Monitoring High-Yields Processes with Defects Count in Nonconforming Items by Artificial Neural Network

Babak Abbasi, Ph.D. Candidate
Department of Industrial Engineering, Sharif University of Technology
P.O. Box 11365-9414 Azadi Ave., Tehran, Iran
Phone: (+98912) 3118168, E-mail: b_abbasi@mehr.sharif.edu

Seyed Taghi Akhavan Niaki, Ph.D., Professor
Department of Industrial Engineering, Sharif University of Technology
P.O. Box 11365-9414 Azadi Ave., Tehran, Iran
Phone: (+9821) 66165740, Fax: (+9821) 66022702, E-mail: Niaki@Sharif.edu

Abstract

In high-yields process monitoring, the Geometric distribution is particularly useful to control the cumulative counts of conforming (CCC) items. However, in some instances the number of defects on a nonconforming observation is also of important application and must be monitored. For the latter case, the use of the generalized Poisson distribution and hence simultaneously implementation of two control charts (CCC & C charts) is recommended in the literature. In this paper, we propose an artificial neural network approach to monitor high-yields processes in which not only the cumulative counts of conforming items but also the number of defects on nonconforming items is monitored. In order to demonstrate the application of the proposed network and to evaluate the performance of the proposed methodology we present two numerical examples and compare the results with the ones obtained from the application of two separate control charts (CCC & C charts).

Key words

1 Corresponding Author
1. Introduction

Traditional attribute control charts, such as $p$ and $c$ charts, are not suitable in automated high-yield manufacturing and continuous production processes. Although there are some procedures to improve the performance of $p$ chart when the proportion of nonconforming is low (see Acosta- Mejia (1999)), these procedures are not applicable to the cases of continuous inspection or when the defect rate is very low (Woodall (1997)). For such a process, the quality level is usually at parts per million (ppm), or almost zero defects, so that even for a sample size of thousands, usually no nonconforming item is observed. For these processes, Goh (1987), Kaminsky et al. (1992), Glushkovsky (1994), Xie and Goh (1995, & 1997), and Nelson (1994) recommend applying the Cumulative Count of Conforming (CCC) chart. In this chart, which is based on the geometric distribution, we count the cumulative conforming items until the first nonconforming item is observed.

In high-quality processes, there are many instances such that when we observe a nonconforming item, the number of defects in the nonconforming item is of an important use too. As an example, we may refer to Xie and Goh (1993) who present a real example of computer hard disks manufacturing process. For these situations, He et al. (2002) show that the Generalized Poisson Distribution (GPD) is very useful and fits these processes very well, especially when the ratio of the sample variance to the sample mean is significantly larger than one. This is the case in high quality processes (see Lambert (1992), Xie and Goh (1993) and Collica et al. (1996) for more details of over desperation processes). He et al. (2002) present control charts based on Generalized Poisson distribution for high quality process with number of defects in nonconforming item. They recommend the application of two control charts simultaneously to monitor the interval
between two nonconforming items (the CCC chart) and the number of defects on the nonconforming item (the c chart). We note that when we apply two separate univariate control charts for multivariate cases simultaneously, the determination of type one error becomes a problem. Another serious problem arises in situations in which both deterioration on the cumulative counts of conforming and improvement on the number of defects on the nonconforming item occur at the same time such that the overall out-of-control condition cannot be detected.

In this paper we present an artificial neural network (ANN) based approach to monitor high-yields processes. We design the network such that it not only controls both the interval between two nonconforming items and the number of defects on the nonconforming item simultaneously, but will also be able to detect different magnitudes of mean-shifts. To do this, first in section two we briefly explain the procedure proposed by He et al. (2002). Then in section three, we explain the fundamentals of neural network, the Perseptron neural networks, and the training process. In section four, we introduce neural network modeling applied for high-yield process monitoring. We present two simulation examples to illustrate the proposed method and to compare its performance with He et al.’s (2002) procedure in section five. Conclusion and recommendations for future research will come in section six.

2. High-Yields Process Monitoring based on Simultaneous CCC and C Charts

When the data from a high-yields process is coming from a generalized Poisson model, He et al. (2002) propose a procedure using two simultaneous control charts. In the
generalized Poisson distribution (GPD), we assume that the overall probability of finding $x$ defects in a product is

$$P_x(\theta, \lambda) = \frac{\theta(\theta + x\lambda)^{x-1} e^{-\theta-x\lambda}}{x!}, \quad x = 0,1,2..., \quad \lambda \geq 0. \tag{1}$$

