APPLICATION SOFTWARE FOR PERFORMANCE MONITORING

By

Rong Chen

B. E. (Automation) Tsinghua University, P.R.China

M. E. (Automation) Tsinghua University, P.R.China

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

We accept this thesis as conforming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

December 1998

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Electrical and Computer Engineering
The University of British Columbia
2075 Wesbrook Place
Vancouver, Canada
V6T 1Z1

Date: Dec. 23, 1998

Abstract

During recent years, control loop performance monitoring has been a topic of interest in both academic and industrial circles because the method gives non-intrusive insight into the operation of control loops. Many useful results have been obtained through research, however, their industrial applications are just starting. The main reason is the lack of good application software for implementation.

In this thesis, an integrated loop monitoring software system was developed. On one hand, it provides comprehensive performance monitoring tools, including process data analysis in both time domain and frequency domain, loop oscillation detection, valve friction detection, and performance index calculation. On the other hand, it integrates all the user-system interfaces from a distributed control system (DCS) into PC networks. Two direct benefits of this approach are: first, it shields the critical DCS system from user errors, avoiding potential damage to the industrial process; second, it presents a friendly graphic user interface (GUI) to the user, making the whole system much easier to use.

The whole software system was developed in an industrial environment, and preliminary evaluation has been carried out.

Table of Contents

List of Tables

List of Figures

Acknowledgment

First, I would like to express my sincere thanks to my supervisors, Prof. Guy A. Dumont and Prof. Michael S. Davies, who have guided me through my whole Master's program at UBC.

Also, I want to express appreciations to Mr. John Ball, who has given me a great deal of help during my stay at Prince George Pulp and Paper Mill.

And finally, I wish to thank all other people in the Control Group at Pulp and Paper Centre for their kind help.

Chapter 1

Introduction

1.1 Introduction

During the development of control theory, much effort has been placed on controller design strategies based upon objective functions and on control algorithms based upon specific mathematical models, while relatively less attention has been paid to control loop performance assessment.

However, performance assessment is very important. If a controller is not running properly, its design objectives will not be met, and the control algorithm can hardly be successful. Also as a result, excessive variability occurs, cost increases, quality degrades. Unfortunately, it is very common that industrial controllers perform poorly due to causes such as improper implementation, lack of proper maintenance, valve wear, and process change.

In modern manufacturing plants, there are usually hundreds or even thousands of automatic control loops but fairly limited human resources, so it is virtually impossible to monitor all control loops without using a formalized assessment tool. Consequently, control malfunctions are often not detected until either a serious failure occurs, or until the resulting variability becomes too high to be ignored by operators. In either case, loss has already existed for a long time before the cause is removed.

On the contrary, if control loop performance monitoring can be automated, then loop malfunctions will be detected and located as soon as they occur. Thus process engineers and instrument staff will be able to respond quickly and maintain better quality control.

Resulting from such concerns, research in the field of performance monitoring has dramatically increased since 1989, when Harris published his ground-breaking paper [11]. The goal of control loop performance monitoring is to use data analysis methods and routine operation data only, in order to assess how well a process is behaving during closed loop operation. Because such methods are not intrusive (no extra disturbance needs to be added into the process) and yet can tell if the process is operating as expected, they are becoming increasingly popular in the process industries.

1.2 Background and Literature Review

1.2.1 Basic Concepts

Control loop performance monitoring deals with the following problems:

(1) Without introducing extraneous disturbing test signals into the system, based only on the routine process data, how to determine whether or not a control loop is behaving well?

(2) If the control loop is not operating well, what is the problem? Is it because the controller is not tuned or chosen appropriately? Or is it because of the process limit itself, while the controller is already doing its best?

Using minimum variance control as a benchmark, one can construct a scalar performance monitoring index to indicate how well a control loop is currently behaving. If the performance index indicates that current control performance is far from the theoretical "best achievable" one (as under the minimum variance control), then there is potential for better control performance by re-tuning the controller or changing the control strategy. If the performance index indicates that under the current control, the system has already reached the theoretical "best achievable" control, while the process output is still not satisfactory as measured by the mean square error, then the only way to achieve better control lies in modifying the process itself such as reducing the process time delay, changing the control variable, or introducing feedforward control.

1.2.2 Process Description and Analysis

Many commonly encountered process industry loops can be described by the following discrete dynamic model:

$$Y_t - \bar{Y}_t = A(z)z^d/B(z)u_t + D_t \tag{1.1}$$

$$d = 1 + integer(T_d/T) \tag{1.2}$$

$$D_t = C(z)/[D(z)(1-z)^d]a_t \tag{1.3}$$

where Y_t is the measured process output,

$\bar{Y}_t$ is the mean of Y_t,

u_t is the deviation of the manipulated variable from a reference value required to keep the process at its mean value,

D_t is the combined effect of all unmeasured disturbances acting on the process output,

$A(z)$, $B(z)$, $C(z)$ and $D(z)$ are polynomials,

z is the backward shift operator $(z^j Y_t = Y_{t-j})$,

d is the number of whole periods of delay in the process,

T_d is the process delay arising from true process deadtime or analysis delay,

T is the control interval,

$\{a_t\}$ is a sequence of independently and identically distributed random variables, with zero mean and standard deviation σ_a.

If the controller is linear and time-invariant, i.e.:

$$u_t = -G_c(z)(Y_t - Y_{sp}) \tag{1.4}$$

where Y_{sp} is a fixed setpoint, and

$Gc(z)$ is the control law,

then the closed control loop system can be easily shown as:

$$Y_t - \bar{Y}_t = F(z)a_t \tag{1.5}$$

where $\bar{Y}_t$ is the mean of Y_t under the feedback control,

$F(z)$ is a monic polynomial; in general case, it has an infinite degree.

Ideally, $\bar{Y}_t = Y_{SP}$, however this may not always be true.

The monic polynomial $F(z)$ can be broken into two parts:

$$F(z) = F_1(z) + F_2(z)z^d \tag{1.6}$$

where $F_1(z)$ is a monic polynomial of order $d-1$:

$$F_1(z) = 1 + f_1 z + \ldots + f_{d-1}z^{d-1} \tag{1.7}$$

So, the closed-loop system can be further interpreted as:

$$Y_t - \bar{Y}_t = F_1(z)a_t + F_2(z)a_{t-d} \tag{1.8}$$

The first and second terms in the right-hand side of (1.8) can be interpreted as the d-step ahead forecast error e_t and the d-step ahead forecast h_t respectively. So, we have:

$$Y_t - \bar{Y}_t = e_t + h_t \tag{1.9}$$

Define the deviation of the measured process output from the setpoint as:

$$y_t = Y_t - Y_{SP} \tag{1.10}$$

If e_t and h_t are independent, then the variance of y_t is given by:

$$var\{y_t\} = var\{e_t\} + var\{h_t\} \tag{1.11}$$

Under minimum variance control, h_t reduces to zero; because e_t depends only on the delay of the process and the disturbance characteristics, it cannot be eliminated by any control methods. So, under the minimum variance control, the variance of y_t becomes:

$$var\{y_t\} = var\{e_t\} \tag{1.12}$$

This is the minimum value which $var\{y_t\}$ can ever reach.

1.2.3 Direct Performance Monitoring

From the above discussion, we know that under minimum variance control, the process output is the error in forecasting the d-step ahead disturbance. Such error is a moving average (MA) time series model of order d:

$$e_t = (1 + f_1 z + \ldots + f_{d-1} z^{d-1}) a_t \tag{1.13}$$

The variance of e_t is given by:

$$var\{e_t\} = (1 + f_1^2 + \ldots + f_{d-1}^2)\sigma_a^2 \tag{1.14}$$

As is well known, a moving average process of order d has the property that its autocorrelation function is 0 beyond lag d. As a result, this property supplies a direct performance monitoring method: by checking the autocorrelation function of the output error, one can easily tell whether or not a control strategy is giving minimum variance control, the theoretical "best achievable" control.

In practice, the sample autocorrelation function is calculated and compared to the approximate upper and lower 95% confidence intervals. If after the lag d, all autocorrelation values lie within such 95% confidence intervals, the current control loop can be regarded as having achieved minimum variance control, and further reductions in the variance of the process output cannot be achieved by changing the control strategy. Instead, it can only be achieved by modifying process itself or reducing the disturbance.

As there exists a direct link between autocorrelation function and power spectrum of the same output error signal, the power spectrum can also be used in a similar way as a direct means of performance monitoring [23].

1.2.4 Performance Monitoring by Performance Index

By checking the autocorrelation function (or corresponding power spectrum) it can be determined whether the current control loop is under minimum variance control, however in practice it is usually difficult for operating staff and control engineers to determine the extent to which the controller is "good" or "bad". It would be convenient and intuitive if there were a single criterion directly related to the controller behavior. This need is met by the construction of a performance index.

From the analysis in section 1.2.2, we know that under the minimum variance control, the process deviation is minimized to its theoretical optimal value. So, if we compare the current control effect with that under the minimum variance control, the resulting ratio will quantify how well the closed loop behaves.

Thus a primitive performance index $PI(d)$ may be defined as:

$$PI(d) = \sigma_y^2/\sigma_{mv}^2 \tag{1.15}$$

where σ_y^2 is the variance of the control deviation under the current control:

$$\sigma_y^2 = var\{e_t\} + var\{h_t\} \tag{1.16}$$

σ_{mv}^2 is the variance of the control deviation under the minimum variance control:

$$\sigma_{mv}^2 = var\{e_t\} \tag{1.17}$$

$PI(d)$ is in $[1, \infty)$.

For computation convenience, we may wish the performance index be bounded within $[0,1]$, so a normalized performance index $NPI(d)$ is calculated as:

$$NPI(d) = 1 - \sigma_{mv}^2/\sigma_y^2 \tag{1.18}$$

$NPI(d)$ is in $[0,1]$.

In practice, it is often useful to assess control loop performance by using a modified performance index $\eta(d)$, which is given as:

$$\eta(d) = 1 - \sigma_{mv}^2/mse(y_t) \tag{1.19}$$

where

$$mse(y_t) = \sigma_y^2 + \bar{d}_y^2 \tag{1.20}$$

where $\bar{d}_y^2$ is the mean deviation from setpoint.

$\eta(d)$ is also in $[0,1]$.

Note:

(1) All the performance indices are written as a function of the process time delay d, emphasizing that they are dependent on d;

(2) Although minimum variance control is a good benchmark to assess the performance of a control loop, it does not necessarily mean minimum variance control should be the best choice for that loop. In certain situations, minimum variance control may cause large excessive control actions which are neither desirable nor tolerable. For processes exhibiting inverse response behavior (non-minimum phase), implementing minimum variance control will result in an unstable controller, which is obviously unacceptable.

1.2.5 Advanced Performance Monitoring

Performance Monitoring in Feedforward/Feedback Control

Desborough et al. [7] constructed a variance table and a generalized performance index to assess the performance of the overall feedforward/feedback control loop.

The basic idea is:

(1) Breaking down disturbances into measurable and unmeasurable components, assuming they are not cross-correlated, to calculate their contributions to the total output variance respectively, and list them in a variance table.

(2) Analyse the table, if the contribution of the measured feedforward variable to the total variance is acceptable, then there is no need to modify the existing feedforward controller or implement a new feedforward controller; if it is unacceptable, then introduce feedforward control and re-tune the controller.

(3) Similar to the SISO performance index, define a generalized performance index to more precisely compare the overall feedforward/feedback control with the minimum variance control benchmark.

The variance table and the generalized performance index can be obtained solely from routine operation data, which makes it attractive for industrial applications. One major concern here is how to separate precisely the measurable disturbances from the unmeasurable ones and how to describe correctly their characteristics, because the variance table and performance index are based on them.

Performance Monitoring in MIMO Control

Huang et al. [13] expanded performance monitoring to the multi-input multi-output (MIMO) case. Their starting point is the single-input single-output (SISO) system. Since univariate minimum variance control is used as a benchmark to assess the closed loop performance, in the MIMO system, multivariate minimum variance control should be able to serve the same purpose.

Their works includes:

(1) Resorting to the interactor matrix concept, prove that the feedback invariance property of minimum variance control in MIMO process can be solved by using the unitary or weighted unitary interactor matrix.

(2) Assert that the interactor matrix (or equally, the transfer function matrix) needs to be known for assessing the MIMO system, just as in SISO system the time delay is required as a priori knowledge in order to assess closed loop performance.

(3) Define a MIMO performance index to assess the current controller performance against that of the multivariate minimum variance controller.

(4) Develop a FCOR (Filtering and CORrelation) algorithm to calculate the above defined MIMO performance index for a class of multivariable processes, which have a diagonal interactor matrix.

Because many industrial processes are inherently multivariable, the above work in MIMO system is certainly desirable. But currently only processes with a diagonal interactor matrix can be handled conveniently, further research work is still needed for the general MIMO case.

1.2.6 Other Approaches to Performance Monitoring

In addition to control loop performance monitoring based on minimum variance control as a benchmark, there are other approaches for assessing the control loop behavior [15, 14, 1].

The PID Approach

PID controllers are the most commonly used controllers in the industry and will very likely remain so in the future. Further, they make up the backbone of many sophisticated control systems. Because of these reasons, the choice of PID controller parameters and assessment of their performance have long been key issues.

The earliest work in this field may be the well-known tuning rules developed by Ziegler and Nichols, which, based on simple tests, gives the tuning rules in terms of simple formulae. Since then, a vast related literature has been published. Recently, Åström [1] proposed a methodology for assessing the performance of such loops.

According to Åström's definition, there are six levels of process knowledge:

- Level 0: Qualitative characterization,
- Level 1: Level 0 plus process time constant and time delay,
- Level 2: Level 1 plus process gain,
- Level 3: Level 2 plus more points on Nyquist curve, possibly with uncertainty regions,
- Level 4: Complete mathematical model with uncertainty regions,
- Level 4A: Knowledge of dynamics that is strictly positive real (SPR) or of first or second order with known model.

Then, depending on the knowledge level of the process dynamics, either crude or accurate assessment of PID controller performance can be made. For crude assessment, only level 2 information about the process is needed. Based on ω_{90}, ω_{180} bandwidth (where the loop phase lag is $-90°$, $-180°$ respectively) or maximum loop gain, one can determine if the PID controller is appropriate for handling the process, and decide what rules should be used to set its parameters. For accurate assessment, level 3 or level 4 information about the process is needed. Based on the known transfer function of the process, the proper PID parameters can be conveniently determined through dominant pole design. Further, the best achievable performance such as an upper bound of achievable bandwidth can be assessed.

Compared with the performance index approach described in section 1.2.5, the PID approach gives frequency domain information in terms of bandwidth, while the performance index approach gives time domain information in terms of standard deviation.

The Linear Quadratic (LQ) Approach

Kammer, et al. [15] presented a model-free method for determining LQ optimality of the current controller as well as the closed loop pole positions that would be obtained by the LQ optimal controller. The approach consists of adding an exogenous signal to the control action, and then analysing the spectrum of the process input and output signals. From this testing, sub-optimal results are obtained, indicating the location of the optimal closed-loop poles. Then, comparing with the current control, it can be decided whether or not the current control law should be replaced by an LQ control algorithm.

The results can be applied through both state and output feedback to all linear and time invariant system without any constraint on the process characteristics, and do not need a parametric model for the process. Unfortunately, one drawback is that because the objective is so general, the needed information is so large that an identification with extraneous exciting signal is a necessity. In many industrial cases this is a serious constraint.

The Expert System Approach

Jofriet et al. [14] developed an expert system (named QCLiP for Queen's/QUNO Control Loop Performance analysis expert system) to analyse control loop performance. The expert system is operated at a supervisory level in a pulp and paper mill, implemented by an advanced intelligent real-time system development and deployment tool Gensym's

G2. The system collects data from a distributed control system, and evaluates the loop performance based on certain rules and exception report cases. (The rules are mainly based on the previously outlined performance index.)

QCLiP has an interpretation hierarchy structure and an analysis toolbox. It can automatically calculate the sample autocorrelation function, estimate the minimum variance spectrum and test the periodic components. It is implemented with a graphical user interface. All these features give QCLiP the potential to become a practical system for continuous on-line loop performance monitoring, although further work is still needed in order to truly reach such an objective.

1.2.7 Related Issues

A control loop is made up of sensors, controllers and actuators. The whole loop performance depends on the performance of sensors and actuators, as well as that of the controller. Unfortunately, for a long time, this issue has been ignored. Often it is incorrectly assumed that the sensors will reflect the process quickly and correctly, and that the actuators will respond instantly and accurately to the signal from the controller. However, this is not always the case. In 1993, a survey indicated that 30% of all control loops in Canadian paper mills were not operating properly because of actuator problems [3].

