Yugoslav Journal of Operations Research
1 (1991), Number 2, 235–243

ECONOMICALLY OPTIMAL c-CONTROL CHARTS

Kun-Jen CHUNG

*Department of Industrial Management,
National Taiwan Institute of Technology,
Taipei, Taiwan, Republic of China*

Abstract. The economic design of statistical process control procedures has attracted the attention of the academic community for over 30 years. However, those models have not gained analogous popularity in industry. One reason for the limited utilization of the economic design of control charts in industry is apparently the mathematical complexity of the associated models and their optimization procedures. The main purpose of this paper is to present a simple search procedure to get the optimal design of c-charts.

Key words and phrases: Control charts, economic design, c-charts, *np*-charts

1. INTRODUCTION

Since the 1950s, considerable attention has been devoted to the economic design of $\overline{X}$-control charts. However, there have been a number of papers dealing with the economic design of p charts and np charts. Ladany (1973), Chiu (1975), Gibra (1978), and Duncan (1978) have developed economic models of the fraction defective control charts when the process is disturbed by a single assignable cause. Chiu (1976), Collani (1989), Gibra (1981) and Montgomery, Heikes and Mance (1975) have presented models on the economic design of fraction defective control charts for process subject to a multiplicity of assignable causes.

Collani (1989) is based on Behl's results (1985) and utilizes the approximation procedure developed in Collani (1987a, 1987b). Collani's optimum solution technique is very complicated and involved. So, Collani (1989) presents a simple graphical algorithm to determine the optimal economic design for a given set of process and economic parameters. The graphical algorithm requires the manual computations and the uses of many tables. The optimal solutions depend on the users. We hardly get a unique optimum solution through the use of the graphical algorithm. So, the graphical algorithm frequently limits the application of Collani's model (1989). For example, if the shift parameter d or the control limit k in Collani (1989) is more than 10, then Figures 1, 2 and 3 in Collani (1989) will be useless. This paper presents a simple computer program for the optimal economic design of c-charts on the cost model of Collani (1989). The results and the execution times of all numerical examples show that our computer program is quite accurate and efficient indeed.

2. THE PROCESS AND CONTROL MODEL

A c-chart consists of three quantities, the sampling interval h, the sample size n and the control limit k with the following decision rule: Every h hours of production take a sample of size n (n consecutive items produced), if the cumulative number of defects of the n items sampled is less than or equal to the control limit k, the process continues to operate, otherwise it is stopped and a process inspection is undertaken. If necessary a renewal is performed, after which the process starts anew in the in-control state.

The quality of the i-th item produced is measured by the random number X_i of its defects. The X_i are assumed to be a sequence of independent random variables each distributed according to a Poisson distribution with mean u and variance u. The production process is said to be in-control, if $u = u_0$ and out-of-control, if $u = u_1$ with $0 < u_0 < u_1$, i.e. a single assignable cause model is assumed, where the assignable cause is represented by a fixed size shift in the mean number of defects on an item produced.

The in-control parameter u_0 and the out-of-control parameter u_1 and hence their quotient $d = u_1/u_0 > 1$, which is called the shift parameter, are assumed to be known. Furthermore we assume that the length of an in-control period τ is distributed according to an exponential distribution with parameter $\lambda > 0$.

Let α be the probability of Type I error and β the probability of Type II error, then we have:

$$\alpha = 1 - \sum_{\nu=0}^{k} \frac{(n \cdot \mu_0)^\nu}{\nu!} e^{-n \cdot \mu_0} \qquad \text{and} \tag{1}$$

$$\beta = \sum_{\nu=0}^{k} \frac{(n \cdot \mu_1)^\nu}{\nu!} e^{-n \cdot \mu_1} = \sum_{\nu=0}^{k} \frac{(d \cdot n \cdot \mu_0)^\nu}{\nu!} e^{-d \cdot n \cdot \mu_0}. \tag{2}$$

In order to formulate the profit function P^*, the following notations will be used. Let

a^*n — be the cost associated with taking and analysing a sample of size n;

e^* — be the total expected cost of a process inspection when the process is operating in-control;

b^* — be the expected benefit per renewal, i.e. the expected additional profit derived from operating for some time τ in control after a renewal, reduced by the total cost of a process inspection (to detect the assignable cause) and the following renewal (to remove the assignable cause);

g_2 — be the expected profit per item produced while operating out-of-control; and

v — be the expected number of items produced per hour of operation.

