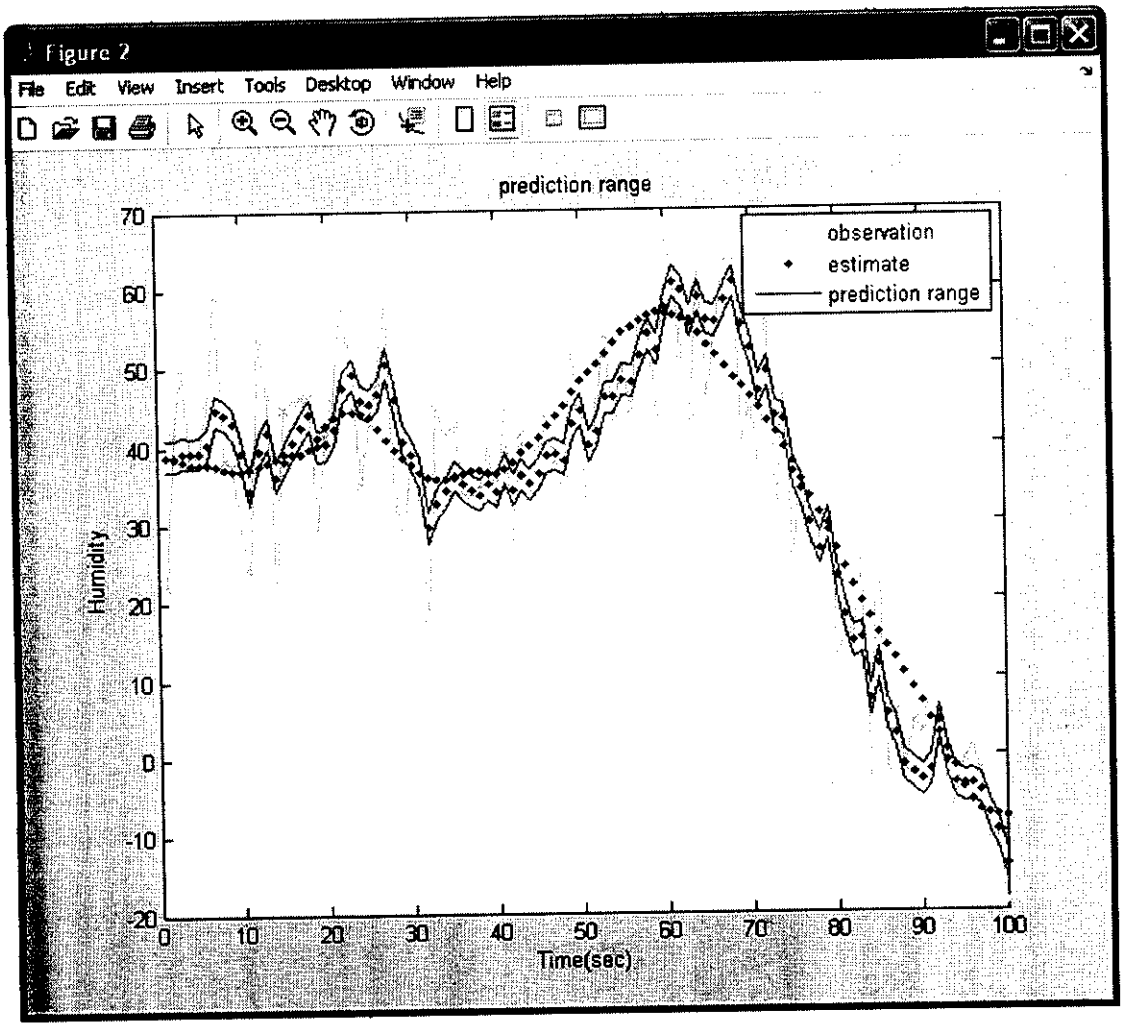

Fig 4.3.2 Range Prediction using Kalman filter.

The Fig 4.3.2 clearly shows that the predicted value is out of range.

4. 4 Huber Method

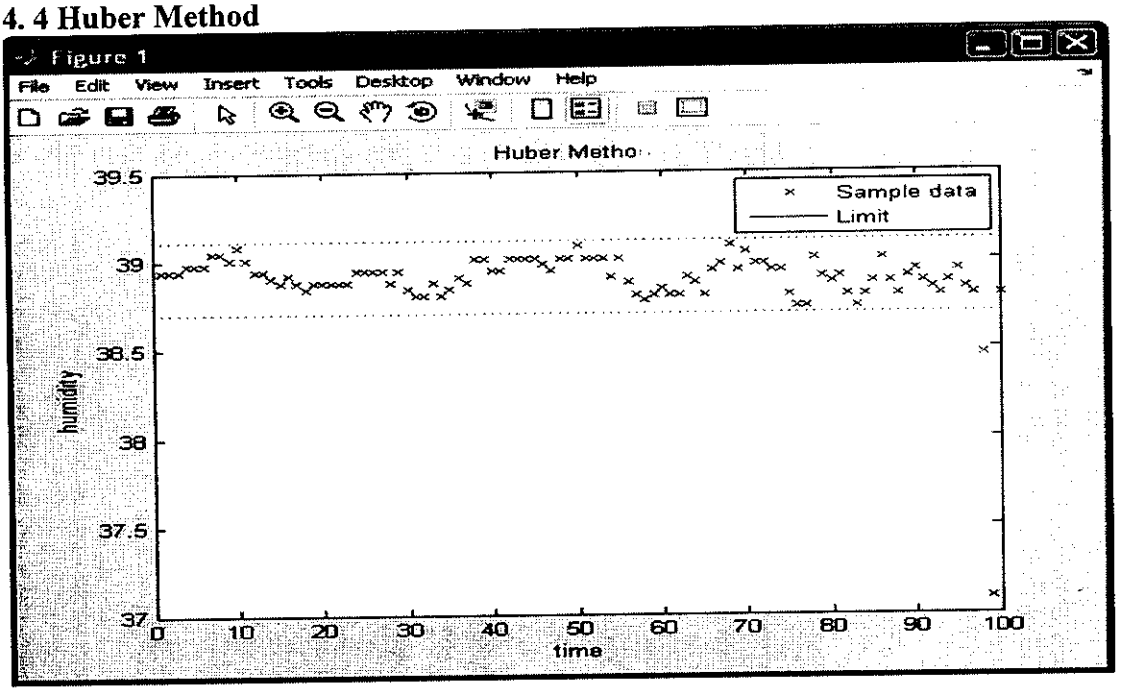

Fig 4.4.1 Huber Method without outliers.

