

A SAS Macro for Implementing a Nonparametric Control Charting Scheme for Individual Observations

Austin R. Brown, Ph.D., Kennesaw State University

ABSTRACT

In process monitoring applications, a commonly encountered scenario is one in which several presumably identical processes are desired to be monitored simultaneously. Such a process is referred to as a "Multiple Stream Process (MSP)." Several parametric control charts have been developed for monitoring MSPs with the majority being derived from the classical Shewhart X-bar chart. These control charts perform best when the assumption of normality is met, but their performance can deteriorate substantially when the underlying data are non-normal. This deterioration is magnified when the sample size is $n=1$. While some work has been done to address the issue of non-normality through the development of a nonparametric MSP chart, a review of the literature shows an apparent gap for nonparametric MSP charts where the sample size is one. Thus, this paper introduces a novel method to address this gap, which is based on the classical nonparametric test, Cochran's Q Test and utilizes the EWMA framework, as well as a SAS macro for implementing the new technique. Derivation of the charting statistic and control limits will be given as well as recommendations for selection of chart parameters for a desired average run length performance in addition to an applied example of usage.

INTRODUCTION

Statistical process control charts have historically been shown to be useful graphical tools for monitoring various business processes for the purpose of ensuring quality. All control charts have the same key characteristics: (1) a plotting statistic, which is calculated using sample data taken from the process; (2) a target value to which the plotting statistic ideally adheres; and (3) control limits which the plotting statistic is compared to at each sampled time point. If the plotting statistic exceeds either an upper or lower control limit (UCL and LCL respectively), then we conclude that the process is out-of-control (OOC). This implies that the true target value may have shifted away from the desired target value. If the opposite occurs, we conclude that the process is in-control (IC). Ideally, if the process has shifted away from the desired target value, we would like to know this in as few samples as possible to avoid prolonged deteriorations in quality. Similarly, if the process is IC, then we would like to avoid receiving false alarms (i.e., plotting statistics incorrectly plotting outside of the control limits) too frequently.

Over the last century, many control charts have been developed to take into consideration all of the different processes which can be monitored. For example, there are control charts for monitoring the mean of some process, the number of nonconforming units in a sample, as well as the proportion of nonconforming units in a sample (Montgomery, 2013). Control charts, much like the various statistical tests available to us, should be carefully chosen to match the characteristics of the monitored process. For example, a control chart developed for monitoring the mean of some process, say the number of minutes spent on hold for customers calling into a customer support center, would not be appropriate for monitoring the proportion of calls a customer service representative fields which are more than five minutes in length. In other words, the control chart chosen should appropriately address all aspects of the process or processes including time what characteristic of the process is to be monitored and how that characteristic is measured (i.e., is it a quantitative variable or categorical variable?).

Traditionally, most control charts have been developed to monitor a single or univariate process and assume that the plotting statistic follows some known distribution. As is the case with statistical tests which depend on distributional assumptions being met, if the monitored process does not meet the distributional assumption of the control chart, then the chart loses efficiency. Here, "efficiency" refers to the speed by which the chart signals OOC when the process really is OOC. Clearly, if we are to use a parametric chart, that is, one which relies on a distributional assumption, then we need to ensure that the monitored process really does follow said distribution to avoid lags in quality. Additionally, sometimes the monitored process is actually several independent and identical processes which we may want to monitor simultaneously rather than independently. Such a process is referred to as a "Multiple Stream Process (MSP)" (Montgomery, 2013). For example, we may be a manager of a customer service call center, and we want to monitor the hold times for customers for each of our 10 customer service representatives. Each representative would be considered an individual stream in this case. Control charts which have been developed for monitoring MSPs traditionally assume the process follows a Normal distribution (Brown and Schaffer, 2020).

