Appendix A: Theorems and proofs

Theorem 1:

The ARL of the two-sided chart can be expressed as a function of the average run lengths of the one-sided charts through the expression

$$\frac{1}{ARL} = \frac{1}{ARL^U} + \frac{1}{ARL^L} \qquad \text{(A1)}$$

where ARL^U and ARL^L denotes the average run lengths for the upper and lower one-sided charts, respectively. This result applies to both Shewhart- and CUSUM-type charts. A proof of expression (A1) is given by using the properties of generating functions.

Proof to Theorem 1:

Generating functions

Let X be a random variable whose possible values are restricted to the nonnegative integers $\{0,1,2,...\}$ and write $c_j = P(X = j)$. The probability generating function (hereafter pgf) is defined as

$$\Pi_X(s) = \sum_{j=0}^{\infty} c_j s^j = \sum_{j=0}^{\infty} P(X = j)s^j = P(X = 0)s^0 + P(X = 1)s^1 + P(X = 2)s^2 + ...$$

where s must be restricted to a region in which the power series is convergent. The power series always converges if

$$|s| \leq 1 \text{, that is, } -1 \leq s \leq 1 . \qquad \text{(A2)}$$

An alternative definition of $\Pi_X(s)$ is

$$\Pi_X(s) = E\left(s^X\right). \qquad \text{(A3)}$$

Properties of generating functions

Let

$$q_j = c_{j+1} + c_{j+2} + ... = P(X = j+1) + P(X = j+2) + ... = P(X > j) \text{ for } j = 0,1,2,... \qquad \text{(A4)}$$

be the ‘tail’ probabilities. Then

$$\sum_{j=0}^{\infty} q_j = (c_1 + c_2 + ...) + (c_2 + c_3 + ...) + ... = c_1 + 2c_2 + 3c_3 + ... = \sum_{j=1}^{\infty} jc_j = \sum_{j=0}^{\infty} jc_j = E(X).$$

Therefore the sequence $\{q_j\}$ for $j = 0,1,2,...$ is of importance, because it constitutes another probability distribution on the integers $0,1,2,...$ with its pgf given by

$$\sum_{j=0}^{\infty} q_j s^j = \frac{1 - \sum_{j=0}^{\infty} c_j s^j}{1 - s} \text{ for } -1 < s < 1 \quad \text{(A5)}$$

which simplifies to

$$Q(s) = \frac{1 - \Pi_X(s)}{1 - s} \text{ for } -1 < s < 1. \quad \text{(A6)}$$

The condition $-1 < s < 1$ is due to the convergence rule (A2), and the fact that expressions (A5) and (A6) will only hold for $s \neq 1$.

Signalling event

Let ε denote a signalling event. Then ε is a *recurrent* event, because if ε occurs on the j^{th} trial, we treat trial $j+1$ as though it were the first trial. Let the random variable Y denote the number of the trial on which event ε occurs for the first time. Let q_j denote the probability of no occurrences in the first j trials of an event ε. Then $q_j = P(\varepsilon \text{ does not occur in the first } j \text{ trials}) = P(Y > j)$. If $Y > j$, there have been no indications of a changed process in the first j points. Let c_j denote the probability that an event ε occurs for the *first time* on the j^{th} trial. Let p_j denote the probability that an event ε occurs on the j^{th} trial. The set of initial conditions is:

$q_0 = P(\varepsilon \text{ does not occur in the first 0 trials}) = 1$	(A7)
$c_0 = 0$	(A8)
$p_0 = 1$	(A9)

Let $Q(s), C(s)$ and $P(s)$ denote the generating functions for the probabilities q_j, c_j and p_j, respectively. The pgf uniquely determines the corresponding probability distribution and in turn the probability distribution on $0,1,2,\ldots$, uniquely determines the pgf. Clearly, there is a 1-1 correspondence between the probability distributions and the pgf's.

Generating function $Q(s)$:

$$Q(s)=\sum_{j=0}^{\infty} q_j s^j = q_0 s^0 + q_1 s^1 + q_2 s^2 + q_3 s^3 + \ldots$$

Applying initial condition (A7), we have that

$$Q(s)=1+q_1 s+q_2 s^2+q_3 s^3+\ldots$$

The generating function is helpful in obtaining moments of the distribution of Y. Particularly,

$$Q(1)=1+q_1+q_2+q_3+\ldots=1+P(Y>1)+P(Y>2)+P(Y>3)+\ldots=E(Y) \qquad \text{(A10)}$$

where $E(Y)$ denotes the average number of trials between consecutive occurrences of signalling events.

Generating function $C(s)$:

$$C(s)=\sum_{j=0}^{\infty} c_j s^j = c_0 s^0 + c_1 s^1 + c_2 s^2 + c_3 s^3 + \ldots$$

Applying initial condition (A8), we have that

$$C(s)=c_1 s+c_2 s^2+c_3 s^3+\ldots$$

Due to the fact that we only consider recurrent events which have finite recurrence times, we have that

$$C(1)=c_1+c_2+c_3+\ldots=1. \qquad \text{(A11)}$$

Generating function $P(s)$:

$$P(s)=\sum_{j=0}^{\infty} p_j s^j = p_0 s^0 + p_1 s^1 + p_2 s^2 + p_3 s^3 + \ldots$$

Applying initial condition (A9), we have that

$$P(s)=1+p_1 s+p_2 s^2+p_3 s^3+\ldots$$

Corollary A1:

The $p's$ can be determined in terms of the $c's$:

$$P(s) = \frac{1}{1 - C(s)}$$

Proof:

Note: In this proof the set of initial conditions holds (see (A7), (A8) and (A9)).

The equation given below holds, since ε is a recurrent event:

$$p_j = c_j p_0 + c_{j-1} p_1 + c_{j-2} p_2 + \ldots + c_1 p_{j-1} + c_0 p_j$$

For $j = 0$: $p_0 = c_0 p_0$

For $j = 1$: $p_1 = c_1 p_0 + c_0 p_1$

For $j = 2$: $p_2 = c_2 p_0 + c_1 p_1 + c_0 p_2$

Continuing this way for $j = 3,4,\ldots$ we have that

$$p_0 + p_1 + p_2 + \ldots = (c_0 p_0) + (c_1 p_0 + c_0 p_1) + (c_2 p_0 + c_1 p_1 + c_0 p_2) + \ldots$$
$$= (p_0 + p_1 + p_2 + \ldots)(c_0 + c_1 + c_2 + \ldots)$$

Applying initial conditions (A8) and (A9) we have that

$$1 + p_1 + p_2 + \ldots = (1 + p_1 + p_2 + \ldots)(0 + c_1 + c_2 + \ldots) \qquad \text{(A12)}$$

Multiplying (A12) through by s^j and summing over j from one to infinity, we obtain

$$\sum_{k=0}^{\infty} p_k = \sum_{k=0}^{\infty} p_k \sum_{i=0}^{\infty} c_i$$

$$\sum_{j=1}^{\infty} \sum_{k=0}^{\infty} p_k s^j = \sum_{j=1}^{\infty} s^j \sum_{k=0}^{\infty} p_k \sum_{i=0}^{\infty} c_i$$

$$\sum_{j=0}^{\infty} p_j s^j = \sum_{j=0}^{\infty} p_j s^j \sum_{k=0}^{\infty} c_k s^k + 1$$

$$P(s) = P(s)C(s) + 1 \qquad \text{(A13)}$$

(refer to expression (A11) for an explanation of why the constant appears in expression (A13)) . Re-writing expression (A13) we obtain

$$P(s) = \frac{1}{1 - C(s)} \qquad \text{(A14)}$$

Corollary A2:

The q's can be determined in terms of the c's:

$Q(s) = \dfrac{1 - C(s)}{1 - s}$ for $-1 < s < 1$.

Proof:

By using equations (A5) and (A6), the q's can be determined in terms of the c's:

$$\sum_{j=0}^{\infty} q_j s^j = \frac{1 - \sum_{j=0}^{\infty} c_j s^j}{1 - s} \tag{A15}$$

which simplifies to

$$Q(s) = \frac{1 - C(s)}{1 - s} \quad \text{for} \quad -1 < s < 1. \tag{A16}$$

From corollary A1 and corollary A2 we have

$$P(s) = \frac{1}{1 - C(s)}$$

and

$$\frac{1}{1 - P(s)} = \frac{1}{(1 - s)Q(s)} \quad \text{for } -1 < s < 1$$

Therefore, we have that

$$P(s) = \frac{1}{1 - C(s)} = \frac{1}{(1 - s)Q(s)} \quad \text{for } -1 < s < 1 \tag{A17}$$

which can be re-written as

$$(1 - s)P(s) = \frac{1}{Q(s)} \quad \text{for } -1 < s < 1 \tag{A18}$$

Taking the limit of $(1 - s)P(s)$ as s approaches unity and applying (A10) we have that

$$\lim_{s \to 1}(1 - s)P(s) = \lim_{s \to 1} \frac{1}{Q(s)} = \frac{1}{Q(1)} = \frac{1}{E(Y)}. \tag{A19}$$

Let ε_U and ε_L denote the signalling events for the upper and lower one-sided charts, respectively. Let ε_{UL} denote a signalling event of the two-sided chart. Let $E_L(Y)$, $E_U(Y)$ and $E_{UL}(Y)$ denote the average recurrence time of ε_L, ε_U and ε_{UL}, respectively. We would like to determine $E_{UL}(Y)$ from $E_L(Y)$ and $E_U(Y)$. Either ε_L or ε_U (but not both) occurs on every trial on which ε_{UL} occurs, and this leads to equation (A20)

$$P_{UL,j} = P_{L,j} + P_{U,j} \tag{A20}$$

Multiplying (A20) through by s^j and summing over j from one to infinity, we obtain the generating functions:

$$P_{UL}(s) = P_L(s) + P_U(s) - 1 \tag{A21}$$

The constant appears, because probabilities sum to unity. Summing over j from one to infinity over the probabilities on the left-hand side of equation (A20), must equal one. Summing over j from one to infinity over the probabilities of the first term on the right-hand side of equation (A20), must equal one *and* summing over j from one to infinity over the probabilities of the second term on the right-hand side of equation (A20), must equal one. Therefore, summing the two terms on the right-hand side equals two. The problem arises: The left-hand side of the equation equals one while the right-hand side of the same equation equals two. This problem is solved by subtracting one from the right-hand side of the equation.

Multiplying (A21) through by $(1-s)$ and taking the limit as s approaches unity, we have that $\lim_{s\to 1}(1-s)P_{UL}(s) = \lim_{s\to 1}(1-s)P_L(s) + \lim_{s\to 1}(1-s)P_U(s) - \lim_{s\to 1}(1-s)$. From (A19) we have that

$$\frac{1}{E_{UL}(Y)} = \frac{1}{E_L(Y)} + \frac{1}{E_U(Y)} - (1-1)$$

which simplifies to

$$\frac{1}{E_{UL}(Y)} = \frac{1}{E_L(Y)} + \frac{1}{E_U(Y)}. \tag{A22}$$

Theorem 2:

Fu, Spiring and Xie (2002) defined the moment generating function (hereafter mgf) as $\phi(t) = 1 + (e^t - 1)\underline{\xi}(I - e^t Q)^{-1}\underline{1}$ and used the mgf to obtain expressions for the first and second moments of the run length variable N. Fu and Lou (2003) defined the probability generating function (hereafter pgf) as $\varphi(t) = 1 + (t-1)\underline{\xi}(I - tQ)^{-1}\underline{1}$ and used the pgf to obtain expressions for the first and second moments of the run length variable N. Although they used different methods, both were able to obtain the following expressions for the first and second moments of the run length variable N:

$$E(N) = \underline{\xi}(I - Q)^{-1}\underline{1}$$

$$E(N^2) = \underline{\xi}(I + Q)(I - Q)^{-2}\underline{1}$$

where Q is the matrix that contains all the transition probabilities of going from a non-absorbing state to a non-absorbing state, I is the identity matrix, $\underline{\xi} = (1,0,0,...,0)$ is a row vector with 1 at the 1^{st} element and zeros elsewhere, $\underline{1} = (1 \quad ... \quad 1)$ is a column vector with all elements equal to unity (refer to the Section 2.3 for more detail about the construction and the dimensions of these matrices).

In this appendix the derivation of the first and second moments of the run length variable N will be done using both the mgf and the pgf.