in which the probability of nonconforming, $(p)$, is equal to $1 - e^{-\theta}$. If $\theta$ increases, the defective rate of the process increases and vice-versa. Hence, $\theta$ shows the defective rate of the process. For the parameter $\lambda$, on one hand, we can see that when $\lambda$ is small; there will be some small but frequent non-zero counts. On the other hand, when $\lambda$ increases, we will observe larger but less frequent non-zero counts. Therefore, we can interpret $\lambda$ as the size of non-zero count. Using these interoperations, He et al. (2002) apply two separate control charts to monitor parameters of a GPD. They use a geometric chart (CCC chart) to monitor parameter $\theta$ and apply a c chart to control $\lambda$. For the CCC chart, after defining the probability of false alarm rate, $\alpha_{ccc}$, the control limits are:

$$\begin{align*}
UCL_{ccc} &= \frac{\ln(\alpha_{ccc})}{\ln(1-p)}, \\
C_{ccc} &= \frac{\ln(0.5)}{\ln(1-p)}, \text{ and } LCL_{ccc} = \frac{\ln(1-\alpha_{ccc})}{\ln(1-p)} \tag{2}
\end{align*}$$

Then we monitor the parameter $\theta$ by plotting the cumulative conforming counts on this chart.

For the c chart, noting that the nonconformities only occur under the condition that the product is nonconforming, we obtain the control limits by conditioning, i.e.:

$$P(k \text{ nonconformities} / \text{nonconforming}) = P(x = k / x > 0) = \frac{\theta(\theta + k\lambda)^{k-1} e^{-\theta-k\lambda}}{k!(1-e^{-\theta})}, \quad k = 0,1,2,...$$

This is called zero-truncated generalized Poisson distribution (Consul (1989)) and the mean and variance of this distribution are:
\[ E(X) = \theta (1 - \lambda)^{-1} (1 - e^{-\theta})^{-1} \]
\[ Var(X) = \left[ \theta (1 - \lambda)^{-3} + \theta^2 (1 - \lambda)^{-2} \right] (1 - e^{-\theta})^{-1} - \theta^2 (1 - \lambda)^{-2} (1 - e^{-\theta})^{-2} \]  

Hence the 3-sigma control limits for an attribute control chart are:

\[ UCL_c = E(X) + 3\sqrt{Var(X)} \, , \, CL_c = E(X) \, , \, \text{and} \, \, LCL_c = E(X) - 3\sqrt{Var(X)} \]  

and the exact probability control limits are:

\[ UCL_c \text{ such that } \sum_{k=1}^{\infty} \frac{\theta(\theta + k\lambda)^{k-1} e^{-\theta-k\lambda}}{k!(1-e^{-\theta})} > \alpha_c \text{ and } \sum_{k=1}^{\infty} \frac{\theta(\theta + k\lambda)^{k-1} e^{-\theta-k\lambda}}{k!(1-e^{-\theta})} < \alpha_c \]  

where \( \alpha_c \) is the probability of type I error. In this chart, we plot the number of defects in nonconforming observations.

If any of the above control charts shows an out-of-control signal, then we classify the process to be in an out-of-control condition. However, for an out-of-control situation on one hand these charts usually are unable to diagnose the parameter(s) that caused the deterioration (see He et al. 2002). On the other hand, in many cases the \( LCL_{ccc} \) becomes zero; therefore this method is not capable of detecting a decreasing counts of conforming items.

3. Multilayer Perceptron Neural Networks (PNN)

PNN is perhaps the most popular network architecture in use today and is discussed at length in most neural network textbooks (e.g., Bishop, 1995). In this type of network, we arrange the units in a layered feed forward topology, where the units each perform a biased weighted sum of their inputs and pass this activation level through a transfer function to produce their output. The network thus has a simple interpretation as a form
of input-output model, with the weights and thresholds (biases) as the free parameters of
the model. Figure (1) shows the topology of PNN with one hidden layer. Such networks
can model functions of almost arbitrary complexity, with the number of layers, and the
number of units in each layer, determining the function’s complexity. Important issues in
Multilayer Perceptrons (MLP) design include specification of the number of hidden
layers and the number of units in these layers (see Haykin, 1994 and Bishop, 1995).

Insert Figure (1) about here

The number of input and output units is defined by the problem and the number of hidden
units to use is far from clear. A good starting point is to use one hidden layer and trade
the number of units in the hidden layer.

3.1. Training Multilayer Perceptrons

Once we select the number of layers and the number of units in each layer, the network's
weights and thresholds must be set to minimize the prediction error made by the network.
This is the role of the training algorithms. This process is equivalent to fitting the model
represented by the network to the training data available. The error of a particular
configuration of the network can be determined by running all the training cases through
the network and comparing the actual output generated with the desired (target) outputs.
Then, by an error function, we combine the differences together to get the network error.
The most common error functions used in the literature are the sum of squared error
(SSE), in which we square and sum together the individual errors of output units in each case.

