CHAPTER 5

HLSRGM BASED SPRT: ORDER STATISTICS

5.1 INTRODUCTION

Failure times of a software reliability growth process are modeled as order statistics of independent, nonidentically distributed exponential random variables. Several classes of models have been proposed to capture this definition of reliability; among the most prominent are models built on the assumptions that waiting times between software failures are exponentially distributed and in addition are, conditional on knowledge of the appropriate parameter set, mutually independent. Such models have been called Exponential Order Statistics (EOS) models by Miller (1986) in his investigation.

To improve and to understand the logic behind process control methods, it is necessary to give some thought to the behaviour of sampling and of averages. If the length of a single failure interval is measured, it is clear that occasionally a length will be found which is towards one end of the tails of the process’s normal distribution. This occurrence, if taken on its own, may lead to the wrong conclusion that the process requires adjustment. If, on the other hand, a sample of four or five is taken, it is extremely unlikely that all four or five failure interval lengths will lie towards one extreme end of the distribution.

If, therefore, we take the average or length of four or five failure intervals, we shall have a much more reliable indicator of the state of the process. Sample failure interval length or means will vary with each sample taken, but the variation will not be as great as that for single failure. In the distribution of mean lengths from samples of four failures, the standard deviation of the means, called the standard error of means, and denoted by the symbol SE, is half the standard deviation of the individual Time between failure taken from the process. When n=4, half the spread of the parent distribution of individual TBF. The smaller spread of the distribution of sample averages provides the basis for a useful means of detecting changes in processes. Any change in the process mean, unless it is extremely large, will be difficult to detect from individual results alone. A large
number of individual readings would, therefore, be necessary before such a change was confirmed.

The distribution of sample means reveals the change much more quickly. Therefore, on a chart for sample means, plotted against time, the change in level would be revealed almost immediately. For this reason sample means rather than individual values are used, where possible and appropriate, to control the centering of processes. If the individual values are not normally distributed, the distribution of the means will tend to have a normal distribution, and the larger the sample size the greater will be this tendency. When samples of size \( n = 4 \) or more are taken from a process which is stable, we can assume that the distribution of the sample means will be very nearly normal, even if the parent population is not normally distributed.

A subgroup or sample is a small set of observations on a process parameter or its output, taken together in time. The two major problems with regard to choosing a subgroup relate to its size and the frequency of sampling. The smaller the subgroup, the less opportunity there is for variation within it, but the larger the sample size the narrower the distribution of the means, and the more sensitive they become to detecting change.

A rational subgroup is a sample of items or measurements selected in a way that minimizes variation among the items or results in the sample, and maximizes the opportunity for detecting variation between the samples. With a rational subgroup, assignable or special causes of variation are not likely to be present, but all of the effects of the random or common causes are likely to be shown. Generally, subgroups should be selected to keep the chance for differences within the group to a minimum, and yet maximize the chance for the subgroups to differ from one another. At this stage it is clear that, in any type of process control charting system, nothing is more important than the careful selection of subgroups.

Order statistics are used in a wide variety of practical situations (Mathai Arak, 2003). Their use in characterization problems, detection of outliers, linear estimation, study of system reliability, life-testing, survival analysis, data
compression and many other fields can be seen from the many books on the
topics, e.g., the recent books (BalaKrishnan.N. et al., 1991), (Arnold.B.C. et al.,
1989).

In this chapter we propose a method based on order statistics of
cumulative quantity between observations of time domain failure data using mean
value function of HLD based on NHPP.

*The content of this chapter is to be published, Dr. R. Satya Prasad, S.
Janardhana Rao, Dr. G. Krishna Mohan, (April, 2014) “HLSRGM Based
and Technology, Vol 3, Issue 2. The reprint is appended toward end of this
thesis.*

### 5.2 ORDERED STATISTICS

Order statistics deals with properties and applications of ordered random
variables and of functions of these variables. The use of order statistics is
significant when failures are frequent or inter failure time is less. Let X denote a
continuous random variable with probability density function (pdf) f(x) and
cumulative distribution function (cdf) F(x), and let (X_1, X_2, ..., X_n) denote a
random sample of size n drawn on X. The original sample observations may be
unordered with respect to magnitude. A transformation is required to produce a
corresponding ordered sample. Let (X_{(1)}, X_{(2)}, ..., X_{(n)}) denote the ordered
random sample such that X_{(1)} < X_{(2)} < ... < X_{(n)}; then (X_{(1)}, X_{(2)}, ..., X_{(n)}) are
collectively known as the order statistics derived from the parent of X. The
various distributional characteristics can be known from Balakrishnan and Cohen