Proof to Theorem 2:

A power series is defied as $f(x) = \sum_{n=0}^{\infty} a_n x^n$. It is also referred to as the generating function. Generating functions are very useful combinatorial enumeration problems. In general we have that $x(1-x)^{-1} = x(1 + x + x^2 + ...)$ for $|x| < 1$. Similarly, in this example we will use the fact that $Q(I-Q)^{-1} = Q(I + Q + Q^2 + ...)$. In addition, generally we have that $x(1-x)^{-2} = x + 2x^2 + 3x^3 + ... = x(1 + 2x + 3x^2 + ...)$ for $|x| < 1$. Similarly, in this example we will use the fact that $Q(I-Q)^{-2} = Q(I + 2Q + 3Q^2 + ...)$.

Moment generating function

(See page 373 of Fu, Spiring and Xie (2002))

The mgf is given by $\phi(t) = 1 + (e^t - 1)\underline{\xi}(I - e^t Q)^{-1}\underline{1}$. It is well-known that if the mgf is differentiable at zero, then the n^{th} moment is given by $\phi^{(n)}(0)$. Therefore, in order to find the first moment, we will have to calculate the first order derivative of the mgf in the point $t = 0$, that is, $E(N) = \phi'(0)$. Similarly, in order to find the second moment, we will have to calculate the second order derivative of the mgf in the point $t = 0$, that is, $E(N^2) = \phi''(0)$.

The first order derivative:

(Differentiation is done by using the well-known product rule).

$$\phi'(t) = e^t \underline{\xi}(I - e^t Q)^{-1}\underline{1} + (e^t - 1)\frac{d}{dt}\left(\underline{\xi}(I - e^t Q)^{-1}\underline{1}\right)$$

At $t = 0$:

$$\phi'(0) = \underline{\xi}(I - Q)^{-1}\underline{1}$$

Therefore we have that $E(N) = \underline{\xi}(I - Q)^{-1}\underline{1}$.

The second order derivative:

(Differentiation is done by using the well-known product rule).

$$\phi''(t) = e^t \underline{\xi}(I - e^t Q)^{-1}\underline{1} + e^t \frac{d}{dt}\left(\underline{\xi}(I - e^t Q)^{-1}\underline{1}\right) + e^t \frac{d}{dt}\left(\underline{\xi}(I - e^t Q)^{-1}\underline{1}\right) + (e^t - 1)\frac{d^2}{dt^2}\left(\underline{\xi}(I - e^t Q)^{-1}\underline{1}\right)$$

At $t = 0$:

$$\phi''(0) = \underline{\xi}(I - Q)^{-1}\underline{1} + \frac{d}{dt}\left(\underline{\xi}(I - e^t Q)^{-1}\underline{1}\right)\Big|_{t=0} + \frac{d}{dt}\left(\underline{\xi}(I - e^t Q)^{-1}\underline{1}\right)\Big|_{t=0}$$

$$\phi''(0) = \underline{\xi}(I - Q)^{-1}\underline{1} + 2\frac{d}{dt}\left(\underline{\xi}(I - e^t Q)^{-1}\underline{1}\right)\Big|_{t=0} \qquad \text{(A23)}$$

Focusing only on the term $\frac{d}{dt}\left(\underline{\xi}(I - e^t Q)^{-1}\underline{1}\right)\Big|_{t=0}$ we obtain:

$$\frac{d}{dt}\left(\underline{\xi}(I-e^tQ)^{-1}\underline{1}\right)\bigg|_{t=0}$$
$$=\frac{d}{dt}\left(\underline{\xi}\left(\sum_{n=0}^{\infty}e^{nt}Q^n\right)\underline{1}\right)\bigg|_{t=0}$$
$$=\underline{\xi}\left(\sum_{n=0}^{\infty}ne^{nt}Q^n\right)\underline{1}\bigg|_{t=0}$$
$$=\underline{\xi}(e^tQ+2e^{2t}Q^2+3e^{3t}Q^3+4e^{4t}Q^4+\ldots)\underline{1}\Big|_{t=0}$$
$$=\underline{\xi}(Q+2Q^2+3Q^3+4Q^4+\ldots)\underline{1}$$
$$=\underline{\xi}Q(I+2Q+3Q^2+4Q^3+\ldots)\underline{1}$$
$$=\underline{\xi}Q(I-Q)^{-2}\underline{1}\,.$$

By substituting $\frac{d}{dt}\left(\underline{\xi}(I-e^tQ)^{-1}\underline{1}\right)\Big|_{t=0}=\underline{\xi}Q(I-Q)^{-2}\underline{1}$ into expression (A23) we obtain

$$\phi''(0)=\underline{\xi}(I-Q)^{-1}\underline{1}+2\underline{\xi}Q(I-Q)^{-2}\underline{1}$$
$$=\underline{\xi}\left((I-Q)^{-1}+2Q(I-Q)^{-2}\right)\underline{1}$$
$$=\underline{\xi}\left((I-Q)^{-1}+2Q(I-Q)^{-1}(I-Q)^{-1}\right)\underline{1}$$
$$=\underline{\xi}\left((I-Q)^{-1}\left(I+2Q(I-Q)^{-1}\right)\right)\underline{1} \qquad \text{(A24)}$$
$$=\underline{\xi}\left((I-Q)^{-1}(I+Q)(I-Q)^{-1}\right)\underline{1} \qquad \text{(A25)}$$
$$=\underline{\xi}(I+Q)(I-Q)^{-2}\underline{1}\,.$$

Therefore we have that

$$\boxed{E\left(N^2\right)=\underline{\xi}(I+Q)(I-Q)^{-2}\underline{1}}$$

To get from expression (A24) to expression (A25) we used the following expansion

$$(I+Q)(I-Q)^{-1}$$
$$=(I+Q)(I+Q+Q^2+Q^3+\ldots)$$
$$=I+Q+Q^2+Q^3+\ldots+Q+Q^2+Q^3+\ldots$$
$$=I+2Q+2Q^2+2Q^3+2Q^4+\ldots$$
$$=I+2Q(I+Q+Q^2+Q^3+\ldots)$$
$$=I+2Q(I-Q)^{-1}\,.$$

Probability generating function

(See page 73 of Fu and Lou (2003))

If N is a discrete random variable taking values of non-negative integers $\{0,1,2,...\}$, then the pgf of N is defined as:

$$\varphi(x) = E\left(x^n\right) = \sum_{n=0}^{\infty} x^n P(N = n) \tag{A26}$$

The pgf is given by $\varphi(t) = 1 + (t-1)\underline{\xi}(I - tQ)^{-1}\underline{1}$. This is obtained by using the definition of the pgf given in expression (A26).

$$\begin{aligned}
\varphi(t) &= \sum_{n=1}^{\infty} t^n P(N = n) \\
&= \sum_{n=1}^{\infty} t^n \left(P(N > n-1) - P(N > n)\right) \\
&= \sum_{n=1}^{\infty} t^n \underline{\xi} Q^{n-1}\underline{1} - \sum_{n=1}^{\infty} t^n \underline{\xi} Q^n \underline{1} \\
&= t\sum_{n=1}^{\infty} t^{n-1} \underline{\xi} Q^{n-1}\underline{1} - \sum_{n=1}^{\infty} t^n \underline{\xi} Q^n \underline{1} \\
&= t\sum_{n=0}^{\infty} t^n \underline{\xi} Q^n \underline{1} - \sum_{n=0}^{\infty} t^n \underline{\xi} Q^n \underline{1} + 1
\end{aligned}$$

The scalar 1 is obtained from the fact that $t^0 \underline{\xi} Q^0 \underline{1} = 1$. By factorizing we obtain

$$\begin{aligned}
\varphi(t) &= 1 + (t-1)\left(\sum_{n=0}^{\infty} t^n \underline{\xi} Q^n \underline{1}\right) \\
&= 1 + (t-1)\underline{\xi}\left(\sum_{n=0}^{\infty} t^n Q^n\right)\underline{1} \\
&= 1 + (t-1)\underline{\xi}(I - tQ)^{-1}\underline{1}.
\end{aligned}$$

It is well-known that if the factorial generating function exists in an interval around $t = 1$, then the r^{th} factorial moment is given by $E\left((X)_r\right) = \varphi_X^{(r)}(1) = \left.\frac{d^r}{dt^r}\varphi_X(t)\right|_{t=1}$ where $(X)_r$ is the falling factorial $(x)_r = x(x-1)(x-2)...(x-r+1)$. Therefore, in order to find the *first factorial moment*, we will have to calculate the first order derivative of the pgf in the point $t = 1$, that is, $E(N) = \varphi'(1)$. This will give us the *first moment* of the run length variable N. Obtaining the second moment of the run length variable N is more difficult. Firstly, we have to find the *second factorial moment* by calculating the second order derivative of the pgf in the point

$t = 1$, that is, $E(N(N-1)) = \varphi''(1)$. Using the fact that $E(N(N-1)) = E(N^2) - E(N)$ we can obtain the *second momen*t of the run length variable N.

The first order derivative:

(Differentiation is done by using the well-known product rule).

$$\varphi'(t) = \underline{\xi}(I - tQ)^{-1}\underline{1} + (t-1)\frac{d}{dt}\left(\underline{\xi}(I - tQ)^{-1}\underline{1}\right)$$

At $t = 1$:

$$\varphi'(1) = \underline{\xi}(I - Q)^{-1}\underline{1}$$

Therefore we have that

$$E(N) = \underline{\xi}(I - Q)^{-1}\underline{1}. \qquad \text{(A27)}$$

The second order derivative:

(Differentiation is done by using the well-known product rule).

$$\varphi''(t) = \frac{d}{dt}\left(\underline{\xi}(I - tQ)^{-1}\underline{1}\right) + \frac{d}{dt}\left(\underline{\xi}(I - tQ)^{-1}\underline{1}\right) + (t-1)\frac{d^2}{dt^2}\left(\underline{\xi}(I - tQ)^{-1}\underline{1}\right)$$

$$\varphi''(t) = 2\frac{d}{dt}\left(\underline{\xi}(I - tQ)^{-1}\underline{1}\right) + (t-1)\frac{d^2}{dt^2}\left(\underline{\xi}(I - tQ)^{-1}\underline{1}\right)$$

At $t = 1$:

$$\begin{aligned}
\varphi''(1) &= 2\frac{d}{dt}\left(\underline{\xi}(I - tQ)^{-1}\underline{1}\right)\Big|_{t=1} \\
&= 2\frac{d}{dt}\left(\underline{\xi}\left(\sum_{n=0}^{\infty} t^n Q^n\right)\underline{1}\right)\Big|_{t=1} \\
&= 2\underline{\xi}\left(\sum_{n=0}^{\infty} n t^{n-1} Q^n\right)\underline{1}\Big|_{t=1} \\
&= 2\underline{\xi}\left(\sum_{n=0}^{\infty} n Q^n\right)\underline{1} \\
&= 2\underline{\xi}\left(0Q^0 + 1Q^1 + 2Q^2 + 3Q^3 + \ldots\right)\underline{1} \\
&= 2\underline{\xi}Q\left(I + 2Q + 3Q^2 + \ldots\right)\underline{1} \\
&= 2\underline{\xi}Q(I - Q)^{-2}\underline{1}.
\end{aligned}$$

Therefore we have that

$$E(N(N-1)) = 2\underline{\xi}Q(I - Q)^{-2}\underline{1}. \qquad \text{(A28)}$$

The second moment is derived by using the fact that

$$E(N(N-1)) = E(N^2) - E(N) \tag{A29}$$

Expression (A29) can be re-written as

$$E(N^2) = E(N) + E(N(N-1)) \tag{A30}$$

From (A27) and (A28) we know that $E(N) = \underline{\xi}(I-Q)^{-1}\underline{1}$ and $E(N(N-1)) = 2\underline{\xi}Q(I-Q)^{-2}\underline{1}$. By substituting this into expression (A30) we obtain $E(N^2) = \underline{\xi}(I-Q)^{-1}\underline{1} + 2\underline{\xi}Q(I-Q)^{-2}\underline{1}$. During the derivation of $E(N^2)$ in the mgf section we have shown that $\underline{\xi}(I-Q)^{-1}\underline{1} + 2\underline{\xi}Q(I-Q)^{-2}\underline{1} = \underline{\xi}(I+Q)(I-Q)^{-2}\underline{1}$. Thus

$$E(N^2) = \underline{\xi}(I+Q)(I-Q)^{-2}\underline{1}. \tag{A31}$$

Finally, though expressions (A27) and (A31) we have $E(N) = \underline{\xi}(I-Q)^{-1}\underline{1}$ and $E(N^2) = \underline{\xi}(I+Q)(I-Q)^{-2}\underline{1}$.

Theorem 3:

In Section 6.1.5 we state that the saddlepoint is the solution to the equation $m(t) = \mu$ where $m(t)$ is the first order derivative of the cumulant generating function (cgf) denoted by $\kappa(t)$. Therefore, the saddlepoint is the solution to the equation $\kappa'(t) = \mu$.