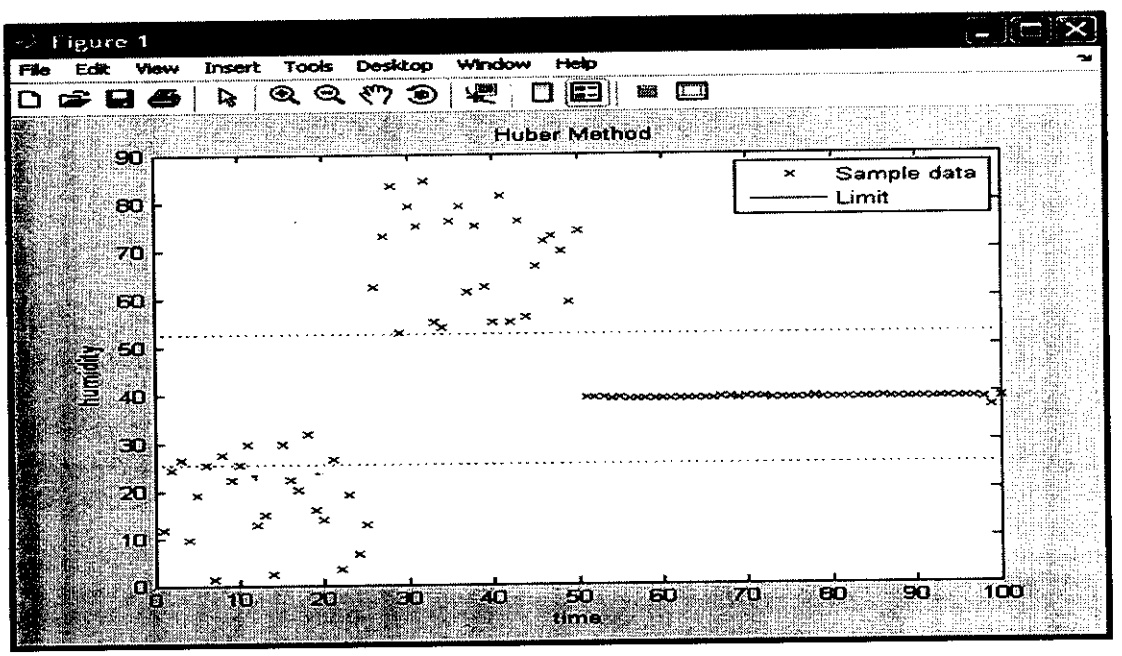

Fig 4.4.2 Huber Method with outlier.

There are no outlier values introduced in original data. The Fig 4.4.1 shows that all values are original values.

50 noisy data are introduced in the original data. Huber method identifies all outlier values, that can be shown in Fig 4.4.2.

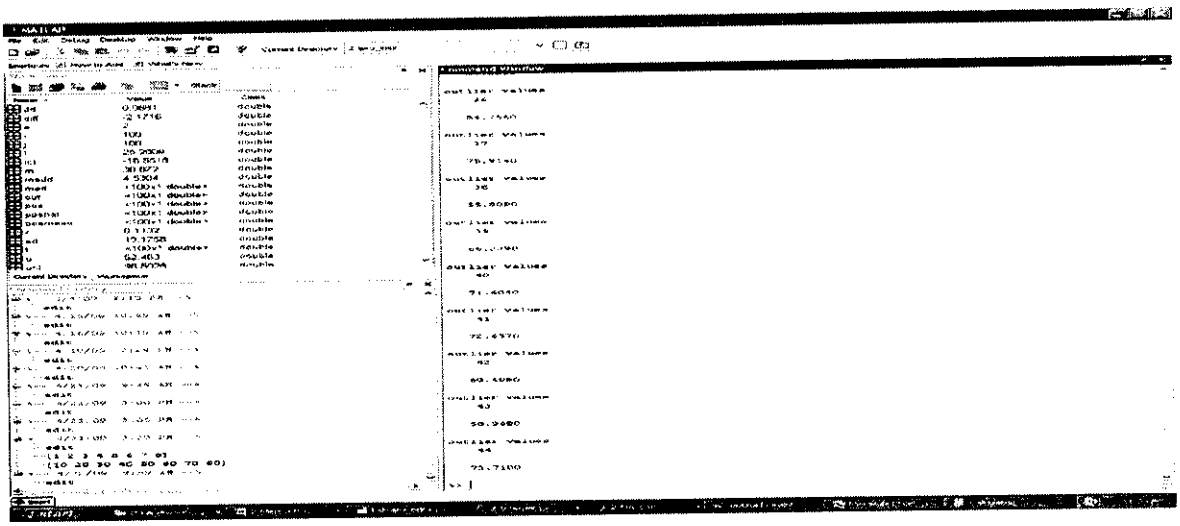

Fig 4.4.3 Huber Method.

No of outlier Data	No of Outlier detected		Percentage	Percentage
	Range [20 30] [50 60]	Range [0 30] [50 80]		
4	4	4	100	100
10	10	10	100	100
14	14	14	100	100
20	20	20	100	100
26	26	26	100	100
32	32	32	100	100
38	38	38	100	100
44	44	44	100	100
50	38	44	76	88
56	Nil	3	0	5.37
64	Nil	Nil	0	0
70	Nil	Nil	0	0
76	Nil	Nil	0	0
80	Nil	Nil	0	0
84	Nil	Nil	0	0
90	Nil	Nil	0	0
100	Nil	Nil	0	0

Table 4.4.1 Performance of Huber Method.