During recent years, some papers concerning this issue have been published [4, 27, 6]. In order to improve the control valve design, Bialkowski [4] presented dynamic specifications for control valves. They include specifications on valve tracking nonlinearities (backlash/stiction), sizing and flow characteristic nonlinearities, dynamic performance specification summary, and their impacts on future valve design.

Taylor [27] described the control valve characteristics that cause process variability in paper mills, including the combination of backlash and stiction in the assembled valve, the response speed as a function of valve size, and the percent overshoot. Taylor tested control valve performance, and expanded these test results to a real mill. From his experiment, he suggested procedures to specify and verify control valve performance so as to assure that a control system with such valves will indeed minimize process variability and improve quality.

Clarke [6] discussed validation standards for both sensors and actuators. In sensor validation, borrowing the idea from metrology, he introduced "uncertainty" as a part of the validity index for the sensor, and classified sensor measurements into four categories:

- clear (the data is fine),
- dazzled (possibly the data is transiently abnormal),
- blurred (the data is abnormal, but believed to have some correspondence to the real measurements),
- blind (the data is completely untrustworthy).

Clarke then discussed when a certain category of measurements can be used for control. In actuator validation, attention must be paid to actuation signal limits and saturation, non-linearity, as well as the highest achievable bandwidth. Therefore, he advocated the use of internal feedback or of an inverse nonlinearity for the purpose of making the nominal actuator approach an ideal one. After examining both sensors and actuators, he stressed the importance of combining the sensor, and the actuator with the controller to constitute a whole valid control loop.

Åström [2] also pointed out limitations on control system performance:

- For a minimum phase system, measurement noise is injected into the system and they can result in large control signals that saturate the actuators. Measurement noise and actuator saturation are thus factors that limit the performance of a minimum phase system.

- For an non-minimum phase system, the limitations are due to system dynamics. Specifically, the upper bounds of the bandwidth of the closed loop system are imposed by time delays and zeros in the right-half plane, while the lower bounds are imposed by poles in the right-half plane.

1.2.8 Industrial Applications

Currently, some industrial applications have been reported in the field of performance monitoring.

Owen, et al. [21] implemented a prototype on-line automatic monitoring system at a paper mill. The system obtains data from the DCS, and analyzes a large number of control loops. The system then locates malfunctioning loops, and diagnoses possible causes.

The approach uses only a small amount of prior information about the loop, such as the time delay, and requires neither identification nor an exact process model. By employing a modified form of the performance index, the system is capable of overcoming difficulties arising from non-stationary disturbances and sensor-caused or valve-caused nonlinearities. The system has been applied in a Paprican's member company mill.

Stanfelj, et al. [25] presented a hierarchical system for monitoring and diagnosing the performance of single-loop control systems. Based primarily on routine operating plant data, the system can

(1) in the first level, identify significant process output deviation,

(2) in the second level, determine the best achievable performance within current control strategy,

(3) in the third level, diagnose the possible reason of poor control performance,

(4) in the fourth level, give suggestions on how to improve current loop performance.

The system has been tested in both simulation and industrial (Shell Canada, Scotford Refinery) environments.

Ogawa [20] developed a data analysis system for control loop performance monitoring. The system can calculate the normalized performance index, time series functions (autocorrelation, partial autocorrelation, etc.) and power spectrum of a certain control loop. The algorithm uses batch-processing. The most distinct feature of this system is it has a very friendly graphic user interface, which is essential for a successful industrial application. The system was developed and implemented in MacMillan Bloedel Paper Ltd.

There are also some commercial performance monitoring softwares such as Matrikon's ProcessDoc, which analyses the plant operating data trend, and audits the control loop performance.

1.3 Thesis Motivation

Because of its calculation simplicity, non-intrusion, and most of all, big potential benefits, performance monitoring has attracted great attention from industry world once after it was presented.

However, the current situation is that although many useful academic results have been developed, their industrial application is lagging behind. There is a lack of good application software to implement these techniques over an extended period in industrial environment. This thesis is devoted to developing software to facilitate the industrial application of performance monitoring.

1.4 Thesis Contribution

In this thesis, an integrated application software system was developed in an industrial environment. The system presents three groups of functions:

- process data analysis in both time domain and frequency domain;
- performance index calculation;
- loop oscillation and valve friction detection.

The system also integrates all the user-system interfaces into PC networks, thus shielding the critical distributed control system (DCS) from user errors while providing a very friendly graphic user interface to facilitate the use of the system.

1.5 Thesis Outline

In Chapter 2, various methods of process data analysis are studied, including setpoint/error analysis, autocorrelation and partial autocorrelation function calculation, power spectrum and cumulative power spectrum estimation. The data pre-processing method is also illustrated. Chapter 3 focuses on computing the performance index. Chapter 4 contains procedures for detecting loop oscillation and valve friction. Chapter 5 describes the performance monitoring application software system in detail, and also gives application examples. In Chapter 6, the thesis results are summarized, conclusions are drawn, and further work is proposed.

Chapter 2

Process Data Analysis

2.1 Introduction

In modern industry, vast amount of routine operation data are collected and stored daily in a computer system. If carefully analysed, these data can present useful information about the performance of the process. In this chapter, we will discuss how to analyse the process data in both time domain and frequency domain.

2.2 Process Data Analysis in the Time Domain

From a mathematical point of view, the industrial sampled data constitutes a time series. For a thorough analysis, interested readers are referred to look in [5] for details. Here, however only the basic data analysis techniques which are commonly used in the process industry are considered.

2.2.1 Setpoint and Error

Setpoint and Error (SPE) plot is a simple data analysis form widely used in the process industry. It plots a loop setpoint and error signals against the time. Very often, the lower and upper limits of the loop are also shown in the plot. Much insight comes from observing sampled process variables. Offsets, cyclic components, and outliers can be intuitively observed. Figure 2.1 – 2.4 show four typical types of process industry data samples. They are respectively of onecycle, twocycle, MA and AR model. In these plots, limits are set according to the maximum allowed process output.

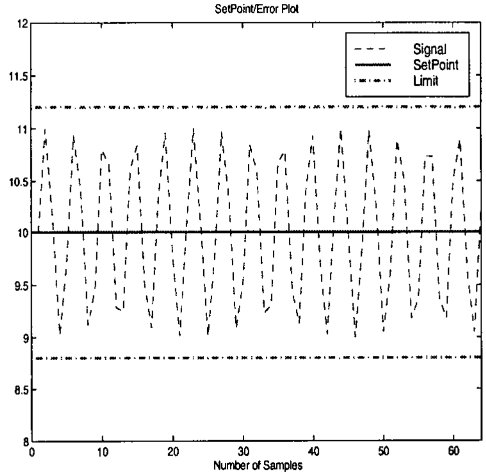


Figure 2.1: SPE Plot (Onecycle Data)

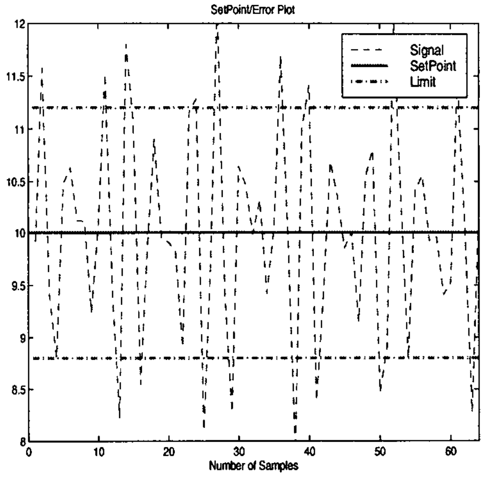


Figure 2.2: SPE Plot (Twocycle Data)

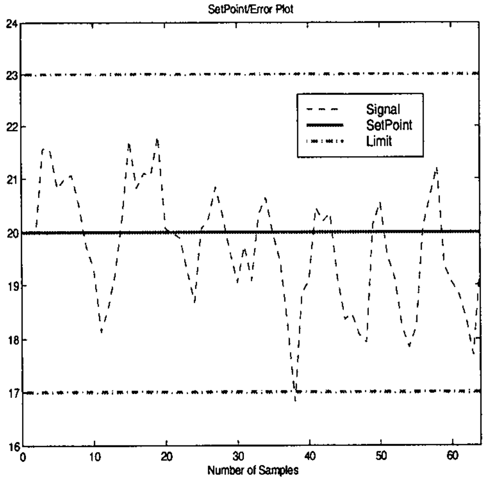


Figure 2.3: SPE Plot (MA Data)

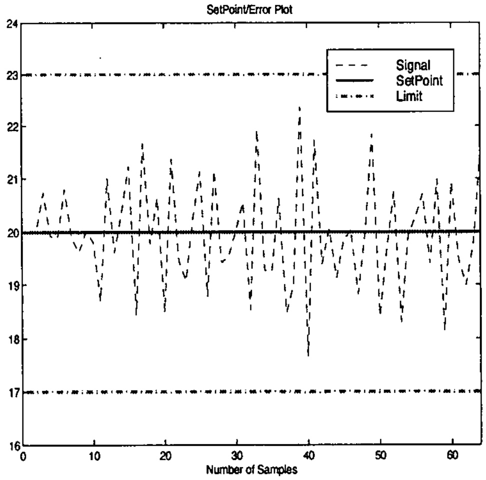


Figure 2.4: SPE Plot (AR Data)

From these plots, we can clearly find the setpoint values, and whether or not the variables exceed the upper and lower limits. Also, we can directly recognize a simple pattern. For example, Figure 2.1 obviously shows a periodical variable. However, for more complex process data as shown in Figure 2.2 – 2.4, one cannot so easily detect patterns. Consequently, we need other tools to analyse these data.

2.2.2 Autocorrelation Function

The autocovariance function $\gamma(d)$ expresses correlation between two observations that are d time periods apart:

$$\gamma(d) = E[(y(t) - \mu_y)(y(t+d) - \mu_y)] \tag{2.21}$$

where $E[\cdot]$ denotes the expectation operation,

$y(t)$ is the observation,

μ_y is the mean of $y(t)$.

$\gamma(0)$ is thus the variance of $y(t)$. The values of the autocovariance function are affected by the variance. If divided by the variance, the autocorrelation function value will be normalized to $[-1, 1]$, which is called the autocorrelation function (ACF):

$$ACF(d) = \gamma(d)/\gamma(0) \tag{2.22}$$

The ACF plot is considered the best indicator of the randomness or periodicity in a data series. If the data is completely random (i.e. white noise), then $ACF(d)$ will be 1 when d is zero, and 0 for all other d values. If the data contains some periodic components, its $ACF(d)$ is then also periodic with period d.

Furthermore, if the data is an n-th order moving average (MA) process, then its $ACF(d)$ vanishes beyond lag n. This property, as discussed in Chapter 1, is the basis for determining whether or not a loop is under minimum variance control just by checking whether or not the $ACF(d)$ value of this loop is zero beyond the loop time delay d.

In practice, the autocorrelation function value of a certain loop is unknown. Instead, we can only calculate an estimate from observations by the following formula:

$$ACF(d) = \frac{1}{N}\sum_{t=1}^{N-d}(y(t) - m_y)(y(t+d) - m_y)/\frac{1}{N}\sum_{t=1}^{N}(y(t) - m_y)^2 \tag{2.23}$$

where d is the lag or time delay,

N is the number of samples,

$y(t)$ is the sample value at time t,

$y(t+d)$ is the sample value at time $t+d$,

m_y is the mean value of all $y(t)$.

In (2.23), if d is not zero, then summations of fewer than N terms are divided by N, which biases the estimate. If $N \gg d$, this is not significant. When d is non-zero, $ACF(d)$ fluctuates asymptotically as Gaussian random variables with mean zero and variance $1/N$. Approximately 95% of the $ACF(d)$ estimates are in $[-1.96/\sqrt{N}, 1.96/\sqrt{N}]$ [5].

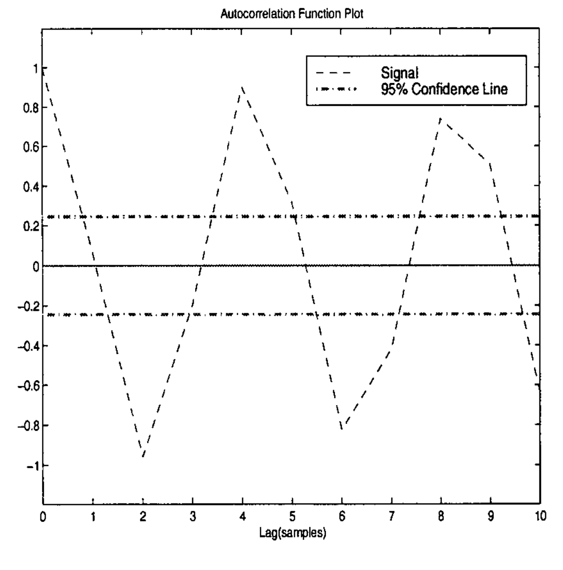


Figure 2.5: ACF Plot (Onecycle Data)

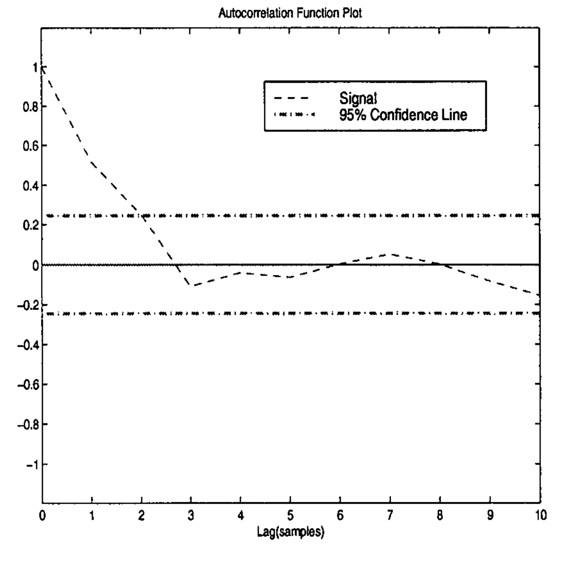


Figure 2.6: ACF Plot (MA Data)

Now, for example we can draw ACF plot for data shown in Figure 2.1 and 2.3 in

Figure 2.5 and 2.6. From Figure 2.5, we can see there is a peak value every 4 sample periods, indicating this is a cyclic variable. This result is consistent with that in Figure 2.1. From Figure 2.6, we find that after 2 sample lags, the $ACF(d)$ values are all within 95% confidence limits, indicating this is a second order MA process.

2.2.3 Partial Autocorrelation Function

The partial autocorrelation function (PACF) is another useful tool. It is the correlation of the residuals of linear regressions. We denote the best linear predictor of $y(t)$ from d adjacent past, $y(t-d), \ldots, y(t-1)$, as $L_p(y(t), d)$, and from d adjacent future, $y(t+1), \ldots, y(t+d)$, as $L_f(y(t), d)$. Time moves backward in the latter case, i.e. we "predict" $y(t)$ from the future observations. The residual (i.e. prediction error) of $y(t)$ from d adjacent observations is:

$$e_f(t,d) = y(t) - L_f(y(t), d) \tag{2.24}$$

$$e_p(t,d) = y(t) - L_p(y(t), d) \tag{2.25}$$

Then the partial autocorrelation function $\phi(d)$ is defined by:

$$\phi(0) = 1 \tag{2.26}$$

$$\phi(1) = Corr(y(1), y(2)) = ACF(1) \tag{2.27}$$

$$\phi(j) = Corr(e_f(1,j), e_p(1+j,j)), \quad 1 < j < d \tag{2.28}$$

$$\phi(d) = Corr(e_f(1,d), e_p(1+d,d)) \tag{2.29}$$

where $Corr(x,y)$ denotes the correlation between two random variables x and y:

$$Corr(x,y) = \frac{E[(x-\mu_x)(y-\mu_y)]}{\sqrt{E[(x-\mu_x)^2]E[(y-\mu_y)^2]}} \tag{2.30}$$

Like the ACF, in the real world we can only estimate the PACF from samples. PACF estimation is very complex and there are many different algorithms to calculate it. Here, we present a simple and convenient recursive algorithm, Durbin-Levinson Algorithm [5].