Proof to Theorem 3:

For the development of saddlepoint methodology see Daniels (1954) for details on density approximations, Lugannani and Rice (1980) and Daniels (1987) for discussions on tail area approximations and Reid (1988) for a review on saddlepoint techniques. Saddlepoint approximations are constructed by performing various operations on the moment generating function (mgf) and the cgf.

Let X be a random variable with a density function denoted by $f(x)$. Let $\phi(t)$ denote the mgf which is defined as $\phi(t) = \int_{-\infty}^{\infty} e^{tx} f(x)dx$. The cgf is just the logarithm of the mgf , i.e. $\kappa(t) = \log(\phi(t))$. From $\phi(t)$ we can obtain $f(x)$ by using the Fourier inversion formula as follows

$$f(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \phi(it)e^{-itx}dt = \frac{1}{2\pi}\int_{-i\infty}^{i\infty} e^{\kappa(t)-tx}dt \qquad \text{(A32)}$$

where $i = \sqrt{-1}$.

By differentiating the integral in (A32) and setting the result equal to zero we obtain

$$\kappa'(t) = x\,. \qquad \text{(A33)}$$

The solution to (A33) is called the saddlepoint and denoted by $\hat{t}$.

Daniels (1954) used the exponential power series expansion to estimate the integral in (A32) and derived the following approximation for $f(x)$

$$f(x) \approx \left(\frac{1}{2\pi\kappa''(\hat{t})}\right)^{\frac{1}{2}} e^{\kappa(\hat{t})-\hat{t}x} \qquad \text{(A34)}$$

Expression (A34) is referred to as the first-order saddlepoint density approximation where $\hat{t}$ is the unique solution to the saddlepoint equation $\kappa'(t) = x$.

For a rigorous account of the underlying mathematical theory of saddlepoint methods, interested readers can refer to Jensen (1995).

Appendix B: Computer programs

Mathcad program 1:

This program calculates the *ARL* and the probability of a signal for the upper one-sided Shewhart sign chart. . These values are shown in Tables 2.5, 2.6 and 2.7 for *n* = 5, 10 and 15, respectively.

$$\text{Probsignalupper}(n,p,a) := \left[\sum_{i=n-a}^{n} \text{combin}(n,i)\cdot p^{i}\cdot(1-p)^{n-i}\right]$$

$$\text{ARLupper}(n,p,a) := \frac{1}{\text{Probsignalupper}(n,p,a)}$$

$n := 10$

$q := 0.1, 0.2 .. 0.9$

$a := 0, 1 .. n$

$\text{Msignal}_{q\cdot 10,\, a} := \text{Probsignalupper}(n,q,a)$

$\text{Mupper}_{q\cdot 10,\, a} := \text{ARLupper}(n,q,a)$

Take note: The output is given in Tables 2.5, 2.6 and 2.7, respectively.

Mathcad program 2:

This program calculates the *ARL* and the probability of a signal for the lower one-sided Shewhart sign chart. These values are shown in Tables 2.5, 2.6 and 2.7 for *n* = 5, 10 and 15, respectively.

$$\text{Probsig}(n,p,a) := \sum_{i=0}^{a} \text{combin}(n,i)\cdot p^{i}\cdot(1-p)^{n-i}$$

$$\text{ARLlower}(n,p,a) := \frac{1}{\text{Probsig}(n,p,a)}$$

$q := 0.1, 0.2 .. 0.9$

$n := 10$

$a := 0, 1 .. n$

$\text{Msig}_{q\cdot 10, a} := \text{Probsig}(n,q,a)$

$\text{Mlower}_{q\cdot 10, a} := \text{ARLlower}(n,q,a)$

Take note: The output is given in Tables 2.5, 2.6 and 2.7, respectively.

Mathcad program 3:

This program calculates the *ARL's* for the upper- and lower one-sided and two-sided Shewhart sign charts with both warning and action limits.

Upper one-sided chart:

$$\text{p0upper}(n,p,w) := \sum_{i=0}^{\frac{(w+n-2)}{2}} \text{combin}(n,i)\cdot p^{i}(1-p)^{n-i}$$

$$\text{p1upper}(n,p,w,a) := \left[\sum_{i=0}^{\frac{(a+n-2)}{2}} \text{combin}(n,i)\cdot p^{i}\cdot(1-p)^{n-i}\right] - \left[\sum_{i=0}^{\frac{(w+n-2)}{2}} \text{combin}(n,i)\cdot p^{i}\cdot(1-p)^{n-i}\right]$$

$$\text{ARLupper}(n,p,w,a,r) := \frac{1-\text{p1upper}(n,p,w,a)^{r}}{\left[1-\text{p1upper}(n,p,w,a)-\text{p0upper}(n,p,w)\cdot\left(1-\text{p1upper}(n,p,w,a)^{r}\right)\right]}$$

$p := 0.5 \quad n := 10$

$a := n\cdot p .. n$

$w := n\cdot p .. n$

$RU1_{a,w} := \text{ARLupper}(n,p,w,a,1)$

Take note: The output is given in Table 2.10.

Lower one-sided chart:

$$\text{p0lower}(n,p,w) := 1 - \sum_{i=0}^{\frac{(n-w)}{2}} \text{combin}(n,i)\cdot p^{i}\cdot(1-p)^{n-i}$$

$$\text{p1lower}(n,p,w,a) := \left[\sum_{i=0}^{\frac{(n-w)}{2}} \text{combin}(n,i)\cdot p^{i}\cdot(1-p)^{n-i}\right] - \left[\sum_{i=0}^{\frac{(n-a)}{2}} \text{combin}(n,i)\cdot p^{i}(1-p)^{n-i}\right]$$

$$\text{ARLlower}(n,p,w,a,r) := \frac{\left(1-\text{p1lower}(n,p,w,a)^{r}\right)}{1-\text{p1lower}(n,p,w,a)-\text{p0lower}(n,p,w)\cdot\left(1-\text{p1lower}(n,p,w,a)^{r}\right)}$$

$RL1_{a,w} := \text{ARLlower}(n,p,w,a,1)$

Take note: The output is given in Table 2.11.

Two-sided chart:

$$\text{ARLtwo}(n,p,w,a,r) := \frac{\text{ARLlower}(n,p,w,a,r)\cdot\text{ARLupper}(n,p,w,a,r)}{\text{ARLlower}(n,p,w,a,r)+\text{ARLupper}(n,p,w,a,r)}$$

$R1_{a,w} := \text{ARLtwo}(n,p,w,a,1)$

Take note: The output is given in Table 2.12.

SAS program 1:

This program calculates the SN_i and T_i statistics shown in Table 2.3.

```
proc iml;

* Number of samples;
nn=15;

* Sample size;
n=5;

signmatrix=j(nn,n,0);
timatrix=j(nn,n,0);

* The known median;
tv = 74;

* The matrix containing the Montgomery (2001) Table 5.2 piston ring data;
matrix = {
74.012 74.015 74.030 73.986 74.000, 73.995 74.010 73.990 74.015 74.001,
73.987 73.999 73.985 74.000 73.990, 74.008 74.010 74.003 73.991 74.006,
74.003 74.000 74.001 73.986 73.997, 73.994 74.003 74.015 74.020 74.004,
74.008 74.002 74.018 73.995 74.005, 74.001 74.004 73.990 73.996 73.998,
74.015 74.000 74.016 74.025 74.000, 74.030 74.005 74.000 74.016 74.012,
74.001 73.990 73.995 74.010 74.024, 74.015 74.020 74.024 74.005 74.019,
74.035 74.010 74.012 74.015 74.026, 74.017 74.013 74.036 74.025 74.026,
74.010 74.005 74.029 74.000 74.020};

* Calculating the SNi statistics;
do k = 1 to nn;
      do l = 1 to n;
            if matrix[k,l]>tv then signmatrix[k,l]=1;
            else if matrix[k,l]<tv then signmatrix[k,l]=-1;
            else signmatrix[k,l]=0;
            end;
end;

signvec=signmatrix[,+];

* Calculating the Ti statistics;
do k = 1 to nn;
      do l = 1 to n;
            if matrix[k,l]>tv then timatrix[k,l]=1;
            else timatrix[k,l]=0;
      end;
end;

tivec=timatrix[,+];
si_ti=signvec||tivec;
create newdata from si_ti[colname = {"Si" "Ti"}];
append from si_ti;

proc print data=newdata;
run;
```

SAS program 2:

This program calculates the *ARL*, *SDRL*, 5th, 25th, 50th, 75th and 95th percentile values for the upper one-sided CUSUM sign chart.

Take note: The programs for the lower one-sided and two-sided CUSUM sign charts are omitted, since they are very similar to this program and can easily be obtained by making minor alterations to this program.

```
proc iml;

* For n even, the reference value k is taken to be even;
* For n odd, the reference value k is taken to be odd;
* Restriction h <= n-k;

* The reference value;
k=1;

* The decision interval;
h=4;

* The sample size;
n=5;

* The z-value will be used in the calculation of the pmf, P(N=z), and the cdf,
P(N<=z);
z=10000;

* The following values will be used to calculate the 5th, 25th, 50th, 75th and
95th percentiles, respectively;
p5p=0.05;
p25p=0.25;
p50p=0.5;
p75p=0.75;
p95p=0.95;

* Calculating the state space;
SRn =do(-n,n,2)`;
S=j(nrow(SRn),1,1);

do i = 1 to nrow(SRn);
      S[i,]=min(h,(max(0,SRn[i,]-k)));
end;

do i = 1 to nrow(S);
      do j = 1 to nrow(S);
            if i=j then S[i,]=S[i,];
            else if S[i,]=S[j,] then S[j,]=999;
      end;
end;

S=S[loc(S<999)];

* Defining the vector eta used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
```

```
eta=j(1,nrow(S)-1,1);

do i = 1 to nrow(S)-1;
      if i = 1 then eta[1,i]=1;
      else eta[1,i]=0;
end;

* Calculating the transition probability matrix;
P = j( nrow(S) , nrow(S), 0);
T = j(n+1,1,1);

do x = 0 to n;
      T[x+1,]=pdf('BINOM',x,0.5,n);
end;

* Calculating the first column of the transition matrix;
do i = 1 to nrow(S)-1;
      small_t = (k - S[i,] + n)/2;
      P[i,1] = sum(T[1:(small_t + 1),]);
end;

* Calculating the middle columns of the transition matrix;
do j = 2 to nrow(S)-1;
      do i = 1 to nrow(S)-1;
            small_t=ceil((S[j,] + k - S[i,] + n) / 2);
            P[i,j]=T[small_t+1,];
      end;
end;

* Calculating the last column of the transition matrix;
do i = 1 to nrow(S)-1;
      P[i,nrow(S)] = 1 - sum ( P[i, 1:(nrow(S)-1)] );
end;

P[nrow(S),nrow(S)]=1;

* Defining the vector one used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
one = j(nrow(S)-1,1,1);

* Defining the matrix Q used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
Q = P[1:nrow(S)-1,1:nrow(S)-1];

* Defining the identity matrix I used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
identity = I(nrow(S)-1);

* Calculating the 5th, 25th, 50th, 75th and 95th percentiles;
pmf=j(z,1,1);
cdf=j(z,1,1);
cdf_5th_p=j(z,1,1);
cdf_25th_p=j(z,1,1);
cdf_50th_p=j(z,1,1);
cdf_75th_p=j(z,1,1);
cdf_95th_p=j(z,1,1);
```

```
do i = 1 to z;
      pmf[i,1] = eta * (Q**(i-1)) * (identity - Q) * one;
      cdf[i,1]=sum(pmf[1:i,1]);
end;

index=j(z,1,1);

do i = 2 to z;
      index[i,]=index[i-1,]+1;
end;

* Calculating the 5th percentile;
do i = 1 to z;
      cdf_5th_p[i,]=cdf[i,];
      if cdf_5th_p[i,]>=p5p then cdf_5th_p[i,]=999;
end;

cdf_5th_p=cdf_5th_p[loc(cdf_5th_p<999)];
if cdf_5th_p[1,]=999 then percentile_p5p=1;
else percentile_p5p=nrow(cdf_5th_p)+1;

* Calculating the 25th percentile;
do i = 1 to z;
      cdf_25th_p[i,]=cdf[i,];
      if cdf_25th_p[i,]>=p25p then cdf_25th_p[i,]=999;
end;

cdf_25th_p=cdf_25th_p[loc(cdf_25th_p<999)];
if cdf_25th_p[1,]=999 then percentile_p25p=1;
else percentile_p25p=nrow(cdf_25th_p)+1;

* Calculating the 50th percentile;
do i = 1 to z;
      cdf_50th_p[i,]=cdf[i,];
      if cdf_50th_p[i,]>=p50p then cdf_50th_p[i,]=999;
end;

cdf_50th_p=cdf_50th_p[loc(cdf_50th_p<999)];
if cdf_50th_p[1,]=999 then percentile_p50p=1;
else percentile_p50p=nrow(cdf_50th_p)+1;

* Calculating the 75th percentile;
do i = 1 to z;
      cdf_75th_p[i,]=cdf[i,];
      if cdf_75th_p[i,]>=p75p then cdf_75th_p[i,]=999;
end;

cdf_75th_p=cdf_75th_p[loc(cdf_75th_p<999)];
if cdf_75th_p[1,]=999 then percentile_p75p=1;
else percentile_p75p=nrow(cdf_75th_p)+1;

* Calculating the 95th percentile;
do i = 1 to z;
      cdf_95th_p[i,]=cdf[i,];
      if cdf_95th_p[i,]>=p95p then cdf_95th_p[i,]=999;
end;
```

```
cdf_95th_p=cdf_95th_p[loc(cdf_95th_p<999)];
if cdf_95th_p[1,]=999 then percentile_p95p=1;
else percentile_p95p=nrow(cdf_95th_p)+1;

* Calculating the average run length (ARL);
ARL = eta * inv(identity-Q) * one;

* Calculating the second moment;
N2 = eta * (identity + Q) * (inv((identity-Q)**2)) * one;

* Calculating the standard deviation;
SDRL = sqrt (N2 - ((ARL)**2) );

* Calculating the two-sided ARL;
ARL_two=(ARL*ARL)/(ARL+ARL);

* Printing the output;
print_cdf=index||cdf;
print_pmf=index||pmf;

print   k [label='Reference value']
        , h [label='Desicion interval']
        , n [label='Sample size']
        , S [label = 'State Space']
        , P [label='Transition probability matrix' format=.3]
        , ARL [label='Average run length' format=.2]
        , ARL_two [label = 'The ARL of the two-sided chart' format=.2]
        , SDRL [label='Standard Deviation of the run length' format=.2]
        , N2 [label='Second moment' format=.2]
        , percentile_p5p [label='Fifth percentile']
        , percentile_p25p [label='25th percentile']
        , percentile_p50p [label='50th percentile']
        , percentile_p75p [label='75th percentile']
        , percentile_p95p [label='95th percentile'];
run;
```