To address the issue of possibly not knowing the distribution of a monitored process, control charts which do not depend on any distributional assumption have been developed which are referred to as "nonparametric" control charts. The reader is directed to Chakraborti (2014) for an overview of many historical examples of nonparametric charts, most of which use traditional nonparametric test statistics in the construction of their plotting statistics. Note, typically the parameter being tested by nonparametric tests is the median rather than the mean. Thus, nonparametric control charts are typically used to monitor the process median. To fill an apparent gap in the literature, Brown and Schaffer (2020) proposed a nonparametric MSP chart called the "Nonparametric Extended Median Test Cumulative Summation (NEMT-CUSUM)" chart. One limitation of the NEMT-CUSUM is that the sample size taken at each time point for each stream must be at minimum 10. Sometimes, our sample size taken at each time point may in fact be a single observation. While control charts have been developed for monitoring individual observations (see Montgomery (2013)), no control chart has been developed for monitoring an MSP when the sample size is one and certainly no nonparametric control chart has been either. Thus, a new nonparametric control chart has been proposed which takes into consideration the situation in which we are monitoring an MSP where the sample size is one for each stream at each time point. This chart was presented at the 2022 Joint Statistical Meetings and is called the "Cochran's Q Exponentially Weighted Moving Average (CQ-EWMA)" control chart (Brown and Whitehead, 2022).

The issue with developing new control charting techniques is that they exist only in theory. Most of them do not have SAS procedures specifically designed for their implementation in real organizations. The CQ-EWMA is no different. Resultingly, the purpose of this paper is to explain in general how the CQ-EWMA works and how a SAS macro called "%rogue_one" can be used for implementing the CQ-EWMA in your own organizations.

FICTICIOUS MONITIONRING SCENARIO

For illustrative purposes, let's assume we are the regional manager of a paper supply company and that we have 10 salespeople working for us. In order for our branch to meet our sales goals, we need our salespeople to be making ample sales daily. At the end of each day, we count up the number of sales each salesperson has made (just the quantity, not the dollar amounts). Suppose we do this each day for a month (20 total observations for each salesperson) and that our target median number of sales per day is 5 for each salesperson.

DATA STRUCTURE

Note, in order for %rogue_one to work properly, the data have to be organized in a specific fashion. Specifically, we need the data to be in a wide format, where each row represents an individual observation for every sampled time point and each column represents an individual salesperson or stream. Note, here a sample number column is not included since the row number serves the same purpose.

Obs	sales1	sales2	sales3	sales4	sales5	sales6	sales7	sales8	sales9	sales10
1	5	6	6	4	7	4	5	6	5	6
2	5	5	4	5	5	5	5	5	5	5
3	6	6	7	3	4	6	6	5	5	5
4	6	4	6	5	4	3	6	6	6	4
5	6	5	4	5	6	3	5	5	4	6
6	5	4	5	6	6	6	5	5	6	4
7	5	4	5	6	4	4	5	5	4	3
8	5	4	6	6	4	5	5	4	6	5
9	5	6	5	5	4	6	6	4	4	3
10	7	3	5	6	4	6	4	5	5	5

Figure 1: Example of Data Structure Necessary for Macro to Function Properly

CQ-EWMA CHART CONSTRUCTION

COMPARING OBSERVED VALUES TO THE MEDIAN

As previously mentioned, most nonparametric control charts monitor the median of a process and the CQ-EWMA is exactly the same in this regard. If the process is in a state of statistical control, then we would expect about half of our observed values to be greater the median and about half to be less than the process median. In our example, we are assuming we know the median number of daily sales is five. In general, the process median is denoted by $\tilde{\mu}_0$. The way the CQ-EWMA chart initially works is by comparing the value observed at each sampled time point, say y_{it} , number of sales made in a day in our example, to the target median (Note, y_{it} represents an observation for the i th stream at the t th time point). If $y_{it} > \tilde{\mu}_0$, then we assign a value of "1" to a new indicator variable called x_{it} . If $y_{it} \leq \tilde{\mu}_0$, then we assign a value of "0" to x_{it} . Then, after all y_{it} 's have been converted to x_{it} 's for a given time point, we sum up the x_{it} 's into a quantity called C_t . In general, assume we have k streams, where $k \geq 10$, and we have taken t samples from each of our k streams. Table 1 demonstrates the general procedure for an arbitrary number of time points sampled.