- Initial conditions:

$$\hat{\phi}_{11} = \hat{\gamma}(1)/\hat{\gamma}(0) \tag{2.31}$$

where $\hat{\gamma}(\cdot)$ is the estimate of the autocovariance function defined in (2.21), and

$$\hat{\gamma}(0) \neq 0$$

$$\hat{\nu}_1 = \hat{\gamma}(0)[1 - \hat{\phi}_{11}^2] \tag{2.32}$$

- Recursive operations:

$$\hat{\phi}_{mm} = [\hat{\gamma}(m) - \sum_{j=1}^{m-1} \hat{\phi}_{m-1,j}\hat{\gamma}(m-j)]/\hat{\nu}_{m-1} \tag{2.33}$$

$$\begin{pmatrix} \hat{\phi}_{m,1} \\ \vdots \\ \hat{\phi}_{m,m-1} \end{pmatrix} = \begin{pmatrix} \hat{\phi}_{m-1,1} \\ \vdots \\ \hat{\phi}_{m-1,m-1} \end{pmatrix} - \hat{\phi}_{mm} \begin{pmatrix} \hat{\phi}_{m-1,m-1} \\ \vdots \\ \hat{\phi}_{m-1,1} \end{pmatrix} \tag{2.34}$$

$$\hat{\nu}_m = \hat{\nu}_{m-1}(1 - \hat{\phi}_{mm}^2) \tag{2.35}$$

- Results:

  $\hat{\phi}_{1,1}, \ldots, \hat{\phi}_{m,m}$ are the estimated partial autocorrelation functions at lags $1, 2, \ldots, m$. where $m = N - 1$,

  N is the number of samples.

While $ACF(d)$ is important for MA process, $PACF(d)$ can provide insight information on judging whether or not the process is of AR type, because for an n-th order AR process, its $PACF(d)$ vanishes after the lag n [5]. Also, the sample $PACF(d)$ of a white noise has the same asymptotic distribution as that of the sample $ACF(d)$. So, the same 95% confidence interval works for both ACF and PACF plots.

Now, we draw PACF plot for data shown in Figure 2.3 and 2.4 in Figure 2.7 and 2.8. From Figure 2.7, we can see $PACF(d)$ of an MA process decays slowly and remains outside the 95% confidence limits after 4 sample lags. But in Figure 2.8, we can see $PACF(d)$ of an AR process decays quickly and stays inside the 95% confidence limits after 2 samples lag, indicating this is a second-order AR process.

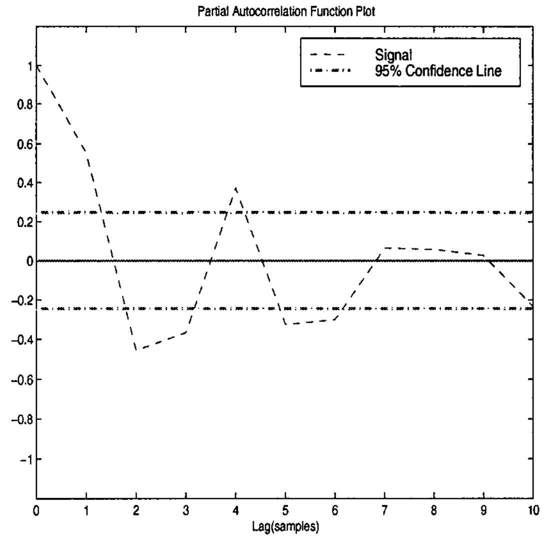


Figure 2.7: PACF Plot (MA Data)

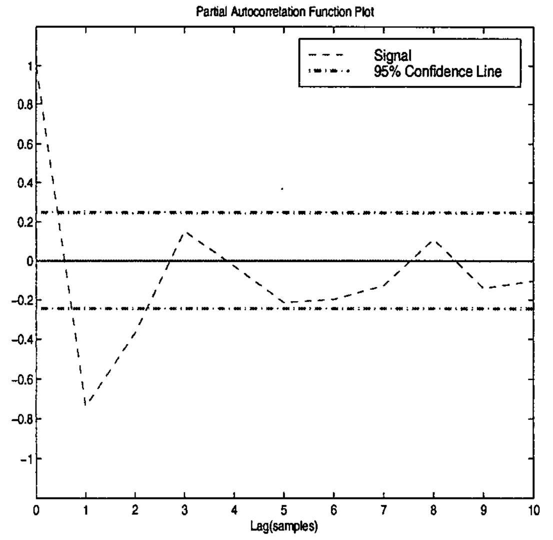


Figure 2.8: PACF Plot (AR Data)

2.3 Process Data Analysis in the Frequency Domain

Very often, data sampled from industrial processes are mixed with load disturbances and stochastic components. In the time domain, it may be hard to distinguish between these components. But in the frequency domain, they can be characterized as a second-order weakly stationary stochastic series [12] and can easily be broken down into discrete frequencies. In this section, we study two commonly used spectral analysis tools: power spectrum estimation (PSE) and cumulative power spectrum estimation (CPSE). These methods can reveal information such as hidden periodicities or close spectral peaks.

2.3.1 Power Spectrum Estimation

Mathematically, the power spectrum is defined as the Fourier transform of the autocorrelation function, and they therefore constitute a Fourier transform pair. In process industries, signals often contain many frequencies, ranging from very slow cycles to very fast cycles. The power spectrum can display these components and indicate what fraction of each variance exists at a particular frequency or period.

In practice, only estimates of the autocorrelation function are available so that after Fourier transforming it, the power spectrum estimation (PSE) is obtained. In a PSE plot, variance is shown versus frequency and the sum of the individual variances at each frequency is equal to the total variance in the data series.

The most popular method of estimating the power spectrum is to use the fast Fourier transform (FFT). If signals are time-varying, then the short-time Fourier transform (SFFT) is preferred, which first uses a window to slide over the signal in the time domain, and then computes the Fourier transform for every portion within each window [19].

Figure 2.9 and 2.10 are the PSE plots for the data shown in Figure 2.1 and 2.2. In Figure 2.9, there is one peak at frequency of 2.5Hz, indicating this is a single cyclic variable. This result is consistent with that of Figure 2.1. While we cannot tell the characteristics of data from Figure 2.2 (time domain), Figure 2.10 (frequency domain) clearly shows there are two peaks at different frequencies, indicating it is a mixture of two cyclic variables. In this case, frequency domain analysis can give us a clearer picture.

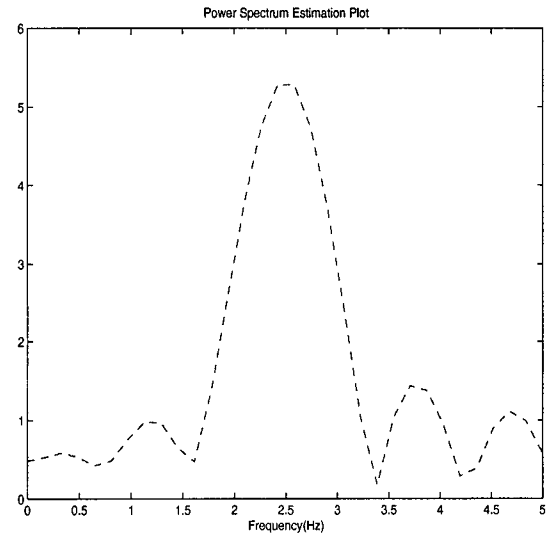


Figure 2.9: PSE Plot (Onecycle Data)

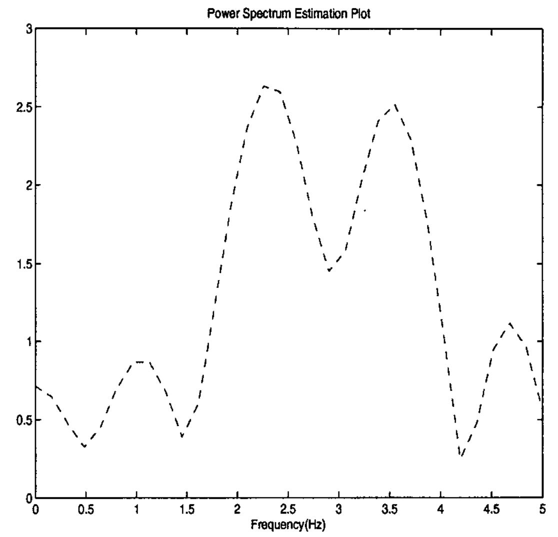


Figure 2.10: PSE Plot (Twocycle Data)

2.3.2 Cumulative Power Spectrum Estimation

The cumulative power spectrum estimation (CPSE) is obtained by summing up all values of the power spectrum estimation from zero frequency to the current frequency. The CPSE plot shows the percentage of total variance versus the frequency. It is very useful in specifying the percent contribution of the individual cycles in the process signal to the total variance.

Figure 2.11 and 2.12 are the CPSE plots for the data shown in Figure 2.1 and 2.2. For instance, from Figure 2.11, approximately 80% of the total variance is seen to be slower than 3 Hz.

2.4 Process Data Pre-Processing

In process industries, original data is continuous in the time domain. In order to implement computer control, the data needs to be first sampled into discrete form.

The sampling theorem says [22] that if the highest frequency contained in a continuous

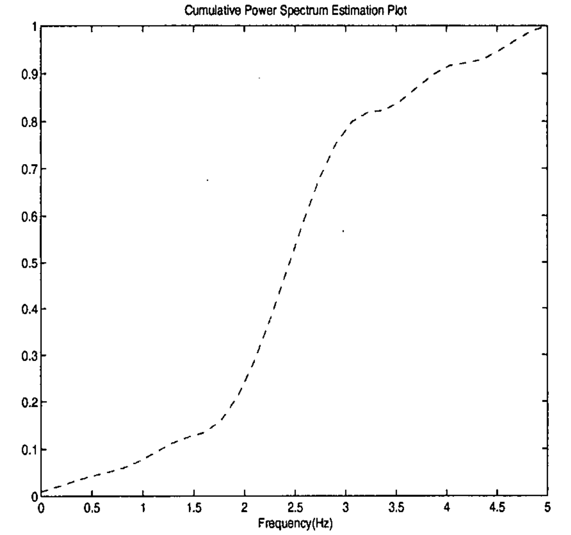


Figure 2.11: CPSE Plot (Onecycle Data)

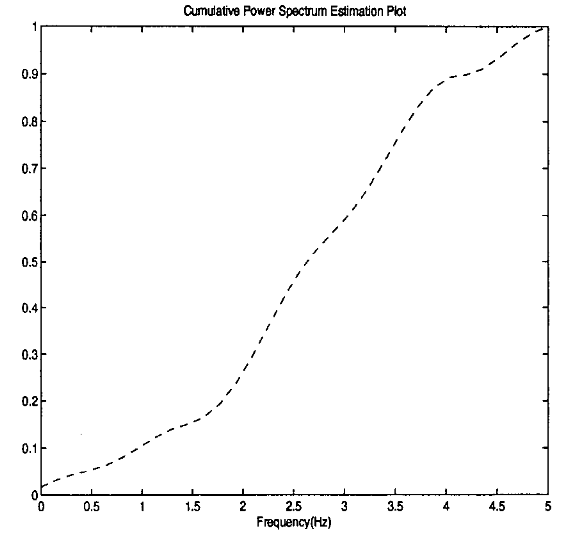


Figure 2.12: CPSE Plot (Twocycle Data)

signal $x_a(t)$ is $F_{max} = B$ and the signal is sampled at a rate $F_s > 2F_{max} \equiv 2B$, then $x_a(t)$ can be exactly recovered from its sample values using the interpolation function

$$g(t) = \frac{sin2\pi Bt}{2\pi Bt}$$

Thus $x_a(t)$ may be expressed as

$$x_a(t) = \sum_{n=-\infty}^{\infty} x_a(\frac{n}{F_s})g(t - \frac{n}{F_s})$$

where $x_a(n/F_s) = x_a(nT) \equiv x(n)$ are the samples of $x_a(t)$, and the sampling rate $F_N = 2B = 2F_{max}$ is called the Nyquist rate.

However, in process applications, the situation is often that a loop is sampled too fast. For example, the sampling period of a Foxboro DCS at the controller level is 1 seconds, i.e. 1 Hz, which is much faster than the Nyquist rate of some loops. For these loops, more data is sampled than is actually needed. In order to reduce the computation load, only one from every several samples will be picked out for further processing. This is called down-sampling.

When down-sampling is performed, the original sampling rate is actually reduced so that the sampling theorem may be violated and "aliasing" may occur. When aliasing occurs, high frequency components will be reflected down from their original position above the Nyquist rate in the frequency domain so that the original signal is distorted. The following figures illustrate such a phenomenon.

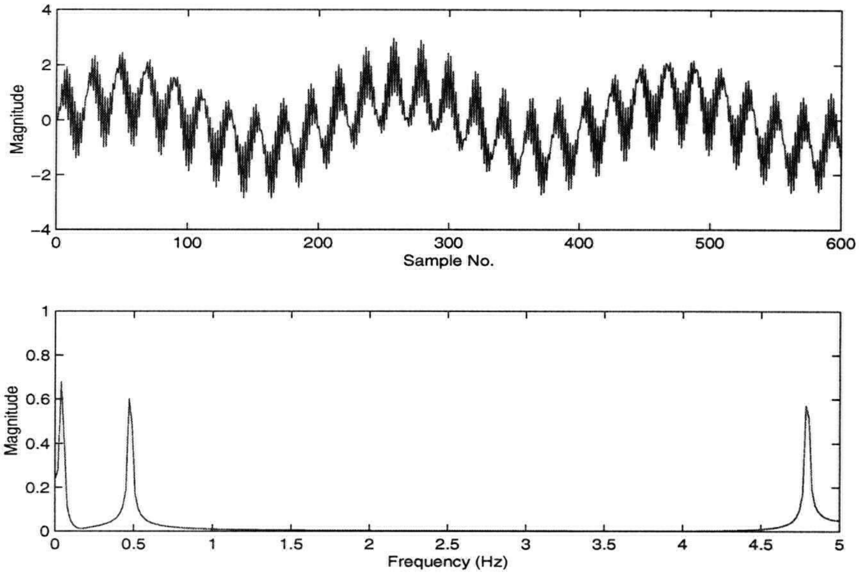


Figure 2.13: Originally Sampled Data

Figure 2.13 shows the original sampled data obtained with a sampling rate of 10 Hz. Assume we are only interested in the components with frequency lower than 0.6 Hz, then the Nyquist rate of the interested signal is $0.6 \times 2 = 1.2$Hz. So, the original sampling rate 10 Hz is too high. Now directly performing down-sampling on the original data with a ratio of 8, the results are shown in Figure 2.14 (note the different frequency ranges in these two figures). Comparing Figure 2.13 and 2.14, one can find that the unwanted high frequency noise component in Figure 2.13 (with frequency of 4.75 Hz) does not disappear in Figure 2.14. Instead, it moves to a different frequency (0.25 Hz). This is caused by

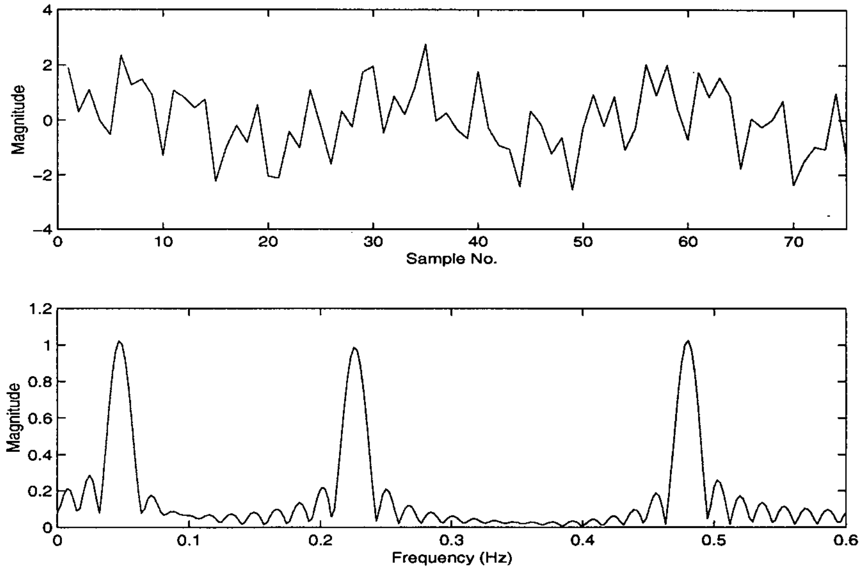


Figure 2.14: Directly Down-Sampled Data

aliasing because the new sampling rate becomes $10 \div 8 = 1.25$ Hz. Although the new sampling rate is still higher than the Nyquist rate of the signal (1.2 Hz), it is less than the Nyquist rate of the noise signal ($4.75 \times 2 = 9.5$Hz). As a result, the original signal is altered and cannot be exactly recovered.

The remedy is to use a lowpass filter to filter the high frequency noise before carrying out down-sampling. First let the original sampled data pass through a 4-th order Bessel lowpass filter with pass band of 1 Hz, and then down-sample the data with a ratio of 8. The results are shown in Figure 2.15. It can be seen that the signal within the frequency range of interest (0 – 0.6 Hz) remains unchanged because the component at 4.75 Hz has been filtered by the Bessel lowpass filter. Such data pre-processing is used in the performance monitoring application system discussed in chapter 5. Because different loops have different bandwidth, the stop bandwidth of the Bessel lowpass filter can be specified correspondently.