SAS program 3:

This program calculates the sign test statistics (SN_i), T_i-statistics, upper CUSUM statistics (S_i^+) and lower CUSUM statistics (S_i^-) for the Montgomery (2001) piston ring data.

```
proc iml;

* Number of samples;
nn=15;

* Sample size;
n=5;

* The known median;
tv = 74;

signmatrix=j(nn,n,0);
timatrix=j(nn,n,0);

* A matrix containing the Montgomery (2001) Table 5.2 piston ring data;
matrix =
{74.012 74.015 74.030 73.986 74.000,
73.995 74.010 73.990 74.015 74.001,
73.987 73.999 73.985 74.000 73.990,
74.008 74.010 74.003 73.991 74.006,
74.003 74.000 74.001 73.986 73.997,
73.994 74.003 74.015 74.020 74.004,
74.008 74.002 74.018 73.995 74.005,
74.001 74.004 73.990 73.996 73.998,
74.015 74.000 74.016 74.025 74.000,
74.030 74.005 74.000 74.016 74.012,
74.001 73.990 73.995 74.010 74.024,
74.015 74.020 74.024 74.005 74.019,
74.035 74.010 74.012 74.015 74.026,
74.017 74.013 74.036 74.025 74.026,
74.010 74.005 74.029 74.000 74.020};

* Calculating the sign test statistics, SNi;
do k = 1 to nn;
      do l = 1 to n;
            if matrix[k,l]>tv then signmatrix[k,l]=1;
            else if matrix[k,l]<tv then signmatrix[k,l]=-1;
            else signmatrix[k,l]=0;
            end;
end;

signvec=signmatrix[,+];

* Calculating the Ti statistics;
do k = 1 to nn;
      do l = 1 to n;
      if matrix[k,l]>tv then timatrix[k,l]=1;
      else timatrix[k,l]=0;
      end;
end;
```

```
tivec=timatrix[,+];

* Calculating the CUSUM statistics;

* Specifying the reference value;
k=2;

* The starting values are set equal to zero;
SNpluszero=0;
SNminuszero=0;

SNplus=j(nn,1,0);
SNminus=j(nn,1,0);

do l = 2 to nn;
      SNplus[1,]=max(0,SNpluszero+(signvec[1,])-k);
      SNminus[1,]=min(0,SNminuszero+(signvec[1,])+k);
      SNplus[l,]=max(0,SNplus[l-1,]+(signvec[l,])-k);
      SNminus[l,]=min(0,SNminus[l-1,]+(signvec[l,])+k);
end;

Nplus=j(nn,1,0);
Nminus=j(nn,1,0);

do l = 1 to nn;
      if SNplus[l,]=0 then Nplus[l,]=0;
      else Nplus[l,]=Nplus[l-1,]+1;
end;

do l = 1 to nn;
      if SNminus[l,]=0 then Nminus[l,]=0;
      else Nminus[l,]=Nminus[l-1,]+1;
end;

* Printing the output;
print signvec [label='The SNi statistics'],
      tivec [label='The Ti statistics'],
      SNplus [label='The upper CUSUM statistics'],
      SNminus [label='The lower CUSUM statistics'];
```

SAS program 4:

This program calculates the *ARL*, *SDRL*, 5th, 25th, 50th, 75th and 95th percentile values for the EWMA sign chart.

```
proc iml;

* Number of subintervals between UCL and LCL;
NN=5;

* Sample size;
n=6;

* p=0.5 when the process is in-control;
p=0.5;

* The EWMA parameter: the multiplier;
L=2;

* The EWMA parameter: the smoothing constant;
lambda=1;

* The z-value will be used in the calculation of the percentiles;
z=10000;

* Calculating the control limits;
UCL = L * 2 * sqrt(n*p*(1-p)) * sqrt(lambda/(2-lambda));
LCL = -UCL;
S=j(NN,1,0);

* The interval between the UCL and LCL are divided into subintervals of width
2*delta;
delta = ((UCL-LCL)/NN)/2;
S[1,1] = UCL - delta;

do i = 2 to NN;
      S[i,1]=S[i-1,1]-2*delta;
end;

Q_a=j(NN,NN,0);
Q_b=j(NN,NN,0);
Q=j(NN,NN,0);

do i = 1 to NN;
      do j = 1 to NN;
      Q_a[i,j]=floor(((((S[j,]-delta) - (1-lambda)*S[i,])/lambda) + n)/2);
      end;
end;

do i = 1 to NN;
      do j = 1 to NN;
      Q_b[i,j]=floor(((((S[j,]+delta) - (1-lambda)*S[i,])/lambda) + n)/2);
      end;
end;

do i = 1 to NN;
```

```
		do j = 1 to NN;
		Q[i,j]=cdf('BINOMIAL',(Q_b[i,j]),p,n)-cdf('BINOMIAL',(Q_a[i,j]),p,n);
		end;
end;

eta=j(1,NN,1);
do i = 1 to NN;
		if i = 1 then eta[1,i]=1;
		else eta[1,i]=0;
end;

* Defining the vector one used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
one=j(NN,1,1);

* Defining the identity matrix I used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
identity = I(NN);

* Calculating the 5th, 25th, 50th, 75th and 95th percentiles;
p5p=0.05;
p25p=0.25;
p50p=0.5;
p75p=0.75;
p95p=0.95;
pmf=j(z,1,1);
cdf=j(z,1,1);
cdf_5th_p=j(z,1,1);
cdf_25th_p=j(z,1,1);
cdf_50th_p=j(z,1,1);
cdf_75th_p=j(z,1,1);
cdf_95th_p=j(z,1,1);

do i = 1 to z;
		pmf[i,1] = eta * (Q**(i-1)) * (identity - Q) * one;
		cdf[i,1]=sum(pmf[1:i,1]);
end;

index=j(z,1,1);

do i = 2 to z;
		index[i,]=index[i-1,]+1;
end;

* Calculating the 5th percentile;
do i = 1 to z;
		cdf_5th_p[i,]=cdf[i,];
		if cdf_5th_p[i,]>=p5p then cdf_5th_p[i,]=999;
end;
cdf_5th_p=cdf_5th_p[loc(cdf_5th_p<999)];
if cdf_5th_p[1,]=999 then percentile_p5p=1;
else percentile_p5p=nrow(cdf_5th_p)+1;

* Calculating the 25th percentile;
do i = 1 to z;
		cdf_25th_p[i,]=cdf[i,];
		if cdf_25th_p[i,]>=p25p then cdf_25th_p[i,]=999;
```

```
end;
cdf_25th_p=cdf_25th_p[loc(cdf_25th_p<999)];
if cdf_25th_p[1,]=999 then percentile_p25p=1;
else percentile_p25p=nrow(cdf_25th_p)+1;

* Calculating the 50th percentile;
do i = 1 to z;
      cdf_50th_p[i,]=cdf[i,];
      if cdf_50th_p[i,]>=p50p then cdf_50th_p[i,]=999;
end;
cdf_50th_p=cdf_50th_p[loc(cdf_50th_p<999)];
if cdf_50th_p[1,]=999 then percentile_p50p=1;
else percentile_p50p=nrow(cdf_50th_p)+1;

* Calculating the 75th percentile;
do i = 1 to z;
      cdf_75th_p[i,]=cdf[i,];
      if cdf_75th_p[i,]>=p75p then cdf_75th_p[i,]=999;
end;
cdf_75th_p=cdf_75th_p[loc(cdf_75th_p<999)];
if cdf_75th_p[1,]=999 then percentile_p75p=1;
else percentile_p75p=nrow(cdf_75th_p)+1;

* Calculating the 95th percentile;
do i = 1 to z;
      cdf_95th_p[i,]=cdf[i,];
      if cdf_95th_p[i,]>=p95p then cdf_95th_p[i,]=999;
end;
cdf_95th_p=cdf_95th_p[loc(cdf_95th_p<999)];
if cdf_95th_p[1,]=999 then percentile_p95p=1;
else percentile_p95p=nrow(cdf_95th_p)+1;

* Calculating the average run length (ARL);
ARL = eta*ginv(identity-Q)*one;

* Calculating the second moment;
N2 = eta * (identity + Q) * (ginv((identity-Q)**2)) * one;

* Calculating the standard deviation;
SDRL = sqrt (N2 - ((ARL)**2) );

* Printing the output;
print_cdf=index||cdf;
print_pmf=index||pmf;

print       UCL [label='Upper control limit'],
            LCL [label='Lower control limit'],
            delta,
            Q,
            ARL [label = 'Average run length' format=.2]
            SDRL [label = 'Standard deviation of the run length' format=.2],
            percentile_p5p [label='Fifth percentile'],
            percentile_p25p [label='25th percentile'],
            percentile_p50p [label='50th percentile'],
            percentile_p75p [label='75th percentile'],
            percentile_p95p [label='95th percentile'];
```

SAS program 5:

This program calculates the signed-rank (SR_i) statistics for the Shewhart-type signed-rank chart using the Montgomery (2001) piston ring data.

```
proc iml;

* Number of samples;
nn=15;

* Sample size;
n=5;

wsrmatrix = j(nn,n,0);
sgnmatrix = j(nn,n,0);
rank_abs_diff = j(nn,n,0);
final_rank_abs_diff=j(nn,n,0);

* The known median;
tv=74;

tvmatrix = j(nn,n,tv);

* A matrix containing the Montgomery (2001) Table 5.2 piston ring data;
obs = {
74.012 74.015 74.030 73.986 74.000,
73.995 74.010 73.990 74.015 74.001,
73.987 73.999 73.985 74.000 73.990,
74.008 74.010 74.003 73.991 74.006,
74.003 74.000 74.001 73.986 73.997,
73.994 74.003 74.015 74.020 74.004,
74.008 74.002 74.018 73.995 74.005,
74.001 74.004 73.990 73.996 73.998,
74.015 74.000 74.016 74.025 74.000,
74.030 74.005 74.000 74.016 74.012,
74.001 73.990 73.995 74.010 74.024,
74.015 74.020 74.024 74.005 74.019,
74.035 74.010 74.012 74.015 74.026,
74.017 74.013 74.036 74.025 74.026,
74.010 74.005 74.029 74.000 74.020};

* Calculating the SRi statistics;
diff=obs-tvmatrix;

do k = 1 to nn;
      do l = 1 to n;
      if diff[k,l]>0 then sgnmatrix[k,l]=1;
      else if diff[k,l]<0 then sgnmatrix[k,l]=-1;
      else if diff[k,l]=0 then sgnmatrix[k,l]=0;
      end;
end;

abs_diff=abs(diff);

do i = 1 to nn;
      rank_abs_diff[i,]=rank(abs_diff[i,]);
```

```
end;

do i = 1 to nn;
      do j = 1 to n;
            do k = 1 to n;
            if abs_diff[i,j]=abs_diff[i,k] then

      final_rank_abs_diff[i,j]=(rank_abs_diff[i,j]+rank_abs_diff[i,k])/2;
            end;
      end;
end;

do i = 1 to nn;
      do j = 1 to n;
            do k = 1 to n;
            if abs_diff[i,j]=abs_diff[i,k] then
            final_rank_abs_diff[i,k]=final_rank_abs_diff[i,j];
            end;
      end;
end;


do k = 1 to nn;
      do l = 1 to n;
            wsrmatrix[k,l]=sgnmatrix[k,l]*final_rank_abs_diff[k,l];
      end;
end;

SRi = wsrmatrix[,+];

* Printing the output;
print SRi [label='The SRi statistics'];
```