The different numbers of outlier values are introduced in the original data. The performance is tabulated. From the table, Huber method has better performance than the other method.

Method 4.5 Linear Regression Technique

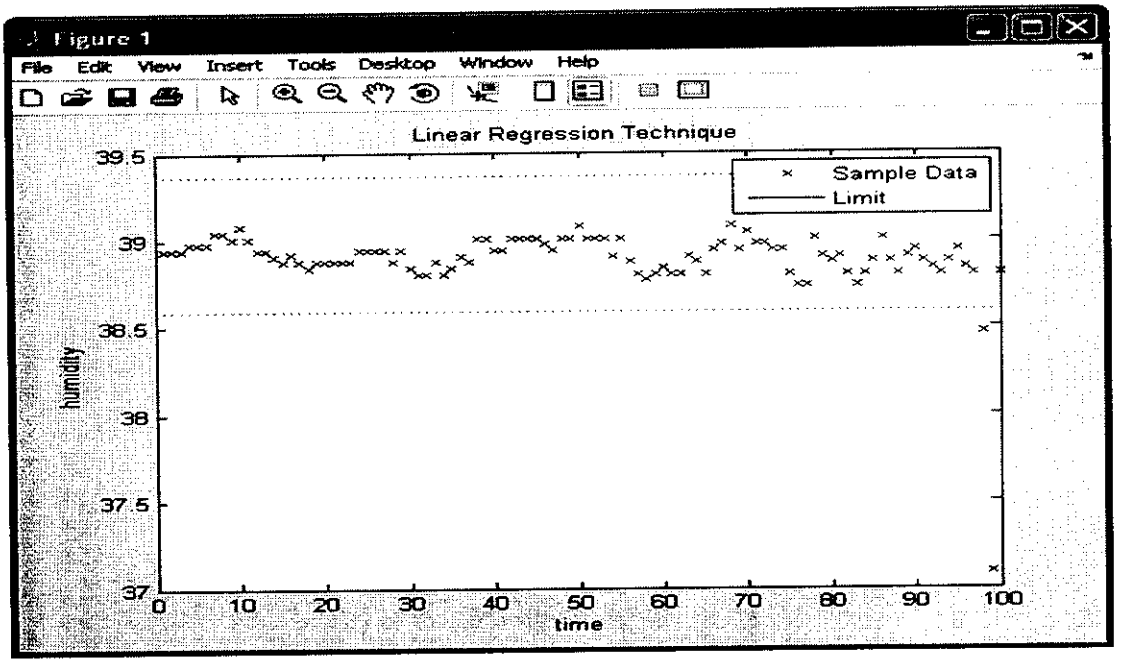

Fig 4.5.1 Linear regression Technique without outliers.

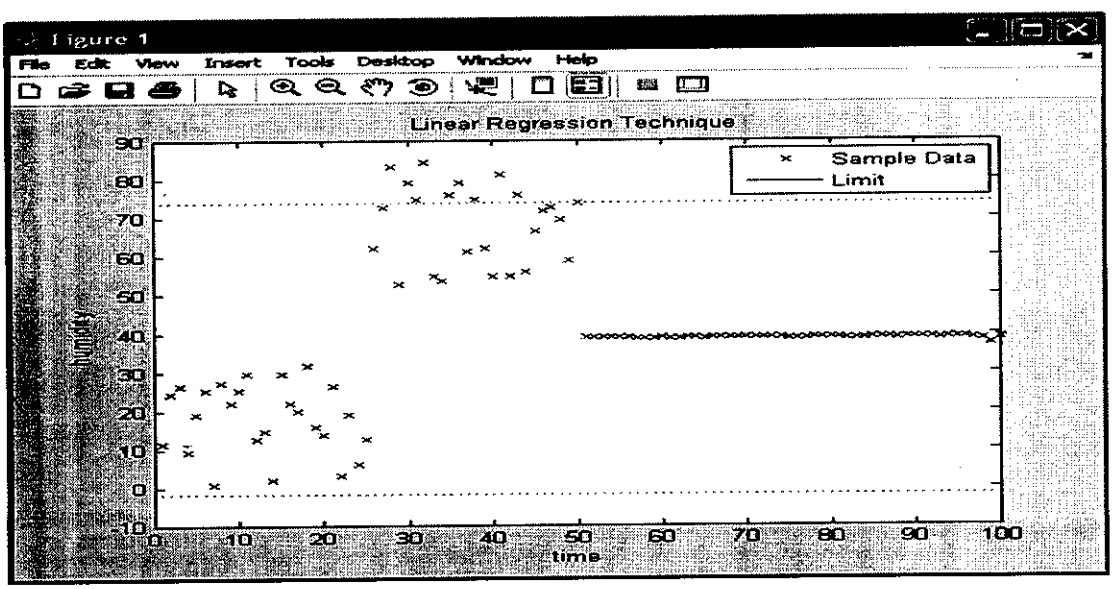

Fig 4.5.2 Linear Regression Technique with outliers.

There are no outlier values introduced in original data. The Fig 4.5.1 shows that all values are original values.

50 noisy data are introduced in the original data. Huber method identifies all outlier values, that can be shown in Fig 4.5.2.