	Sample Number			
Stream	1	2	...	t
1	x_{11}	x_{12}		x_{1t}
2	x_{21}	x_{22}		x_{2t}
:	:	:	...	:
k	x_{k1}	x_{k2}		x_{kt}
Column Totals	C_1	C_2	...	C_t

Table 1. Comparing Individual Observations to the Median

What may become clear here is that, if we can assume each of our streams are independent of each other (and this seems reasonable in our sales example), that the C_t values all follow a Binomial Distribution, which suggests:

$$\begin{aligned} E[C_t] &= k(0.50) \\ \text{Var}[C_t] &= k(0.50)^2. \end{aligned} \tag{1}$$

To convert the sales data we observed in Figure 1 into x_{it} values we need in Table 2, we can do so by making use of an array within the `DATA` step:

```
/* Tabulate Xt Values Using an Array */
data xt(drop=i sales1-sales10);
set paper;
array sales[10] sales1-sales10;
array xt[10];
do i=1 to 10;
    if sales[i] > &mu0 then xt[i] = 1;
    else xt[i] = 0;
end;
run;
```

The resulting data table, `xt`, is shown in Figure 2.

Obs	xt1	xt2	xt3	xt4	xt5	xt6	xt7	xt8	xt9	xt10
1	0	1	1	0	1	0	0	1	0	1
2	0	0	0	0	0	0	0	0	0	0
3	1	1	1	0	0	1	1	0	0	0
4	1	0	1	0	0	0	1	1	1	0
5	1	0	0	0	1	0	0	0	0	1
6	0	0	0	1	1	1	0	0	1	0
7	0	0	0	1	0	0	0	0	0	0
8	0	0	1	1	0	0	0	0	1	0
9	0	1	0	0	0	1	1	0	0	0
10	1	0	0	1	0	1	0	0	0	0

Figure 2: Indicator Variable Recoding

CALCULATING COLUMN TOTALS, STANDARDIZED VALUES & CUMULATIVE SUMS

Next, we need to sum the columns to create our C_t values. But to more be more efficient in the amount of code necessary for the macro to work as intended, multiple steps can be combined into one `DATA` step. In the CQ-EWMA framework, the C_t 's, being binomial random variables, are standardized into values called Z_t 's:

$$Z_t = \frac{C_t - k(0.50)}{0.50\sqrt{k}}. \quad (2)$$

From here, the Z_t 's are cumulatively summed into values denoted Q_t , which is the value to be used in the traditional EWMA framework:

$$Q_t = \sum_{i=1}^t Z_t. \quad (3)$$

We can perform all three tasks in one step. However, we need the names of the columns of the `xt` data table to do so. To achieve this task, we can make use of another macro function called `%get_cols_delim`:

```

/* Create Macro to Get Column Names */
%macro get_cols_delim(lib,mem,mvar,type,delim=" ");
%global &mvar;
  proc sql noprint;
    select
      name
    into
      :&mvar separated by '&delim'
    from
      dictionary.columns
    where
      libname eq upcase("&lib")
      and memname eq upcase("&mem")
    %if %upcase(&type) ne ALL %then
      and upcase(type) eq upcase("&type");
  ;
  quit;
%put &mvar = &&&mvar;
%mend get_cols_delim;

```

In this macro, "lib" is the library where the data table, `xt`, is stored, "mem" is the name of the data table itself, "mvar" is what you decide to name the macro variable containing the column names, "type" refers to the type of variable you want the name of (e.g., numeric, character, or all), and "delim" is the delimiter separating the column names in the macro variable. For the given data table, we can see that this macro executes properly by executing the following code:

```
%get_cols_delim(WORK,xt,col_names,all,delim=",");
```

Now, we can use the macro variable `&col_names` in the DATA step described before:

```

/* Calculate Ct, Zt, Qt values using Data Step */
data cts(KEEP=Ct Zt Qt);
set xt;
Ct = sum(&col_names);
Zt = (Ct - %sysevalf(&k*0.50)) / (%sysevalf(&k*0.25)**0.50);
if _N_ = 1 then Qt=Zt;
else if _N_ > 1 then Qt+Zt;
run;

```

The data contained in the `cts` can be used to actually calculate the plotting statistic and control limits.