SAS program 6:

This program calculates the $FAR's$ and $ARL_0's$ for the two-sided Shewhart signed-rank chart and the two-sided Shewhart signed-rank-like chart, respectively.

```
proc iml;

*Number of simulations;
numsim=10000;

ARLmatrix=j(numsim,1,0);

* Starting the simulation study;
do ss = 1 to numsim;

* The upper control limit;
ucl = 10;
ucl = ucl-1;

* The lower control limit;
lcl = -ucl;

* Number of samples (must theoretically go to infinity;
nn=10000;

* Sample size;
n=10;

wsrmatrix = j(nn,n,0);
sgnmatrix = j(nn,n,0);
rank_abs_diff = j(nn,n,0);
final_rank_abs_diff=j(nn,n,0);

* The median of the standard normal distribution;
tv=0;

tvmatrix = j(nn,n,tv);
obs = j(nn,n,0);

* Genrating oservations from a standard normal distribution;
call randgen(obs,'normal');

diff=obs-tvmatrix;

do k = 1 to nn;
      do l = 1 to n;
            if diff[k,l]>0 then sgnmatrix[k,l]=1;
            else if diff[k,l]<0 then sgnmatrix[k,l]=-1;
            else if diff[k,l]=0 then sgnmatrix[k,l]=0;
      end;
end;

abs_diff=abs(diff);

do i = 1 to nn;
```

```
        rank_abs_diff[i,]=rank(abs_diff[i,]);
end;

do i = 1 to nn;
        do j = 1 to n;
                do k = 1 to n;
                        if abs_diff[i,j]=abs_diff[i,k] then

        final_rank_abs_diff[i,j]=(rank_abs_diff[i,j]+rank_abs_diff[i,k])/2;
                end;
        end;
end;

do i = 1 to nn;
        do j = 1 to n;
                do k = 1 to n;
                        if abs_diff[i,j]=abs_diff[i,k] then
                        final_rank_abs_diff[i,k]=final_rank_abs_diff[i,j];
                end;
        end;
end;


do k = 1 to nn;
        do l = 1 to n;
                wsrmatrix[k,l]=sgnmatrix[k,l]*final_rank_abs_diff[k,l];
        end;
end;

SRi = wsrmatrix[,+];
count = j(nn,1,0);

do l = 1 to nn;
        if SRi[l,]>=ucl then count[l,]=999;
        if SRi[l,]<=lcl then count[l,]=999;
end;

do ll = 1 to nn;
        if count[ll,]=999 then goto skip;
end;

skip: ARL = ll;
ARLmatrix[ss,1]=ARL;
end;

* The simulated average run length (ARL);
simulatedARL = ARLmatrix[+,]/nrow(ARLmatrix);

* The simulated false alarm rate (FAR);
FAR = 1/simulatedARL;

ucl = ucl+1;

print           n [label='Sample size'],
                ucl [label = 'Upper control limit'],
                simulatedARL [label = 'ARL' format=.3],
                FAR [label = 'FAR' format=.3];
```

SAS program 7:

This program calculates the *ARL*, *SDRL*, 5th, 25th, 50th, 75th and 95th percentile values for the upper one-sided CUSUM signed-rank chart.

Take note: The programs for the lower one-sided and two-sided CUSUM signed-rank charts are omitted, since they are very similar to this program and can easily be obtained by making minor alterations to this program.

```
proc iml;

* The reference value;
k=2;

* The decision interval;
h=8;

* The sample size;
n=4;

* The z-value will be used in the calculation of the pmf, P(N=z), and the cdf,
P(N<=z);
z=10000;

* The following values will be used to calculate the 5th, 25th, 50th, 75th and
95th percentiles, respectively;
p5p=0.05;
p25p=0.25;
p50p=0.5;
p75p=0.75;
p95p=0.95;

* Calculating the state space;
SRn =do((-n*(n+1)/2),(n*(n+1)/2),2)`;
S=j(nrow(SRn),1,1);

do i = 1 to nrow(SRn);
      S[i,]=min(h,(max(0,SRn[i,]-k)));
end;

do i = 1 to nrow(S);
      do j = 1 to nrow(S);
            if i=j then S[i,]=S[i,];
            else if S[i,]=S[j,] then S[j,]=999;
      end;
end;

S=S[loc(S<999)];

if h > (SRn[nrow(SRn),]-k) then print "Not possible";

* Defining the vector eta used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
eta=j(1,nrow(S)-1,1);
```

```
do i = 1 to nrow(S)-1;
      if i = 1 then eta[1,i]=1;
      else eta[1,i]=0;
end;

* Calculating the transition probability matrix;
P = j( nrow(S) , nrow(S), 0);

* Wilcoxon siged-rank probabilities for a sample size of 4;
if n = 4 then do;
      T4 = j((n*(n+1)/2)+1,1,1);
      T4[1:3,]=1;
      T4[4:8,]=2;
      T4[9:11,]=1;
      T = T4/(2**n);
end;

* Wilcoxon siged-rank probabilities for a sample size of 5;
if n = 5 then do;
      T5 = j((n*(n+1)/2)+1,1,1);
      T5[1:3,]=1;
      T5[4:5,]=2;
      T5[6:11,]=3;
      T5[12:13,]=2;
      T5[14:16,]=1;
      T = T5/(2**n);
end;

* Wilcoxon siged-rank probabilities for a sample size of 6;
if n = 6 then do;
      T6 = j((n*(n+1)/2)+1,1,1);
      T6[1:3,]=1;
      T6[4:5,]=2;
      T6[6,]=3;
      T6[7:9,]=4;
      T6[10:13,]=5;
      T6[14:16,]=4;
      T6[17,]=3;
      T6[18:19,]=2;
      T6[20:22,]=1;
      T = T6/(2**n);
end;


* Wilcoxon siged-rank probabilities for a sample size of 7;
if n = 7 then do;
      T7 = j((n*(n+1)/2)+1,1,1);
      T7[1:3,]=1;
      T7[4:5,]=2;
      T7[6,]=3;
      T7[7,]=4;
      T7[8:9,]=5;
      T7[10,]=6;
      T7[11:12,]=7;
      T7[13:17,]=8;
      T7[18:19,]=7;
      T7[20,]=6;
```

```
      T7[21:22,]=5;
      T7[23,]=4;
      T7[24,]=3;
      T7[25:26,]=2;
      T7[27:29,]=1;
      T = T7/(2**n);
end;

* Wilcoxon siged-rank probabilities for a sample size of 8;
if n = 8 then do;
      T8 = j((n*(n+1)/2)+1,1,1);
      T8[1:3,]=1;
      T8[4:5,]=2;
      T8[6,]=3;
      T8[7,]=4;
      T8[8,]=5;
      T8[9,]=6;
      T8[10,]=7;
      T8[11,]=8;
      T8[12,]=9;
      T8[13,]=10;
      T8[14,]=11;
      T8[15,]=12;
      T8[16:18,]=13;
      T8[19,]=14;
      T8[20:22,]=13;
      T8[23,]=12;
      T8[24,]=11;
      T8[25,]=10;
      T8[26,]=9;
      T8[27,]=8;
      T8[28,]=7;
      T8[29,]=6;
      T8[30,]=5;
      T8[31,]=4;
      T8[32,]=3;
      T8[33:34,]=2;
      T8[35:37,]=1;
      T = T8/(2**n);
end;

* Wilcoxon siged-rank probabilities for a sample size of 9;
if n = 9 then do;
      T9 = j((n*(n+1)/2)+1,1,1);
      T9[1:3,]=1;
      T9[4:5,]=2;
      T9[6,]=3;
      T9[7,]=4;
      T9[8,]=5;
      T9[9,]=6;
      T9[10,]=8;
      T9[11,]=9;
      T9[12,]=10;
      T9[13,]=12;
      T9[14,]=13;
      T9[15,]=15;
      T9[16,]=17;
```

```
      T9[17,]=18;
      T9[18,]=19;
      T9[19:20,]=21;
      T9[21,]=22;
      T9[22:25,]=23;
      T9[26,]=22;
      T9[27:28,]=21;
      T9[29,]=19;
      T9[30,]=18;
      T9[31,]=17;
      T9[32,]=15;
      T9[33,]=13;
      T9[34,]=12;
      T9[35,]=10;
      T9[36,]=9;
      T9[37,]=8;
      T9[38,]=6;
      T9[39,]=5;
      T9[40,]=4;
      T9[41,]=3;
      T9[42:43,]=2;
      T9[44:46,]=1;
      T = T9/(2**n);
end;

* Wilcoxon siged-rank probabilities for a sample size of 10;
if n = 10 then do;
      T10 = j((n*(n+1)/2)+1,1,1);
      T10[1:3,]=1;
      T10[4:5,]=2;
      T10[6,]=3;
      T10[7,]=4;
      T10[8,]=5;
      T10[9,]=6;
      T10[10,]=8;
      T10[11,]=10;
      T10[12,]=11;
      T10[13,]=13;
      T10[14,]=15;
      T10[15,]=17;
      T10[16,]=20;
      T10[17,]=22;
      T10[18,]=24;
      T10[19,]=27;
      T10[20,]=29;
      T10[21,]=31;
      T10[22,]=33;
      T10[23,]=35;
      T10[24,]=36;
      T10[25,]=38;
      T10[26:27,]=39;
      T10[28:29,]=40;
      T10[30:31,]=39;
      T10[32,]=38;
      T10[33,]=36;
      T10[34,]=35;
      T10[35,]=33;
```

```
        T10[36,]=31;
        T10[37,]=29;
        T10[38,]=27;
        T10[39,]=24;
        T10[40,]=22;
        T10[41,]=20;
        T10[42,]=17;
        T10[43,]=15;
        T10[44,]=13;
        T10[45,]=11;
        T10[46,]=10;
        T10[47,]=8;
        T10[48,]=6;
        T10[49,]=5;
        T10[50,]=4;
        T10[51,]=3;
        T10[52:53,]=2;
        T10[54:56,]=1;
        T = T10/(2**n);
end;

* Calculating the first column of the transition matrix;
do i = 1 to nrow(S)-1;
        small_t = (k - S[i,] + (n*(n+1)/2) ) / 2;
        P[i,1] = sum(T[1:(small_t + 1),]);
end;

* Calculating the middle columns of the transition matrix;
do j = 2 to nrow(S)-1;
        do i = 1 to nrow(S)-1;
                small_t=(S[j,] + k - S[i,] + (n*(n+1)/2)) / 2;
                P[i,j] = T[small_t + 1,];
        end;
end;

* Calculating the last column of the transition matrix;
do i = 1 to nrow(S)-1;
        small_t=((S[nrow(S),] + k - S[i,] + (n*(n+1)/2)) / 2)-1;
        P[i,nrow(S)] = 1- sum(T[ 1:(small_t + 1),]);
end;

P[nrow(S),nrow(S)]=1;

* Defining the vector one used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
one = j(nrow(S)-1,1,1);

* Defining the matrix Q used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
Q = P[1:nrow(S)-1,1:nrow(S)-1];

* Defining the identity matrix I used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
identity = I(nrow(S)-1);

* Calculating the 5th, 25th, 50th, 75th and 95th percentiles;
pmf=j(z,1,1);
```

```
cdf=j(z,1,1);
cdf_5th_p=j(z,1,1);
cdf_25th_p=j(z,1,1);
cdf_50th_p=j(z,1,1);
cdf_75th_p=j(z,1,1);
cdf_95th_p=j(z,1,1);

do i = 1 to z;
      pmf[i,1] = eta * (Q**(i-1)) * (identity - Q) * one;
      cdf[i,1]=sum(pmf[1:i,1]);
end;

index=j(z,1,1);

do i = 2 to z;
      index[i,]=index[i-1,]+1;
end;

* Calculating the 5th percentile;
do i = 1 to z;
      cdf_5th_p[i,]=cdf[i,];
      if cdf_5th_p[i,]>=p5p then cdf_5th_p[i,]=999;
end;

cdf_5th_p=cdf_5th_p[loc(cdf_5th_p<999)];

if cdf_5th_p[1,]=999 then percentile_p5p=1;
else percentile_p5p=nrow(cdf_5th_p)+1;

* Calculating the 25th percentile;
do i = 1 to z;
      cdf_25th_p[i,]=cdf[i,];
      if cdf_25th_p[i,]>=p25p then cdf_25th_p[i,]=999;
end;

cdf_25th_p=cdf_25th_p[loc(cdf_25th_p<999)];

if cdf_25th_p[1,]=999 then percentile_p25p=1;
else percentile_p25p=nrow(cdf_25th_p)+1;

* Calculating the 50th percentile;
do i = 1 to z;
      cdf_50th_p[i,]=cdf[i,];
      if cdf_50th_p[i,]>=p50p then cdf_50th_p[i,]=999;
end;

cdf_50th_p=cdf_50th_p[loc(cdf_50th_p<999)];

if cdf_50th_p[1,]=999 then percentile_p50p=1;
else percentile_p50p=nrow(cdf_50th_p)+1;

* Calculating the 75th percentile;
do i = 1 to z;
      cdf_75th_p[i,]=cdf[i,];
      if cdf_75th_p[i,]>=p75p then cdf_75th_p[i,]=999;
end;
```

```
cdf_75th_p=cdf_75th_p[loc(cdf_75th_p<999)];

if cdf_75th_p[1,]=999 then percentile_p75p=1;
else percentile_p75p=nrow(cdf_75th_p)+1;

* Calculating the 95th percentile;
do i = 1 to z;
      cdf_95th_p[i,]=cdf[i,];
      if cdf_95th_p[i,]>=p95p then cdf_95th_p[i,]=999;
end;

cdf_95th_p=cdf_95th_p[loc(cdf_95th_p<999)];

if cdf_95th_p[1,]=999 then percentile_p95p=1;
else percentile_p95p=nrow(cdf_95th_p)+1;

* Calculating the average run length (ARL);
ARL = eta * inv(identity-Q) * one;

* Calculating the second moment;
N2 = eta * (identity + Q) * (inv((identity-Q)**2)) * one;

* Calculating the standard deviation;
SDRL = sqrt (N2 - ((ARL)**2) );

* Printing the output;
print_cdf=index||cdf;
print_pmf=index||pmf;

print       k [label='Reference value'],
            h [label='Desicion interval'],
            n [label='Sample size'],
            S [label = 'State Space'],
            P [label='Transition probability matrix' format=fract.],
            ARL [label='ARL' format=.2],
            SDRL [label='SDRL' format=.2],
            percentile_p5p [label='5th'],
            percentile_p25p [label='25th'],
            percentile_p50p [label='50th'],
            percentile_p75p [label='75th'],
            percentile_p95p [label='95th'];
run;
```