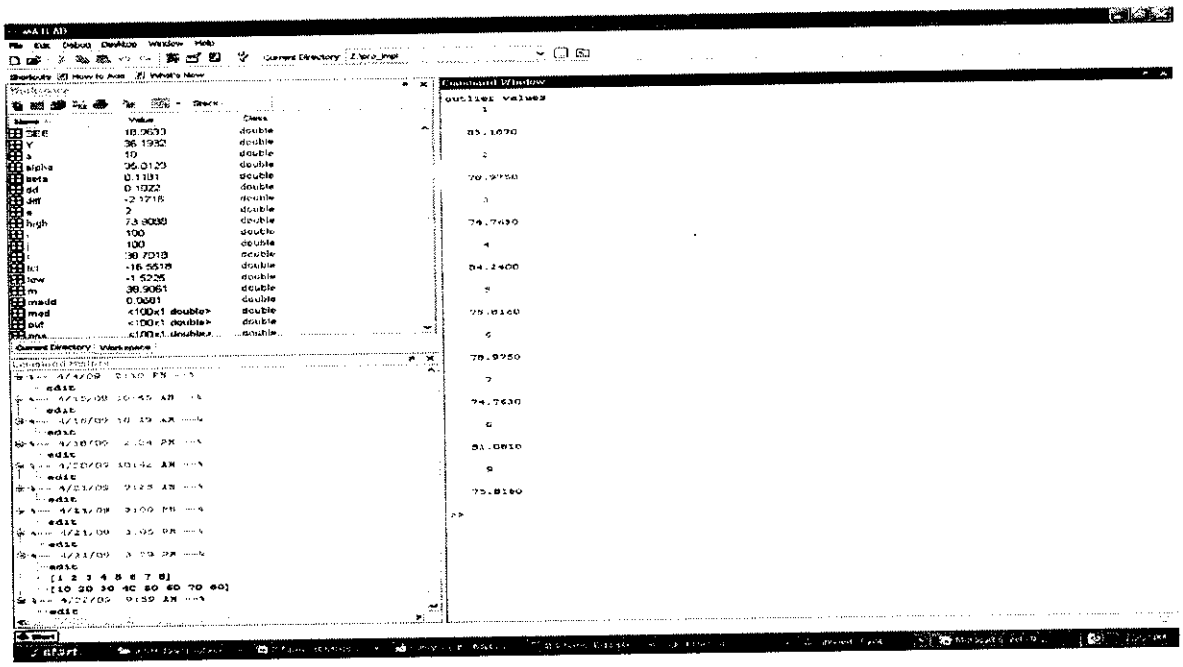

Fig 4.5.3 Linear Regression Technique.

No of outlier Data	Outlier detected		Percentage	Percentage
	Range [20 30] [50 60]	Range [0 30] [50 80]		
4	4	4	100	100
10	9	8	90	80
14	11	10	78.56	71.43
20	11	9	55	45
26	13	11	50	42.31
32	11	10	34.38	31.25
38	9	11	23.68	28.95
44	8	7	18.18	15.90
50	5	9	10	18
56	7	12	12.5	21.43
64	13	11	20.31	17.19
70	12	16	17.14	22.86
76	21	19	27.63	25
80	31	26	38.75	32.50
84	39	28	16.25	33.33
90	45	32	50	35.56
100	50	50	50	50

Table 4.5.1 Performance of Linear Regression Technique.

The different numbers of outlier values are introduced in the original data. The performance is tabulated.