The inter-failure time data represent the time lapse between every two
consecutive failures. On the other hand if a reasonable waiting time for failures is
not a serious problem, we can group the inter-failure time data into non
overlapping successive sub groups of size 4 or 5 and add the failure times within
each sub group. For instance if a data of 100 inter-failure times are available we
can group them into 20 disjoint subgroups of size 5. The sum total in each
subgroup would denote the time lapse between every 5\textsuperscript{th} failure and it would be 5\textsuperscript{th} order statistic in a sample of size 5.

In general for inter-failure data of size ‘n’, if r (any natural number less than ‘n’) and preferably a factor of n, we can conveniently divide the data into ‘k’ disjoint subgroups (k=n/r) and the cumulative total in each subgroup indicate the time between every r\textsuperscript{th} failure. The probability distribution of such a time lapse would be that of the r\textsuperscript{th} ordered statistic in a subgroup of size r, which would be equal to r\textsuperscript{th} power of the distribution function of the original variable m(t) (David. H. A. et al., 2003).

The whole process involves the mathematical model of the mean value function and knowledge about its parameters. If the parameters are known they can be taken as they are, for the further analysis. If the parameters are not known they have to be estimated using a sample data by any admissible, efficient method of estimation. This is essential because the control limits depend on mean value function, which in turn depends on the parameters. If software failures are quite frequent keeping track of inter-failures is a tedious task. If failures are more frequent order statistics are preferable.

5.3 MODEL DESCRIPTION AND PARAMETER ESTIMATION.

To calculate the parameter values and control limits using Order Statistics approach, we considered Half Logistic Distribution.

The mean value function of HLD from equation 2.2.5 is

\[ m(t) = \frac{a(1-e^{-bt})}{(1+e^{-bt})} \]

To get m(t) value for r\textsuperscript{th} Order Statistics, take m(t) to the power ‘r’

\[ [m(t)]^r = \left[ \frac{a(1-e^{-bt})}{(1+e^{-bt})} \right]^r \quad (5.3.1) \]
Derivation with respect to \( t \) of equation 5.3.1

\[
m'(t_k) = a' r \left( 1 - e^{-b_{t_k}} \right)^{r-1} \left[ \frac{b e^{-b_{t_k}} \left( 1 + e^{-b_{t_k}} \right) - \left( 1 - e^{-b_{t_k}} \right) \left( -b e^{-b_{t_k}} \right)}{\left( 1 + e^{-b_{t_k}} \right)^2} \right]
\]

\[
= 2 b a' r e^{-b_{t_k}} \left( 1 - e^{-b_{t_k}} \right)^{r-1} \frac{1 + e^{-b_{t_k}}}{\left( 1 + e^{-b_{t_k}} \right)^{r+1}}
\]

\( L = e^{-m(t_k)} \prod_{k=1}^{n} m'(t_k) \)

Taking logarithms on both the sides...

\[
\log L = \log e^{-m(t_k)} \prod_{k=1}^{n} m'(t_k)
\]

\[
= -m(t_k) + \sum_{k=1}^{n} \log \left[ \frac{2 b a' r e^{-b_{t_k}} \left( 1 - e^{-b_{t_k}} \right)^{r-1}}{\left( 1 + e^{-b_{t_k}} \right)^{r+1}} \right]
\]

\[
= -a' \left[ 1 - e^{-b_{t_k}} \right] r + \sum_{k=1}^{n} \left[ \log 2 + \log b + r \log a + \log \left( -b t_k \right) + (r-1) \log \left( 1 - e^{-b_{t_k}} \right) + (r+1) \log \left( 1 + e^{-b_{t_k}} \right) \right]
\]

Taking the partial derivation with respect to ‘\( a \)’ and equating to ‘0’.