SAS program 8:

This program calculates the *ARL*, *SDRL*, 5th, 25th, 50th, 75th and 95th percentile values for the EWMA signed-rank chart.

```
proc iml;

* Number of subintervals between UCL and LCL;
NN=5;

* Sample size;
n=10;

* p=0.5 when the process is in-control;
p=0.5;

* The EWMA parameter: the multiplier;
L=1;

* The EWMA parameter: the smoothing constant;
lambda=0.2;

* The z-value will be used in the calculation of the percentiles;
z=10000;

* The in-control standard devation of the signed-rank statistic;
stdev=sqrt ( n * (n+1) * (2*n+1) / 6);

* Wilcoxon siged-rank probabilities for a sample size of 4;
if n = 4 then do;
      T4 = j((n*(n+1)/2)+1,1,1);
      T4[1:3,]=1;
      T4[4:8,]=2;
      T4[9:11,]=1;
      T = T4/(2**n);
end;

* Wilcoxon signed-rank probabilities for a sample size of 5;
if n = 5 then do;
      T5 = j((n*(n+1)/2)+1,1,1);
      T5[1:3,]=1;
      T5[4:5,]=2;
      T5[6:11,]=3;
      T5[12:13,]=2;
      T5[14:16,]=1;
      T = T5/(2**n);
end;

* Wilcoxon signed-rank probabilities for a sample size of 6;
if n = 6 then do;
      T6 = j((n*(n+1)/2)+1,1,1);
      T6[1:3,]=1;
      T6[4:5,]=2;
      T6[6,]=3;
      T6[7:9,]=4;
```

```
      T6[10:13,]=5;
      T6[14:16,]=4;
      T6[17,]=3;
      T6[18:19,]=2;
      T6[20:22,]=1;
      T = T6/(2**n);
end;


* Wilcoxon signed-rank probabilities for a sample size of 7;
if n = 7 then do;
      T7 = j((n*(n+1)/2)+1,1,1);
      T7[1:3,]=1;
      T7[4:5,]=2;
      T7[6,]=3;
      T7[7,]=4;
      T7[8:9,]=5;
      T7[10,]=6;
      T7[11:12,]=7;
      T7[13:17,]=8;
      T7[18:19,]=7;
      T7[20,]=6;
      T7[21:22,]=5;
      T7[23,]=4;
      T7[24,]=3;
      T7[25:26,]=2;
      T7[27:29,]=1;
      T = T7/(2**n);
end;

* Wilcoxon signed-rank probabilities for a sample size of 8;
if n = 8 then do;
      T8 = j((n*(n+1)/2)+1,1,1);
      T8[1:3,]=1;
      T8[4:5,]=2;
      T8[6,]=3;
      T8[7,]=4;
      T8[8,]=5;
      T8[9,]=6;
      T8[10,]=7;
      T8[11,]=8;
      T8[12,]=9;
      T8[13,]=10;
      T8[14,]=11;
      T8[15,]=12;
      T8[16:18,]=13;
      T8[19,]=14;
      T8[20:22,]=13;
      T8[23,]=12;
      T8[24,]=11;
      T8[25,]=10;
      T8[26,]=9;
      T8[27,]=8;
      T8[28,]=7;
      T8[29,]=6;
      T8[30,]=5;
      T8[31,]=4;
```

```
      T8[32,]=3;
      T8[33:34,]=2;
      T8[35:37,]=1;
      T = T8/(2**n);
end;

* Wilcoxon signed-rank probabilities for a sample size of 9;
if n = 9 then do;
      T9 = j((n*(n+1)/2)+1,1,1);
      T9[1:3,]=1;
      T9[4:5,]=2;
      T9[6,]=3;
      T9[7,]=4;
      T9[8,]=5;
      T9[9,]=6;
      T9[10,]=8;
      T9[11,]=9;
      T9[12,]=10;
      T9[13,]=12;
      T9[14,]=13;
      T9[15,]=15;
      T9[16,]=17;
      T9[17,]=18;
      T9[18,]=19;
      T9[19:20,]=21;
      T9[21,]=22;
      T9[22:25,]=23;
      T9[26,]=22;
      T9[27:28,]=21;
      T9[29,]=19;
      T9[30,]=18;
      T9[31,]=17;
      T9[32,]=15;
      T9[33,]=13;
      T9[34,]=12;
      T9[35,]=10;
      T9[36,]=9;
      T9[37,]=8;
      T9[38,]=6;
      T9[39,]=5;
      T9[40,]=4;
      T9[41,]=3;
      T9[42:43,]=2;
      T9[44:46,]=1;
      T = T9/(2**n);
end;

* Wilcoxon signed-rank probabilities for a sample size of 10;
if n = 10 then do;
      T10 = j((n*(n+1)/2)+1,1,1);
      T10[1:3,]=1;
      T10[4:5,]=2;
      T10[6,]=3;
      T10[7,]=4;
      T10[8,]=5;
      T10[9,]=6;
      T10[10,]=8;
```

```
      T10[11,]=10;
      T10[12,]=11;
      T10[13,]=13;
      T10[14,]=15;
      T10[15,]=17;
      T10[16,]=20;
      T10[17,]=22;
      T10[18,]=24;
      T10[19,]=27;
      T10[20,]=29;
      T10[21,]=31;
      T10[22,]=33;
      T10[23,]=35;
      T10[24,]=36;
      T10[25,]=38;
      T10[26:27,]=39;
      T10[28:29,]=40;
      T10[30:31,]=39;
      T10[32,]=38;
      T10[33,]=36;
      T10[34,]=35;
      T10[35,]=33;
      T10[36,]=31;
      T10[37,]=29;
      T10[38,]=27;
      T10[39,]=24;
      T10[40,]=22;
      T10[41,]=20;
      T10[42,]=17;
      T10[43,]=15;
      T10[44,]=13;
      T10[45,]=11;
      T10[46,]=10;
      T10[47,]=8;
      T10[48,]=6;
      T10[49,]=5;
      T10[50,]=4;
      T10[51,]=3;
      T10[52:53,]=2;
      T10[54:56,]=1;
      T = T10/(2**n);
end;

* Calculating the control limits;
UCL = L * stdev * sqrt(lambda/(2-lambda));
LCL = -UCL;

S=j(NN,1,0);

* The interval between the UCL and LCL are divided into subintervals of width
2*delta;
delta = ((UCL-LCL)/NN)/2;

S[1,1] = UCL - delta;

do i = 2 to NN;
      S[i,1]=S[i-1,1]-2*delta;
```

```
end;

Q_a=j(NN,NN,0);
Q_b=j(NN,NN,0);
Q=j(NN,NN,0);

do i = 1 to NN;
      do j = 1 to NN;
      Q_a[i,j]=floor(((((S[j,]-delta) - (1-lambda)*S[i,])/lambda) +
n*(n+1)/2)/2);
      end;
end;

do i = 1 to NN;
      do j = 1 to NN;
      Q_b[i,j]=floor(((((S[j,]+delta) - (1-lambda)*S[i,])/lambda) +
n*(n+1)/2)/2);
      end;
end;

do i = 1 to NN;
      do j = 1 to NN;
      lower = Q_a[i,j];
      upper = Q_b[i,j];
      if lower < 0 then if lower*upper > 0 then Q[i,j]=0;
      if lower > n*(n+1)/2 then lower_term = 1;
      else if lower < 0 then lower_term = 0;
      else lower_term = sum(T[1:(lower+1),]);
      if upper > n*(n+1)/2 then upper_term = 1;
      else if upper < 0 then upper_term = 0;
      else upper_term = sum(T[1:(upper+1),]);
      Q[i,j] = upper_term - lower_term;
      end;
end;

* Defining the vector eta used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
eta=j(1,NN,1);
do i = 1 to NN;
      if i = 1 then eta[1,i]=1;
      else eta[1,i]=0;
end;

* Defining the vector one used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
one=j(NN,1,1);

* Defining the identity matrix I used in the formula for the ARL given by
E(N)=eta*inv(I-Q)*one;
identity = I(NN);

* Calculating the 5th, 25th, 50th, 75th and 95th percentiles;
p5p=0.05;
p25p=0.25;
p50p=0.5;
p75p=0.75;
p95p=0.95;
```

```
pmf=j(z,1,1);
cdf=j(z,1,1);
cdf_5th_p=j(z,1,1);
cdf_25th_p=j(z,1,1);
cdf_50th_p=j(z,1,1);
cdf_75th_p=j(z,1,1);
cdf_95th_p=j(z,1,1);

do i = 1 to z;
      pmf[i,1] = eta * (Q**(i-1)) * (identity - Q) * one;
      cdf[i,1]=sum(pmf[1:i,1]);
end;

index=j(z,1,1);

do i = 2 to z;
index[i,]=index[i-1,]+1;
end;

* Calculating the 5th percentile;
do i = 1 to z;
      cdf_5th_p[i,]=cdf[i,];
      if cdf_5th_p[i,]>=p5p then cdf_5th_p[i,]=999;
end;

cdf_5th_p=cdf_5th_p[loc(cdf_5th_p<999)];
if cdf_5th_p[1,]=999 then percentile_p5p=1;
else percentile_p5p=nrow(cdf_5th_p)+1;

* Calculating the 25th percentile;
do i = 1 to z;
      cdf_25th_p[i,]=cdf[i,];
      if cdf_25th_p[i,]>=p25p then cdf_25th_p[i,]=999;
end;

cdf_25th_p=cdf_25th_p[loc(cdf_25th_p<999)];
if cdf_25th_p[1,]=999 then percentile_p25p=1;
else percentile_p25p=nrow(cdf_25th_p)+1;

* Calculating the 50th percentile;
do i = 1 to z;
      cdf_50th_p[i,]=cdf[i,];
      if cdf_50th_p[i,]>=p50p then cdf_50th_p[i,]=999;
end;

cdf_50th_p=cdf_50th_p[loc(cdf_50th_p<999)];
if cdf_50th_p[1,]=999 then percentile_p50p=1;
else percentile_p50p=nrow(cdf_50th_p)+1;

* Calculating the 75th percentile;
do i = 1 to z;
      cdf_75th_p[i,]=cdf[i,];
      if cdf_75th_p[i,]>=p75p then cdf_75th_p[i,]=999;
end;

cdf_75th_p=cdf_75th_p[loc(cdf_75th_p<999)];
```

```
if cdf_75th_p[1,]=999 then percentile_p75p=1;
else percentile_p75p=nrow(cdf_75th_p)+1;

* Calculating the 95th percentile;
do i = 1 to z;
      cdf_95th_p[i,]=cdf[i,];
      if cdf_95th_p[i,]>=p95p then cdf_95th_p[i,]=999;
end;

cdf_95th_p=cdf_95th_p[loc(cdf_95th_p<999)];
if cdf_95th_p[1,]=999 then percentile_p95p=1;
else percentile_p95p=nrow(cdf_95th_p)+1;

* Calculating the average run length (ARL);
ARL = eta*ginv(identity-Q)*one;

* Calculating the second moment;
N2 = eta * (identity + Q) * (ginv((identity-Q)**2)) * one;

* Calculating the standard deviation;
SDRL = sqrt (N2 - ((ARL)**2) );

* Printing the output;
print_cdf=index||cdf;
print_pmf=index||pmf;

test = inv(identity - Q);

print        test,
             NN [label = 'Number of intervals between LCL and UCL'],
             n [label = 'Sample size'],
             L [label = 'L: EWMA parameter'],
             lambda [label = 'lambda: EWMA parameter'],
             UCL [label='Upper control limit'],
             LCL [label='Lower control limit'],
             delta,
             Q [format=fract.],
             ARL [label = 'Average run length' format=.2]
             SDRL [label = 'Standard deviation of the run length' format=.2],
             percentile_p5p [label='Fifth percentile'],
             percentile_p25p [label='25th percentile'],
             percentile_p50p [label='50th percentile'],
             percentile_p75p [label='75th percentile'],
             percentile_p95p [label='95th percentile'];
```