4.6 Modified z-score Technique

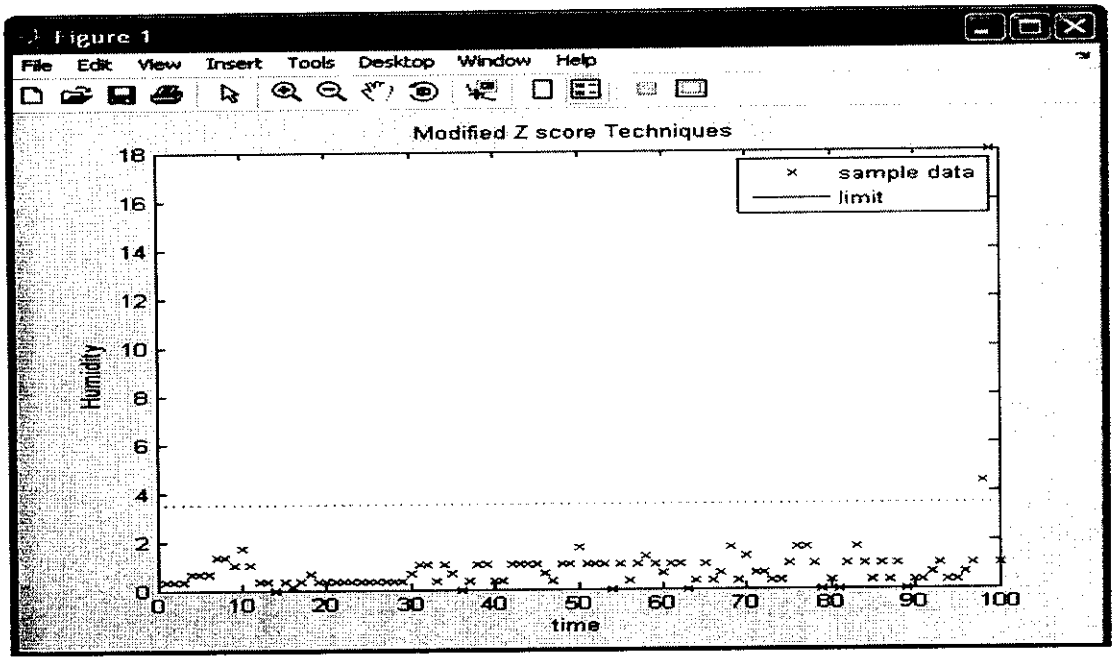

Fig 4.6.1 Modified z-score Technique without outliers

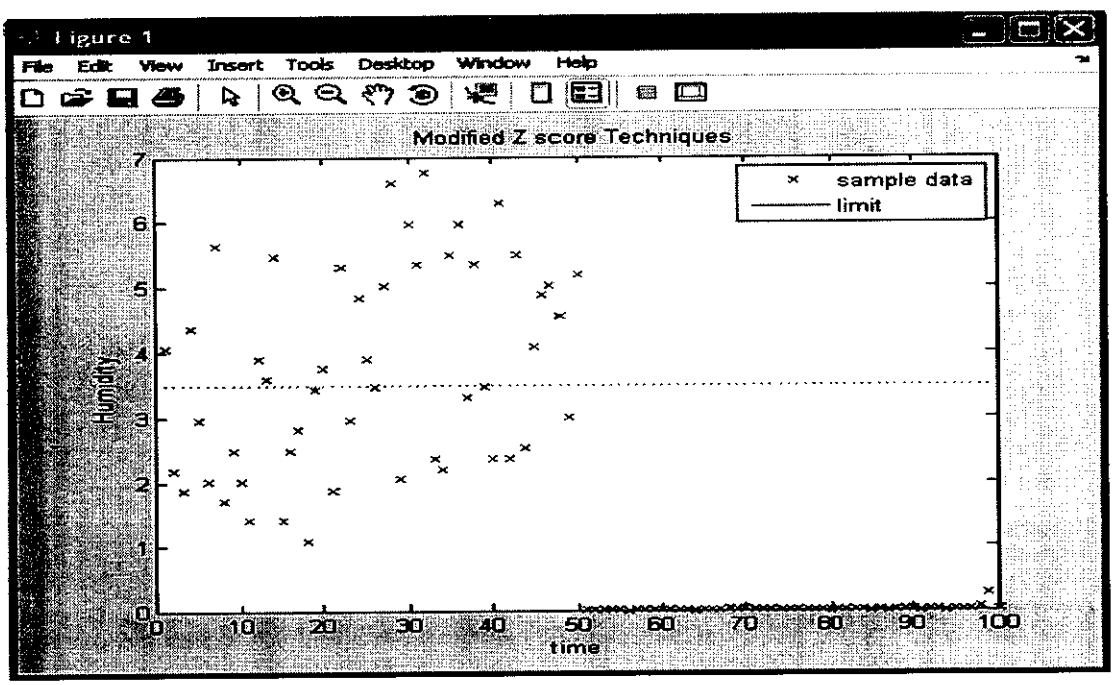

Fig 4.6.2 Modified z-score Technique with outliers.

There are no outlier values introduced in original data. The Fig 4.6.1 shows that all values are original values.

50 noisy data are introduced in the original data. Huber method identifies all outlier values, that can be shown in Fig 4.6.2.

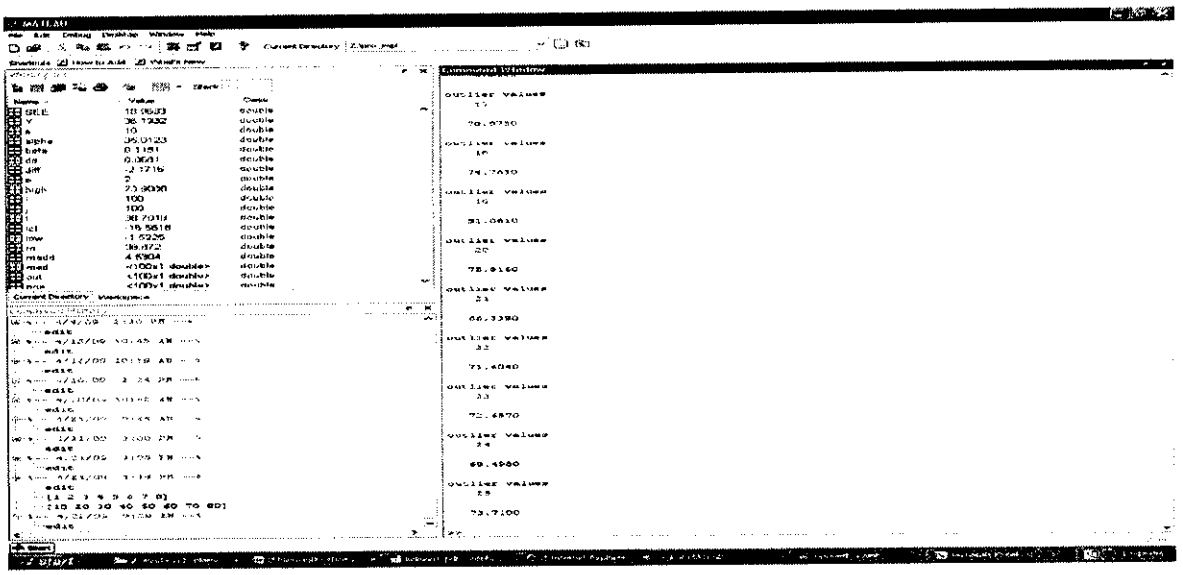

Fig 4.6.3 Modified z-score Technique

No of outlier Data	Outlier detected		Percentage	Percentage
	Range [20 30] [50 60]	Range [0 30] [50 80]		
4	4	4	100	100
10	10	10	100	100
14	14	14	100	100
20	20	20	100	100
26	26	26	100	100
32	32	32	100	100
38	38	38	100	100
44	44	44	100	100
50	2	25	4	50
56	Nil	Nil	0	0
64	Nil	Nil	0	0
70	Nil	Nil	0	0
76	Nil	Nil	0	0
80	Nil	Nil	0	0
84	Nil	Nil	0	0
90	Nil	Nil	0	0
100	Nil	Nil	0	0

Table 4.6.1 Performance of Modified z-score Technique

The different numbers of outlier values are introduced in the original data. The performance is tabulated.