CALCULATING CQ-EWMA PLOTTING STATISTIC AND CONTROL LIMITS

In the EWMA framework, the plotting statistic is a function of the quantitative variable being monitored. In our case, the quantitative variable being monitored is Q_t . The plotting statistic, denoted as r_t , is defined as:

$$r_t = \lambda Q_t + (1 - \lambda)r_{t-1}, \quad (4)$$

where $0 < \lambda < 1$ and $r_0 = 0$. To determine the control limits, we need to have knowledge of the variance of r_t . Here, it is a little different from the traditional EWMA charting technique in that the Q_t 's are clearly dependent upon each other. It can be shown that the variance for r_t is:

$$\begin{aligned} \text{Var}[r_t] &= \lambda^2 \left[\sum_{j=1}^t (1-\lambda)^{2j-2} j + 2 \sum_{j=1}^t (1-\lambda)^{2(j-1)} j \left[\frac{1 - (1-\lambda)^{t-j}}{\lambda} \right] \right] \\ &= \sigma_r^2. \end{aligned} \quad (5)$$

Thus, we can define an upper and lower control limit as:

$$\begin{aligned} UCL &= L\sigma_r, \\ LCL &= -L\sigma_r, \end{aligned} \quad (6)$$

where L denotes the half-widths of the control limits. To first calculate the values of r_t , the Q_t values were isolated in their own data table.

```
data qt;
set cts;
keep Qt;
run;
```

From there, the Q_t values were transposed using `TRANSPPOSE` procedure in order to later make use of `ARRAY` processing:

```
/* Transpose */
proc transpose data=qt out=rt1(DROP=_NAME_);
var Qt;
run;
```

We once again make use of the `%get_cols_delim` macro function in order to extract the names of the columns in the `rt1` data table outputted from `PROC TRANSPPOSE` in order to assign them to an `ARRAY` in the `DATA` step. Here, we are also making use of `DO LOOPS` to calculate the r_t values in a new data table called `rt`:

```
/* Get Column Names of Transposed Data Table */
%get_cols_delim(WORK,rt1,col_names1,all,delim=" ");
/* Calculate Rt */
data rt(DROP=&col_names1 i);
set rt1;
array ewma[&n] &col_names1;
array rtz[&n];
do i=1 to &n;
    if i=1 then rtz[i] = ewma[i]*&lambda;
    else rtz[i] = ewma[i]*&lambda + &oml*rtz[i-1];
end;
run;
```

We can then transpose the `rt` data table back to long format to be rejoined with the `cts` data table notably making use of the `%get_cols_delim` macro function again.

```
/* Transpose to Long */
%get_cols_delim(WORK,rt,col_names2,all,delim=" ");
proc transpose data=rt out=rt2(DROP=_NAME_ _LABEL_
                             RENAME=COL1=rt);
var &col_names2;
run;
```

```
/* Append to cts */
```

```
data cts1;  
merge cts rt2;  
run;
```

Now, we are nearly ready to calculate the control limits. First, we need to calculate the variance as given by (5). To do so, the variance and covariance pieces of (5) were calculated separately using a sample number variable created with the `MONOTONIC` function. In a single `DATA` step, the resulting variance terms were used to calculate the control limits given by (6), and then remerged with `cts`. Notably here, `IF-THEN-DO` logic was utilized in order to ensure computations are taking place correctly.

```
/* Calculate Control Limits & Merge with cts1 */
```

```
data var(drop=i);  
do i=1 to &n;  
    sample=i;  
output;end;  
run;  
  
data var1(drop=v1 v2 v3 v4);  
set var;  
v1 = &oml**(2*sample-2)*sample;  
v2 = 2*&oml**(2*sample-2)*sample*((1-&oml**(&n-sample))/&lambda);  
if _N_ = 1 then do;  
    var=&lambda**2*(v1+v2);  
    v3 = v1;  
    v4 = v2;  
end;  
else if _N_ > 1 then do;  
    v3+v1;  
    v4+v2;  
    var = &lambda**2*(v3+v4);  
end;  
UCL = &L*sqrt(var);  
LCL = (-1)*&L*sqrt(var);  
merge cts1;  
run;
```

From here, we can compare the values of r_t to the control limits in order to determine if the process is in a state of statistical control or not. We do this in two ways: with a data table similar to the output generated by the `SHEWHART` procedure and with a chart generated by the `SGPLOT` procedure.

```
/* Determine if any points are outside of the control limits */
```

```
data cqewma;  
set var1;  
if rt > UCL | rt < LCL then Violation = "Yes";  
else Violation = "No";  
run;  
  
data violations;  
set cqewma;  
where Violation = "Yes";  
run;
```

```

/* Generate Plot */
proc sgplot data=cqewma;
series x=Sample y=UCL/lineattrs=(color=blue);
series x=Sample y=LCL/lineattrs=(color=blue);
series x=Sample y=rt/markers lineattrs=(color=red)
                                markerattrs=(color=red);

axis label="Time Point";
yaxis label="Plotting Statistic Value";
title "CQ-EWMA";
run;
title;

```

The generated chart is given by Figure 3.