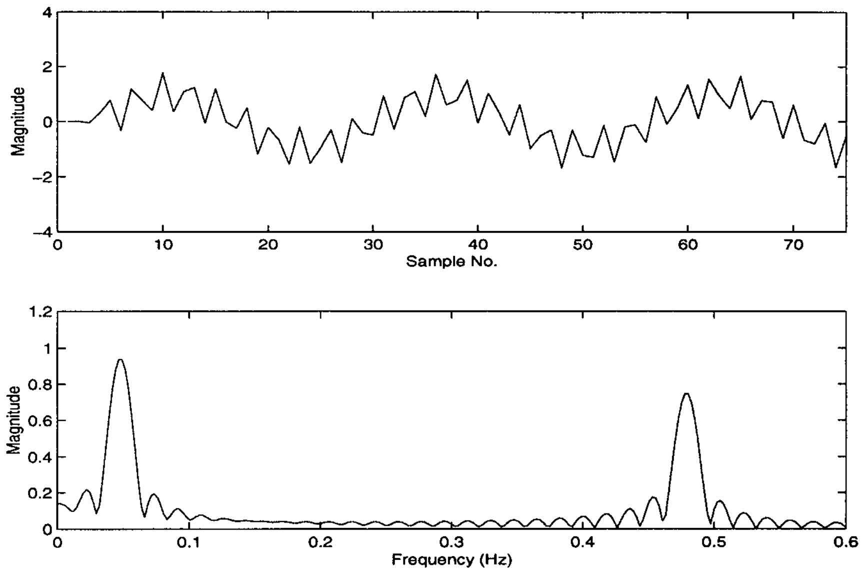


Figure 2.15: Filtered and Down-Sampled Data

2.5 Conclusion

In this chapter, several simple data analysis methods are discussed, including the setpoint and error (SPE) plot, the autocorrelation function (ACF) plot and the partial autocorrelation function (PACF) plot in the time domain, the power spectrum estimation (PSE) plot and the cumulative power spectrum estimation (CPSE) plot in the frequency domain. Generally speaking, the SPE plot is an intuitive and simple method to find out offsets, variances and simple cyclic components; the ACF plot is useful in looking for single periodic component and determining MA model orders; the PACF plot is good for determining AR model orders; and the PSE plot is particularly suitable of breaking down different frequency components; the CPSE plot is used to indicate the percentage distribution of total variance at different frequencies. Used in combination, all these can assist us to analyse effectively most data obtained from an industrial process. In addition, for some industrial loops, a DCS sampling rate is too fast so that down-sampling is needed. Before carrying out down-sampling, the data must first go through a lowpass filter to avoid aliasing.

In addition to these methods, statistical process control (SPC) tools are also often used in industries. Interested readers are referred to [17].

Chapter 3

Performance Index Calculation

3.1 Introduction

As discussed in section 1.2.4, the performance index is a scalar measure used to indicate how well a control loop is operating. A practical form, $\eta(d)$, is given by:

$$\eta(d) = 1 - \sigma^2_{mv}/(\sigma^2_y + \bar{d}^2_y) \tag{3.36}$$

where σ^2_{mv} is the loop variance deviation under the minimum variance control,
σ^2_y is the loop variance deviation under the current control,
$\bar{d}^2_y$ is the mean variance deviation from setpoint.

In (3.36), σ^2_y and $\bar{d}^2_y$ are easy to calculate from loop data, and the only problem is obtaining σ^2_{mv}. This is the major issue that will be discussed in this chapter.

3.2 Performance Index Calculation via an ARMA Model

As is well known, under minimum variance control, for a linear discrete time invariant SISO system, the loop output error can be fitted by a moving average (MA) time series model of order d^{-1}, where d is the process time delay:

$$e_t = (1 + f_1 z + \ldots + f_{d-1} z^{d-1}) a_t \tag{3.37}$$

So, the variance of e_t is given by:

$$var\{e_t\} = (1 + f_1^2 + \ldots + f_{d-1}^2)\sigma_a^2 \tag{3.38}$$

This is σ^2_{mv}, the loop deviation under minimum variance control:

$$\sigma^2_{mv} = (1 + f_1^2 + \ldots + f_{d-1}^2)\sigma_a^2 \tag{3.39}$$

So, now the problem of how to calculate σ^2_{mv} becomes that of finding MA model coefficients $f_1, \ldots, f_{d-1}$ and the noise deviation σ_a^2. Two methods are discussed below.

3.2.1 Durbin-Levison Algorithm

The Durbin-Levison Algorithm [5], which was used to calculate partial autocorrelation function in section 2.2.3, can also be used to obtain MA model coefficients $f_1, \ldots, f_{d-1}$ and noise deviation σ_a. Recall the algorithm:

- Initial conditions:

$$\hat{\phi}_{11} = \hat{\gamma}(1)/\hat{\gamma}(0) \tag{3.40}$$

where

$$\hat{\gamma}(0) \neq 0$$

$$\hat{\nu}_1 = \hat{\gamma}(0)[1 - \hat{\phi}_{11}^2] \tag{3.41}$$

- Recursive operations:

$$\hat{\phi}_{mm} = [\hat{\gamma}(m) - \sum_{j=1}^{m-1} \hat{\phi}_{m-1,j}\hat{\gamma}(m-j)]/\hat{\nu}_{m-1} \tag{3.42}$$

$$\begin{pmatrix} \hat{\phi}_{m,1} \\ \vdots \\ \hat{\phi}_{m,m-1} \end{pmatrix} = \begin{pmatrix} \hat{\phi}_{m-1,1} \\ \vdots \\ \hat{\phi}_{m-1,m-1} \end{pmatrix} - \hat{\phi}_{mm} \begin{pmatrix} \hat{\phi}_{m-1,m-1} \\ \vdots \\ \hat{\phi}_{m-1,1} \end{pmatrix} \tag{3.43}$$

$$\hat{\nu}_m = \hat{\nu}_{m-1}(1 - \hat{\phi}_{mm}^2) \tag{3.44}$$

- Results:

  $\hat{\phi}_{1,1}, \ldots, \hat{\phi}_{m,m}$ are the estimated partial autocorrelation functions at lags $1, 2, \ldots, m$.
  where $m = N - 1$,

  N is the number of samples.

In addition to the above results, it can be also shown that $\hat{\phi}_{m,1}, \ldots, \hat{\phi}_{m,m}$ are the estimate of the coefficients of the following AR model:

$$\Phi(q^{-1})y_t = a_t \tag{3.45}$$

where $\Phi(q^{-1}) = 1 + \phi_{m,1}q^{-1} + \ldots + \phi_{m,m}q^{-m}$.

and, $\hat{\nu}_m^2$ is the estimate of the noise deviation σ_a^2.

In order to calculate the performance index, the MA model is needed rather than the AR model of the process. This can be done through polynomial comparison. Substituting the MA model:

$$y_t = \Psi(q^{-1})a_t \tag{3.46}$$

where $\Psi(q^{-1}) = 1 + \psi_1 q^{-1} + \ldots + \psi_n q^{-n}$.

to the AR model (3.45) gives:

$$\Phi(q^{-1})\Psi(q^{-1})a_t = a_t \tag{3.47}$$

Thus:

$$\Phi(q^{-1})\Psi(q^{-1}) = 1 \tag{3.48}$$

If written out:

$$(1 + \phi_{m,1}q^{-1} + \ldots + \phi_{m,m}q^{-m})(1 + \psi_1 q^{-1} + \ldots + \psi_n q^{-n}) = 1 \tag{3.49}$$

Comparing the same degree of q^{-1} of the both sides, we have:

$$\phi_{m,1} + \psi_1 = 0 \tag{3.50}$$

$$\phi_{m,2} + \psi_2 + \phi_{m,1}\psi_1 = 0 \tag{3.51}$$

$$\vdots$$

And obtain:

$$\psi_1 = -\phi_{m,1} \tag{3.52}$$

$$\psi_2 = -\phi_{m,1}\psi_1 - \phi_{m,2} \tag{3.53}$$

$$\vdots$$

Estimate σ_{mv}^2 as:

$$\sigma_{mv}^2 = (1 + \psi_1^2 + \ldots + \psi_{d-1}^2)\hat{\nu}_m^2 \tag{3.54}$$

and performance index $\eta(d)$ as:

$$\eta(d) = 1 - \sigma_{mv}^2/(\sigma_y^2 + \bar{d}_y^2) \tag{3.55}$$

Note:

(1) In order to calculate the performance index $\eta(d)$, the process time delay d must be known in advance;

(2) When estimating the AR model coefficients in Durbin-Levison Algorithm, the order of AR model n must satisfy: $n \gg d$ to ensure the model accuracy.

3.2.2 Mayne-Firoozan Algorithm

The Mayne-Firoozan Algorithm [18] is an extension of the Durbin-Levison Algorithm. It considers the problem of the parameter estimation for an ARMA (mixed autoregressive-moving average) model:

$$A(q^{-1})y_t = C(q^{-1})a_t \tag{3.56}$$

where

$$A(q^{-1}) = 1 + a_1q^{-1} + \dots a_nq^{-n} \tag{3.57}$$

$$C(q^{-1}) = 1 + c_1q^{-1} + \dots c_nq^{-n} \tag{3.58}$$

Its basic idea is:

- Step 1: using linear least square estimation method to obtain an asymptotically unbiased parameter estimate $\hat{A}(q^{-1})$,
- Step 2: using $\hat{A}(q^{-1})$ to calculate the residual sequence $\{\varepsilon_t\}$,
- Step 3: estimating $\hat{A}(q^{-1})$ again and $\hat{C}(q^{-1})$,
- Step 4: determining the filtered data sequence $\{\overline{y}_t\}$ using $\hat{C}(q^{-1})\overline{y}_t = y_t$,
- Step 5: using the residual sequence $\{\varepsilon_t\}$ and the filtered data sequence $\{\overline{y}_t\}$ to get the final asymptotically unbiased and efficient estimator of $\hat{A}(q^{-1})$ and $\hat{C}(q^{-1})$.

With the ARMA model, the MA model coefficients can be found through the same polynomial comparison method shown in Durbin-Levison Algorithm, and estimate σ_a through $\{\varepsilon_t\}$. Then, using (3.36) to get performance index $\eta(d)$.

3.3 Performance Index Calculation via a Laguerre Network

The above approaches to calculate performance index are via an ARMA model, which means the degree of the model has to be chosen. This is not always an easy choice.

For example, in the Durbin-Levison Algorithm, the choice of the AR model order directly affects the accuracy of the performance index. The model parameter identification procedure can also be very complicated, as shown in the Mayne-Firoozan Algorithm.

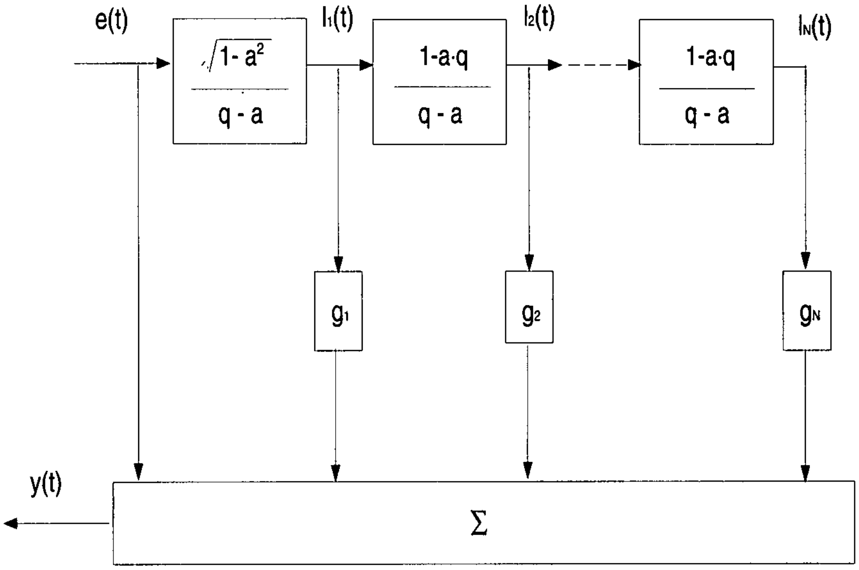


Figure 3.16: Block Diagram of Laguerre Network

By comparison, the use of a Laguerre network has some attractive and convenient properties. A block diagram representation of the Laguerre network which models a control loop is shown in Figure 3.16. A discrete Laguerre model is usually written as [8]:

$$L_i(q) = \frac{\sqrt{1-a^2}}{q-a}(\frac{1-aq}{q-a})^{i-1} \qquad i = 1, \cdots \tag{3.59}$$

where a is the Laguerre filter time scale.

Because Laguerre functions are orthonormal and complete in $L_2[0,\infty)$, they can represent the impulse response of any stable sampled linear time invariant system $h(t)$ with an infinite expansion:

$$h(t) = \sum_{i=1}^{\infty} g_i l_i(t) \tag{3.60}$$

where g_i is the i-th Laguerre gain.

$l_i(t)$ is the output of i-th Laguerre filter.

In practice, the above infinite expansion is truncated after N filters, which expressed in transfer function form is:

$$H(q^{-1}) = \sum_{i=1}^{N} g_i L_i(q^{-1}) \tag{3.61}$$

In (3.61), $L_i(q^{-1})$ depends only on the time scale a, which is often chosen by trial as 0.2 and 0.3. So once the number of Laguerre filters N is set, the only parameters remaining to be determined are the g_i, which can be found through recursive extended least squares (RELS) estimation. After determining the Laguerre network, the performance index is calculated. The whole procedure is shown below [16]:

- Step 1: Represent the discrete Laguerre network in state-space form:

$$l(t+1) = Al(t) + be(t) \tag{3.62}$$

$$y(t) = c^T l(t) + e(t) \tag{3.63}$$

where A is a matrix whose elements are given by:

$$A = \begin{cases} a_{ij} = a & i = j \\ a_{ij} = 0 & i < j \\ a_{ij} = (-a)^{i-j-1}(1-a^2) & i > j \end{cases} \tag{3.64}$$

b is a vector whose elements are given by:

$$b_i = (-a)^{i-1}\sqrt{1-a^2} \qquad i = 1, \cdots, N \tag{3.65}$$

c is a vector whose elements are given by:

$$c = [g_1 \; g_2 \cdots \; g_N]^T \tag{3.66}$$

$l(t)$ is a vector whose elements are given by:

$$l(t) = [l_1(t)\ l_2(t)\ \cdots\ l_N(t)]^T \tag{3.67}$$

- Step 2: Estimate the parameters of the model recursively by using:

$$l(t) = Al(t-1) + b\eta(t-1) \tag{3.68}$$

$$P(t) = P(t-1) - \frac{P(t-1)l(t)l^T(t)P(t-1)}{1 + l^T(t)P(t-1)l(t)} \tag{3.69}$$

$$\hat{c}(t) = \hat{c}(t-1) + P(t)l(t)[y(t) - \hat{c}^T(t-1)l(t)] \tag{3.70}$$

$$\eta(t) = y(t) - \hat{c}^T(t)l(t) \tag{3.71}$$

where the initial values $l(0), \hat{c}(0)$ are randomly set, the initial values of $P(0)$ are set to be very large.

- Step 3: Terminate the recursive process when parameters of the model have converged, and now the residual $\eta(t)$ gives an estimate of the white noise $e(t)$:

$$\hat{\sigma}_\eta^2 = \hat{\sigma}_e^2 \tag{3.72}$$

- Step 4: Calculate the loop output variance under minimum variance control:

$$\sigma_{mv}^2 = \hat{\sigma}_\eta^2[1 + (\hat{c}^T b)^2 + (\hat{c}^T Ab)^2 + \cdots + (\hat{c}^T A^{d-2} b)^2] \tag{3.73}$$

where d is the loop time delay.

- Step 5: Compute the perform index $\eta(d)$:

$$\eta(d) = 1 - \sigma_{mv}^2/(\sigma_y^2 + \bar{d}_y^2) \tag{3.74}$$

Compared to the Durbin-Levison algorithm, which is batch-processing, this algorithm is recursive and is easier to carry out on-line. Compared to Mayne-Firoozan algorithm, its computation load is much less.

3.4 Simulation Results

Consider the following process loop, (process time delay $d = 4$ is assumed known):

$$y(t) = \frac{14.647}{(1 - 0.35z^{-1})} u(t-4) + \frac{1}{(1 - 1.35z^{-1} + 0.35z^{-2})} a(t) \tag{3.75}$$

The control law is given as following, and is proved to be close to minimum variance control [11].

$$u(t) - u(t-1) = -0.13y(t) + 0.011y(t-1) \tag{3.76}$$

The loop noise signal $a(t)$ is simulated using pseudo random white noise sequence with zero mean and 0.6 variance, i.e. $\{a(t)\} \sim WN(0, 0.6^2)$.