Based on the model described in this section and the parameters just defined, Collani (1989) has shown that the expected profit function P^* per item in the long run is

$$P^*(h,n,k) = \frac{e^*}{v}\frac{1}{h}\left\{\frac{((b^*/e^*)(e^{\lambda h}-1)-\alpha)(1-\beta)}{e^{\lambda h}-\beta} - (a^*/e^*)n\right\} + g_2. \quad (3)$$

The problem is to determine the sampling interval h^*, the sampling plan (n^*, k^*) which maximizes the objective function $P^*(h,n,k)$. By setting $x = \lambda h$, $b = b^*/e^*$ and $a = a^*/e^*$, the above standardization of P^* leads to so-called Standardized Profit Function

$$P(x,n,k) = \frac{1}{x}\left\{\frac{b(e^x-1)-\alpha}{e^x-\beta}(1-\beta) - an\right\}. \quad (4)$$

Obviously this problem is equivalent to determine x^*, n^* and k^*, maximizing $P(x,n,k)$ where x^* is called standardized sampling interval. The parameters a, b can be described as the "relative sampling cost per item", and the "relative benefit per renewal", respectively. The parameter a is clearly positive valued and in practice should always be very much less than 1. In practice, b should also always be positive and will generally be much greater than 1.

3. THE EXPLICIT EQUATION FOR x

It can be shown that the following two relations hold

(a): $1 > \frac{1}{x} - \frac{1}{e^x-1} > 0$ *for all* $x > 0$.

(b): $\lim_{x\to 0+}\left(\frac{1}{x} - \frac{1}{e^x-1}\right) = \frac{1}{2}$.

Arnold and Collani (1989, pp. 153) indicate that all economic approaches considered heretofore lead to optimum values of x which are in general not greater than 0.1. By relations (a) and (b), we take 0.5 as a correction number. Replacing $1/(e^x-1)$ in equation (4) by $1/x - 1/2$, we get equation (5).

$$\overline{P}(x,n,k) = (1-\beta)\frac{(b+\alpha/2)x-\alpha}{((1+\beta)/2)x^2+(1-\beta)x} - \frac{an}{x}. \quad (5)$$

For a given (n,k), setting the partial derivative of $\overline{P}$ with respect to x equals to zero yields $r_1x^2 + r_2x + r_3 = 0$, where

$$r_1 = (1+\beta)\left(\frac{1}{2}\left(b+\frac{\alpha}{2}\right)(1-\beta) - \frac{1}{4}an(1+\beta)\right),$$
$$r_2 = -(1-\beta^2)(an+\alpha),$$
$$r_3 = -(1-\beta)^2(an+\alpha).$$

So

$$x = \frac{-r_2 + \sqrt{r_2^2 - 4r_1r_3}}{2r_1} = x(n, k). \tag{6}$$

4. THE SEARCH PROCEDURE

In the process of obtaining the optimal solution n^*, k^* and x^*, both n and k are integers. The value of n has no upper bound. The stopping rule for n is described in the program listing. Let $[nu_0]$ be the greatest integer less than or equal to nu_0. Then the search procedure is as follows.

(i) Given $1 \leq n$, and $[nu_0] \leq k \leq [nu_0] + 6\sqrt{nu_0} = U(n)$.

(ii) Calculate α and β from equations (1) and (2).

(iii) Calculate x from equation (6).

(iv) Calculate $P(x, n, k)$ from equation (4).

(v) Find $PP(n) = \underset{[nu_0] \leq k \leq U(n)}{\text{maximum}} (PP(n))$.

(vi) Calculate $PP^* = \underset{1 \leq n}{\text{maximum}} (PP(n)) = P(x^*, n^*, k^*)$ (say).