### 3.2. The Back Propagation Algorithm

The best-known example of a neural network-training algorithm is the back propagation algorithm (see Patterson (1996), Haykin (1994), and Fausett (1994)). Modern second-order algorithms such as conjugate gradient descent and Levenberg-Marquardt (see Bishop (1995)) are substantially faster for many problems, however back propagation still has advantages in some circumstances, and is the easiest algorithm to understand. Therefore, we have chosen to use this algorithm. There are also heuristic modifications of back propagation, which work well for some problem domains.

In back propagation, the gradient vector of the error surface is calculated. This vector points along the line of steepest descent from the current point, so we know that if we move along it a "short" distance, we will decrease the error. A sequence of such moves, slowing as we near the bottom, will eventually find a minimum of some sort. The difficult part is to decide how large the steps should be.

Large steps may converge more quickly, but may also overstep the solution or go off in the wrong direction (if the error surface is very eccentric). A classic example of this in neural network training is where the algorithm progresses very slowly along a steep, narrow, valley, bouncing from one side across to the other. In contrast, although very small steps may go in the correct direction, they also require a large number of iterations. In practice, the step size is proportional to the slope (so that the algorithms settle down in a minimum) and to a special constant: the learning rate. The correct setting for the
learning rate is application-dependent, and is typically chosen by experiment; it may also be time varying, getting smaller as the algorithm progresses.

The algorithm therefore progresses iteratively, through a number of epochs. On each epoch, we submit the training cases in turn to the network and target actual outputs; then, compare and calculate the error. This error, together with the error surface gradient, is used to adjust the weights, and then the process repeats. The initial network configuration is random and training stops when a given number of epochs elapse, or when the error reaches an acceptable level, or when the error stops improving.

4. Using ANN to Monitor a High-Yields Process

Many quality engineers and researchers are familiar with the successful applications of Artificial Neural networks used to monitor univariate and multivariate processes (see Cheng (1997), Guh and Hsieh (1999), and Niaki and Abbasi (2005)). However, there is no research on the application of ANN to monitor high-yields processes. In these processes, the probability of finding a nonconforming item is very low and the number of defects in a nonconforming item is very important to be monitored. Therefore, we propose an ANN for the following purposes:

1- To avoid using two separate control charts

2- To detect out-of-control situations more effectively

3- To employ the network to data that do not come from a generalized Poisson distribution (If we have enough historical data for the training phase, we can use ANN robustly).

4- To ease implementation of high-yields process monitoring for quality engineers
In order to design and train a proper network, first we need to prepare training-data sets. We may generate these data in two general ways. If the data is coming from a generalized Poisson distribution, then we estimate its parameters and generate data accordingly. However, if the historical data do not follow a generalized Poisson distribution, then we can use them for training purposes directly. We set the network target values equal to zero and one for in-control and out-of-control processes, respectively. Moreover, the generated number of training-data sets for in-control processes is equal to this number for out-of-control processes.

After training-data sets preparation, we design an appropriate PPN by trial and error on the number of layers and neurons in hidden layers. In all experimentations, we try networks with one hidden layer; changing its number of neurons. The number of neurons in the input and output layers equal to two and one respectively. We train the network until it reaches an acceptable SSE (Haykin (1994)). In the detection phase, we need to determine a cutting-value by which the process is identified as in or out-of control. In other words, if the output of the network is less than the cutting-value, then the process is in-control, otherwise it is in an out-of-control situation. In this paper, we select the cutting-value based on a specified in-control Average Run Length \((ARL_0)\) obtained by simulation and the bisection method.

The idea of the bisection method is based on the fact that a function will change sign when it passes through zero. By evaluating the function at the middle of an interval and replacing whichever limit has the same sign, the bisection method can halve the size of the interval in each iteration and eventually find the root. For example, to find a root
of \( f(x) = 0 \) in the interval of \((a_0, b_0)\) with which \( f(a_0) f(b_0) < 0 \) we pick tolerance \( \varepsilon \) and then apply the following algorithm:

\[
x_{k+1} = \frac{b_k + a_k}{2} \quad \text{for } k = 0, 1, 2, \ldots
\]

If \( \left| f(x_{k+1}) \right| < \varepsilon \) then we have found the root. Stop iterations.
Else
\[
\text{If } (f(x_{k+1}) f(b_k) < 0) \text{ then } a_{k+1} = x_{k+1} \text{ and } b_{k+1} = b_k
\]
\[
\text{Else } a_{k+1} = a_k \text{ and } b_{k+1} = x_{k+1}
\]

For the problem at hand we have \( f(x) = ARL_0(x) - ARL_0 \) in which \( x \) is the cutting-value.