SAS program 9:

This program calculates the values for the CUSUM Mann-Whitney chart shown in Table 6.9.

```
proc iml;

* The data used by Sullivan and Woodall (1996) and Zhou, Zou and Wang (2007);
x_obs={
-0.69, 0.56, -0.96, -0.11, -0.25, 0.45, -0.26, 0.68, 0.22, -2.10, 0.65, -1.49,
-2.49, -1.11, 0.23, 2.16, 1.95, 1.54, 0.67, 1.09, 1.37, 0.69, 2.26, 1.86,
0.62, -1.04, 2.30, 0.07, 1.49, 0.52};

* Reference value;
k = 2;

* Calculating the Mann-Whitney (MW) values;
MW=j(nrow(x_obs)-1,1,.);

do r = 1 to nrow(x_obs-1);
      keep=0;
            do i = 1 to r;
                  count = j(nrow(x_obs),1,.);
                        do j = r+1 to nrow(x_obs);
                              if x_obs[i,]>x_obs[j,] then count[j,]=1;
                              t_sum=count[+,];
                        end;
                  keep = keep // t_sum;
            end;
      MW[r,] = keep[+,];
end;

t = j(nrow(MW),1,.);

do l = 1 to nrow(MW);
      t[l,]=l;
end;

MW = t||MW;

* Calculating the expected value of MWt, i.e. E(MWt);
exp = j(nrow(MW),1,.);

do l = 1 to nrow(MW);
      exp[l,]=(MW[l,1]*((nrow(x_obs))-MW[l,1]))/2;
end;

* Calculating the standard deviation of MWt, i.e. stdev(MWt);
stdev = j(nrow(MW),1,.);

do l = 1 to nrow(MW);
      stdev[l,]=sqrt((MW[l,1]*((nrow(x_obs))-MW[l,1])*(nrow(x_obs)+1))/12);
end;

* Calculating the standardized MWt values, i.e. SMWt;
SMW = j(nrow(MW),1,.);
```

```
do l = 1 to nrow(MW);
      SMW[l,]=(MW[l,2]-exp[l,])/stdev[l,];
end;

* Starting values for CUSUM;
S_plus = j (nrow(MW),1,.);
S_plus[1,]=0-SMW[1,]-k;
if S_plus[1,]<0 then S_plus[1,]=0;

S_minus = j (nrow(MW),1,.);
S_minus[1,]=0+SMW[1,]-k;
if S_minus[1,]<0 then S_minus[1,]=0;

* Calculating the CUSUM statistics;
do l = 2 to nrow(S_plus);
      S_plus[l,]=S_plus[l-1,]+SMW[l,]-k;
      if S_plus[l,]<0 then S_plus[l,]=0;
end;

do l = 2 to nrow(S_minus);
      S_minus[l,]=S_minus[l-1,]+SMW[l,]+k;
      if S_minus[l,]>0 then S_minus[l,]=0;
end;

MW = MW[,2];

print        x_obs [label='Xi-values' format=.2],
             MW [label='Mann-Whitney statistics'],
             exp [label='Expected values of MW-statistics'],
             stdev [label='Standard deviation values of the MW-statistics'
format=.3],
             SMW [label='Standarddized values for the MW-statistics'
format=.3],
             S_plus [label='Upper CUSUM statistics' format=.3],
             S_minus [label='Lower CUSUM statistics' format=.3];
```

SAS program 10:

This program calculates the charting statistics for the Shewhart-type signed-rank-like chart.

```
proc iml;

* Number of samples;
nn=15;

* Sample size;
n=5;

wsrmatrix = j(nn,n,0);
sgnmatrix = j(nn,n,0);
rank_abs_diff = j(nn,n,0);
final_rank_abs_diff=j(nn,n,0);

* Median of the reference sample;
tv=74.001;

tvmatrix = j(nn,n,tv);

* A matrix containing the Montgomery (2001) Table 5.2 piston ring data;
obs = {
74.012 74.015 74.030 73.986 74.000,
73.995 74.010 73.990 74.015 74.001,
73.987 73.999 73.985 74.000 73.990,
74.008 74.010 74.003 73.991 74.006,
74.003 74.000 74.001 73.986 73.997,
73.994 74.003 74.015 74.020 74.004,
74.008 74.002 74.018 73.995 74.005,
74.001 74.004 73.990 73.996 73.998,
74.015 74.000 74.016 74.025 74.000,
74.030 74.005 74.000 74.016 74.012,
74.001 73.990 73.995 74.010 74.024,
74.015 74.020 74.024 74.005 74.019,
74.035 74.010 74.012 74.015 74.026,
74.017 74.013 74.036 74.025 74.026,
74.010 74.005 74.029 74.000 74.020};

* Calculating the SRLi statistics;
diff=obs-tvmatrix;

do k = 1 to nn;
      do l = 1 to n;
      if diff[k,l]>0 then sgnmatrix[k,l]=1;
      else if diff[k,l]<0 then sgnmatrix[k,l]=-1;
      else if diff[k,l]=0 then sgnmatrix[k,l]=0;
      end;
end;

abs_diff=abs(diff);

do i = 1 to nn;
      rank_abs_diff[i,]=rank(abs_diff[i,]);
```

```
end;

do i = 1 to nn;
      do j = 1 to n;
            do k = 1 to n;
                  if abs_diff[i,j]=abs_diff[i,k] then

      final_rank_abs_diff[i,j]=(rank_abs_diff[i,j]+rank_abs_diff[i,k])/2;
            end;
      end;
end;

do i = 1 to nn;
      do j = 1 to n;
            do k = 1 to n;
                  if abs_diff[i,j]=abs_diff[i,k] then
                  final_rank_abs_diff[i,k]=final_rank_abs_diff[i,j];
            end;
      end;
end;


do k = 1 to nn;
      do l = 1 to n;
            wsrmatrix[k,l]=sgnmatrix[k,l]*final_rank_abs_diff[k,l];
      end;
end;

SRi = wsrmatrix[,+];

print SRi [label='Signed-rank-like statistics'];
```

Mathematica program 1:

Chakraborti and Van de Wiel (2003) wrote a Mathematica program which deals with the computation of the upper and lower control limits of the Mann-Whitney control chart for either a specified in-control *ARL* or a specified ρ^{th} percentile of conditional *ARL* (that is: that level of which one wants to be $100(1-\rho)\%$ sure that it is exceeded for his/her specific reference sample). In addition, it contains procedures for approximation of the ARL_0 and percentiles when control limits are given as well as procedures for computations under out-of-control situations. This Mathematica program can be reached using the website www.win.tue.nl/~markvdw. This program has user friendly parameters which I changed to suit my examples in Section 6.1.

References

Albers, W. and Kallenberg, W.C.M. (2004). "Empirical nonparametric control charts: estimation effects and corrections." *Journal of Applied Statistics,* **31**, 345-360.

Amin, R.W. (1987). "Variable sampling interval control charts." Unpublished Ph.D dissertation, Virginia Polytechnic Institute and State University, Department of Statistics.

Amin, R.W. and Hemasinha, R. (1993). "The switching behavior of $\overline{X}$ charts with variable sampling intervals." *Communications in Statistics: Theory and Methods,* **22** (7), 2081-2102.

Amin, R.W., Reynolds Jr. M.R. and Bakir, S.T. (1995). "Nonparametric quality control charts based on the sign statistic." *Communications in Statistics: Theory and Methods,* **24** (6), 1597-1623.

Amin, R.W. and Widmaier, O. (1999). "Sign control charts with variable sampling intervals." *Communications in Statistics: Theory and Methods,* **28**, 1961-1985.

Bain, L.J. and Engelhardt, M. (1992). *Introduction to Probability and Mathematical Statistics, Second Edition,* Duxbury Press.

Bakir, S.T. (2003). "A quality control chart based on signed-ranks." *Joint Statistical Meeting – Section on Quality and Productivity,* 423-429.

Bakir, S.T. (2004). "A distribution-free Shewhart quality control chart based on signed-ranks." *Quality Engineering,* **16**, 613-623.

Bakir, S.T. (2006). "Distribution-free quality control charts based on signed-rank-like statistics." *Communications in Statistics: Theory and Methods*, **35**, 743-757.

Bakir, S.T. and Reynolds, Jr. M.R. (1979). "A nonparametric procedure for proses control based on within-group ranking." *Technometrics*, **21** (2), 175-183.

Barnard, G.A. (1959). "Control charts and stochastic processes." *Journal of the Royal Statistical Society, B,* **21**, 239-271.

Bartlett, M.S. (1953). "Recurrence and first passage times." *Proceedings of the Cambridge Philosophical Society,* **49**, 263-275.

Borror, C.M., Montgomery, D.C. and Runger, G.C. (1999). "Robustness of EWMA control charts to non-normality." *Journal of Quality Technology*, **31** (3), 309-316.

Brook, D. and Evans, D.A. (1972). "An approach to the probability distribution of CUSUM run length." *Biometrika*, **59** (3), 539-549.

Chakraborti, S. (2000). "Run length, average run length and false alarm rate of Shewhart $\overline{X}$ chart: Exact derivations by conditioning." *Communications in Statistics: Simulation and Computation,* **29** (1), 61-81.

Chakraborti, S. (2006). "Parameter estimation and design considerations in prospective applications of the $\overline{X}$ chart." *Journal of Applied Statistics,* **33**, 439-459.

Chakraborti, S. (2007). "Run length distribution and percentiles: The Shewhart $\overline{X}$ chart with unknown parameters." *Quality Engineering,* **19**, 119-127.

Chakraborti, S. and Eryilmaz, S. (2007). "A nonparametric Shewhart-type signed-rank control chart based on runs." *Communications in Statistics: Simulation and Computation,* **36**, 335-356.

Chakraborti, S., Eryilmaz, S. and Human, S.W. (2006). "A Phase II Shewhart-type nonparametric control chart based on runs." (Submitted).

Chakraborti, S. and Van der Laan, P. (1996). "Precedence tests and confidence bounds for complete data: an overview and some results." *Journal of the Royal Statistical Society Series D: The Statistician,* **45** (3), 351-369.

Chakraborti, S. and Van der Laan, P. (1997). "An overview of precedence tests for censored data." *Biometrical Journal,* **39** (1), 99-116.