\[
\frac{\partial \log L}{\partial a} = -ra'^{-1} \left( 1 - e^{-b_{t_k}} \right)^{r} + \sum_{k=1}^{n} \left[ 0 + \frac{r}{a} + 0 - 0 = 0 - 0 \right]
\]

\[
= \frac{r}{a} \left[ -a' \left( 1 - e^{-b_{t_k}} \right)^{r} \right] + n
\]

\[
\frac{r}{a} \left[ -a' \left( 1 - e^{-b_{t_k}} \right)^{r} \right] + n = 0
\]

\[
-a' \left( 1 - e^{-b_{t_k}} \right)^{r} + n = 0
\]

\[
a' = n \left( \frac{1 - e^{-b_{t_k}}}{1 + e^{-b_{t_k}}} \right)^{r}
\]

\( (5.3.4) \)
Taking the partial derivation with respect to ‘b’ and equating to ‘0’.

\[
\frac{\partial \log L}{\partial b} = -a' \cdot r \left( \frac{1-e^{-br_k}}{1+e^{-br_k}} \right)^{r-1} \left[ \frac{2b \cdot e^{-br_k}}{(1+e^{-br_k})^2} \right]
\]

\[
= -a' \cdot r \left( \frac{1-e^{-br_k}}{1+e^{-br_k}} \right)^{r-1} \left[ \frac{2b \cdot e^{-br_k}}{(1+e^{-br_k})^2} \right] + \sum_{k=1}^{n} \frac{1}{b} \cdot (0+0-2b \cdot (1-e^{-br_k} + r+1)(e^{-br_k}))
\]

\[
= \frac{-2br \cdot e^{-br_k} \cdot (1-e^{-br_k})^{r-1}}{(1+e^{-br_k})^2} + \sum_{k=1}^{n} \frac{1}{b} - t_k + \sum_{k=1}^{n} b \cdot e^{-br_k} \cdot (1-e^{-br_k})^{r-1}
\]

\[
g(b) = \frac{-2br \cdot e^{-br_k} \cdot (1-e^{-br_k})^{r-1}}{(1+e^{-br_k})^2} + \sum_{k=1}^{n} \frac{1}{b} - nt_k + \sum_{k=1}^{n} b \cdot e^{-br_k} \cdot (1-e^{-br_k})^{r-1}
\]

\[
= \frac{2br \cdot e^{-br_k} \cdot (1-e^{-br_k})^{r-1}}{(1+e^{-br_k})^2} \cdot \sum_{k=1}^{n} \left[ \frac{2be^{-br_k} \cdot r - e^{-br_k}}{1-e^{-br_k}} \right] + \sum_{k=1}^{n} b \cdot e^{-br_k} \cdot (1-e^{-br_k})^{r-1}
\]

\[
= \frac{2nb \cdot e^{-br_k}}{(1-e^{-br_k}) \cdot (1-e^{-br_k})} + \sum_{k=1}^{n} \left[ \frac{2be^{-br_k} \cdot r - e^{-br_k}}{1-e^{-br_k}} \right] + \sum_{k=1}^{n} b \cdot e^{-br_k} \cdot (1-e^{-br_k})^{r-1}
\]

\[
\therefore g(b) = \frac{2nb \cdot e^{-br_k}}{1-e^{-br_k}} + \sum_{k=1}^{n} \left[ \frac{2be^{-br_k} \cdot r - e^{-br_k}}{1-e^{-br_k}} \right] + \sum_{k=1}^{n} b \cdot e^{-br_k} \cdot (1-e^{-br_k})^{r-1}
\]

(5.3.5)
Again taking the partial derivation with respect to ‘b’ and equating to ‘0’.

\[
\frac{\partial^2 \log L}{\partial b^2} = 2n \left[ \frac{\left( e^{bt_k} - b e^{bt_k} \right) \left( 1 - e^{-bt_k} \right)}{\left( 1 - e^{-bt_k} \right)^2} \right] + n \left( \frac{-1}{b^2} \right) - 0
\]

\[
= 2n \frac{e^{bt_k} \left( 1 - bt_k \right) \left( 1 - e^{-bt_k} \right) - 2b^2 e^{-bt_k}}{\left( 1 - e^{-bt_k} \right)^2} + \frac{n}{b^2}
\]

\[
+ 2r \sum_{i=1}^{n} \frac{1 \left( e^{bt_i} - e^{bt_k} \right) - b \left( t_i e^{bt_i} + t_i e^{-bt_i} \right)}{\left( e^{bt_i} - e^{-bt_i} \right)^2} - 2n \sum_{i=1}^{n} \frac{1 \left( e^{2bt_i} - 1 \right) - b t_i \left( e^{2bt_i} \right)}{\left( e^{2bt_i} - 1 \right)^2}
\]