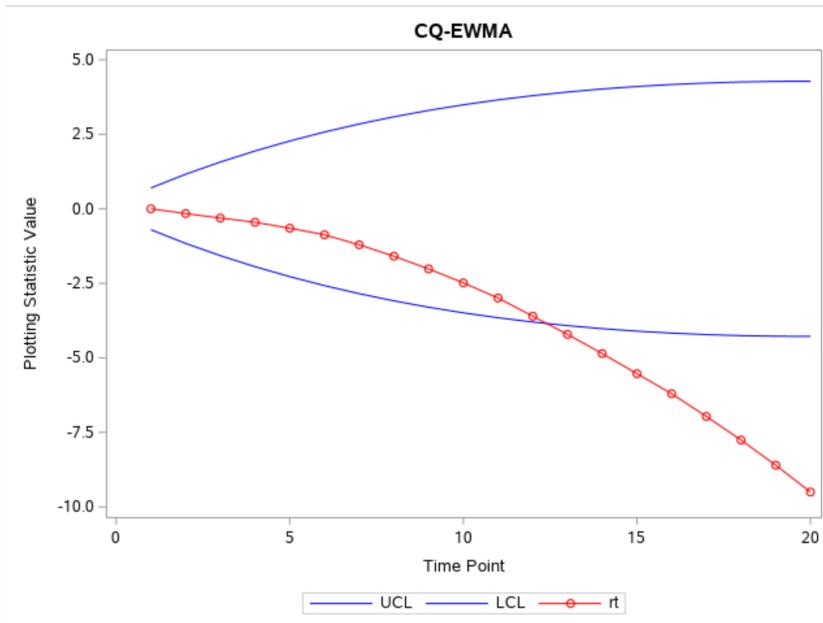


Figure 3: CQ-EWMA Control Chart for Regional Paper Supplier

Here, we can see that the chart is signaling an out-of-control point around point 13. We can confirm this by also making use of the `violations` data table, shown in Figure 4.

Obs	sample	var	UCL	LCL	Ct	Zt	Qt	rt	Violation
1	13	2.03038	3.91851	-3.91851	4	-0.63246	-15.8114	-4.21179	Yes
2	14	2.13733	4.02040	-4.02040	3	-1.26491	-17.0763	-4.85501	Yes
3	15	2.22695	4.10382	-4.10382	3	-1.26491	-18.3412	-5.52932	Yes
4	16	2.29924	4.16989	-4.16989	4	-0.63246	-18.9737	-6.20154	Yes
5	17	2.35444	4.21965	-4.21965	1	-2.52982	-21.5035	-6.96664	Yes
6	18	2.39299	4.25406	-4.25406	3	-1.26491	-22.7684	-7.75673	Yes
7	19	2.41548	4.27399	-4.27399	2	-1.89737	-24.6658	-8.60218	Yes
8	20	2.42259	4.28029	-4.28029	2	-1.89737	-26.5631	-9.50023	Yes

Figure 4: Violations Data Table

After removing the extraneous data tables from the `WORK` library using the `DATASETS` procedure, the `%rogue_one` macro is complete and has the following form:

```
%rogue_one(dt,mu0,L,lambda,lib);
```

Here, "dt" is the name of the initial data table to be analyzed, "mu0" is the target median value, "L" is the half-width (recommended to be 2.75 by Brown and Whitehead (2022)), "lambda" is the weighting parameter (recommended to be 0.05 by Brown and Whitehead (2022)), and "lib" is the name of the library where "dt" is saved. In our regional paper supplier example, the code we would use is:

```
%rogue_one(paper,5,2.75,0.05,work);
```