First, calculate the performance index $\eta(d)$ via an ARMA model (using Durbin-Levison algorithm). The MA model coefficients are estimated as:

$$\psi_1 = 0.8636 \qquad \psi_2 = 0.1898 \qquad \psi_3 = -0.1658$$

The noise signal variance is estimated as:

$$\nu_m^2 = 0.3587$$

So, the loop deviation given by the minimum variance control is:

$$\sigma_{mv}^2 = (1 + \psi_1^2 + \psi_2^2 + \psi_3^2)\hat{\nu}_m^2 = 0.6490$$

The loop deviation σ_y^2 and the mean deviation from setpoint $\bar{d}_y^2$ under the current control can be easily calculated from loop data, the results are:

$$\sigma_y^2 = 0.7260$$

$$\bar{d}_y^2 = 0.0023$$

And, the performance index $\eta(d)$ can now be computed out:

$$\eta(d) = 1 - \sigma_{mv}^2/(\sigma_y^2 + \bar{d}_y^2) = 0.1089$$

Next, we calculate the performance index $\eta(d)$ using the Laguerre network representation. The Laguerre filter number N is chosen as 6, the filter time scale a is chosen as 0.2, and the network coefficients are estimated as:

$$\hat{c}^T b = 0.8360 \qquad \hat{c}^T A b = 0.2010 \qquad \hat{c}^T A^2 b = -0.2017$$

The noise signal variance is estimated as:

$$\hat{\sigma}_\eta^2 = 0.3611$$

So, the loop deviation given by the minimum variance control is:

$$\sigma_{mv}^2 = \hat{\sigma}_\eta^2[1 + (\hat{c}^T b)^2 + (\hat{c}^T A b)^2 + \cdots + (\hat{c}^T A^{d-2} b)^2] = 0.6428$$

The loop deviation σ_y^2 and the mean deviation from setpoint $\bar{d}_y^2$ under the current control are obtained as before. And, the performance index $\eta(d)$ is now computed out as:

$$\eta(d) = 1 - \sigma_{mv}^2/(\sigma_y^2 + \bar{d}_y^2) = 0.1175$$

The control law is proved to be close to minimum variance control [11], so the loop performance index should be small. The results obtained from both methods are also small, indicating both methods are right.

The process time delay d must be known in advance, otherwise neither method can give out an accurate performance index. Table 3.1 shows the corresponding performance index values when different time delay values are used.

Table 3.1: Performance Index for Different Time Delay d

	d=2	d=3	d=4	d=5	d=6
ARMA model method	0.6510	0.3301	0.1089	0.1003	0.0392
Laguerre network method	0.6718	0.3485	0.1175	0.1040	0.0571

The performance index value is not very sensitive to the Laguerre filter number N and the filter time scale a. Table 3.2 shows the different results when N and a vary.

Table 3.2: Performance Index for Different N and a

	$N = 5$	$N = 6$
$a = 0.2$	0.1303	0.1175
$a = 0.3$	0.1326	0.1281

3.5 Conclusion

The performance index is a scalar value which reflects how well a control loop is behaving and is therefore useful in control loop monitoring automation. This chapter describes methods of calculating the performance index. Two approaches are introduced: one using the ARMA model and the other using a Laguerre network. The two methods use the same Harris index definition to calculate the performance index $\eta(d)$:

$$\eta(d) = 1 - \sigma_{mv}^2/(\sigma_y^2 + \bar{d}_y^2)$$

where σ_{mv}^2 is the loop deviation under the minimum variance control,

σ_y^2 is the loop deviation under the current control,

$\bar{d}_y^2$ is the mean deviation from setpoint.

Two methods only differ in how to estimate σ_{mv}^2. The ARMA approach calculates it by coefficients of the MA model, while the second method uses Laguerre filter gains. Given the loop time delay, both can give out a relatively accurate performance index value. The ARMA model method first needs to estimate the AR model coefficients and then convert them to MA model coefficients by polynomial comparison. The computation load is significant. Also, it is batch algorithm so that it is not directly suitable for on-line implementation. By comparison, the Laguerre network method is quicker in computation as it only needs to estimate the Laguerre filter time scale. In addition, it is easier for on-line implementation because it is an recursive algorithm.

If the process time delay is unknown, neither method can calculate the performance index accurately. Accurate identification of the process time delay is therefore a very important issue that needs extensive research work.

Chapter 4

Loop Oscillation and Valve Friction Detection

4.1 Introduction

It has been found that oscillations occur in many process control loops. Thus they may have severe effects on the performance of the loop: increasing energy consumption, causing raw material loss and product quality degradation. In addition to outside load disturbances and improper controller parameters, control valve friction is a major cause of loop oscillation: it has been claimed that about 30% of all process loops are oscillating because of valve friction [3].

A control loop performance monitoring system must be able to detect loop oscillation and diagnose the cause. This chapter will present procedures for detecting loop oscillation and valve friction.

4.2 Loop Oscillation Detection

4.2.1 Loop Oscillation Causes

The major causes for loop oscillation are [9]:

- Load disturbances near the ultimate frequency (where the phase lag is $-180°$). While low frequency load disturbances are eliminated by the controller, and high frequency load disturbances are filtered out by the process itself, such middle range frequency load disturbances cannot be treated by either the controller or the process, instead they are typically amplified by feedback.

- A badly tuned controller. Especially when an nonlinear loop is subjected to a change in operating point, too high a controller gain will likely result in a loop oscillation.

- Valve friction. When friction occurs, a valve introduces nonlinearity to the whole loop, and may lead to stick slip and finally causes loop oscillations. The patterns of such oscillations vary, depending on the type of valve and the type of friction. Often this effect is incorrectly believed result from controller mis-tuning.

4.2.2 Loop Oscillation Detection

One approach to loop oscillation detection is quite intuitive: if within a certain period of time, the process output crosses the setpoint too many times and each time with a too big overshoot, then a loop oscillation is concluded to be present.

The whole procedure is as follows [9]:

- Step 1: Calculate the integrated absolute error (IAE) using:

$$IAE = \int_{t_{i-1}}^{t_i} |e(t)|dt \tag{4.77}$$

where t_{i-1} and t_i are two consecutive times of zero crossings,

$e(t)$ is process output error.

- Step 2: If IAE exceeds a certain limit IAE_{lim}, then one load disturbance is counted to have occurred.

The IAE_{lim} can be computed by:

$$IAE_{lim} = \frac{2}{\omega_u} \tag{4.78}$$

where ω_u is the ultimate frequency.

In most cases, ω_u is unknown and replaced by ω_i:

$$\omega_i = \frac{2\pi}{T_i} \tag{4.79}$$

where T_i is the integral time of the controller.

- Step 3: If over a supervision time T_{sup}, the number of detected load disturbances exceeds a certain limit n_{lim}, then a loop oscillation is considered to be present.

  n_{lim} is usually chosen as 10, and the supervision time T_{sup} is chosen as:

  $$T_{sup} \geq n_{lim} \frac{T_u}{2} \tag{4.80}$$

  where T_u is the ultimate oscillation period.

  In practice, if T_u is unknown, it can be replaced by the integral time of the controller T_i.

In the above procedure it is assumed that the controller has integral action so that the mean process output error is zero. If the controller does not contain an integral action, a similar approach can still be obtained [9].

Figure 4.17 shows two simulated control loops. Applying the above procedure to them, the following results were obtained:

For loop 1,

$$Ti = 10sec$$

So,

$$IAE_{lim} = \frac{2}{\omega_i} = \frac{2}{2\pi/T_i} \approx 3.18$$

n_{lim} is chosen as 10, so the supervision time must satisfy:

$$T_{sup} \geq n_{lim} \frac{T_i}{2} = 50sec$$

As the sampling frequency is one sample per second, we choose 300 samples so that $T_{sup} = 300sec$. It is easy to calculate that the total number of IAE whose value exceeds IAE_{lim} is 14, greater than n_{lim}, so it is concluded that oscillation exists in loop 1. Similarly it is also concluded that oscillation exists in loop 2. However, as will be seen in section 4.3, the reasons for the oscillation in those two loops are different.

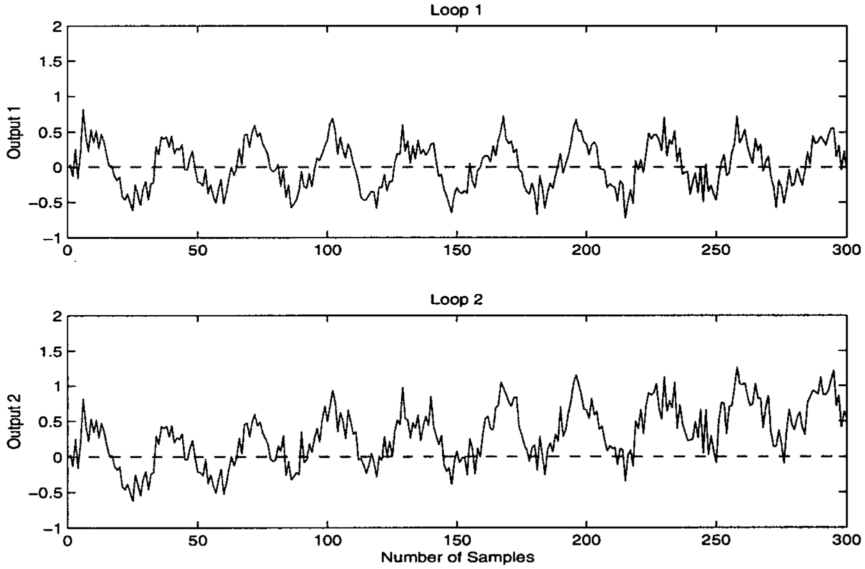


Figure 4.17: Loops with Oscillation

4.3 Valve Friction Detection

After loop oscillation is detected, the next step is naturally to find out its origin so that corrective actions can be taken. As mentioned, valve friction is one major cause for loop oscillation and moreover this kind of oscillation is often mistaken as due to poorly tuned controller parameters. As a result, controller parameters are sometimes re-tuned to very conservative values and the whole loop performance is degraded. In contrast, if there is a method to determine whether or not valve friction is present and to estimate the friction degree, then an appropriate solution can be found: If the friction level is high, then it may be the origin of the loop oscillation so that the first step should be valve maintenance; If the friction level is not high, then a mis-tuned controller or an external perturbation is more likely to be responsible for the oscillation.

A method to detect valve friction is now discussed [26]. For each type of valve used in control loops, there is a characteristic function (or characteristic table) describing the valve input-output relation. With an ideal valve, its output will track the valve reference signal given by the controller, in conformity with the valve characteristic function. But in practice, the real valve output tracks the reference signal with delay and/or overshoot. As a result, the valve output is different from the one indicated by the valve characteristic function (See Figure 4.18). The more friction that exists in a valve, the more deviation appears between the real valve output and the indicated value. The degree of valve friction is found by measuring such deviation.

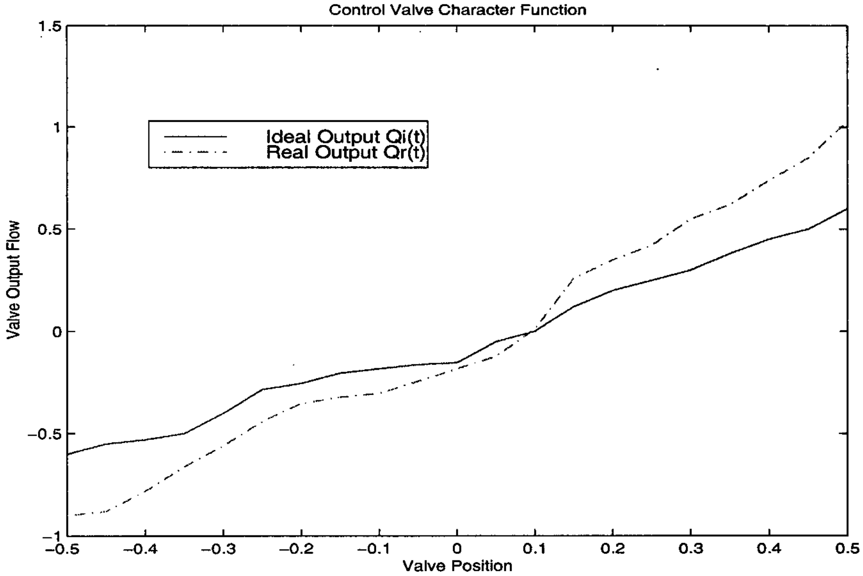


Figure 4.18: Control Valve Input-Output Relation

The whole procedure is:

- Step 1: Calculate D, the overall deviation between real valve output and ideal valve output over a supervision time:

$$D = \sum_{t=0}^{t=T_{sup}} |Q_r(t) - Q_i(t)| \tag{4.81}$$

where T_{sup} is the supervision time,

$Q_r(t)$ is the real valve output,

$Q_i(t)$ is the ideal valve output.

- Step 2: Determine D_{max}, the maximum acceptable overall deviation between real valve output and ideal valve output over a supervision time:

$$D_{max} = \sum_{t=0}^{t=T_{sup}} |Q_{max}(t) - Q_i(t)| \tag{4.82}$$

where $Q_{max}(t)$ is the maximum allowed valve output deviation in each valve position.

- Step 3: Define a valve friction index F_r:

$$F_r = \frac{D}{D_{max}} \tag{4.83}$$

If $F_r \ll 1$, then the real valve output is close to the ideal one, which means the valve is operating in conformity with its characteristic function; If $F_r > 1$, then the real valve output has deviated far away from the ideal one, which indicates high friction in the valve; If $F_r \approx 1$, then the valve is near its maximum allowed deviation.

The method is now used to diagnose two loops shown in Figure 4.17. For loop 1, $F_r \approx 0.6$, which means the control valve does not have too much friction and the oscillation reason should be either external perturbation or bad tuned controller parameters. For loop 2, $F_r \approx 3.5$, which indicates high valve friction has caused the loop oscillation. Therefore, the two loops need different remedies in order to eliminate oscillations from them.

4.4 Conclusion

Loop oscillation is a common phenomenon encountered in process loops and can have a severe impact on the loop performance. The detection of its existence is an important issue in control loop performance monitoring. This chapter introduced a procedure for carrying out this task. First the load disturbance is detected through loop output integrated absolute error (IAE), and then the total number of load disturbance occurrences within a supervision time is checked. If this number exceeds a certain limit, then an oscillation is believed to have occurred. This procedure is simple and easy to implement, and yet may yield good results.

After a loop oscillation is detected, the next step is to diagnose its cause. One algorithm to detect valve friction was presented here. It measures the deviation distance between the real valve output and the ideal one described by its characteristic input-output function. Based on this, a friction index F_r is defined. When F_r is greater than 1, valve friction is likely to exist. The method is simple and does not require a model, as long as the characteristic input-output function of the valve is known.

However, loop oscillation may be also resulted from external load disturbance, or mistuned controller parameters. When a loop oscillates, the interaction with other loops will export the oscillation so that it is often very difficult to determine which loop is the source of the oscillation. So, many challenges remain.

Chapter 5

Performance Monitoring Application Software

5.1 Introduction

Control loop performance monitoring is very useful for assessing the loop operation and ensuring the product quality, however, very few industrial application software systems are available. This gap must be filled.

As part of this thesis, an integrated performance monitoring software system has been developed in an industrial environment and evaluated. Its source code exceeds 10,000 lines.

In the work of developing a software system, the algorithm part usually accounts for about 35%, the system specification and structure design part accounts approximately for 25%, the user interface part accounts for 20%, and the installation and validation part usually accounts for 20%. However, this chapter will only introduce the system from a general perspective, and with emphasis on its applications.

5.2 System Development Environment and Phases

5.2.1 Development Environment

The system was developed in a Windows 95/NT environment, using Visual C++ 5.0 Developer Studio and Microsoft Foundation Classes (MFC). This provides an excellent development environment for the performance monitoring system.

Windows programming is different from traditional programming. Generally speaking, it has the following major characteristics:

- Multi-Task: Windows is a multi-task operating environment, which means that under Windows, multiple programs can run at the same time.
- Event-Driven: A Windows program is event-driven; each user action (called an event) will typically result in a particular segment of the program being executed.
- Graphical Interface: Windows supports a complete and elegant graphical interface to the user, which makes the user operation intuitively easy.
- Hardware Isolation: No direct access to the hardware resources (for example, keyboard, screen, printer, etc.) is permitted.

The above four attributes enable a well designed Windows program to be run-time effective, system reliable and user friendly.

C++ is an object-oriented programming language. It has four distinct features:

- Inheritance, such as super- and sub-classes;
- Generic Programming, such as template technology;
- Polymorphism, such as operator overloading;
- Dynamic Binding, such as virtual function.

These features greatly increase the reliability, reusability and maintainability of a program, especially of a large program written in C++. C++ therefore has now become a widely used program language.