In general, the values n^*, k^* and x^* will be the optimal design values of the c-chart. When the value of n is large, $6\sqrt{nu_0}$ will be quite large. Hence, the search procedure requires the stopping rule for k to accelerate the process of getting the optimal design. Let nbound and kbound be two positive integers. The stopping rules on n and k are described as follows.

The stopping rule for k: Suppose that $P^*_{n_0,k_0} = P(n_0, k_0, x(n_0, k_0))$ is the current optimum value and $P^*_{n_0,k_0} \geq P^*_{n_0,k_0+j}$ for all $j = 1, 2, \ldots$, kbound. The search procedure will execute the next n_0 (that is $n_0 + 1$) and terminate on k_0.

The stopping rule for n: Suppose that $P^*_{n_0} = P(n_0, k_0, x(n_0, k_0))$ is the current optimum value and $P^*_{n_0} \geq P^*_{n_0+j}$ for all $j = 1, 2, \ldots$, nbound. The search procedure will be $n^* = n_0$, $k^* = k_0$ and $x^* = x(n_0, k_0)$.

Montgomery (1982, pp. 41) remarks: "The second phase of the optimization finds the optimal k and h for each value of n in the interval $\max(1, n^* - 10) \leq n \leq n^* + 10$."

Based on the above statement, we take nbound = kbound = 10 in the program listing. Usually our search procedure will get the optimal design of c-charts. However, if u_0 is small, $6\sqrt{nu_0}$ may be quite small as well. We may let nbound be larger. We recommend nbound = 30 and kbound = 10 to assure to get the optimal design of c-charts when u_0 is small. In general, the larger nbound will not influence the efficiency of the search procedure when u_0 is small. The details about the search procedure are described in the program listing.

5. NUMERICAL EXAMPLES

EXAMPLE 1: $u_0 = 0.10$, $d = 4.00$, $a = 0.0025$ and $b = 100$.

Following our search procedure, we get the optimal solution is $n^* = 19$, $k^* = 5$, $x^* = 0.028$ and $P(19, 5, 0.028) = 95.651$. Collani (1989) reports as an optimal plan for Example 1 the values $(19, 5, 0.033)$ for a maximum expected profit per item. However, the actual optimal plan for Example 1 is $(19, 5, 0.020)$ and $P(19, 5, 0.020) = 95.403$ following the search procedure in Collani (1989). Collani's calculation is in error because the actual value of β is 0.2307 but not 0.1057. Our optimal solution is better than Collani's solution.

EXAMPLE 2: $p_0 = u_0 = 0.02$, $d = 4.00$ (i.e. $p_1 = 0.08$), $a = 0.002$ and $b = 100$.

Following our search procedure, we get the optimal solution is $n^* = 64$, $k^* = 3$, $x^* = 0.047$ and $P(64, 3, 0.047) = 92.679$. However, Collani's optimal solution is $n^* = 63$, $k^* = 3$, $x^* = 0.047$ and $P(63, 3, 0.047) = 92.676$. So, our optimal solution is better than Collani's optimal solution.

EXAMPLE 3: $u_0 = 1.00$, $d = 2.00$, $a = 0.03$ and $b = 100$.

Following our search procedure, we get the optimal solution is $n^* = 6$, $k^* = 0.059$ and $P(6, 9, 0.059) = 90.991$. However, Collani's optimal solution is $n^* = 6$, $k^* = 8$, $x^* = 0.022$ and $P(6, 8, 0.022) = 83.567$. Collani's optimal profit is 8.16% lower than the true optimum.

EXAMPLE 4: $u_0 = 4.00$, $d = 5.00$, $a = 0.16$ and $b = 100$.

Following our search procedure, we get the optimal solution is $n^* = 1$, $k^* = 10$, $x^* = 0.058$ and $P(1, 10, 0.058) = 94.291$. On the other hand, we follow Collani's search procedure (1989). We get $a_0 = 0.04$, $b = 1.0000$, $C = 1.0021$, $\alpha = 0.3712$ and $\beta = 0.0000$. Since $C > 0.9999$, Figure 3 in Collani (1989) becomes useless. Hence, Figures 1, 2 and 3 in Collani (1989) do not provide an approximately optimal solution.