(5.3.6)

/*NR METHOD FOR a & b OF 4th ORDER STATISTICS APPROACH*/

#include <stdio.h>
#include <conio.h>
#include <math.h>
#define N 21
#define r 4

double g(double b, double t[], double tn, double tk);
double gdash(double b, double t[], double tn);

main()
{
    int i;

double tk=0,c1,c2,c3,c4;

double t_avg,g1,g2,a;

double t[N] = { 1576, 4149, 5827, 10071, 11836, 15280, 16860, 19572, 23827, 28257, 31886, 34467, 40751, 48262, 53223, 56160, 61565, 69815, 82822, 91190, 97698};

double b[N];
tk=0, t_avg=0.0;
printf("n Input observations :n");
for(i=0;i<N;i++)
{
    tk=tk+t[i];
}
t_avg=tk/N;
b[0]=3.0/t_avg;
printf("nb=%fn",b[0]);
i=-1;
do
{
    i=i+1;
    g1=g(b[i],t,t[N-1],tk);
    g2=gdash(b[i],t,t[N-1]);
    b[i+1]=b[i]-(g1/g2);
    printf("nb[%d]=  %lf b[%d]=  %lf",i,b[i],i+1,b[i+1]);
    printf("\tt|b[%d]-b[%d]|    =  %lf",i+1,i,fabs(b[i+1]-b[i]));
}while((fabs(b[i+1]-b[i])>=0.0001));
c1=(1-exp(-b[i+1]*t[N-1]));
c2=(1+exp(-b[i+1]*t[N-1]));
c3=pow((c1/c2),r);
c4=N/c3;
a=pow(c4,(1.0/r));
printf("n\nb[%d] = %lf is the M.L.E of b = %lf a=%lf",i+1,b[i+1],b[i+1],a);
getch();
}
/*Function for calculating g(b)*/
double g(double b, double t[N], double tn,double tk)
{
    int k;
    double v2=0.0,v3=0.0,v4=0.0,c1,g_val;
    for(k=0;k<N-1;k++)
    {
        v2=v2+t[k];
        v3=v3+(b/(exp(b*t[k])-exp(-b*t[k])));
        v4=v4+(b/(exp(2*b*t[k])-1));
    }
    c1=(2*N*b*exp(-b*tn))/(1-exp(-2*b*tn));
    g_val=c1+(N/b)-(N*v2)+(2*r*v3)-(2*v4);
    return g_val;
}

/*function for calculating g'(b)*/
double gdash(double b,double t[N], double tn)
{
    int k;
    double gdash_val,v1=0.0,v2=0.0,c1,c2,c3,c4;
    for(k=0;k<N;k++)
    {
        v1=v1+(exp(b*t[k])-exp(-b*t[k])-exp(b*t[k])+exp(-b*t[k]))/
           (exp(b*t[k])-exp(-b*t[k]))*(exp(b*t[k])-exp(-b*t[k]));
        v2=v2+(exp(2*b*t[k])-1-(2*b*t[k])*exp(2*b*t[k]))/(exp(2*b*t[k])-1)*
                (exp(2*b*t[k])-1);
    }
    c1=(1-(b*tn))*(1-exp(-2*b*tn));
    c2=(2*b*b*exp(-2*b*tn));
    c3=(1-exp(-2*b*tn))*(1-exp(-2*b*tn));
c4=(2*N*(c1-c2))/(c3);
gdash_val=c4-(N/b*b)+(2*r*v1)-(2*v2);
return gdash_val;
}