Until recently, Windows programming was very time-consuming and tedious with many functions for a programmer to remember. Microsoft Foundation Classes (MFC), a set of pre-defined classes which bind most of these functions into some application-specific groups, make today's Windows programming much easier and more efficient.

5.2.2 Development Phases

The system was developed in conformity to the standard software development process [24]. Four phases have been carried out to produce the final software system:

- Software Specification: define the functionality of the software and constraints on its operation;
- Software Development: write software code to implement the specification;
- Software Validation: validate to ensure that the software really does what the customer wants;
- Software Evolution: evolve the software to meet the changing customer needs.

5.3 System Structure and Functions

5.3.1 System Structure

The system is a multiple document interface (MDI) application program so that it can monitor multiple loops at the same time. Figure 5.19 shows the system structure. For each loop, the data are first sampled by the DCS – the "DCS data"; then they are collected and stored in a Foxboro Data for Windows database – the "real-time or historian data". The performance monitoring system accesses the data from Data for Windows; pre-processes it (as discussed before, first filtering it with a 4-th order Bessel lowpass filter, then down-sampling it); and then performs three kinds of operations on the data: process data analysis, performance index calculation, loop oscillation and valve friction detection. Finally the results are shown on the screen and can be saved to a file and/or printed.

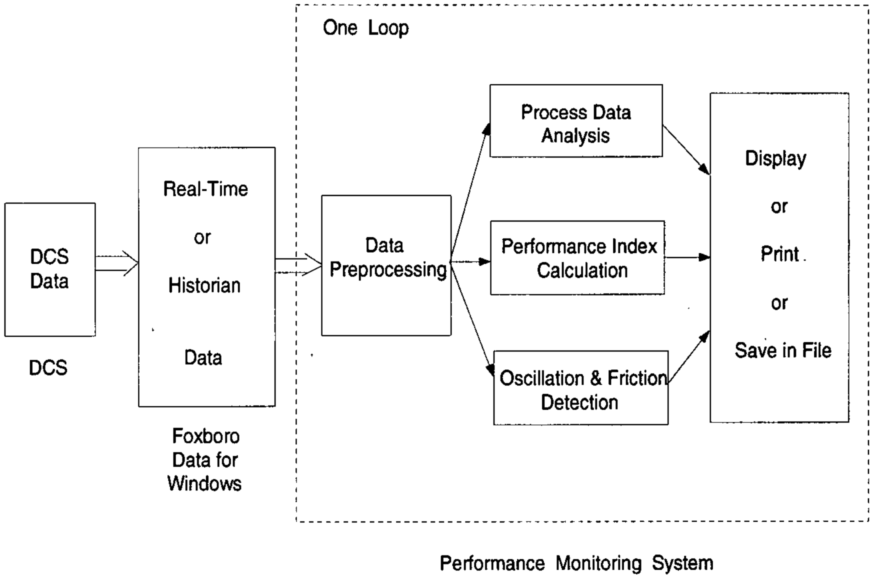

One Loop
DCS Data
DCS
Real-Time or Historian Data
Foxboro Data for Windows
Data Preprocessing
Process Data Analysis
Performance Index Calculation
Oscillation & Friction Detection
Display or Print or Save in File
Performance Monitoring System

Figure 5.19: Performance Monitoring System Structure

5.3.2 System Functions

The system has two groups of functions: system service, performance monitoring.

System Service Function

Figure 5.20 shows the whole system function menu, where *Plot*, *Edit*, *View*, *Window*, *Help* are system service functions. They are described below in detail:

- Plot: contains functions related for plot/file operation, including drawing a new plot, saving it in a file; calling an existing plot from a file; printing out a plot; directly calling four newly used plots; etc.

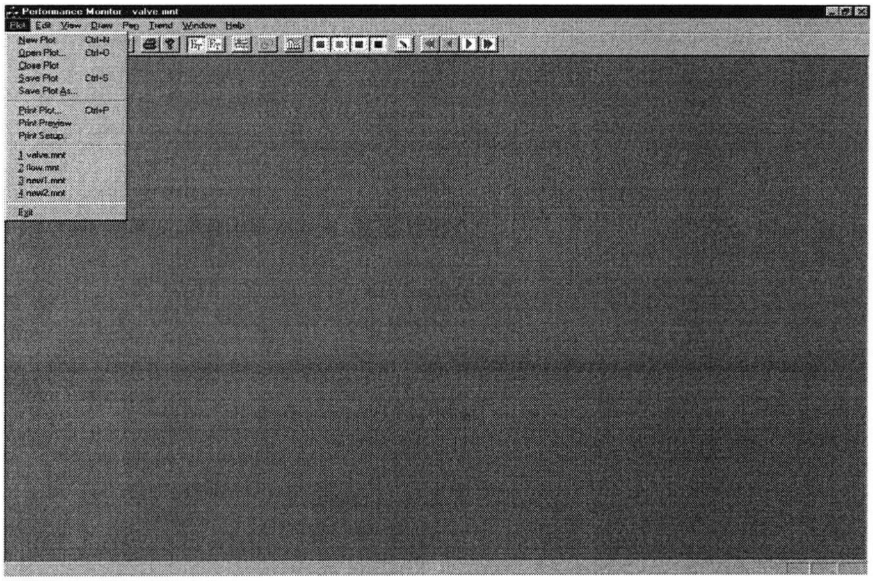


Figure 5.20: Performance Monitoring System Function

- Edit: contains functions related to plot editing, including cutting or copying a plot or a portion of a plot and later pasting it. This function is not currently implemented.

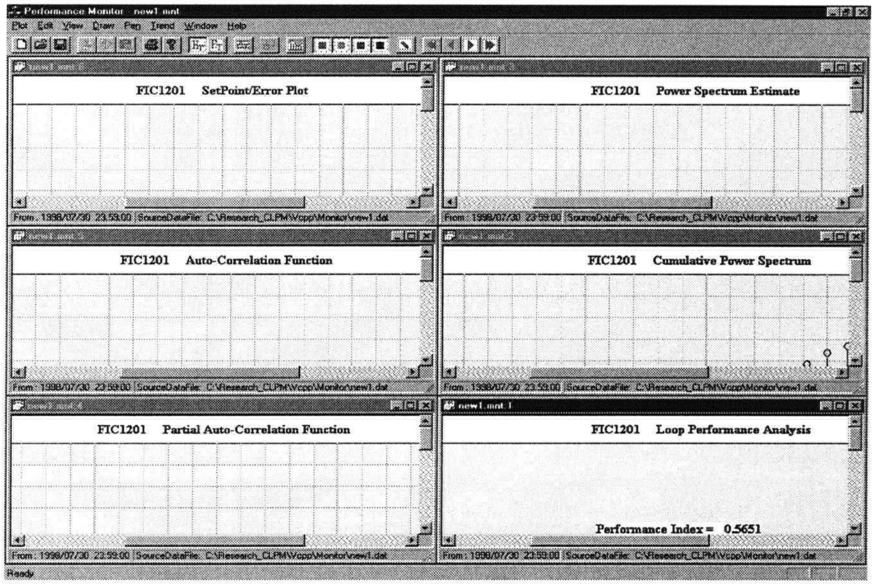
- View: contains functions for viewing the plot. By changing the view scale, the plot can be shown on the screen in different sizes.

- Window: contains functions related to windows management. As there are several windows on the screen simultaneously, the user should be able to tile them, cascade them or choose any one of them as the active window. All these operations are defined here.

- Help: contains information explaining the system's operation.

Performance Monitoring Function

In Figure 5.20, *Draw*, *Pen*, *Trend* are performance monitoring functions. They provide the following operations:

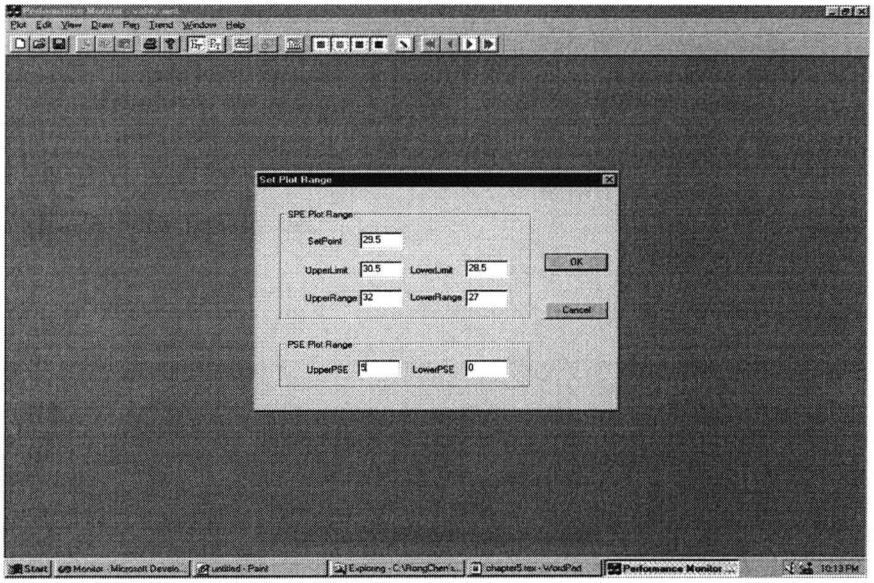

Performance Monitor - new1.mnl
Plot Edit View Draw Pen Trend Window Help
FIC1201 SetPoint/Error Plot
FIC1201 Power Spectrum Estimate
FIC1201 Auto-Correlation Function
FIC1201 Cumulative Power Spectrum
FIC1201 Partial Auto-Correlation Function
FIC1201 Loop Performance Analysis
From: 1998/07/30 23:59:00 | SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\new1.dat
new1.mnl:1
Ready

Performance Index = 0.5651

Figure 5.21: Six Plots in the Performance Monitoring System

- Draw: As shown in Figure 5.21, the system will automatically draw six plots for each loop, they are: setpoint/error plot, autocorrelation function plot, partial autocorrelation function plot, power spectrum estimation plot, cumulative power spectrum estimation plot, loop performance analysis results. The plot range can be adjusted by the user in order to show each plot more clearly, this is done by a dialogue between the user and the system. Figure 5.22 shows such a dialogue. Also, the plots can be drawn using either real-time data or historical data. All these operations are specified in Draw.

- Pen: To distinguish different plots, the user may choose different pens with different



SPE Plot Range

SetPoint 29.5

UpperLimit 30.5 LowerLimit 28.5

UpperRange 32 LowerRange 27

PSE Plot Range

UpperPSE 5 LowerPSE 0

Figure 5.22: A Dialogue for Changing the Plot Range

colors, widths, types to draw them respectively. These operations are defined here.

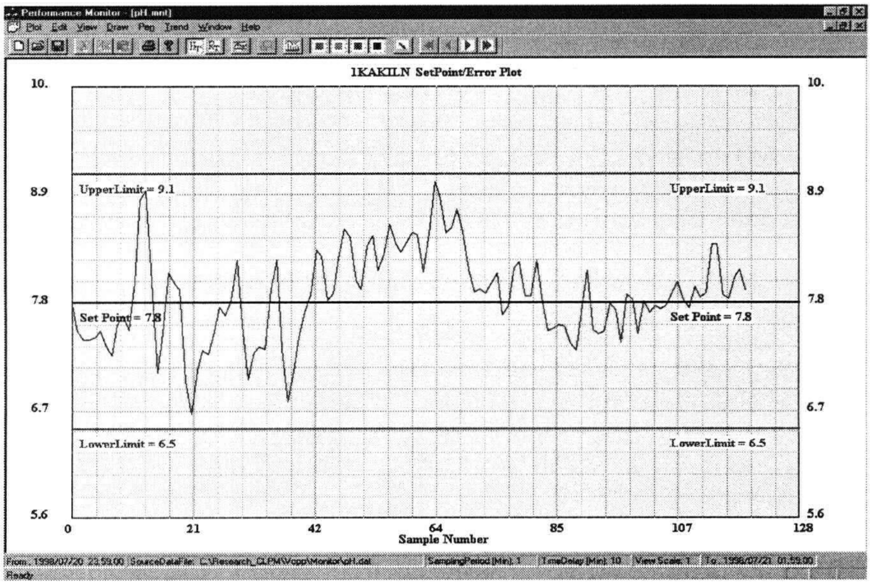
- Trend: If a plot is drawn using real-time data, then the plot will be automatically updated. If the plot is drawn using historical data, then the user might wish to browse the plot from current time backward or forward. Using operations defined here (Backward/Forward One/Ten Sample(s)), this can be easily done.

5.4 Application Cases

In this section, we will illustrate how the performance monitoring system works via two application cases.

5.4.1 pH Control Monitoring

First apply the system to monitor a pH control loop. It is the effluent pH control in a biobasin for pulp preparation with 93% sulfuric acid. Pulp is mixed with the requisite amount of bleaching agent to carry out bleaching reaction, and then is transferred to a subsequent washing operation. During the reaction, pH value must be maintained at a certain level. A pH probe is in the line just before entering the biobasin and a disturbance flow of spill lagoon pumpback is added to the stream after the pH probe. The loop is controlled by a Foxboro PID controller. The time delay of the loop is 10 minutes. The original sampling period is 10 seconds, which is too fast for this loop. So, the data has gone through a 4−th order Bessel lowpass filter with a stop-band of $0.5 minute^{-1}$ and then down sampled to a period of 1 minute.

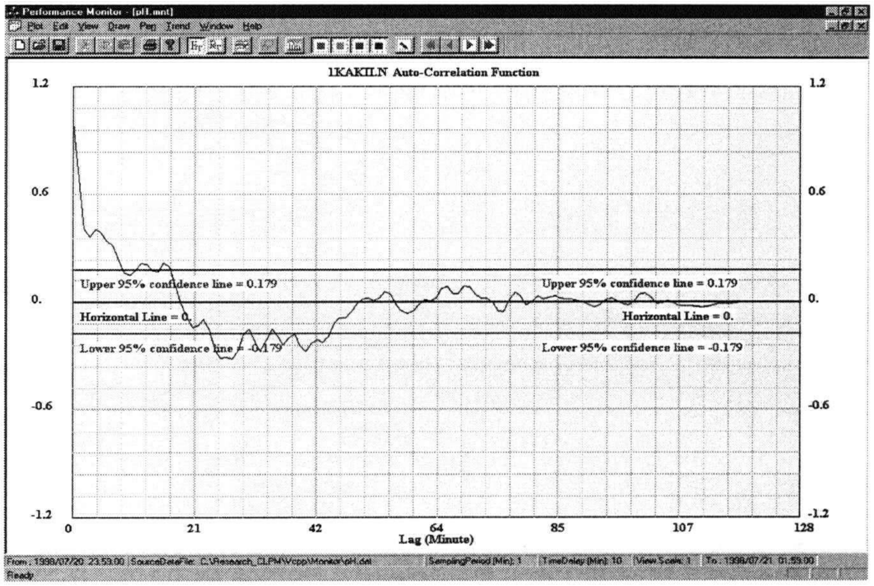


From: 1998/07/20 23:59:00 SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\pH.dat SamplingPeriod (Min): 1 TimeDelay (Min): 10 ViewScale: 1 To: 1998/07/21 01:59:00

Ready

Figure 5.23: Setpoint/Error Plot for a pH Control Loop

Figure 5.23 shows the setpoint/error plot for the down-sampled control loop. From the plot we can see the control effect is fairly good, with fluctuations within the lower and upper limits (here limits are set according to the maximum allowed process output). For more information about the loop operation, other monitoring results are examined.


Performance Monitor - [pH.mnt]
Plot Edit View Draw Pen Trend Window Help
1KAKILN Auto-Correlation Function
1.2
0.6
0.
-0.6
-1.2
1.2
0.6
0.
-0.6
-1.2
Upper 95% confidence line = 0.179
Horizontal Line = 0.
Lower 95% confidence line = -0.179
Upper 95% confidence line = 0.179
Horizontal Line = 0.
Lower 95% confidence line = -0.179
0
21
42
64
85
107
128
Lag (Minute)

From : 1998/07/20 23:59:00 SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\pH.dat SamplingPeriod(Min): 1 TimeDelay(Min): 10 ViewScale: 1 To : 1998/07/21 01:59:00

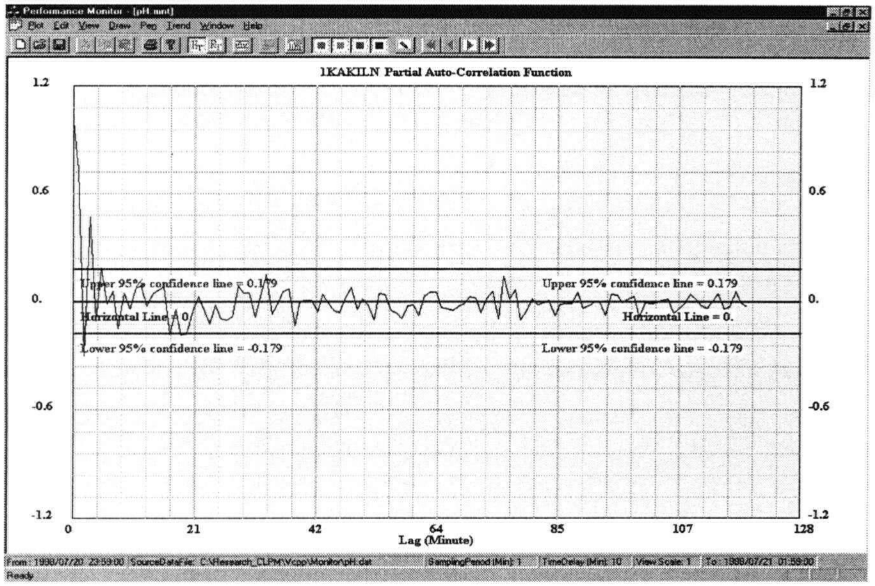
Ready

Figure 5.24: Autocorrelation Function Plot for a pH Control Loop

Figure 5.24 and 5.25 are respectively the autocorrelation function (ACF) and partial autocorrelation function (PACF) plots for the loop. From Figure 5.24 it can be seen that the ACF plot does not enter the 95% confidence limits for a large lag (greater than 40), which means the loop cannot be properly modeled as a moving average(MA) process.