Optimum designs for 2200 numerical examples were obtained by using our search procedure. The parameters values of d and a range $2 \leq d \leq 11$ and $0.0001 \leq a \leq 0.2000$, respectively. The value of b is fixed as 100. The value of u_0 is 0.02, 0.10, 1.00, 4.00 or 16.00, respectively. Table 1 shows the results for $u_0 = 0.10$, $b = 100$ and $d = 4.00$. All calculations were executed on a personal computer ENSONTECH (PC-386). The execution times of all numerical examples of Table 1 are between 0.38 and 3.46 seconds. The execution times of all other numerical examples for $b = 100$, $u_0 = 0.02$, 0.10, 1.00, 4.00, 16.00 and $2 \leq d \leq 11$ are summarized as follows.

(A) $u_0 = 0.02 = p_0$, $b = 100$ and $2 \leq d \leq 11$.

The execution times are between 0.16 and 24.50 seconds. The execution times of the great many numerical examples are within 2.00 seconds.

(B) $u_0 = 0.10$, $b = 100$ and $2 \leq d \leq 11$.

The execution times are between 0.38 and 24.88 seconds. The execution times of the great many numerical examples are within 2.00 seconds as well.

Table 1: The Optimal Solutions for $u_0 = 0.10$, $d = 4.00$, and $b = 100$

a	n	k	x	α	β	Profit	Execution times (CPU seconds)*
0.0001	42	12	0.008	0.0004	0.1454	98.889	3.46
0.0005	29	8	0.016	0.0031	0.1830	97.759	2.31
0.0010	26	7	0.021	0.0053	0.1863	97.003	2.14
0.0015	22	6	0.023	0.0075	0.2256	96.458	1.76
0.0020	23	6	0.028	0.0094	0.1892	96.016	1.87
0.0025	19	5	0.028	0.0132	0.2307	95.651	1.54
0.0030	19	5	0.030	0.0132	0.2307	95.326	1.59
0.0035	15	4	0.029	0.0186	0.2581	95.028	1.32
0.0040	15	4	0.030	0.0186	0.2581	94.774	1.26
0.0045	16	4	0.035	0.0237	0.2351	94.539	1.38
0.0050	16	4	0.037	0.0237	0.2351	94.317	1.37
0.0055	16	4	0.038	0.0237	0.2351	94.103	1.37
0.0060	12	3	0.035	0.0338	0.2942	93.905	1.10
0.0065	12	3	0.036	0.0338	0.2942	93.736	1.05
0.0070	12	3	0.037	0.0338	0.2942	93.571	1.10
0.0075	12	3	0.038	0.0338	0.2942	93.411	1.04
0.0080	12	3	0.039	0.0338	0.2942	93.255	1.04
0.0085	12	3	0.040	0.0338	0.2942	93.102	1.04
0.0090	13	3	0.046	0.0431	0.2381	92.955	1.15
0.0095	13	3	0.047	0.0431	0.2381	92.815	1.16
0.0100	13	3	0.048	0.0431	0.2381	92.677	1.15
0.0150	9	2	0.048	0.0629	0.3027	91.655	0.93
0.0200	5	1	0.042	0.0902	0.4060	90.867	0.66
0.0250	5	1	0.045	0.0902	0.4060	90.290	0.66
0.0300	5	1	0.048	0.0902	0.4060	89.747	0.66
0.0350	6	1	0.063	0.1219	0.3084	89.244	0.72
0.0400	2	0	0.047	0.1813	0.4493	88.910	0.49
0.0450	1	0	0.025	0.0952	0.6703	88.704	0.44
0.0500	1	0	0.025	0.0952	0.6703	88.506	0.44
0.0550	1	0	0.026	0.0952	0.6703	88.311	0.44
0.0600	1	0	0.026	0.0952	0.6703	88.120	0.44
0.0650	1	0	0.027	0.0952	0.6703	87.931	0.44
0.0700	1	0	0.027	0.0952	0.6703	87.746	0.44
0.0750	1	0	0.028	0.0952	0.6703	87.564	0.44
0.0800	1	0	0.028	0.0952	0.6703	87.385	0.43
0.0850	1	0	0.029	0.0952	0.6703	87.208	0.44
0.0900	1	0	0.029	0.0952	0.6703	87.034	0.44
0.0950	1	0	0.029	0.0952	0.6703	86.863	0.44
0.1000	1	0	0.030	0.0952	0.6703	86.694	0.44
0.1200	1	0	0.031	0.0952	0.6703	86.040	0.38
0.1400	1	0	0.033	0.0952	0.6703	85.418	0.38
0.1600	1	0	0.034	0.0952	0.6703	84.824	0.38
0.1800	1	0	0.036	0.0952	0.6703	84.255	0.38
0.2000	1	0	0.037	0.0952	0.6703	83.708	0.44