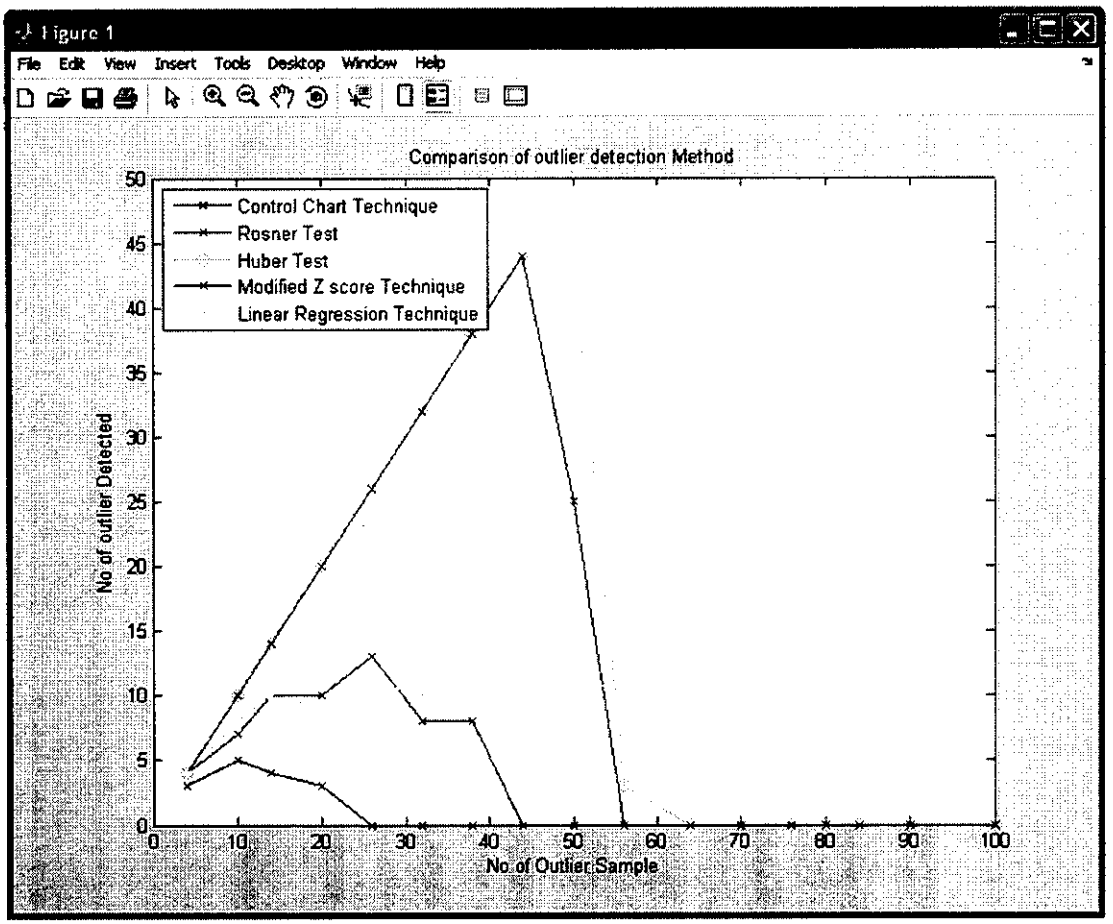

Fig 4.7 Comparison of Outlier Detection Techniques.

The performance graph shows that the Huber test and modified z-score technique has better performance than the other methods.

CHAPTER 6

CONCLUSION AND FUTURE OUTLOOK

Various outlier detection techniques are implemented and performances of each method are tabulated. The Kalman filter did not predict the exact value. In case of any outlier data then the prediction range is more deviated. The LRT identified a small amount outlier data. The CCT has failed to detect even very less number of outliers present in the original data. The modified z-score technique and Huber technique detects the outliers among the original data.

Among the various detection technique, modified z-score technique and Huber technique detects the outlier data better than the other methods i.e. CCT, Rosner technique, Kalman filter, LRT.

Future outlook

The detection method identifies only the local outliers. The outlier detection is happened only after the data gets collected at the sink. So the transmission energy and transmission cost are wasted. If we detect the outlier data at sensor itself, both transmission cost and energy get saved.

APPENDIX 1

Coding for Control Chart Technique

/* IMPLEMENTATION OF CONTROL CHART TECHNIQUE */

/* ---------- */

```
x=[38.9401 38.9401  38.9401  38.9742  38.9742  38.9742 39.0422
39.0422  39.0082  39.0763 ...
   39.0082 38.9401  38.9401  38.9061  38.872   38.9190 38.872
38.8379  38.872   38.872 ...
   38.872  38.872   38.872   38.9401  38.9401  38.9401 38.9401
38.872   38.9401  38.8379 ...
   38.8039 38.8039  38.872   38.8039  38.8379  38.9061 38.872
39.0082  39.0082  38.9401 ...
   38.9401 39.0082  39.0082  39.0082  39.0082  38.9742 38.9401
39.0082  39.0082  39.0763 ...
   39.0082 39.0082  39.0082  38.9061  39.0082  38.872  38.8039
38.7698  38.8039  38.8379 ...
   38.8039 38.8039  38.9061  38.872   38.8039  38.9401 38.9742
39.0763  38.9401  39.0422 ...
   38.9742 38.9742  38.9401  38.9401  38.8039  38.7357 38.7357
39.0082  38.9061  38.872 ...
   38.9061 38.8039  38.7357  38.8039  38.872   39.0082 38.872
38.8039  38.9061  38.9401 ...
   38.872  38.8379  38.8039  38.872   38.9401  38.8379 38.8039
38.4629  37.0933  38.8039 ];

xbar=mean(x);
sd=std(x);
ucl=xbar+3*sd;
lcl=xbar-3*sd;
z=1;
for i=1:100;
    if lcl<x(i) && ucl>x(i)
    else
        disp('outlier values');
        disp(z);
        disp(x(i));
        z=z+1;
    end
end
t=1:100;
t=t';
plot(t,x,'Xb',t,lcl,'-r',t,ucl,'-r');
xlabel('time');
ylabel('humidity');
title('Control Chart Technique');
legend('Sample Data','Limit')
```