From Figure 5.25 it is seen that the PACF plot decays quickly. After 4 lags it is within the 95% confidence limits. So the loop can be approximately expressed as a fourth order autoregressive(AR) process.

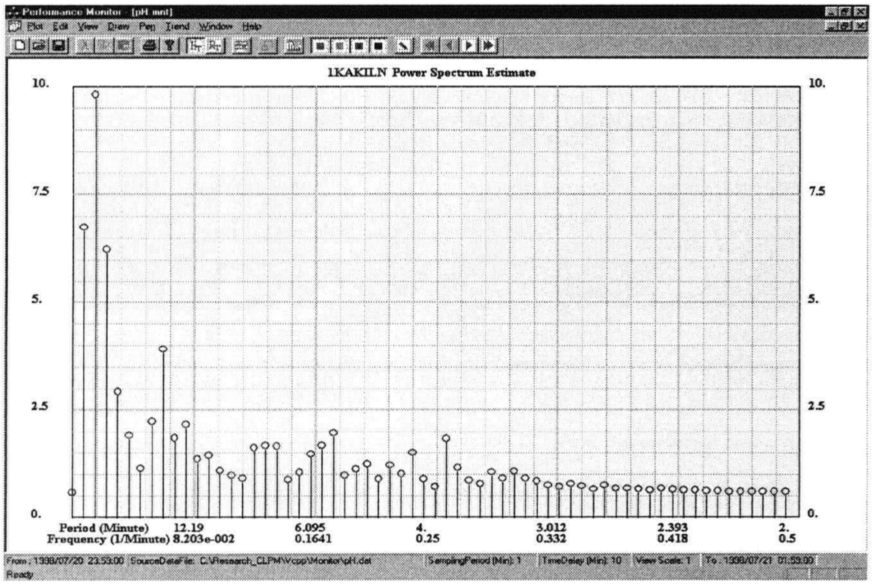
Next, we move to the frequency domain to examine the loop. Figure 5.26 and 5.27 are respectively the power spectrum estimation (PSE) and cumulative power spectrum estimation (CPSE) plots for the loop. From Figure 5.26 it can be seen that all major disturbances reside in low frequencies (lower than $0.1 minute^{-1}$). By examining Figure 5.27


Performance Monitor - [pH.mnt]
Plot Edit View Draw Pen Trend Window Help
1KAKILN Partial Auto-Correlation Function
1.2
0.6
0.
-0.6
-1.2
Upper 95% confidence line = 0.179
Horizontal Line = 0.
Lower 95% confidence line = -0.179
0
21
42
64
85
107
128
Lag (Minute)

From: 1998/07/20 23:59:00 | SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\pH.dat | SamplingPeriod (Min): 1 | TimeDelay (Min): 10 | View Scale: 1 | To: 1998/07/21 01:59:00

Ready

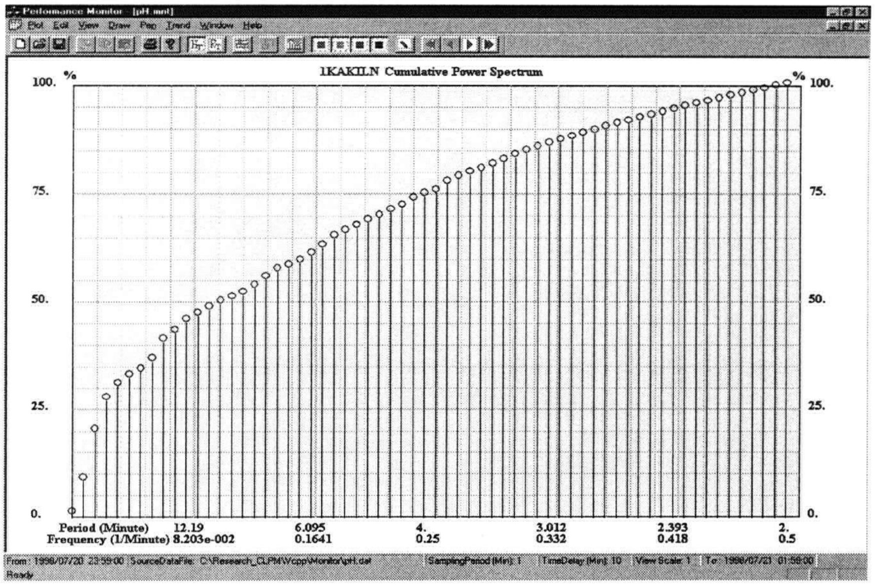
Figure 5.25: Partial Autocorrelation Function Plot for a pH Control Loop


Performance Monitor - [pH.mnt]
Plot Edit View Draw Pen Trend Window Help
1KAKILN Power Spectrum Estimate
10.
7.5
5.
2.5
0.
Period (Minute) 12.19 6.095 4. 3.012 2.393 2.
Frequency (1/Minute) 8.203e-002 0.1641 0.25 0.332 0.418 0.5

From: 1998/07/20 23:53:00 | SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\pH.dat | SamplingPeriod (Min): 1 | TimeDelay (Min): 10 | View Scale: 1 | To: 1998/07/21 01:53:00

Ready

Figure 5.26: Power Spectrum Estimation Plot for a pH Control Loop


Performance Monitor - [pH.mnt]
Plot Edit View Draw Pen Trend Window Help
1KAKILN Cumulative Power Spectrum
%
100.
75.
50.
25.
0.
Period (Minute) 12.19 6.095 4. 3.012 2.393 2.
Frequency (1/Minute) 8.203e-002 0.1641 0.25 0.332 0.418 0.5
Ready

From : 1998/07/20 23:59:00 SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\pH.dat SamplingPeriod (Min): 1 TimeDelay (Min): 10 View Scale: 1 To: 1998/07/21 01:59:00

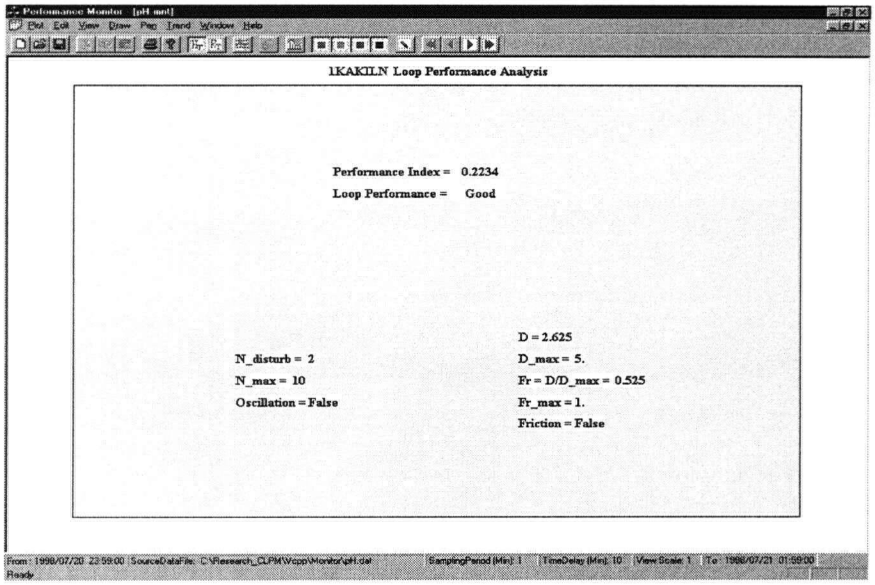
Figure 5.27: Cumulative Power Spectrum Estimation Plot for a pH Control Loop

it can be found that about 50% of the total disturbances are below $0.1 minute^{-1}$. This indicates that the major disturbance of the pH control loop comes from a slow source.

Finally, diagnose the loop more closely as in Figure 5.28. The performance index is $0.2234 \ll 1$, which indicates the control is good. Also, the diagnosis results show no loop oscillation and valve friction.

5.4.2 Flow Control Monitoring

The second application case is a white water flow control loop. It is used to control the pulp consistency. Pulp consistency, defined as mass or weight percentage of bone dry fiber in a pulp stock, can affect the final paper quality and need control accurately. It can be regulated by controlling the white water flow. A PID controller is applied to the white water flow loop. The time delay of the loop is 4 minutes. The original sampling period is 10 seconds, which is too fast for such a flow loop. So, the data has gone through a 4-th


Performance Monitor - [pH.mnt]
Plot Edit View Draw Pen Trend Window Help

1KAKILN Loop Performance Analysis

Performance Index = 0.2234
Loop Performance = Good

D = 2.625
N_disturb = 2
D_max = 5.
N_max = 10
Fr = D/D_max = 0.525
Oscillation = False
Fr_max = 1.
Friction = False

From : 1998/07/20 23:59:00 | SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\pH.dat | SamplingPeriod (Min): 1 | TimeDelay (Min): 10 | View Scale: 1 | To : 1998/07/21 01:59:00
Ready

Figure 5.28: Performance Analysis Results for a pH Control Loop

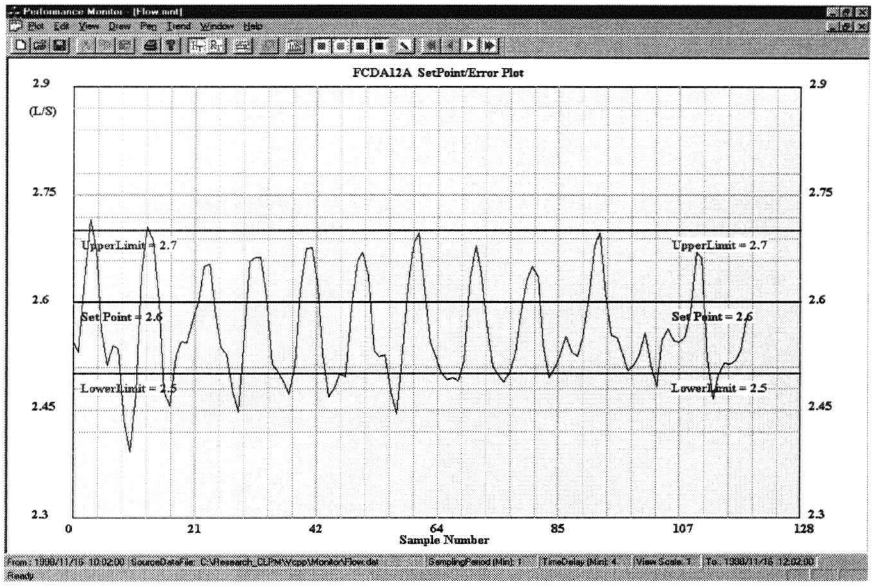
order Bessel lowpass filter with a stop-band of $0.5 minute^{-1}$ and then down sampled to a period of 1 minute.

Figure 5.29 shows the setpoint/error plot for the control loop. From the plot it is seen that the control effect is not good and an oscillation obviously exists. Other monitoring results help to determine the cause.

Figure 5.30 and 5.31 are respectively the autocorrelation function (ACF) and partial autocorrelation function (PACF) plots for the loop. From Figure 5.30 we can see the ACF plot fluctuates and does not stay in the 95% confidence limits for a very big lag (over 64 lags). The plot also shows that the peak values occur periodically, which indicates that the error is caused by a periodic component.

In Figure 5.31 the PACF plot decays very quickly, indicating the loop can be described by an AR model.

In the frequency domain, Figure 5.32 and 5.33 are respectively the power spectrum

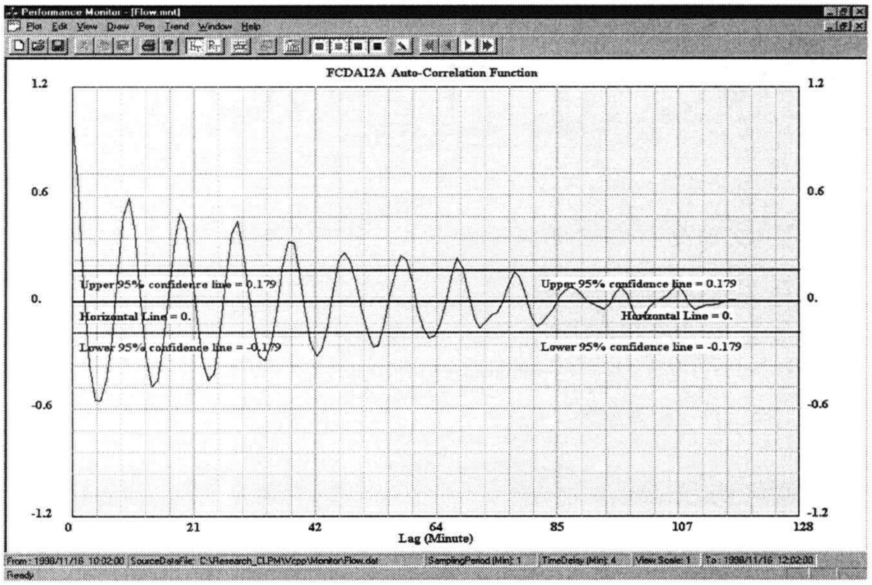


Performance Monitor - [Flow.mnt]
Plot Edit View Draw Pen Trend Window Help
FCDA12A SetPoint/Error Plot
2.9
(L/S)
2.75
2.6
2.45
2.3
UpperLimit = 2.7
Set Point = 2.6
LowerLimit = 2.5
0
21
42
64
85
107
128
Sample Number

From : 1998/11/16 10:02:00 SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\Flow.dat SamplingPeriod (Min): 1 TimeDelay (Min): 4 View Scale: 1 To : 1998/11/16 12:02:00

Ready

Figure 5.29: Setpoint/Error Plot for a Flow Control Loop

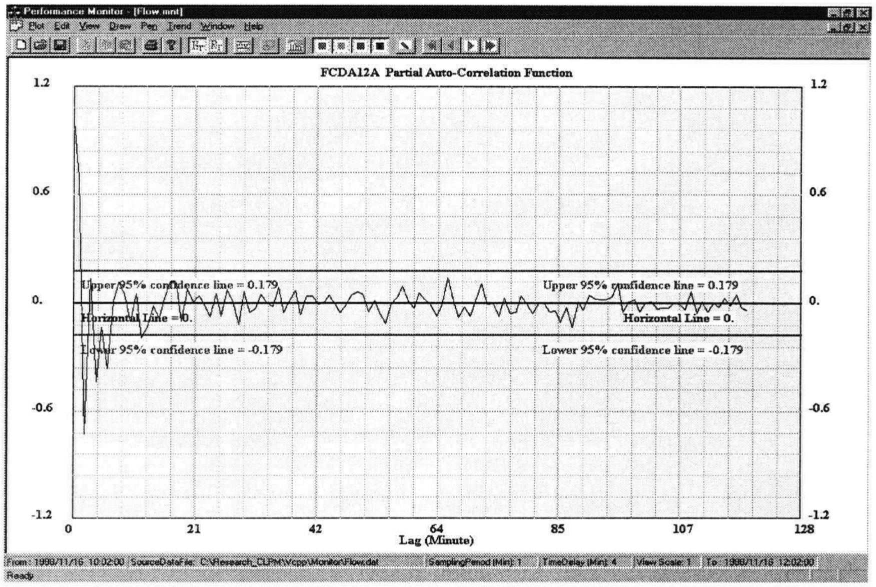


Performance Monitor - [Flow.mnt]
Plot Edit View Draw Pen Trend Window Help
FCDA12A Auto-Correlation Function
1.2
0.6
0.
-0.6
-1.2
Upper 95% confidence line = 0.179
Horizontal Line = 0.
Lower 95% confidence line = -0.179
0
21
42
64
85
107
128
Lag (Minute)