Now we are ready to test the proposed ANN and use it in real-world situations.

5. Performance Evaluation

In order to understand the proposed method better and to evaluate its performance, in this section we provide two simulation examples.

5.1. Simulation Example 1

Suppose that we have a process in which the historical data follow a generalized Poisson model, the maximum likelihood estimation (mle) of its parameters is \( \hat{\theta} = 0.01, \hat{\lambda} = 0.8 \), and we intend to design and train a neural network to monitor this process He et al. (2002)). To do this, first we generate 400 training-data sets for in-control (target is zero) and 100 sets for each of out-of-control cases that we want to identify (target is one). For an example of an out-of-control case, suppose we want to detect shifted values of 0.03 and 0.9 for \( \theta \) and \( \lambda \), respectively. In other words, we generate out-of-control training-data sets coming from a GPD with \((0.03,0.8),(0.01,0.9),(0.03,0.9)\) as its parameters.
Table (1) shows both the cumulative counts of conforming and the number of defects in nonconforming items of the training-data sets generated in 51 in-control replications of this process.

**Insert Table (1) about here**

We design a network with two neurons in its input layer, one neuron in its output layer, and find the number of neurons in its hidden layer by trial and error.

In the training phase, the SSE value of the network becomes 0.021 after 1550 epochs. Applying the bisection method and simulation for \( ARL_0 = 20 \) the cutting-value becomes 0.715. At this stage, we may test the trained network and compare its performance with CCC & C charts. To do this we use two criteria; the first one is the usual ARL criterion and the other one is the average number of items inspected to obtain an out-of-control alarm (Airl).

Table (2) shows ARL and Table (3) shows Airl values of both the trained ANN and CCC & C control charts for different shift values of the parameters in 10000 simulation replications.

**Insert Table (2) about here**

**Insert Table (3) about here**
The results show that the proposed ANN method performs better than the CCC & C control charts almost in every scenario. In situations in which CCC & C charts show better performance, the difference is not significant.

5.2. Simulation Example 2

In this numerical example suppose an in-control process follow a generalized Poisson distribution with mle estimation of its parameters as \( \theta = 0.14 \) and \( \lambda = 0.87 \). We intend to set up a neural network to identify out-of-control situations. In order to do this, first we generate 400 training-data set for in-control and 100 sets for each of out of control cases that we are interested in identifying. For the out-of-control cases, suppose we want to detect shifted values of 0.24 and 1.07 for \( \theta \) and \( \lambda \), respectively. In other words, we generate out-of-control training-data sets coming from a GPD with \((0.24,0.87),(0.14,1.07),(0.24,1.07)\) as its parameters. Table (4) shows 51 in-control generated samples of this process.

We design a network with two neurons in its input layer, one neuron in its output layer, and find the number of neurons in its hidden layer by trial and error method. The SSE becomes 0.039 after 1200 epochs. Applying the bisection method and simulation the cutting-value becomes 0.45 for an \( ARL_0 = 20 \). Table (5) shows the ARL and Table (6) shows the AIRL of both ANN and CCC & C control charts for different magnitudes of parameters shifts in 10000 simulation replications.
The results show that the proposed ANN method performs better than the CCC & C control charts in every scenario. Moreover, we see that the CCC & C charts are unable to detect the shifts in $\theta$ in scenarios where the cumulative counts of conforming items decreases. The out-of-control ARLs in these situations are even more than the in-control ARLs.

6. Conclusion and Recommendations for Future Research

Because traditional attribute control charts such as $p$ and $c$ charts are not suitable in automated high-yield manufacturing and continuous production processes, the cumulative count of conforming (CCC) control chart is presented. Moreover, in situations in which the number of defects in nonconforming item is also an important issue the simultaneous use of the CCC and $c$ charts is recommended. One of the pitfalls of this control charting method, which is based on the generalized Poisson distribution, is that in many cases its LCL becomes zero; therefore this method is unable to detect a decreasing number of conforming items and increasing number of defects in nonconforming item. In this paper, we proposed a preceptron artificial neural network approach to detect out-of-control condition in high-yields processes. The proposed method is easy to understand and to employ. Two numerical examples show that the performance of the proposed procedure is better than the CCC & c charts in terms of in & out-of-control ARL and
AIRL. Besides, the proposed method is robust to the underlying distribution of the process data. For the future research, we recommend the following:

a) In processes in which the data are not coming from a generalized Poisson distribution, the usual CCC and c charts are not applicable, but the proposed method does not require GPD data. A performance evaluation study is needed for this situation.

b) For an out-of-control signal, we may design an ANN to diagnose the parameter(s) that has (have) shifted as well.

c) We may design an ANN to both detect the out-of-control conditions and the magnitudes of parameters’ shifts.

7. References