Coding for Kalman Filter

/* IMPLEMENTATION OF ROSNER TEST */

/* ---------- */

```
function [pos,posmeas,poshat,l,u,e]=kalmannew(y,ds)
dt=1;
e=2;

measnoise = 10; % position measurement noise
accelnoise = 0.5; % acceleration noise

a = [1 dt; 0 1]; % transition matrix
c = [1 0]; % measurement matrix
x =[y(1);0]; % initial state vector
xhat = x; % initial state estimate

Q = accelnoise^2 * [dt^4/4 dt^3/2; dt^3/2 dt^2]; % process noise
covariance
P = Q; % initial estimation covariance
R = measnoise^2; % measurement error covariance

% set up the size of the innovations vector
Inn = zeros(size(R));

pos = []; % true position array
poshat = []; % estimated position array
posmeas = []; % measured position array
l=[];%lower limit
u=[];%upper limit


for i=2:ds+1;
        % Simulate the process
    ProcessNoise = accelnoise * randn * [(dt^2/2); dt];
    x = a * x + ProcessNoise;
    % Simulate the measurement
    MeasNoise = measnoise * randn;
    z = c * x + MeasNoise;
    % Innovation
    Inn = z - c * xhat;
    % Covariance of Innovation
    s = c * P * c' + R;
    % Gain matrix
    K = a * P * c' * inv(s);
    % State estimate
    xhat = a * xhat + K * Inn;
    % Covariance of prediction error
    P = a * P * a' + Q - a * P * c' * inv(s) * c * P * a';
    % Save some parameters in vectors for plotting later

    pos = [pos; x(1)];%true
    posmeas = [posmeas; z];%measured
    poshat = [poshat; xhat(1)];%estimated
    l=poshat-e;
    u=poshat+e;

    x=[y(i-1);0];
    xhat=x;
end
```

Coding for Rosner Test

```
/* IMPLEMENTATION OF ROSNER TEST */
/* ------------------------------------ */

x=[38.9401 38.9401  38.9401  38.9742  38.9742  38.9742 39.0422
39.0422  39.0082  39.0763 ...
   39.0082 38.9401  38.9401  38.9061  38.872   38.9190 38.872
38.8379  38.872   38.872 ...
   38.872  38.872   38.872   38.9401  38.9401  38.9401 38.9401
38.872   38.9401  38.8379 ...
   38.8039 38.8039  38.872   38.8039  38.8379  38.9061 38.872
39.0082  39.0082  38.9401 ...
   38.9401 39.0082  39.0082  39.0082  39.0082  38.9742 38.9401
39.0082  39.0082  39.0763 ...
   39.0082 39.0082  39.0082  38.9061  39.0082  38.872  38.8039
38.7698  38.8039  38.8379 ...
   38.8039 38.8039  38.9061  38.872   38.8039  38.9401 38.9742
39.0763  38.9401  39.0422 ...
   38.9742 38.9742  38.9401  38.9401  38.8039  38.7357 38.7357
39.0082  38.9061  38.872 ...
   38.9061 38.8039  38.7357  38.8039  38.872   39.0082 38.872
38.8039  38.9061  38.9401 ...
   38.872  38.8379  38.8039  38.872   38.9401  38.8379 38.8039
38.4629  37.0933  38.8039 ];

out=[];
xbar=mean(x);
sd=std(x);
z=1;
l=-1.96;
u=1.96;
for i=1:100;
    diff=x(i)-xbar;
    r=abs(diff/sd);
    out=[out;r];
    if r<1.96
    else
        disp('outlier values');
        disp(z);
        disp(x(i));
        z=z+1;

    end
end
t=1:100;
t=t';
plot(t,out,'Xb',t,l,'r',t,u,'r');
xlabel('time');
ylabel('humidity');
title('Rosner Test');
legend('sample data','limit');
```

Coding for Huber Test

/* IMPLEMENTATION OF HUBER TEST */

/* ---------- */

```
x=[38.9401 38.9401  38.9401  38.9742  38.9742  38.9742 39.0422
39.0422  39.0082  39.0763 ...
   39.0082 38.9401  38.9401  38.9061  38.872   38.9190 38.872
38.8379  38.872   38.872 ...
   38.872  38.872   38.872   38.9401  38.9401  38.9401 38.9401
38.872   38.9401  38.8379 ...
   38.8039 38.8039  38.872   38.8039  38.8379  38.9061 38.872
39.0082  39.0082  38.9401 ...
   38.9401 39.0082  39.0082  39.0082  39.0082  38.9742 38.9401
39.0082  39.0082  39.0763 ...
   39.0082 39.0082  39.0082  38.9061  39.0082  38.872  38.8039
38.7698  38.8039  38.8379 ...
   38.8039 38.8039  38.9061  38.872   38.8039  38.9401 38.9742
39.0763  38.9401  39.0422 ...
   38.9742 38.9742  38.9401  38.9401  38.8039  38.7357 38.7357
39.0082  38.9061  38.872 ...
   38.9061 38.8039  38.7357  38.8039  38.872   39.0082 38.872
38.8039  38.9061  38.9401 ...
   38.872  38.8379  38.8039  38.872   38.9401  38.8379 38.8039
38.4629  37.0933  38.8039 ];

med=[];
t=1;
m=median(x);

for i=1:size(x,2);
    dd=abs(x(i)-m);
    med=[med;dd];
end
madd=median(med);
l=m-1.5*madd;
u=m+1.5*madd;
for j=1:size(x,2);
    if l<x(j) && u>x(j)
    else
        disp('outlier values');
        disp(t);
        disp(x(j));
        t=t+1;
    end
end
t=1:100;
t=t';
plot(t,x,'Xb',t,l,'r',t,u,'r');
xlabel('time');
ylabel('humidity');
title('Huber Method');
legend('Sample data','Limit');
```