* All calculations were executed on a personal computer ENSONTECH (PC-386).

(C) $u_0 = 1.00$, $b = 100$ and $2 \leq d \leq 11$.

The execution times are between 0.88 and 8.63 seconds. As in (A) and (B), the execution times of the great many numerical examples are still within 2.00 seconds.

(D) $u_0 = 4.00$, $b = 100$ and $2 \leq d \leq 11$.

The execution times are between 2.41 and 8.57 seconds. The execution times of the great many numerical examples are within 4.00 seconds.

(E) $u_0 = 16.00$, $b = 100$ and $2 \leq d \leq 11$.

The execution times are between 15.55 and 19.61 seconds.

To sum up, the results and the execution times of all numerical examples show that our search procedure is very accurate and efficient. Equation (21) in Collani (1989) indicates that the optimal sampling plan is more or less independent of the quantity b or the sampling interval x, but only on a, d and u_0. This means that the range of model parameters (in all the examples $b = 100$) is large enough to ensure that the search procedure performs well under a wide range of possible model parameters. Therefore, we conclude that our search procedure is not only quite accurate and efficient, but also simpler to solve than Collani's search procedure (1989).

REFERENCES

[1] Arnold, B. F. and Collani, E. v. (1989), *On the robustness of $\overline{X}$-charts*, Statistics **20**, 149–159.

[2] Behl, M. (1985), *Kostenoptimale Prüfpläne für die laufende Kontrolle eines qualitativen Merkmals*, Metrika **32**, 219–251.

[3] Chiu, W. K. (1975), *Economic design of attribute control charts*, Technometrics **17**, 81–87.

[4] Chiu, W. K. (1976), *Economic design of np-charts for processes subject to a multiplicity of assignable causes*, Management Science **23**, 404–411.

[5] Collani, E. v. (1986), *A simple procedure to determine the economic design of an $\overline{X}$-control chart*, Journal of Quality Technology **18**, 145–151.

[6] Collani, E. v. (1987a), *Optimal inspection and sampling procedures*, Proceedings of the IXth International Conference on Production Research, Cincinnati, vol. 1, 389–395.

[7] Collani, E. v. (1987b), *The economic design of $\overline{X}$-control charts*, Proceedings of IASTED International Symposium on Reliability and Quality Control, Paris, 186–189.

[8] Collani, E. v. (1989), *Economically optiml c- and np-control charts*, Metrika **36**, 215–232.

[9] Duncan, A. J. (1978), *The economic design of p-charts to maintain current control of a process: some numerical results*, Technometrics **20**, 235–243.

[10] Gibra, I. N. (1978), *Economically optimal determination of the parameters of np-control charts*, Journal of Quality Technology **10**, 12–19.

[11] Gibra, I. N. (1981), *Economic design of attribute control charts for multiple assignable causes*, Journal of Quality Technology **13**, 93–99.

[12] IMSL (1982) IMSL Library Reference Manual, IMSL Inc., Houston, TX.

[13] Ladany, S. P. (1973), *Optimal use of control charts for controlling current production*, Management Science **19**, 763–772.

[14] Montgomery, D. C. (1982), *Economic design of an $\overline{X}$ control chart*, Journal of Quality Technology **14**, 40–43.

[15] Montgomery, D. C., Heikes, R. G. and Mance, J. F. (1975), *Economic design of fraction defective control charts*, Management Science **21**, 1272–1284.