From : 1998/11/16 10:02:00 SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\Flow.dat SamplingPeriod (Min): 1 TimeDelay (Min): 4 View Scale: 1 To : 1998/11/16 12:02:00

Ready

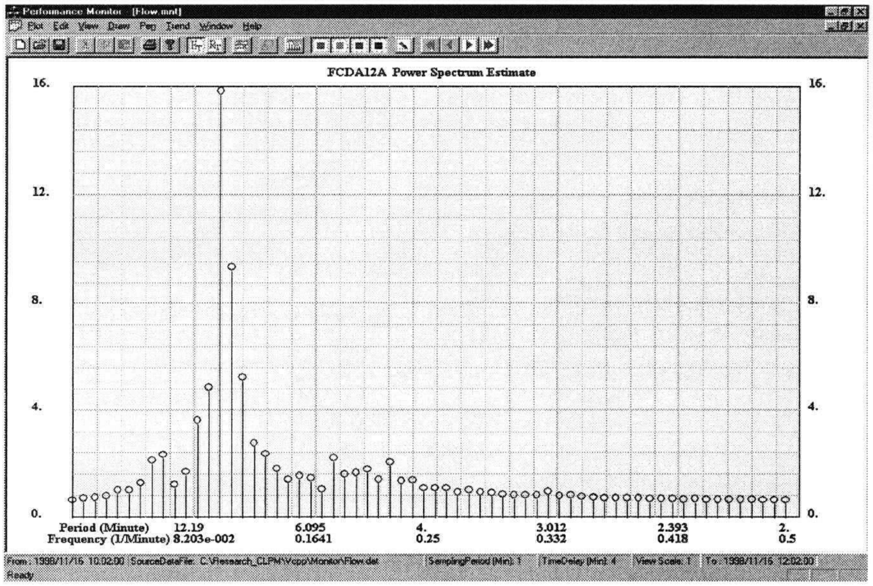
Figure 5.30: Autocorrelation Function Plot for a Flow Control Loop


Performance Monitor - [Flow.mnt]
Plot Edit View Draw Pen Trend Window Help
FCDA12A Partial Auto-Correlation Function
1.2
0.6
0.
-0.6
-1.2
Upper 95% confidence line = 0.179
Horizontal Line = 0.
Lower 95% confidence line = -0.179
Upper 95% confidence line = 0.179
Horizontal Line = 0.
Lower 95% confidence line = -0.179
0
21
42
64
85
107
128
Lag (Minute)

From: 1998/11/16 10:02:00 SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\Flow.dat SamplingPeriod (Min): 1 TimeDelay (Min): 4 View Scale: 1 To: 1998/11/16 12:02:00

Ready

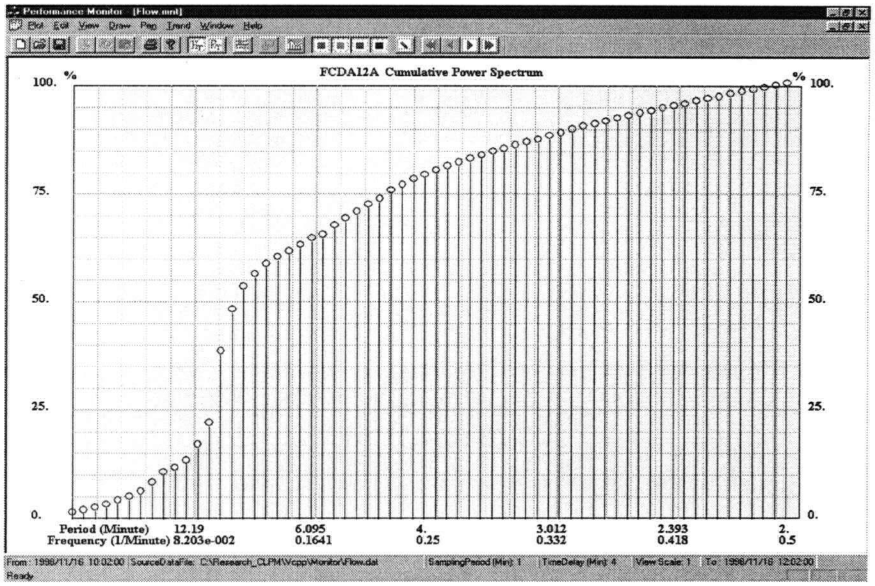
Figure 5.31: Partial Autocorrelation Function Plot for a Flow Control Loop


Performance Monitor - [Flow.mnt]
Plot Edit View Draw Pen Trend Window Help
FCDA12A Power Spectrum Estimate
16.
12.
8.
4.
0.
Period (Minute) 12.19 6.095 4. 3.012 2.393 2.
Frequency (1/Minute) 8.203e-002 0.1641 0.25 0.332 0.418 0.5

From: 1998/11/16 10:02:00 SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\Flow.dat SamplingPeriod (Min): 1 TimeDelay (Min): 4 View Scale: 1 To: 1998/11/16 12:02:00

Ready

Figure 5.32: Power Spectrum Estimation Plot for a Flow Control Loop

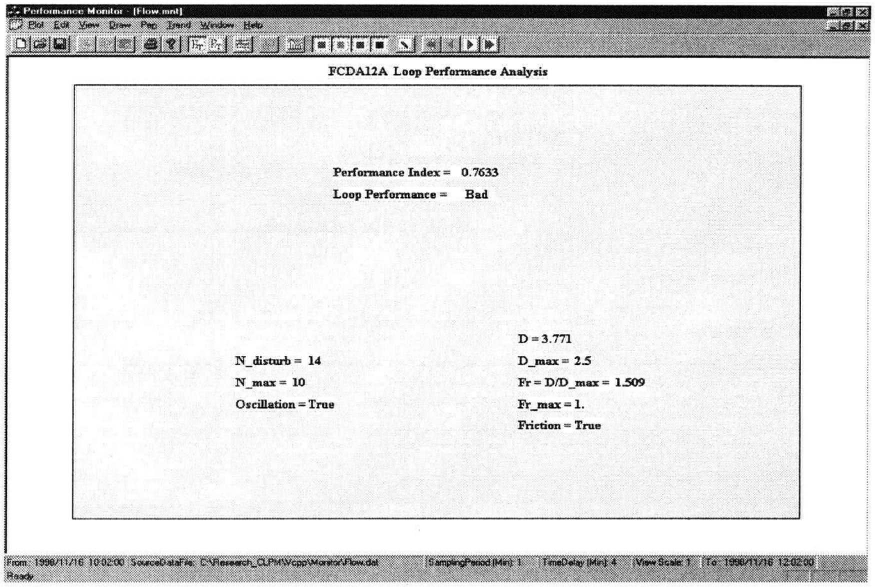

Performance Monitor - [Flow.mnl]
Plot Edit View Draw Pep Trend Window Help
FCDA12A Cumulative Power Spectrum
%
100.
75.
50.
25.
0.
Period (Minute) 12.19 6.095 4. 3.012 2.393 2.
Frequency (1/Minute) 8.203e-002 0.1641 0.25 0.332 0.418 0.5

From: 1998/11/16 10:02:00 | SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\Flow.dat | SamplingPeriod (Min): 1 | TimeDelay (Min): 4 | ViewScale: 1 | To: 1998/11/16 12:02:00
Ready

Figure 5.33: Cumulative Power Spectrum Estimation Plot for a Flow Control Loop

estimation (PSE) and cumulative power spectrum estimation (CPSE) plots for the loop. In Figure 5.32 note that there are peak values around $0.1minute^{-1}$ in the power spectrum, which means here exist major disturbances. From Figure 5.33, it can be seen that over 50% of total disturbances are lower than $0.1minute^{-1}$, and that around $0.1minute^{-1}$ there exists a sharp increase because here exist major disturbances.

Figure 5.34 shows the specific loop diagnosis results. The performance index is $0.7633 \approx 1$, indicating the control is poor. The diagnosis results show that both loop oscillation and valve friction exist. Probably valve friction is one cause of loop oscillation. So the first step for improving the control loop performance should be checking the control valve.


Performance Monitor - [Flow.mnt]
Plot Edit View Draw Pep Trend Window Help

FCDA12A Loop Performance Analysis

Performance Index = 0.7633
Loop Performance = Bad

D = 3.771

N_disturb = 14
N_max = 10
Oscillation = True

D_max = 2.5
Fr = D/D_max = 1.509
Fr_max = 1.
Friction = True

From: 1998/11/16 10:02:00 SourceDataFile: C:\Research_CLPM\Vcpp\Monitor\Flow.dat | SamplingPeriod (Min): 1 | TimeDelay (Min): 4 | View Scale: 1 | To: 1998/11/16 12:02:00
Ready

Figure 5.34: Performance Analysis Results for a Flow Control Loop

5.5 Conclusion

In this chapter, a performance monitoring system was presented. The system was developed in Windows 95/NT environment, using Visual C++ 5.0 Developer Studio and Microsoft Foundation Classes (MFC). These development environment and tools greatly facilitate software development. The system was developed according to well-established software development phases.

The system is a multiple document interface (MDI) application program and can monitor multiple loops simultaneously. The system receives loop data via Foxboro Data for Windows and then analyses the loop performance. There are two groups of function: system service function and performance monitoring function. The former provides management services for plots, files and windows, the latter provides operations for performance monitoring and related accessory functions.

Two application cases were also given out. One is a pH control loop, the other is a flow control loop. Through the two cases, procedures on diagnosis of a loop using the six different monitoring plots were expounded. The underlying ideas of the six plots were explained in Chapter 2, 3, 4 respectively.

Chapter 6

Conclusions

6.1 Summary of Thesis Work

Since Harris published his paper on control loop performance monitoring in 1989, the process control community has given great attention to this field. If the loop performance can be assessed solely from routine operating data, especially if it can be assessed by a relatively simple index, process industries will be able to make sure that huge numbers of control loops are working properly with a very limited number of control staff.

Research since then has considered feedback-only and feedforward plus feedback control, SISO system and MIMO systems, and from basic performance indexes to generalized ones. Currently feedback-only loop in the SISO system is the best understood area. Such loops are the most widely used ones in the process industry and consist the backbone of all advanced control systems.

To carry out control loop performance monitoring task, three approaches are commonly used: (1) direct process data analysis; (2) performance index calculation; (3) loop oscillation and valve friction detection.

Direct process data analysis includes time domain and frequency domain data analysis. In the time domain, autocorrelation and partial autocorrelation functions are used to examine the randomness, periodicity and correlation inside the loop data; in the frequency domain, power spectrum and cumulative power spectrum display different frequency components inside the loop data and reveal their contribution to the total output error. Before data analysis, data sometimes needs to be preprocessed. The lowpass filter and down-sampling methods were introduced here.

The performance index can be used to indicate quantitatively how well a control loop is operating. There are different methods for calculating the performance index. One way is through an ARMA model, another is via a Laguerre network. They use the same Harris index definition but differ in how to estimate the loop deviation under the minimum variance control. Usually, the Laguerre method is more computationally efficient and easier for on-line implementation.

Loop oscillation is a common phenomenon and has severe effects. In order to removed the oscillation, it must be detected accurately and early. By checking the number of load disturbance occurrences, oscillation can be detected.

Valve friction is one major cause of loop oscillation. For each kind of control valve, its normal input-output function is usually known. By comparing the achieved valve input-output relation to this, a valve friction can be found.

Compared with academic research achievements, industrial applications are lagging behind because of a lack of the application software. The primary purpose of this thesis is to improve the situation. A system based on Windows 95/NT was developed using Visual C++ and MFC. It is a multiple document interface (MDI) application program and can monitor multiple loops simultaneously. Loop data is entered from a Foxboro Data for Windows and then the loop performance is analysed. There are two groups of functions, where the system service function provides management services for plots, files and windows; the performance monitoring function provides operations for performance monitoring and related accessory functions. Preliminary applications have shown the usage of the system.

6.2 Future Work

6.2.1 Related to the Thesis Work

- Time Delay Estimation. Loop time delay is a very important variable for performance monitoring, as both direct process data analysis and performance index calculation depend on it. However, currently there are very few results [16] on accurate estimation of loop time delay.

- On-Line Testing. Although the system has been applied to two application cases, it is not yet implemented on-line. The Foxboro Data for Windows module is not working normally at present, so the system cannot be put into daily use as it cannot gather data constantly via the Foxboro Data for Windows. This however will probably be done in the near future.

6.2.2 General

- Improvements to the Harris index in order to obtain more accurate performance diagnosis.

- Extending the current SISO performance monitoring method to apply to general MIMO systems (although there have been some results for special MIMO systems [13, 10]) and to develop corresponding software system to implement it in industry.

Bibliography

[1] K. J. Åström "Assessment of Achievable Performance of Simple Feedback Loops". *International Journal of Adaptive Control and Signal Processing*, 5(3):3–19, 1991.

[2] K. J. Åström "Limitations on Control System Performance". *IEEE Conference on Decision and Control*, 1997.

[3] W. L. Bialkowski. "Dreams vs. Reality: A View from Both Sides of the Gap". *Proceedings of Control Systems'92*, 1992.

[4] W. L. Bialkowski. "Dynamic Specifications: the Answer to the Control Performance Issue". *Proceedings of Control Systems'92*, 1992.

[5] P. J. Brockwell and R. A. Davis. "Time Series: Theory and Methods". *Springer*, 1991.

[6] D. W. Clarke. "Sensor, Actuator, and Loop Validation". *IEEE Control Systems*, pages 39–45, 1995.

[7] L. Desborough and T. J. Harris. "Performance Assessment Measure for Univariate Feedforward/Feedback Control". *Can. J. Chem. Eng.*, 71:605–616, 1993.

[8] Y. Fu and G. A. Dumont. "Optimum Laguerre Time Scale and its On-Line Estimation". *IEEE Trans. Automat. Contr.*, 38(6):934–938.

[9] T. Hagglund. "Automatic Monitoring of Control Loop Performance". *Control Systems 94, Conference on Control Systems in Pulp and Paper Industry*, 1994.

[10] T. J. Harris and L. Desborough. "Performance Assessment of Multivariable Feedback Controllers". *Proceedings of 1995 AIChE Annual Meeting*, pages 12–17, Nov. 1995.

[11] T.J. Harris. "Assessment of Closed Loop Performance". *Can. J. Chem. Eng.*, 67:856–861, 1989.

[12] S. Haykin. "Nonlinear Methods of Spectral Analysis". *Springer-Verlag*, 1979.

[13] B. Huang and S. L. Shah. "Performance Assessment of Multivariable Control Loops on a Paper-Machine Headbox". *Can. J. Chem. Eng.*, 75:134–142, 1997.

[14] P. Jofriet. "An Expert System for Control Loop Performance". *Pulp and Paper Canada*, (6):70–73, 1996.

[15] L. C. Kammer, R. R. Bitmead, and P. L. Bartlett. "Optimal Controller Properties from Closed-Loop Experiments". *Automatica*, 34(1), 1998.

[16] C. B. Lynch and G. A. Dumont. "Closed Loop Performance Monitoring". *IEEE Trans. Control System Technology*, 4(2):185–192, 1996.

[17] P. Lyonnet. "Tools of total quality: an introduction to statistical process control". *Chapman and Hall*, 1991.

[18] D. Q. Mayne and F. Firoozan. "Linear Estimation of ARMA Systems". *Proceedings of 7th IFAC Conference*, pages 1907–1912, 1978.

[19] R. L. Motard and B. Joseph. "Wavelet Applications in Chemical Engineering". *Kluwer Academic Publishers*, 1994.

[20] S. Ogawa. "A Data Analysis and Graphical Presentation System for Control Loop Performance Assessment". *Process Control Electrical and Information Conference*, pages 483–494, 1998.

[21] J. G. Owen, D. Read, H. Blekkenhorst, and A. A. Roche. "A Mill Prototype for Automatic Monitoring of Control Loop Performance". *Control Systems*, pages 171–178, 1996.

[22] J. G. Proakis and D. G. Manolakis. "Digital Signal Processing". *Prentice Hall*, 1996.

[23] C. Pryor. "Auto-covariance and Power Spectrum Analysis, Derive New Information from Process Data". *Control Eng.*, (2):103–106, 1982.

[24] I. Sommerville. "Software Engineering". *Addison-Wesley Publishing Company*, 1995.

[25] N. Stanfelj. "Monitoring and Diagnosing Process Control Performance: The Sigle-Loop Case". *Ind. Eng. Chem. Res.*, 32:301–314, 1993.

[26] O. Taha, G. A. Dumont, and M. S. Davies. "Detection and Diagnosis of Oscillations in Control Loops". *Proceedings of 35th IEEE Conference on Decision and Control*, pages 1–6, 1996.

[27] G. Taylor. "The Role of Control Valves in Process Performance". *Proceedings of 80th Annual Meeting, Technical Section Canadian Pulp and Paper Association*, 1994.