ADDRESSING AN OUT-OF-CONTROL POINT

We can see in the previous example that we have evidence of an out-of-control process. We can also see from the chart that the sequence of points is moving in the negative direction. Contextually, this would mean that at least one of our salespeople is not meeting the daily sales goals necessary for our branch to meet our quota. In this case, it would be recommended to use the `xt` table to determine which of our salespeople may need assistance in achieving their sales goals. To do this, I would recommend using a procedure similar to that proposed by Brown and Schaffer (2020). That is, starting from the "inflection-point," that is the point in the r_t series where the points start moving toward the control limit they will ultimately surpass, sum up the columns from that point to the end of the series. In our case, it looks like the series immediately starts moving toward the lower control limit. So we would sum up all of the columns in the `xt` table using the `PRINT` procedure.

```
/* Summing up Columns of xt */
proc print data=xt;
sum xt1-xt10;
run;
```

We can see the results in Figure 5.

Obs	xt1	xt2	xt3	xt4	xt5	xt6	xt7	xt8	xt9	xt10
1	0	1	1	0	1	0	0	1	0	1
2	0	0	0	0	0	0	0	0	0	0
3	1	1	1	0	0	1	1	0	0	0
4	1	0	1	0	0	0	1	1	1	0
5	1	0	0	0	1	0	0	0	0	1
6	0	0	0	1	1	1	0	0	1	0
7	0	0	0	1	0	0	0	0	0	0
8	0	0	1	1	0	0	0	0	1	0
9	0	1	0	0	0	1	1	0	0	0
10	1	0	0	1	0	1	0	0	0	0
11	0	0	0	0	0	1	0	0	1	1
12	1	0	0	0	0	0	0	0	0	0
13	1	1	1	1	0	0	0	0	0	0
14	1	0	0	0	0	0	1	1	0	0
15	1	1	0	0	1	0	0	0	0	0
16	0	1	0	1	0	1	0	0	0	1
17	0	0	0	0	1	0	0	0	0	0
18	0	0	0	0	0	1	0	1	0	1
19	0	0	1	0	0	1	0	0	0	0
20	0	0	0	0	0	0	1	0	0	1
	8	6	6	6	5	8	5	4	4	6

Figure 5: Column Totals from xt Data Table

Here, since our plotting statistic went below the lower control limit, we would be interested in determining which of our salespeople met their daily sales goals the least frequent number of days. From Figure 5, it would seem that salespeople 5, 7, 8, and 9 likely contributed to the out-of-control point.

CONCLUSIONS

Statistical process control charts can be useful and effective tools in process monitoring. Because processes can take many forms, we need to be careful in selecting a charting technique most appropriate for the process being monitored. If we are monitoring a MSP where our sample size at each time point is $n = 1$ and we don't know or can't confirm that our data come from a Normal distribution, the CQ-EWMA chart is a nice option. In this paper, we saw how the `%rogue_one` macro is constructed and saw how it provides the end user with graphical and tabular output similar to that of `PROC SHEWHART`. This output helps the end user make decisions about the state of their monitored process.

REFERENCES

- Brown, A.R., and Whitehead, B.J. (2022). *A Nonparametric Control Chart for Monitoring Individual Observations of Multiple Stream Processes based on Cochran's Q Test*. [Poster]. 2022 Joint Statistical Meetings, Washington, D.C.
- Brown, A. R., and J.R. Schaffer. (2020). A nonparametric CUSUM control chart for multiple stream processes based on a modified extended median test. *Communications in Statistics - Theory and Methods*, 50 (24), 6067–6080. <https://doi.org/10.1080/03610926.2020.1738492>
- Chakraborti, S. (2014). Nonparametric (Distribution-free) quality control charts. *Wiley StatsRef: Statistics Reference Online*, ed. N. Balakrishnan, T. Colton, B. Everitt, W. Piegorisch, F. Ruggeri, J. L. Teugels. <https://doi.org/10.1002/9781118445112.stat02699>.
- Montgomery, D. C. (2013). *Statistical quality control: A modern introduction*. Hoboken, NJ: Wiley.

ACKNOWLEDGMENTS

I am very grateful to my friend and colleague Bryce Whitehead, without whom this work would not be possible. I am also grateful to my colleagues at Kennesaw State University for encouraging me to pursue this work.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Austin R. Brown, Ph.D.
Kennesaw State University
College of Computing and Software Engineering
School of Data Science and Analytics
abrow708@kennesaw.edu

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.