/*NR METHOD FOR a & b OF 5th ORDER STATISTICS APPROACH*/
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define N 17
#define r 5

double g(double b, double t[], double tn,double tk);
double gdash(double b, double t[], double tn);

main()
{
    int i;
    double tk=0,c1,c2,c3,c4;
    double t_avg,g1,g2,a;
    double t[N] = { 2610, 4436, 8163, 11836, 15685, 17995, 22226, 28257, 32346, 
                    39856, 46147, 53223, 58996, 67374, 80106, 91190, 98692};
    double b[N];
    tk=0, t_avg=0.0;
    printf("n Input observations :\n");
    for(i=0;i<N;i++)
    {
        tk=tk+t[i];
    }
    t_avg=tk/N;
    b[0]=3.0/t_avg;
printf("nb=%f\n",b[0]);
i=-1;
do {
i=i+1;
g1=g(b[i],t[N-1],tk);
g2=gdash(b[i],t,N-1);
b[i+1]=b[i]-(g1/g2);
printf("nb[%d]= %lf b[%d]= %lf",i,b[i],i+1,b[i+1]);
printf("n|b[%d]-b[%d]| = %lf",i+1,i,fabs(b[i+1]-b[i]));
} while((fabs(b[i+1]-b[i])>=0.0001));
c1=(1-exp(-b[i+1]*t[N-1]));
c2=(1+exp(-b[i+1]*t[N-1]));
c3=pow((c1/c2),r);
c4=N/c3;
a=pow(c4,(1.0/r));
printf("nb[%d] = %lf is the M.L.E of b = %lf a=%lf",i+1,b[i+1],b[i+1],a);
getch();
}

/*Function for calculating g(b)*/
double g(double b, double t[N], double tn,double tk) {
int k;
double v2=0.0,v3=0.0,v4=0.0,c1,g_val;
for(k=0;k<N-1;k++) {
v2=v2+t[k];
\[ v_3 = v_3 + \frac{b}{(\exp(b \cdot t[k]) - \exp(-b \cdot t[k]))}; \]
\[ v_4 = v_4 + \frac{b}{(\exp(2b \cdot t[k]) - 1)}; \]

\}
\[ c_1 = \frac{2N \cdot b \cdot \exp(-b \cdot t[n])}{1 - \exp(-2b \cdot t[n])}; \]
\[ g_{\text{val}} = c_1 + \frac{(N/b) - (N \cdot v_2 + (2r \cdot v_3) - 2 \cdot v_4)}{c_3}; \]
\[ \text{return } g_{\text{val}}; \]

/*function for calculating \( g'(b) \)*/

double gdash(double b, double t[N], double t[n])
{
    int k;
    double gdash_val, v1 = 0.0, v2 = 0.0, c1, c2, c3, c4;
    for(k=0;k<N;k++)
    {
        v1 = v1 + \frac{(\exp(b \cdot t[k]) - \exp(-b \cdot t[k]) - (b \cdot t[k]) \cdot \exp(b \cdot t[k]) + \exp(-b \cdot t[k])}/(\exp(b \cdot t[k]) - \exp(-b \cdot t[k]));
        v2 = v2 + \frac{(\exp(2b \cdot t[k]) - 1 - (2b \cdot t[k]) \cdot \exp(2b \cdot t[k]))/(\exp(2b \cdot t[k]) - 1) \cdot \exp(2b \cdot t[k]) - 1)}{c_3};
    }
    c1 = (1 - (b \cdot t[n])) \cdot (1 - \exp(-2b \cdot t[n]));
    c2 = (2b \cdot b \cdot \exp(-2b \cdot t[n]));
    c3 = (1 - \exp(-2b \cdot t[n])) \cdot (1 - \exp(-2b \cdot t[n]));
    c4 = (2N \cdot (c1 - c2))/(c3);
    gdash_val = c4 - \frac{(N/b \cdot b) + (2r \cdot v_1) - (2 \cdot v_2)}{c_4};
    return gdash_val;
}
5.4 SPRT ANALYSIS OF DATA SETS

The developed SPRT methodology is for a software failure data which is of the form \([t, N(t)]\). Where, \(N(t)\) is the failure number of software system or its sub system in ‘t’ units of time. In this section we evaluate the decision rules based on the considered mean value function for Four different data sets of the above form, borrowed from (Lyu, 1996). The procedure adopted in estimating the parameters is a MLE. Based on the estimates of the parameter ‘b’ in each mean value function, we have chosen the specifications of \(b_0 = b - \delta, \ b_1 = b + \delta\) equidistant on either side of estimate of b obtained through a Data Set to apply SPRT such that \(b_0 < b < b_1\). Assuming the value of \(\delta = 0.0001\), the choices are given in the following table.