Coding for Linear Regression Technique

```
/* IMPLEMENTATION OF LINEAR REGRESSION TECHNIQUE */
/* ---------------------------------- */

x=[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 ...
   16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 ...
   31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 ...
   46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 ...
   61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 ...
   76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 ...
   91 92 93 94 95 96 97 98 99 100];

y=[38.9401 38.9401  38.9401  38.9742  38.9742  38.9742 39.0422
39.0422  39.0082  39.0763 ...
   39.0082 38.9401  38.9401  38.9061  38.872   38.9190 38.872
38.8379  38.872   38.872 ...
   38.872  38.872   38.872   38.9401  38.9401  38.9401 38.9401
38.872   38.9401  38.8379 ...
   38.8039 38.8039  38.872   38.8039  38.8379  38.9061 38.872
39.0082  39.0082  38.9401 ...
   38.9401 39.0082  39.0082  39.0082  39.0082  38.9742 38.9401
39.0082  39.0082  39.0763 ...
   39.0082 39.0082  39.0082  38.9061  39.0082  38.872  38.8039
38.7698  38.8039  38.8379 ...
   38.8039 38.8039  38.9061  38.872   38.8039  38.9401 38.9742
39.0763  38.9401  39.0422 ...
   38.9742 38.9742  38.9401  38.9401  38.8039  38.7357 38.7357
39.0082  38.9061  38.872 ...
   38.9061 38.8039  38.7357  38.8039  38.872   39.0082 38.872
38.8039  38.9061  38.9401 ...
   38.872  38.8379  38.8039  38.872   38.9401  38.8379 38.8039
38.4629  37.0933  38.8039 ];
xbar=mean(x);
ybar=mean(y);
tot=0;
t=0;
tt=0;
temp=0;
ttt=0;
xnot=10;
for i=1:100;
    xt=(x(i)-xbar);
    yt=(y(i)-ybar);
    tot=tot+xt*yt;
    t=t+xt*xt;
    tt=tt+y(i)*y(i);
    temp=temp+y(i);
    ttt=ttt+x(i)*y(i);
end

beta=tot/t;
alpha=ybar-beta*xbar;
Y=alpha+beta*xnot;
SEE=sqrt((tt-alpha*temp-beta*ttt)/98);
```

```
ttemp=1+(1/100)+((xnot-xbar)*(xnot-xbar)/t);
low=Y-1.96*SEE*sqrt(ttemp);
high=Y+1.96*SEE*sqrt(ttemp);


z=1:100;
z=z';
plot(z,y,'Xb',z,low,'-r',z,high,'-r');
xlabel('time');
ylabel('humidity');
title('Linear Regression Technique');
legend('Sample Data','Limit');
disp('outlier values');
a=1;
for i=1:100;
    if y(i)>low  && y(i)<high
    else
        disp(a);
        disp(y(i));
        a=a+1;
    end
end
```

Coding for Modified Z-Score Technique

```
/* IMPLEMENTATION OF LINEAR REGRESSION TECHNIQUE */
/* ---------------------------------- */

x=[38.9401 38.9401  38.9401  38.9742  38.9742  38.9742 39.0422
39.0422  39.0082  39.0763 ...
   39.0082 38.9401  38.9401  38.9061  38.872   38.9190 38.872
38.8379  38.872   38.872 ...
   38.872  38.872   38.872   38.9401  38.9401  38.9401 38.9401
38.872   38.9401  38.8379 ...
   38.8039 38.8039  38.872   38.8039  38.8379  38.9061 38.872
39.0082  39.0082  38.9401 ...
   38.9401 39.0082  39.0082  39.0082  39.0082  38.9742 38.9401
39.0082  39.0082  39.0763 ...
   39.0082 39.0082  39.0082  38.9061  39.0082  38.872  38.8039
38.7698  38.8039  38.8379 ...
   38.8039 38.8039  38.9061  38.872   38.8039  38.9401 38.9742
39.0763  38.9401  39.0422 ...
   38.9742 38.9742  38.9401  38.9401  38.8039  38.7357 38.7357
39.0082  38.9061  38.872 ...
   38.9061 38.8039  38.7357  38.8039  38.872   39.0082 38.872
38.8039  38.9061  38.9401 ...
   38.872  38.8379  38.8039  38.872   38.9401  38.8379 38.8039
38.4629  37.0933  38.8039 ];


med=[];
y=1;
out=[];
m=median(x);

for i=1:size(x,2);
    dd=abs(x(i)-m);
    med=[med;dd];
end
madd=median(med);

for j=1:size(x,2);
    z=abs(0.6745*(x(j)-m)/madd);
    out=[out;z];
    if z>3.5
         disp('outlier values');
        disp(y);
        disp(x(j));
        y=y+1;
    else

    end
end
t=1:100;
t=t';
plot(t,out,'Xb',t,3.5,'r');
xlabel('time');
ylabel('humidity');
title('Modified Z score Techniques');
legend('sample data','limit');
```

REFERENCES

[1] Akyildiz I.F., Sankarasubramaniam W. Su, Y. and Cayirci E. (2002), 'A Survey on Sensor Networks,' IEEE Communications Magazine.

[2] A. Jain, E. Y. Chang, and Y.-F. Wang. "Adaptive stream resource management using Kalman filters". In Proceedings *of SIGMOD*, 2004.

[3] E. Elnahrawy and B. Nath. "Cleaning and querying noisy sensors". In Proceedings of MSWiM, 2003.

[4] Jayashree L.S, Arumugam S, Meenakshi A.R. ,"A Communication efficient framework for outlier free data reporting in data gathering sensor networks", International Journal of Network Management,437-445,2007.

[5] S.C.Gupta. V.K.Mapoor., "Fundamentals of Applied Statistics", volume 62, sultan chand and son's publishers, 1997.

[6] S. Mukhopadhyay, D. Panigrahi, and S. Dey. "Model based error correction for wireless sensor networks". In Proceedings *of SECON*, 2004.

[7] S. R. Jeffery, G. Alonso, M. J. Franklin, W. Hong, and J. Widom. "Declarative support for sensor data cleaning". In Proceedings *of PerCom*, 2006.

[8] Yon Zhen Zhuang, Lei Chen, X.sean Wang, Jie Lian. "A weighted Moving Average based approach for clean sensor data". IEEE, ICDCS (*International conference on Distributed Computing Systems*), 2007.

[9] Y. Zhuang and L. Chen. , "In-network outlier cleaning for data collection in sensor networks". In Proceedings of CleanDB, 2006.

[10] Zuriana Abu Bakar, Rosmayati Mohemad, Akbar Ahmad. "A Comparative Study for Outlier Detection Techniques in Data Mining". IEEE, CIS, 2006.

[11] www.cee.vt.edu/ewr/environmental/teach/smprimer/outlier/outlier.html.

[12] www.exploringdata.cqu.edu.au/boxplot.html.

[13] Intel Lab Data. http://db.csail.mit.edu/labdata/labdata.html.