Chakraborti, S., Van der laan, P. and Bakir, S.T. (2001). "Nonparametric control charts: An overview and some results." *Journal of Quality Technology* **33** (3), 304-315.

Chakraborti, S., Van der Laan, P. and Van de Wiel, M.A. (2004). "A class of distribution-free control charts." *Journal of the Royal Statistical Society. Series C: Applied Statistics*, **53**, 443-462.

Chakraborti, S. and Van de Wiel, M.A. (2003). "A nonparametric control chart based on the Mann-Whitney statistic." SPOR Report 2003-24, Technical University of Eindhoven, Eindhoven.

Champ, C.W. and Chou, S.P. (2003). "Comparison of standard and individual limits Phase I Shewhart $\overline{X}$, R and S charts." *Quality and Reliability Engineering International*, **19**, 161-170.

Champ, C.W. and Jones, L.A. (2004). "Designing Phase I $\overline{X}$ chart with small sample sizes." *Quality and Reliability Engineering International,* **20**, 497-510.

Champ, C.W. and Woodall, W.H. (1987). "Exact results for Shewhart control charts with supplementary runs rules." *Technometrics* **29** (4), 393-399.

Champ, C.W. and Woodall, W.H. (1997). "Signal probabilities of runs rules supplementing a Shewhart control chart." *Communications in Statistics - Simulation and Computation,* **26** (4), 1347-1360.

Chou, S.P. and Champ, C.W. (1995). "A comparison of two Phase I control charts." *Proceedings of the Quality and Productivity Section of the American Statistical Association*, 31-35.

Crowder, S.V. (1987). "Simple methods for studying run length distributions of exponentially weighted moving average control charts." *Technometrics,* **29**, 401-407.

Daniels, H.E. (1954). "Saddlepoint approximations in statistics." *Annals of Mathematical Statistics,* **25**, 631-650.

Daniels, H.E. (1987). "Tail probability approximations." *International Statistical Review,* **55** (1), 37-48.

Ewan, W. D. (1963). "When and how to use CUSUM charts." *Technometrics*, **5** (1), 1-22.

Ewan, W.D. and Kemp, K.W. (1960). "Sampling inspection of continuous processes with no autocorrelation between successive results." *Biometrika*, **47**, 363-380.

Fix, E. and Hodges, J.L. (1955). "Significance probabilities of the Wilcoxon test." *Annals of Statistics,* **26**, 301-312.

Fu, J.C. and Lou, W.Y.W. (2003). *Distribution Theory of Runs and Patterns and Its Applications: A Finite Markov Chain Imbedding Technique,* Singapore: World Scientific Publishing.

Fu, J.C., Spiring, F.A. and Xie, H. (2002). "On the average run lengths of quality control schemes using a Markov chain approach." *Statistics and Probability Letters,* **56**, 369-380.

Gan, F.F. (1993). "An optimal design of CUSUM control charts for binomial counts." *Journal of Applied Statistics,* **20**, 445-460.

Gibbons, J.D. and Chakraborti, S. (2003). *Nonparametric Statistical Inference,* 4^{th} Edition, Revised and Expanded, Marcel Dekker, New York.

Goldsmith, R.L. and Whitfield, H. (1961). "Average run lengths in cumulative chart quality control schemes." *Technometrics,* **3** (1), 11-20.

Hawkins, D.M. (1977). "Testing a sequence of observations for a shift in location." *Journal of the American Statistical Association*, **72**, 180-186.

Hawkins, D.M. and Olwell, D.H. (1998). *Cumulative sum charts and charting for quality improvement. New York: Springer*

Hawkins, D.M., Qiu, P. and Kang, C.W. (2003). "The change-point model for statistical process control." *Journal of Quality Technology*, **35**, 355-366.

Hawkins, D.M. and Zamba, K.D. (2005). "Statistical process control for shifts in mean or variance using a change-point formulation." *Technometrics*, **47**, 164-173.

Hollander, M. and Wolfe, D.A. (1973). *Nonparametric statistics.* New York: John Wiley.

Human, S.W., Chakraborti, S. and Smit, C.F. (2007). "Design of S^2, S and R control charts for Phase I application." Submitted.

Human, S.W., Chakraborti, S. and Smit, C.F. (2008). "Nonparametric Shewhart-type sign control charts based on runs." Submitted.

Janacek, G.J. and Meikle, S.E. (1997). "Control charts based on medians." *The Statistician*, **46** (1), 19-31.

Jensen, J.L. (1995). *Saddlepoint Approximations,* Oxford Statistical Science Series, Clarendon Press, Oxford.

Johnson, N.L. (1961). “Simple theoretical approach to cumulative sum control charts.” *Journal of the American Statistical Association,* **56**, 835-840.

Jones, L.A. (2002). “The statistical design of EWMA control charts with estimated parameters.” *Journal of Quality Technology,* **34** (3), 277-288.

Jones, L.A. and Champ, C.W. (2002). “Phase I control charts for times between events.” *Quality and Reliability Engineering International,* **18**, 479-488.

Kemp, K.W. (1971). “Formal expressions which can be applied to CUSUM charts.” *Journal of the Royal Statistical Society, Series B: Methodological,* **33**, 331-360.

Khoo, M.B.C. (2004). “Design of runs rules schemes.” *Quality Engineering* **16** (1), 27-43.

King, E.P. (1954). “Probability limits for the average chart when process standards are unspecified.” *Industrial Quality Control*, **10**, 62-64.

Klein, M. (2000). “Two alternatives to the Shewhart $\overline{X}$ control chart.” *Journal of Quality Technology*, **32** (4), 427-431.

Koning, A.J. (2006). “Model-based control charts in Phase I statistical process control.” *Statistica Neerlandica,* **60** (3), 327-338.

Koning, A.J. and Does, R.J.M.M. (2000). “CUSUM charts for preliminary analysis of individual observations.” *Journal of Quality Technology*, **32**, 122-132.

Lehmann, E.L. (1975). *Nonparametrics: Statistical Methods Based on Ranks*. San Francisco: McGraw-Hill.

Lucas, J.M. (1985). "Counted data CUSUMS." *Technometrics,* **27** (2), 129-144.

Lucas, J.M. and Crosier, R.B. (1982). "Fast initial response for CUSUM quality-control schemes: Give your CUSUM a head start." *Technometrics*, **24** (3), 199-205.

Lucas, J.M. and Saccucci, M.S. (1987). "Exponentially weighted moving average control schemes: Properties and enhancements." *Faculty Working Series Paper 87-5*, Drexel University, Department of Quantitative Methods.

Lucas, J.M. and Saccucci, M.S. (1990). "Exponentially weighted moving average control schemes: Properties and enhancements." *Technometrics,* **32** (1), 1-12.

Lugannani, R. and Rice, S. (1980). "Saddlepoint approximation for the distribution of the sum of independent random variables." *Advances in Applied Probability,* **12**, 475-490.

Mathisen, H.C. (1943). "A method of testing the hypothesis that two samples are from the same population." *Annals of Mathematical Statistics,* **14**, 188-194.

Montgomery, D.C. (2001). *Introduction to Statistical Quality Control,* 4th Edition, John Wiley & Sons, New York.

Montgomery, D.C. (2005). *Introduction to Statistical Quality Control 5th ed.* New York: John Wiley.

Montgomery, D.C., Gardiner, J.S. and Pizzano, B.A. (1987). "Statistical process control methods for detecting small process shifts." *Frontiers in Statistical Quality Control,* **3,** 163-178.

Moses, L.E. (1963). "Rank tests of dispersion." *The Annals of Mathematical Statistics*, **34** (3), 973-983.

Nelson, L.S. (1984). "The Shewhart control chart - tests for special causes." *Journal of Quality Technology*, **16**, 237-239.

Page, E.S. (1954). "Continuous inspection schemes." *Biometrika*, **41**, 100-115 .

Page, E.S. (1955). "Control charts with warning lines." *Biometrika,* **42**, 243-257.

Page, E. S. (1961). "Cumulative sum charts." *Technometrics*, **3** (1), 1-9.

Page, E. S. (1962). "A modified control chart with warning lines." *Biometrika*, **49**, 171-176.

Park, C. and Reynolds, Jr. M.R. (1987). "Nonparametric procedures for monitoring a location parameter based on linear placement statistics." *Sequential Analysis*, **6**, 303-323.

Pettitt, A.N. (1979). "A nonparametric approach to the change-point problem." *Applied Statistics*, **28**, 126-135.

Radson, D. and Boyd, A.H. (2005). "Graphical representation of run length distributions." *Quality Engineering*, **17**, 301-308.

Randles, R.H. and Wolfe, D.A. (1979). *Introduction to the Theory of Nonparametric Statistics,* John Wiley and Sons, New York.

Reid, N. (1988). "Saddlepoint methods and statistical inference (with discussion)." *Statistical Science*, **3**, 213-238.

Rendtel, U. (1990). "CUSUM schemes with variable sampling intervals and sampling sizes." *Statistical Papers*, **31**, 103-118.

Reynolds, M.R., Amin, R.W. and Arnold, J.C. (1990). "CUSUM charts with variable sampling intervals." *Technometrics*, **32** (4), 371-384.

Roberts, S.W. (1958). "Properties of control chart zone tests." *The Bell System Technical Journal* **37**, 83-114.

Roberts, S.W. (1959). "Control chart tests based on geometric moving averages." *Technometrics* **1**, 239-250.

Roberts, S.W. (1966). "A comparison of some control chart procedures." *Technometrics,* **8** (3)**,** 411-430.

Robinson, P.B. and Ho, T.Y. (1978). "Average run lengths of geometric moving average charts by numerical methods." *Technometrics*, **20**, 85-93.

Saccucci, M.S., Amin, R.W. and Lucas, J.M. (1992). "Exponentially weighted moving average control schemes with variable sampling intervals." *Communications in Statistics: Simulation and Computation,* **21** (3), 627-657.

Sheu, S. and Yang, L. (2006). "The generally weighted moving average median control chart." *Quality Technology and Quantitative Management,* **3** (4), 455-471.

Shewhart, W.A. (1926). "Quality control charts." *Bell Systems Technical Journal*, 593-603.

Shewhart, W.A. (1931). *Economic Control of Quality of Manufactured Product,* New York: Van Nostrand.

Shewhart, W.A. (1939). "Economic control of quality of manufactured product." *Princeton: Van Nostrand Reinhold.*

Shewhart, W.A. (1941). "Contributions of statistics to the science of engineering." *Fluid Mechanics and Statistical Methods in Engineering,* University of Pennsylvania Press, Philadelphia, 97-124.

Shmueli, G. and Cohen, A. (2003). "Run-length distribution for control charts with runs and scans rules." *Communications in Statistics: Theory and Methods,* **32** (2), 475-495.

Sullivan, J.H. and Woodall, W.H. (1996). "A control chart for preliminary analysis of individual observations." *Journal of Quality Technology*, **28** (3), 265-278.

Van de Wiel, M.A. (2007). Personal communication.

Van der Laan, P. and Chakraborti, S. (1999). "Best precedence tests against Lehmann alternatives." *Bulletin International Statistical Institute*, **58** (3), 395-396.

Van Dobben de Bruyn, C.S. (1968). *Cumulative Sum Tests, New York: Hafner.*

Weindling, J.I., Littauer, S.B. and Oliveira, J.T. (1979). "Mean action time of the $\overline{X}$ control chart with warning limits." *Journal of Quality Technology*, **2** (2), 79-85.

Western Electronic Company. (1956). *Statistical Quality Control Handbook,* Western Electric Corporation, Indianaplois.

White, C. H., Keats, J. B. and Stanley, J. (1997). "Poisson CUSUM versus c-chart for defect data." *Quality Engineering*, **9**, 673-679.

Woodall, W.H. (1983). "The distribution of the run length of one-sided CUSUM procedures for continuous random variables." *Technometrics*, **25**, 295-301

Woodall, W. and Adams, B. (1993). "The statistical design of CUSUM charts." *Quality Engineering*, **5**, 559-570.

Woodall, W.H. and Montgomery, D.C. (1999). "Research issues and ideas in statistical process control." *Journal of Quality Technology*, **31** (4), 376-386.

Woodall, W.H. (2000). "Controversies and contradictions in statistical process control." *Journal of Quality Technology*, **32** (4), 341-350.

Wu, C., Zhao, Y. and Wang, Z. (2002). "The median absolute deviations and their applications to Shewhart $\overline{X}$ control charts." *Communications in Statistics: Simulation and Computation,* **31** (3), 425-442.

Zhou, C., Zou, C. and Wang, Z. "Nonparametric control chart for preliminary analysis of individual observations." Preprint, http://202.113.29.3/keyan/pre/preprint06/06-17.pdf, December 2007.