Table 5.4.1: a, b Estimates, Specifications of \(b_0, b_1\) for Time domain in 4\(^{th}\) & 5\(^{th}\) order

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Order</th>
<th>Estimate of ‘a’</th>
<th>Estimate of ‘b’</th>
<th>(b_0)</th>
<th>(b_1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS2</td>
<td>4</td>
<td>2.140695</td>
<td>0.000518</td>
<td>0.000418</td>
<td>0.000618</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>1.762340</td>
<td>0.000651</td>
<td>0.000551</td>
<td>0.000751</td>
</tr>
<tr>
<td>SYS3</td>
<td>4</td>
<td>2.672358</td>
<td>0.000802</td>
<td>0.000702</td>
<td>0.000902</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>2.101649</td>
<td>0.000763</td>
<td>0.000663</td>
<td>0.000863</td>
</tr>
<tr>
<td>SYS1</td>
<td>4</td>
<td>2.414736</td>
<td>0.000727</td>
<td>0.000627</td>
<td>0.000827</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>1.933182</td>
<td>0.001227</td>
<td>0.001127</td>
<td>0.001327</td>
</tr>
<tr>
<td>DS2</td>
<td>4</td>
<td>2.280082</td>
<td>0.011644</td>
<td>0.011544</td>
<td>0.011744</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>1.855771</td>
<td>0.009497</td>
<td>0.009397</td>
<td>0.009597</td>
</tr>
</tbody>
</table>

Using the selected \(b_0, b_1\) and subsequently the \(m_0(t), m_1(t)\) for the model, we calculated the decision rules given by Equations 2.4.2.4 and 2.4.2.5, sequentially at each ‘t’ of the data sets taking the strength (\(\alpha, \beta\)) as (0.05, 0.2). These are presented for the model in Table 5.4.2 for 4\(^{th}\) order and in Table 5.4.3 for 5\(^{th}\) order. The following consolidated tables reveals the iterations required to come to a decision about the software of each Data Set.
Table 5.4.2: SPRT analysis for 4 data sets of Time domain data of 4th order

<table>
<thead>
<tr>
<th>Data Set</th>
<th>T</th>
<th>N(t)</th>
<th>Acceptance region (≤)</th>
<th>Rejection Region (≥)</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS2</td>
<td>1576</td>
<td>1</td>
<td>-0.638751</td>
<td>2.442881</td>
<td>Accept</td>
</tr>
<tr>
<td></td>
<td>4149</td>
<td>2</td>
<td>5.840916</td>
<td>11.187336</td>
<td></td>
</tr>
<tr>
<td>SYS3</td>
<td>89</td>
<td>1</td>
<td>-1.555141</td>
<td>2.767475</td>
<td>Reject</td>
</tr>
<tr>
<td></td>
<td>193</td>
<td>2</td>
<td>-1.558258</td>
<td>2.777885</td>
<td></td>
</tr>
<tr>
<td></td>
<td>269</td>
<td>3</td>
<td>-1.559066</td>
<td>2.793303</td>
<td></td>
</tr>
<tr>
<td>SYS1</td>
<td>227</td>
<td>1</td>
<td>-1.411762</td>
<td>2.516505</td>
<td>Reject</td>
</tr>
<tr>
<td></td>
<td>444</td>
<td>2</td>
<td>-1.408873</td>
<td>2.569541</td>
<td></td>
</tr>
<tr>
<td></td>
<td>759</td>
<td>3</td>
<td>-1.299596</td>
<td>2.811027</td>
<td></td>
</tr>
<tr>
<td></td>
<td>39</td>
<td>1</td>
<td>-23.532863</td>
<td>42.065257</td>
<td></td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>2</td>
<td>-24.372214</td>
<td>44.342369</td>
<td></td>
</tr>
<tr>
<td></td>
<td>68</td>
<td>3</td>
<td>-24.733295</td>
<td>45.548811</td>
<td></td>
</tr>
<tr>
<td></td>
<td>97</td>
<td>4</td>
<td>-26.144561</td>
<td>51.754244</td>
<td></td>
</tr>
<tr>
<td></td>
<td>104</td>
<td>5</td>
<td>-26.512602</td>
<td>53.719483</td>
<td></td>
</tr>
<tr>
<td></td>
<td>113</td>
<td>6</td>
<td>-27.013749</td>
<td>56.529453</td>
<td></td>
</tr>
<tr>
<td></td>
<td>158</td>
<td>7</td>
<td>-30.617901</td>
<td>75.600785</td>
<td></td>
</tr>
<tr>
<td>DS2</td>
<td>173</td>
<td>8</td>
<td>-32.569514</td>
<td>83.929867</td>
<td>Continue</td>
</tr>
<tr>
<td></td>
<td>194</td>
<td>9</td>
<td>-36.322273</td>
<td>97.433344</td>
<td></td>
</tr>
<tr>
<td></td>
<td>229</td>
<td>10</td>
<td>-46.174613</td>
<td>125.547123</td>
<td></td>
</tr>
<tr>
<td></td>
<td>237</td>
<td>11</td>
<td>-49.215504</td>
<td>133.146250</td>
<td></td>
</tr>
<tr>
<td></td>
<td>243</td>
<td>12</td>
<td>-51.720214</td>
<td>139.177993</td>
<td></td>
</tr>
<tr>
<td></td>
<td>260</td>
<td>13</td>
<td>-59.961147</td>
<td>157.983005</td>
<td></td>
</tr>
<tr>
<td></td>
<td>271</td>
<td>14</td>
<td>-66.289464</td>
<td>171.666561</td>
<td></td>
</tr>
</tbody>
</table>
Table 5.4.3: SPRT analysis for 4 data sets of Time domain data of 5th order