[16] Williams, W. W., Looney, S. W. and Peters, M. H. (1985), *Use of curtailed sampling plans in the economic design of np-control charts*, Technometrics **27**, 57–63.

```
C      Program Listing For The c Chart Design.

       EXTERNAL DPOIDF,UMACH
       INTEGER  NO,nStop,kStop,NOUT,K1,nBound
       IMPLICIT DOUBLE PRECISION (A-H,K-Z)

       b=100.0
       u=0.10
cccccccccccccc
       nBound=10
       kBound=10
cccccccccccccc
                  CALL UMACH(2,NOUT)
       OPEN ( 2,FILE='c-np.DAT',STATUS='OLD')
       OPEN (4,FILE='nk10.PRN')
500    READ (2,15,END=5) a,d
15     FORMAT (F6.4,1X,F6.4)
       WRITE(4,30)
30     FORMAT (///10X,'THIS IS OPTIMAL SOLUTIONS OF THE c CHART'//)
       CALL GETTIM (IH1,IM1,IS1,IHS1)
       WRITE (4,98)IH1,IM1,IS1,IHS1
98     FORMAT (5X,'THE START TIME IS   ',I2,':',I2,':',I2,':',I2//)
       write(4,40)
40     format(1x,'      ',2x,'n',4x,'k',4x,'x',5x,'alpha',3x,'Beta',8x,'P
      *')

cccccccccccccccccccccccccccccccccccccccc
c                                      c
c      Find The Optimal Solution.      c
c                                      c
cccccccccccccccccccccccccccccccccccccccc

         MINS=-100000.00
         nStop=0
         DO 100 n1 = 1.,1000.
            S1=-100000.00
            kStop=0
            u1=d*u
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
            k2=float(int(n1*u))
            k4=float(int(n1*u+6*dsqrt(n1*u)))
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
            DO 200 k1 = K2,k4,1.0
                CALL UMACH(2,NOUT)

                theta0=n1*u
                theta1=n1*u1

                beta=DPOIDF(k1,theta1)
                p=1-beta
                IF(P .LT. 0.01)GOTO 200
                ARFA=1.-DPOIDF(k1,theta0)
                R1= (1+beta)*(0.5*(b+arfa/2.)* (1-beta)-a*n1*
      *            (1+beta)/4.)
                R2= -(1-beta*beta)*(a*n1+arfa)
                R3= -(1-beta)*(1-beta)*(a*n1+arfa)
                x = (-R2+DSQRT(R2*R2-4*R1*R3))/(2*R1)
c
                IF(x .LE. 0.0 .or. x .gt. 100)GOTO 300
                   b0 = dexp(x)
```

```
                S = ((b*(b0-1.)-arfa)*(1-beta)/(b0-beta)-a*n1)/x
             IF ( S .gt. S1) THEN
                ARFA1=ARFA
                S1=S
                N2=N1
                k21=k1
                x2=x
                P2=P
                kStop=0
             ELSE
                kStop=kStop+1
             ENDIF
             IF (kStop .GT. kBound) GOTO 300
 200       CONTINUE
 300       IF (S1 .gt. MINS) THEN
             ARFA2=ARFA1
             MINS=S1
             N=N2
             x3=x2
             k=k21
             P3=P2
             nStop=0
           ELSE
             nStop=nStop+1
           ENDIF
           IF (nStop .GT. nBound) GOTO 400
 100    CONTINUE

400   Beta3 = 1-P3
      WRITE (4,50) N,k,x3,arfa2,Beta3,MINS
 50   FORMAT(4X,3X,f4.0,1X,f4.0,F6.3,2X,F6.4,2x,F6.4,F11.5//)
      CALL GETTIM (IH2,IM2,IS2,IHS2)
      WRITE (4,99)IH2,IM2,IS2,IHS2
99    FORMAT (5X,'THE END TIME IS   ',I2,':',I2,':',I2,':',I2/)
      GOTO 500
 5    CLOSE(2,STATUS='KEEP')
      CLOSE(4,STATUS='KEEP')
      STOP
      END
```