<table>
<thead>
<tr>
<th>Data Set</th>
<th>T</th>
<th>N(t)</th>
<th>Acceptance region (≤)</th>
<th>Rejection Region (≥)</th>
<th>Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS2</td>
<td>2610</td>
<td>1</td>
<td>1.045866</td>
<td>5.367692</td>
<td>Accept</td>
</tr>
<tr>
<td></td>
<td>93</td>
<td>1</td>
<td>-1.183010</td>
<td>2.105074</td>
<td></td>
</tr>
<tr>
<td>SYS3</td>
<td>243</td>
<td>2</td>
<td>-1.188486</td>
<td>2.115602</td>
<td>Reject</td>
</tr>
<tr>
<td></td>
<td>345</td>
<td>3</td>
<td>-1.194028</td>
<td>2.129161</td>
<td></td>
</tr>
<tr>
<td>SYS1</td>
<td>342</td>
<td>1</td>
<td>-1.953645</td>
<td>3.504983</td>
<td>Accept</td>
</tr>
<tr>
<td></td>
<td>571</td>
<td>2</td>
<td>-1.949675</td>
<td>3.795651</td>
<td></td>
</tr>
<tr>
<td></td>
<td>968</td>
<td>3</td>
<td>-1.230700</td>
<td>5.403833</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1986</td>
<td>4</td>
<td>6.675843</td>
<td>18.963992</td>
<td></td>
</tr>
</tbody>
</table>
From the Table 5.4.2 and 5.4.3, a decision of either to accept, reject the system or continue is reached much in advance of the last time instant of the data.
5.5 CONCLUSION.

The above consolidated tables of HLSRGM as exemplified for four Data Sets indicate that the model is performing well in arriving at a decision. The model has given a decision of rejection for 2 Data Sets i.e. SYS3 and SYS1 at 3\textsuperscript{rd} instances for both, acceptance for 1 Data Set i.e. SYS2 at 2\textsuperscript{nd} instance and continue for 1 Data set i.e DS2 in 4\textsuperscript{th} order respectively. In 5\textsuperscript{th} order a decision of acceptance for 2 Data Sets i.e. SYS2 and SYS1 at 1\textsuperscript{st} and 4\textsuperscript{th} instance, rejection for 1 Data Set i.e SYS3 at 3\textsuperscript{rd} instance and continue for 1 Data set i.e. DS2. In both 4\textsuperscript{th} and 5\textsuperscript{th} orders, the decisions are of almost same. Therefore, we may conclude that, applying SPRT on data sets we can come to an early conclusion of reliability / unreliability of software.